ulm university universität
**u**ulm

# Zeit-Platz Tradeoffs im Beweissystem der Resolution

Institut für Theoretische Informatik
an der Fakultät für Ingenieurwissenschaften, Informatik und Psychologie
der Universität Ulm

**Masterarbeit**
in Mathematik

zur Erlangung des akademischen Grades
Master of Science

vorgelegt von

**Florian Wörz**
geboren am 17.11.1991 in Ulm-Söflingen

im Juli 2018

| | |
|---|---|
| **Erstgutachter:** | Prof. Dr. Jacobo Torán |
| **Zweitgutachter:** | Prof. Dr. Uwe Schöning |

Private E-Mail-Adresse des Autors: Flo.Woerz@googlemail.com.

Haupttext gesetzt mit 12 pt der Latin Modern und 1,5-fachem Zeilenabstand in LaTeX $2_\varepsilon$.
Grafiken mit TikZ, PGFPlots und Adobe Photoshop erstellt.

*Datum des Satzes:* 2. Juli 2018.

Für Sassa.

“ *We live on an island surrounded by a sea of ignorance. As our island of knowledge grows, so does the shore of our ignorance.* ”

John Archibald Wheeler, *Scientific American 267(3), Sept. 1992*

## Eidesstattliche Erklärung

Hiermit versichere ich an Eides statt, die vorliegende Masterarbeit selbstständig und ohne unzulässige fremde Hilfe, nur unter Verwendung der von mir angegebenen Quellen und Hilfsmittel verfasst zu haben.
Die Arbeit hat in dieser oder vergleichbarer Form noch keinem anderem Prüfungsgremium vorgelegen.

Blaustein, den 05. 07. 2018

Florian Wörz

# Danksagungen

Es ist mir eine besondere Freude, meine Dankbarkeit gegenüber meinem Betreuer Herrn Prof. Dr. Jacobo Torán auszudrücken. Seine Vorlesungen über verschiedenste Themen – von *Quantum Computing* über *Algorithmische Spieltheorie* bis zur *Komplexitätstheorie* – haben mir die Vielfalt der theoretischen Informatik gezeigt, wie es kein anderer Dozent in einem anderen „Teilgebiet der Mathematik“ vermochte. Sein Vorlesungsstil, der häufig eine Brücke zwischen rein formalen Beweisen und einem dazugehörigen Verständnis durch passende Beispiele (an passenden Stellen während des Beweises untergebracht) schlägt, hat den Stil dieser Arbeit maßgeblich mitgeprägt.

Durch Vergabe dieses sehr anspruchsvollen Themas mit den zugehörigen Dissertationen/Papern hat er mir insbesondere einen sehr tiefen Einblick in die (sogar sehr aktuelle) Forschung gegeben und ich konnte am eigenen Leibe erfahren, was es bedeutet, sich gedanklich *an die Grenze des menschlichen Wissens*† zu begeben (das Eingangszitat von John Archibald Wheeler soll hieran erinnern).

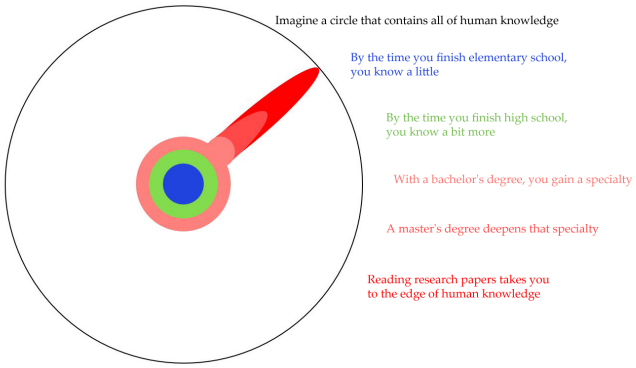



Abbildung 0.1.: Entlehnt von `http://matt.might.net/articles/phd-school-in-pictures/`: „Imagine a circle that contains all of human knowledge. By the time you finish elementary school, you know a little. By the time you finish high school, you know a bit more. With a bachelor's degree, you gain a specialty. A master's degree deepens that specialty. Reading research papers takes you to *the edge of human knowledge*.“

Für hieraus resultierende Fragen aller Art hatte Jacobo immer ein offenes Ohr. Vielen Dank für die vielen Ratschläge, zahlreichen Besprechungen und Ermutigungen – um dies mit den Worten seiner Tochter* zu beschreiben: „He would cheer me on, he would push me, but never too much, so that it stayed fun“.

†Den Ursprung dieser passenden Formulierung habe ich im Comic der Abbildung 0.1 dargestellt.

*Siehe *Surprising My Dad At Boston Marathon*
URL: `https://www.youtube.com/watch?v=_w_vSVR5Vj4&t=32s`.

Zudem danke ich Herrn Prof. Dr. Uwe Schöning, dass er sich bereit erklärt hat, als Zweitgutachter für die vorliegende Arbeit zu fungieren und mir somit und durch die Idee, § 16c der Rahmenprüfungsordnung zu nutzen, ermöglicht hat, meine Masterarbeit in Mathematik in der theoretischen Informatik zu schreiben. Ich hatte die Ehre, bei ihm eine Vorlesung über sat-Solving [ST2013] besuchen zu dürfen, welche mein Interesse an der theoretischen Informatik maßgeblich geprägt hat. Auf geniale Weise führte die Vorlesung einen Balanceakt zwischen Intuition und aktueller Forschung durch. Hierdurch wurden mir die Augen geöffnet, wie Forschung funktioniert.

Auch möchte ich dem Institut für Theoretische Informatik an der Universität Ulm dafür danken, dass mir ermöglicht wurde, den *Workshop on Proof Complexity*, der Teil der *Federated Logic Conference 2018 (FLoC '18)* an der University of Oxford ist, zu besuchen.

Des Weiteren habe ich Herrn Assoc. Prof. Dr. Jakob Nordström zu danken, der mich indirekt durch meinen Erstgutachter auf dem Schloss Dagstuhl Seminar 18051 *Proof Complexity at Schloss Dagstuhl January – February 2018* auf eine Idee zur Beweisvereinfachung in [BNT2013] des ursprünglichen Tradeoff-Resultats aus [BBI2016] aufmerksam machte (siehe hierzu Abschnitt 1.2).

Weiter danke ich Christopher „Chris“ Beck für die persönliche Korrespondenz [Bec2018], die ich mit ihm führen konnte und in der er mir einige Vortragsnotizen über das Thema anvertraut hat.

Einen besonderen Dank verdient Sarah Gegenheimer, für die zahlreichen Zeiten, in denen ich zweifelte, ob das Tradeoff-Resultat mitsamt der Vereinfachung zu einem Ergebnis führen wird. Ohne ihren Beistand wäre diese Arbeit niemals entstanden! Hierfür werde ich ihr für immer dankbar sein: Ich konnte dieser Tatsache durch die Widmung der vorliegenden Arbeit nicht ansatzweise Rechnung tragen.

Außerdem möchte ich meiner Familie für Ihre nie endende Unterstützung während meiner Masterstudienzeit, selbst als ich an der University of Helsinki studiert habe, danken. Speziell während der anstrengenden Zeit beim Schreiben der Masterarbeit war ich für ihre Stütze und ihren moralischen Beistand sehr dankbar.

Abschließend habe ich der Talanx-Stiftung im Stifterverband für die Deutsche Wissenschaft für Ihr dreifaches Stipendium während meines Masterstudiums sowie Google für deren Developer Scholarship zu danken.

# Inhaltsverzeichnis

# KAPITEL 1

# Einleitung

## 1.1 Motivation

Das $\mathsf{P} \stackrel{?}{=} \mathsf{NP}$-Problem [Coo2006] – eines der sieben Millennium-Probleme [CJW2006] des Clay Mathematics Institute – wird von vielen Experten als das tiefgreifendste ungelöste Problem der theoretischen Informatik, gar der ganzen Mathematik bezeichnet. Scott Aaronsons berühmtes Zitat[1] drückt die Signifikanz dieses Problems *anschaulich*[2] aus:

> If $\mathsf{P} = \mathsf{NP}$, then the world would be a profoundly different place than we usually assume it to be. There would be no special value in "creative leaps," no fundamental gap between solving a problem and recognizing the solution once it's found. Everyone who could appreciate a symphony would be Mozart; everyone who could follow a step-by-step argument would be Gauss; everyone who could recognize a good investment strategy would be Warren Buffett.

Unter anderem um dieses Problem zu lösen, wurde das Studium der Beweiskomplexität von Cook und Reckhow in den Jahren 1974/75/79 durch [CR1974a], [Rec1975] und [CR1979] angestoßen: Das Forschungsgebiet untersucht das Konzept der Komplexität vom Standpunkt der Logik ([Pud2008]). Eine typische **theoretische** Fragestellung dieses Gebietes ist „*Welche Theoreme können mit beschränkten Kapazitäten für die Berechnung bewiesen werden?*“ bzw. „*Was ist der kürzeste Beweis eines speziellen Theorems in einem gegebenen formalen System (einem Beweissystem)?*“ ([Bon2018]). Eines der bekanntesten Resultate ist das Cook-Reckhow-Theorem 3.3.9, welches eine Querverbindung zwischen Beweiskom-

[1] Aus: Reasons to believe. In: *Shtetl-Optimized – The Blog of Scott Aaronson.*
URL: `https://www.scottaaronson.com/blog/?p=122`
Siehe auch [AB2009, § 2.7.1 The philosophical importance of NP; § 2.7.3 What if P = NP?] für eine ähnliche Formulierung.

[2] Es sollte angemerkt werden, dass Aaronson seine Aussage später relativiert hat und betont hat, dass es sich um eine heuristische Erklärung handelt: „$\mathsf{P} = \mathsf{NP}$ might make it easier to automate human creativity, but it's far from clear that it's either necessary or sufficient: at any rate, that's a separate discussion. [...] I wasn't talking about the difference between humans and machines; I was talking about the difference between finding and verifying.“

plexität und Komplexitätstheorie herstellt: Superpolynomielle untere Schranken für die Länge von Beweisen aller aussagenlogischer Beweissysteme zu zeigen ist äquivalent damit $\mathsf{NP} \neq \mathsf{co\text{-}NP}$ (also insb. $\mathsf{P} \neq \mathsf{NP}$) zu zeigen. Vor allem in den letzten 10–15 Jahren fand eine gegenseitige „Befruchtung“ zwischen Beweiskomplexität und Disziplinen wie Kryptographie, algebraischer Komplexitätstheorie oder kombinatorischer Optimierung statt, was andeutet, wie vielversprechend dieses Forschungsgebiet ist ([Bon2018]).

Ebenso ist Beweiskomplexität auch aus **praktischer** Sicht äußerst interessant: Sie erlaubt die Analyse der Möglichkeiten und Grenzen von modernsten Algorithmen für das Erfüllbarkeitsproblem der Aussagenlogik, sogenannten SAT-Solvern, die in zahlreichen Anwendungen in der Industrie unerlässlich sind (vgl. [BHMW2009] sowie Seite 28) und Formeln mit mehreren hunderttausend Variablen (trotz der vermuteten Hartnäckigkeit des Problems) routinemäßig lösen müssen. Die oft benutzen komplizierten Heuristiken werden dabei vernachlässigt und das Problem wird zurückgeführt auf die Untersuchung der Beweise (in Form der Niederschrift der Begründungen für die Berechnungen), die diese Algorithmen produzieren – also auf ein Problem in der Beweiskomplexität: Die Untersuchung des zugrundeliegenden Beweissystems. Dies stellt oft die einzige Möglichkeit zur Untersuchung der Algorithmen bzw. Algorithmenfamilien dar (vgl. [Bec2017]).

Untere (sogar scharfe) Abschätzungen für die Laufzeit von Algorithmen (bzw. der Beweislänge) konnten bereits für eine Vielzahl von expliziten, harten, UNSAT-CNF-Formeln gezeigt werden. Beispielhaft sind die exponentiellen unteren Schranken u. a. aus [Hak1985, Urq1987, CS1988] zu erwähnen (siehe Abschnitt 4.2). Diese unteren Schranken implizieren, dass jeder optimale (sogar nicht-deterministische) Algorithmus, der auf diesen Beweissystemen beruht, selbst mit allen richtigen Entscheidungen zwingend mindestens den Anteil der in diesen Schranken spezifizierten Berechnungsressourcen benötigt (siehe [Nor2013]).

Esteban und Torán formalisierten in [ET2001] schließlich den Begriff des Platzes für Resolutionsbeweise (siehe Abschnitt 4.1 für historische Bemerkungen). Unter alleiniger Betrachtung dieses Komplexitätsmaßes konnten ebenfalls scharfe untere und obere Schranken von $\Theta(n)$ für explizite UNSAT-CNF-Formeln der Größe $n$ gezeigt werden: Untere Schranken finden sich z. B. in den Artikeln [ET2001, ABRW2002, AD2008]. Obere Schranken ergeben sich aus dem Fakt, dass alle UNSAT-Formeln mit $n$ Variablen baumartige Resolutionswiderlegungen mit Platz höchstens $n$ haben ([ET2001]). Für die Praxis sind diese baumartigen Widerlegungen jedoch typischerweise zu lang ([Nor2015, Bec2018]), sodass die Frage sinnvoll erscheint, ob eine Widerlegung mit Benutzung von wenig Platz und *simultan* kurzer praktikabler Länge möglich ist.

Eine relativ neue Forschungsrichtung im Gebiet der Beweiskomplexität beschäftigt sich daher mit sog. *Zeit-Platz Tradeoffs* für verschiedene Beweissysteme: Für moderne SAT-

Solver stellt Platz ebenso wie Zeit einen Kapazitätsengpass dar (vgl. auch [ST2013, § 6.3], [JMNŽ2012]) – speziell für clause learning Algorithmen, z. B. `GRASP` aus [MS1999], die eine Vielzahl von abgeleiteten Klauseln während der Berechnung im Speicher behalten müssen. Die Balance zwischen diesen zwei Komplexitätsmaßen bestimmt den Erfolg dieser sehr erfolgreichen conflict-driven clause learning (`CDCL`) Algorithmen, die sich beginnend u. a. mit [MMZ+2001] und [ZMMM2001] aus den ebenfalls sehr erfolgreichen `DPLL`-Algorithmen entwickelt haben: Schnellere Laufzeiten (um `DPLL` zu „schlagen“) ziehen möglicherweise die Notwendigkeit für exorbitant großen Speicherplatz nach sich – durch passende Heuristiken wird versucht die „*richtigen*“ Klauseln im Speicher zu behalten (siehe bspw. [ST2013, § 4.4.1, S. 138]).

Eine äußerst praktische Fragestellung ist daher, wie es [BBI2016] formuliert, ob diese Zeit-Platz Tradeoffs ein grundsätzliches „Handicap“ auf Resolution basierender Algorithmen darstellen (dies beinhaltet `DPLL`/`CDCL`-Solver unter Ausnutzung bekannter Korrespondenzen zwischen diesen Algorithmen und dem Beweissystem der Resolution, wie z. B. in [BKS2004, PD2010, PD2011, ST2013] und Abschnitt 4.3 dargelegt), oder lediglich ein „Artefakt“ einzelner Algorithmen sind (d. h. durch eine Anpassung der Heuristik evtl. überwunden werden können). Untere Zeit/Platz-Schranken aus der Beweiskomplexität werden durch Ausnutzung dieser Korrespondenz zu entsprechenden unteren Schranken der Algorithmen – die Tradeoffs sind also ein grundlegendes „Handicap“ dieser Algorithmen.

Aufbauend auf [Hak1985, ABRW2002, BG2003, BN2008, Nor2009, BN2011, Nor2013] konnte in [BBI2016] der erste Zeit-Platz Tradeoff für superlinearen Platz im Beweissystem der Resolution gezeigt werden. Superlinear bedeutet, dass wir es erlauben, mehr Platz zu benutzen, als die Eingabeformel einnimmt – was für SAT-Solver der Fall ist. Deshalb stellt dies das erste Tradeoff-Resultat dar, das eine derart hohe Praxisrelevanz und Aussagekraft für SAT-Solver hat. Hiermit wurde zudem folgende theoretische Frage von Ben-Sasson ([Ben2007, NH2013], [Nor2013, Open Problem 17]; vgl. Problem 7.0.2) *negativ* beantwortet: „*Do all* CNF *formulas have resolution proofs of linear space that are at most polynomially larger than the resolution proof length possible when space is unrestricted?*“ Formaler:

> Does there exist [a constant] $c$ such that any CNF [formula in $N$ variables] with a [shortest] refutation of [length] $T$ also has a refutation of [length] $T^c$ in space $\mathcal{O}(N)$?

Unsere Konstruktion eines Gegenbeispiels in dieser Arbeit modifiziert und vereinfacht dabei den Beweis aus [BBI2016] in Haupttheorem 7.0.1 unter Benutzung von Ideen aus [BNT2013] für das Beweissystem `PCR` (Polynomial Calculus Resolution), wodurch das Tradeoff-Resultat schließlich verbessert werden konnte.

Im nachfolgenden Abschnitt haben wir unseren Beitrag genau erörtert.

## 1.2 Unser Beitrag

Durch die vorliegende Arbeit verbessern wir das in [BBI2016] erzielte erste Tradeoff-Resultat für Zeit und superlinearen Platz im Beweissystem der Resolution, inspiriert durch die angedeuteten Ideen für das Beweissystem PCR (Polynomial Calculus Resolution) aus [BNT2013] unter Zuhilfenahme der Ausführungen in [Bec2017].

- Speziell modifizieren wir die für den Tradeoff benötigten Tseitin-Formeln durch eine geschickte Änderung der Konstruktion der ihr zugrunde liegenden Graphen. Durch diese Ideen aus [BNT2013, Bec2017] erhalten wir ein Tradeoff-Resultat für 8-CNF-Formeln, was eine signifikante Verbesserung gegenüber der ehemals benötigten *unbeschränkten* Weite der Formeln in [BBI2016] darstellt.
- Die obige Konstruktionsänderung erlaubt uns zudem, die technische Analyse drastisch zu vereinfachen, was einen klareren Blick auf die eigentliche Beweistechnik des Tradeoffs erlaubt.
- Weiterhin verallgemeinern wir die in [BNT2013] erzielten Resultate mithilfe der Ideen aus der erst kürzlich veröffentlichten Dissertation [Bec2017]: Hierdurch gilt unser Tradeoff-Resultat nicht nur für eine spezifische Familie von Graphen wie in [BBI2016], sondern für alle Graphenfamilien, denen eine erweiterte isoperimetrische Eigenschaft zugrunde liegt.
- Durch genauere Abschätzungen und exakterer Verwendung der $\mathcal{O}$-Notation gelingt es uns abschließend [BNT2013, Theorem 4] vom Beweissystem PCR auf Resolution zu übertragen und in Haupttheorem 7.0.1 (i) zu verschärfen. Hierin einbezogen sind einige Korrekturen von Nachlässigkeiten im Originalbeweis der Proposition 7.1.11 aus [BBI2016].
- Durch ein sorgfältigeres Studium der erweiterten Isoperimetrieeigenschaften von Gittergraphen als in [BNT2013, Bec2017], gelang es uns, die schwerwiegenden Fehler in der Definition der für erweiterte Isoperimetrie benötigten Knotenmengen mittlerer Größe zu korrigieren, sie für alle benötigten Graphengrößen wohldefiniert zu gestalten, und durch Anwendung von Techniken aus der Analysis den Isoperimetrie-Beweis zu verbessern und wiederherzustellen (siehe hierzu Kommentar 7.3.29). Hierzu war es nötig, zahlreiche Argumentationsfehler in der Dissertation [Bec2017] zu beheben. Schließlich führen wir alle hierdurch notwendigen Beweisänderungen der darauf folgenden Resultate bzw. Modifikationen der benötigten Definitionen durch. Wir verweisen insbesondere auf Kommentar 7.3.29, der zeigt, dass die in [BNT2013, Bec2017] verwendeten Parameter des Gittergraphen zu Problemen im Beweis führen.

- Des Weiteren konnten wir die minimale Größe der benutzten Graphen zur Konstruktion der Tseitin-Formeln im Kontrast zu [BBI2016] um einen polynomiellen Faktor verkleinern: Unser Resultat gilt für Gittergraphen der Länge $8n^3 \leq \ell \leq 2^n$, während in [BBI2016] Parameter $2n^4 \leq \ell < (4n)^{-1} \left(\frac{9}{8}\right)^n$ nötig waren. Das Ergebnis in Haupttheorem 7.0.1 gilt zudem für alle $n \in \mathbb{N}_{\geq 5}$, was die Aussage „if $n$ is sufficiently large“ in [BBI2016, Theorem 15] zusätzlich konkretisiert.
- Eine leichte Unsauberkeit in [BNT2013, Bec2017] wird durch eine Modifikation der Epocheneinteilung beseitigt. Detaillierte Ausführungen hierzu haben wir auf Seite 128f. gegeben.
- Zahlreiche Grafiken und Beispiele erleichtern das Verständnis der Hauptideen der Beweise.

Wir verweisen für weitere Details zu Korrekturen auf Kommentar 7.3.29.

## 1.3 Gliederung der Arbeit

Der Hauptteil der Arbeit ist in zwei große Teile gegliedert:

- Im ersten Teil, bestehend aus den Kapiteln 3 und 4, liefern wir eine umfangreiche Einführung in die wichtigsten Konzepte der Beweiskomplexität und motivieren das Thema Tradeoffs.
- Im zweiten Teil der Arbeit, bestehend aus den Kapiteln 5–7, wollen wir schließlich den Zeit-Platz Tradeoff im Beweissystem der Resolution beweisen, wofür wir zu Beginn in zwei einführenden Kapiteln die notwendigen Grundlagen liefern, um eine schnelle Formulierung und einen stringenten Beweis des Theorems im darauffolgenden Kapitel zu ermöglichen.

Wir schließen mit dem letzten Teil der Arbeit, der Appendices und Verzeichnisse beinhaltet. Im Anhang, bestehend aus den Kapiteln A–D, stellen wir grundlegendes Hintergrundmaterial für den Komfort des Lesers zusammen. Das Material des Anhangs kann in vielen Standardwerken zur theoretischen Informatik gefunden werden, ist aber aufgrund der thematischen Vielfalt teilweise über viele Fundstellen verteilt. Wir empfinden es daher als nützlich, eine gemeinsame Referenz für diese Resultate im Anhang bereitzustellen.

Nachfolgend geben wir eine kurze Beschreibung der einzelnen Kapitel:

**Kapitel 2** Vor dem Hauptteil der Arbeit erörtern wir im nachfolgenden Kapitel die in der Arbeit verwendeten Notationen und Begrifflichkeiten. Dies erleichtert den Einstieg in den Hauptteil durch Erinnerung an alle relevanten Konzepte der Komplexitätstheorie, Logik und Graphentheorie. Einem in der theoretischen Informatik versierten Leser ist es möglich, dieses Kapitel zu überspringen.

### Teil I

**Kapitel 3** Wir geben eine Einführung in die Beweiskomplexität, motivieren typische Forschungsfragen des Gebietes, stellen Beispiele für Beweissysteme vor und führen insbesondere das Beweissystem der Resolution sowie verschiedene Repräsentationsarten eines Resolutionsbeweises ein.

**Kapitel 4** Zunächst werden die Komplexitätsmaße Zeit und Platz im Beweissystem der Resolution sauber und ausführlich eingeführt. Untere und obere Schranken für Komplexitätsmaße sowie der Zusammenhang zwischen Komplexitätsmaßen und SAT-Solvern geben eine erste Motivation für das Studium von Zeit-Platz Tradeoffs. Es folgt eine kurzes Beispiel eines Tradeoff-Resultates außerhalb der Beweiskomplexität und eine Vorstellung von ausgewählten bekannten Tradeoffs in der Beweiskomplexität.

### Teil II

**Kapitel 5** Der sauberen Einführung von Tseitin-Formeln mit zahlreichen Beispielen und den später wichtigen Aussagen über diese Formeln ist dieses Kapitel gewidmet.

**Kapitel 6** Für die Formulierung des späteren Tradeoff-Resultates benötigen wir noch einige Hilfswerkzeuge: Wir studieren daher in diesem Kapitel die benötigten Graphen, Substitutionsformeln bzw. die XORification von Formeln und Zufallseinschränkungen von Formeln in einem erweiterten Sinne.

**Kapitel 7** Dieses Kapitel beinhaltet Haupttheorem 7.0.1, welches unser gewünschter Zeit-Platz Tradeoff ist. Wir beweisen zunächst die im Haupttheorem behaupteten oberen Schranken in zwei eigenen Abschnitten, bevor wir den Tradeoff in einem weiteren Abschnitt zeigen. Dies geschieht der Übersichtlichkeit halber in fünf Einzelschritten.

### Appendix und Verzeichnisse

**Anhang A** Wir setzen die Sprachen TAUT, SAT und UNSAT in einen Zusammenhang.

**Anhang B** Wir geben zwei Charakterisierungen der Komplexitätsklasse NP und erörtern den Begriff der polynomiell many-one-Reduzierbarkeit. Wir zeigen zudem mögliche Beziehungen von P, NP und co-NP.

**Anhang C** Wir erörtern, wie jede Formel durch die Tseitin-Transformation in eine satäquivalente 3-CNF-Formel umgewandelt werden kann.

**Anhang D** Um Beweissysteme untereinander zu vergleichen, ist ein umfangreiches Wissen über verschiedene Relationsarten unumgänglich. Besonders wichtig ist der Begriff der kanonischen Partialordnung zu einer gegebenen Quasiordnung, den wir in diesem Anhang erläutern.

# KAPITEL 2

## Präliminarien und Notationen

An Paul R. Halmos Worte „*The beginner ... should not be discouraged if ... he finds that he does not have the prerequisites for reading the prerequisites*“ erinnernd, wollen wir in diesem einführenden Kapitel zunächst die verwendeten Notationen und Bezeichnungen klären und festhalten, dass wir Kenntnisse aus einführenden Logik- und Komplexitätstheorie-Vorlesungen als bekannt voraussetzen werden.

Alle in dieser Arbeit auftretenden Symbole und Bezeichner sind entweder im Verlaufe dieses Kapitels oder an der Stelle ihres ersten Auftretens explizit definiert. Wir verweisen auch auf das Symbolverzeichnis ab Seite 183.

Wir verwenden Standardabkürzungen der deutschen Sprache sowie die gängigen mathematischen Abkürzungen *o. B. d. A.* für *ohne Beschränkung der Allgemeinheit*, sowie *m. a. W.* für *mit anderen Worten*.

Unsere Notationen und Darstellungen orientieren sich vor allem an [AB2009, Bru2014, Dal2013, HL2011, Hof2007, Nor2001, Nor2008, Pap1994, Sch2013, Sch2018b, ST2013], sowie [Tor2017, Vol2013].

### Mengen und Körper

Mit $\mathbb{N}$ sei die Menge der *natürlichen Zahlen* $1, 2, 3, \ldots$ bezeichnet. Wir setzen $\mathbb{N}_0 := \mathbb{N} \cup \{0\}$. Ist $m \in \mathbb{N}$, so setzen wir $\mathbb{N}_{\geq m} := \{n \in \mathbb{N} : n \geq m\}$. Für $n \in \mathbb{N}$ setzen wir $[n] := \{1, \ldots, n\}$. Die Menge der *ganzen Zahlen* $\mathbb{Z}$ ist definiert über $\mathbb{Z} := \mathbb{N}_0 \cup \{-n : n \in \mathbb{N}\}$.

Die Menge der *rationalen Zahlen* wird mit $\mathbb{Q}$ bezeichnet. Die Menge der *reellen Zahlen* bezeichnen wir mit $\mathbb{R}$, die der *positiven reellen Zahlen* mit $\mathbb{R}^+ := \{x \in \mathbb{R} \mid x > 0\}$, die der *nicht-negativen reellen Zahlen* mit $\mathbb{R}_0^+ := \{x \in \mathbb{R} \mid x \geq 0\}$. Die *komplexen Zahlen* bilden die Menge $\mathbb{C}$, wobei wir die *imaginäre Einheit* mit *aufrechtem* i bezeichnen.

Mit $\mathbb{F}_2 := \mathrm{GF}(2) = \mathbb{Z}/2\mathbb{Z}$ bezeichnen wir den (bis auf Isomorphie) eindeutigen *Körper mit zwei Elementen*, also den Körper der Restklassen ganzer Zahlen modulo 2. Dabei repräsentiere 0 die Restklasse $2\mathbb{Z}$ der geraden Zahlen und 1 die Restklasse $1 + 2\mathbb{Z}$ der ungeraden Zahlen.

Das Symbol $\uplus$ bezeichne die *disjunkte Vereinigung* von Mengen. Sind $A$ und $B$ zwei

Mengen, so verwenden wir die Schreibweise $A \subset B$, um eine (*möglicherweise nicht echte*) *Mengeninklusion* zu notieren, d. h. für alle $x \in A$ gilt auch $x \in B$. Eine entsprechende echte Inklusion notieren wir mit $A \subsetneq B$. Ist $M$ eine Menge, so bezeichnen wir mit $\binom{M}{k}$ die *Menge aller $k$-elementigen Teilmengen* von $M$. Mit $\mathfrak{P}(M)$ notieren wir die *Potenzmenge* einer Menge $M$.

### Folgen

Ist $(x_i)_{i\in I}$ eine Folge, $I$ eine Indexmenge, und $X$ eine beliebige Menge, so folgen wir dem gängigen Notationsmissbrauch und schreiben $(x_i)_{i\in I} \subset X$, wenn wir meinen, dass $(x_i)_{i\in I}$ eine *Folge in $X$* ist. Formal korrekter wäre $\{x_i \mid i \in I\} \subset X$.

### Asymptotische Notation durch Landau-Symbole

Es sei $f\colon \mathbb{N} \to \mathbb{R}_0^+$ eine Funktion. Mit $\mathcal{O}(f(n))$ bezeichnen wir die Klasse der Funktionen $g\colon \mathbb{N} \to \mathbb{R}_0^+$, für die es ein $c \in \mathbb{R}^+$ und ein $N \in \mathbb{N}$ gibt, sodass für alle $n \geq N$ die Ungleichung $g(n) \leq cf(n)$ gilt, in Zeichen

$$\mathcal{O}(f(n)) := \{g\colon \mathbb{N} \to \mathbb{R}_0^+ \mid \exists c \in \mathbb{R}^+ \ \exists N \in \mathbb{N} \ \forall n \geq N : \ g(n) \leq cf(n)\}.$$

Entsprechend definieren wir die Klassen

$$\begin{aligned}
\Omega(f(n)) &:= \{g\colon \mathbb{N} \to \mathbb{R}_0^+ \mid \exists c \in \mathbb{R}^+ \ \exists N \in \mathbb{N} \ \forall n \geq N : \ g(n) \geq cf(n)\},\\
o(f(n)) &:= \{g\colon \mathbb{N} \to \mathbb{R}_0^+ \mid \forall c \in \mathbb{R}^+ \ \exists N \in \mathbb{N} \ \forall n \geq N : \ g(n) \leq cf(n)\},\\
\omega(f(n)) &:= \{g\colon \mathbb{N} \to \mathbb{R}_0^+ \mid \forall c \in \mathbb{R}^+ \ \exists N \in \mathbb{N} \ \forall n \geq N : \ g(n) \geq cf(n)\},
\end{aligned}$$

sowie

$$\Theta(f(n)) := \mathcal{O}(f(n)) \cap \Omega(f(n)).$$

Ist eine Funktion $g$ Element einer dieser Klassen, bspw. $g(n) \in \mathcal{O}(f(n))$, so folgen wir dem weit verbreiteten Notationsmissbrauch und schreiben $g(n) = \mathcal{O}(f(n))$. Ebenso benutzen wir obige Notation in Abschätzungen, welche nur von links nach rechts gelesen werden können.

### Asymptotische Terminologie

Eine Funktion $f$ ist *polynomiell* in $n$, falls $f(n) = \mathcal{O}(n^k)$ für ein $k \in \mathbb{N}$ gilt. Auch hier folgen wir dem gängigen Notationsmissbrauch und schreiben für die entsprechende obere Schranke $f(n) = n^{\mathcal{O}(1)}$. Hierbei ist der Exponent $\mathcal{O}(1)$ als Kurzschreibweise für eine nicht-negative Konstante oder als durch eine Konstante beschränkte Funktion zu interpretieren.

Ebenfalls schreiben wir $f(n) \leq \mathrm{poly}(n)$, falls ein Polynom $p\colon \mathbb{N} \to \mathbb{R}_0^+$ existiert, sodass $f(n) = \mathcal{O}(p(n))$ gilt. Dabei sind wir nicht an dem genauen Polynom interessiert und

schreiben in (Un)gleichungsketten auch mehrfach poly(·), obwohl wir damit ggfs. unterschiedliche Polynome meinen – der große Vorteil hierin liegt an der leichteren Lesbarkeit dieser (Un)gleichungsketten, da polynomielle Faktoren „absorbiert“ werden (man vergleiche diese Schreibweise mit dem „Absorbieren“ von Faktoren in einer Konstante $C$ von Ungleichungszeichen zu Ungleichungszeichen bei Abschätzungen in der Analysis).

Eine Funktion $f$ ist *polylogarithmisch* in $n$, falls $f(n) = \mathcal{O}\Big((\log n)^k\Big)$ für ein $k \in \mathbb{N}$ gilt. Wir schreiben hierfür auch $f(n) \leq \operatorname{polylog}(n)$ ähnlich wie oben. Alle polylogarithmischen Funktionen sind $o(n^\varepsilon)$ für alle $\varepsilon > 0$, d. h. eine polylogarithmische Funktion wächst langsamer als jede Funktion $n^\varepsilon$ für einen positiven Exponenten $\varepsilon$.

Weiterhin nennen wir die Funktion $f$

- *sublinear* in $n$, falls $f(n) = o(n)$ gilt,
- *superlinear* in $n$, falls $f(n) = \omega(n)$ gilt,
- *quasi-* oder *pseudo-polynomiell* in $n$, falls $f(n) = \mathcal{O}\left(\exp\left((\log n)^k\right)\right)$ für ein $k \in \mathbb{N}$ gilt, was wir ebenfalls kurz als $f(n) = \exp\left((\log n)^{\mathcal{O}(1)}\right)$ schreiben wollen,
- *superpolynomiell* in $n$, falls $f(n) = \omega(n^k)$ für alle $k \in \mathbb{N}$ ist,
- *exponentiell* in $n$, falls $f(n) = \Omega\Big(\exp(n^\delta)\Big)$ für ein $\delta \in \mathbb{R}^+$ gilt.

### Formale Sprachen

Es sei $\Sigma = \{a, b, \dots\}$ ein endliches *Alphabet* (d. h. eine endliche Menge von Symbolen). Ein *endliches Wort* über $\Sigma$ ist eine Folge $w = a_1 \dots a_n$, wobei $a_i \in \Sigma$ für $i = 1, \dots, n$ ist. Wir schreiben $|w|$ für die *Länge* von $w$, d. h. für die Anzahl der Symbole im String $w$, also $n$ in obigem Fall. Das *leere Wort* bezeichnen wir mit $\varepsilon$ und meinen damit die Folge der Länge 0. Die *Menge aller endlichen Wörter* über $\Sigma$, die *Kleenesche Hülle* von $\Sigma$, bezeichnen wir mit $\Sigma^*$ (das leere Wort ist ebenfalls Element dieser Menge). Die *positive Hülle* $\Sigma^+$ einer Sprache $\Sigma$ ist die Menge aller endlichen Wörter exklusive des leeren Wortes.

Eine *Sprache* bzw. ein *Entscheidungsproblem* über $\Sigma$ ist eine Teilmenge $L \subset \Sigma^*$. Das *Komplement* einer Sprache $L$ definieren wir durch $\overline{L} := \Sigma^* \setminus L$.

### Boolesche Variablen und Wahrheitswerte

Wir nehmen an, dass eine abzählbar unendliche Menge $\mathcal{V}$ von *Booleschen Variablen* existiert, die Werte aus $\{\mathtt{0}, \mathtt{1}\}$ annehmen können (vgl. [Sch2018b, Definition 2.2]). Für gewöhnlich bezeichnen wir Boolesche Variablen in dieser Arbeit mit $x, y, z, x_1, x_2, \dots$.

Die Elemente der Menge $\{\mathtt{0}, \mathtt{1}\}$ wollen wir *Wahrheitswerte* nennen. Wir identifizieren das Element `0` mit `False` und `1` mit `True`, was die Bezeichnung „Wahrheitswerte“ erklärt.

### Boolesche Formeln und Variablen

Wir definieren den Begriff *Boolesche Formel* bzw. *aussagenlogische Formel* (engl. *propositional formula*) oder auch kurz *Formel* induktiv:

1. Jede Variable aus der Menge $\mathcal{V}$ ist eine Boolesche Formel.
2. Ist $F$ eine Boolesche Formel, so ist auch $\neg F$ eine Boolesche Formel. Statt $\neg F$ werden wir auch $\overline{F}$ schreiben.
3. Sind $F$ und $G$ Boolesche Formeln, so sind auch $F \vee G$, sowie $F \wedge G$ Boolesche Formeln.
4. Die Konstanten[3] 0 und 1 sind Boolesche Formeln.

Dabei nennen wir die Symbole $\neg, \vee, \wedge$ *logische Operatoren* (engl. *connectives*). Es sei angemerkt, dass wir Hilfssymbole wie Klammern und Kommata ebenfalls zulassen. Standardgemäße Vereinfachungen zum Weglassen der Klammern, wie sie in jedem Lehrbuch zur Logik erörtert werden, verwenden wir, ohne sie hier im Detail zu erklären. Sind $F$ und $G$ zwei Formeln, wollen wir zudem die Schreibweise $F \to G$ als Abkürzung für die Formel $\neg F \vee G$ verwenden. In analoger Weise sei $F \leftarrow G$ definiert. Wir schreiben $F \leftrightarrow G$, falls $F \to G$ und $F \leftarrow G$ gilt. Des Weiteren verwenden wir $F \oplus G$ als Abkürzung für $\neg(F \leftrightarrow G)$. Schließlich bezeichnen wir die *Menge der aussagenlogischen Formeln* (über $\mathcal{V}$) mit PROP.

Mit $\mathrm{Var}(F)$ bezeichnen wir die Menge der Variablen, die in der Booleschen Formel $F$ vorkommen. Die formale Definition erfolgt erneut induktiv:

1. $\mathrm{Var}(0) = \mathrm{Var}(1) = \varnothing$.
2. $\mathrm{Var}(x) = \{x\}$ für $x \in \mathcal{V}$.
3. $\mathrm{Var}(\neg F) = \mathrm{Var}(F)$.
4. $\mathrm{Var}((F \vee G)) = \mathrm{Var}(F) \cup \mathrm{Var}(G), \quad \mathrm{Var}((F \wedge G)) = \mathrm{Var}(F) \cup \mathrm{Var}(G)$.

**Wahrheitsbelegungen, Einschränkungen und Logik**

Eine *Belegung* (engl. *truth assignment*) $\alpha$ einer Formel ist eine Abbildung

$$\alpha\colon \mathrm{Dom}(\alpha) \to \{\mathtt{0}, \mathtt{1}\},$$

wobei der Definitionsbereich von dieser Abbildung $\mathrm{Dom}(\alpha) \subset \mathcal{V}$ eine endliche Teilmenge der Booleschen Variablen ist. Angelehnt an unsere bisherige Notation bezeichnen wir mit $\mathrm{Var}(\alpha) := \mathrm{Dom}(\alpha)$ den Definitionsbereich von $\alpha$, also diejenigen Variablen aus $\mathcal{V}$, denen ein Wert unter $\alpha$ zugeordnet wird.

Wir notieren eine Belegung $\alpha$ mit $\mathrm{Var}(\alpha) = \{x_1, \ldots, x_k\}$ durch die Mengenschreibweise $\alpha = \{x_1 = \alpha(x_1), \ldots, x_k = \alpha(x_k)\}$.

Falls $\alpha(x) = \mathtt{1}$ für eine Variable $x \in \mathrm{Var}(\alpha)$ gilt, sprechen wir davon, dass $x$ durch $\alpha$ auf `True` gesetzt wird. Gilt entsprechend $\alpha(x) = \mathtt{0}$, wird $x$ von $\alpha$ auf `False` gesetzt. Andernfalls

[3] Präziser wählen wir eine beliebige Variable $x_0 \in \mathcal{V}$ und setzen $0 := x_0 \wedge \neg x_0$ sowie $1 := x_0 \vee \neg x_0$. Es gibt durchaus andere Wege diese Konstanten zu definieren. Dabei ist lediglich darauf zu achten, dass die Formel 0 unerfüllbar und die Formel 1 eine Tautologie ist (wir verweisen auf die kommenden Unterabschnitte für die Definitionen dieser Begriffe). Da es unendlich viele Elemente in $\mathcal{V}$ gibt, können wir o. B. d. A. annehmen, dass 0 und 1 die einzigen Formeln sind, welche die Variable $x_0$ enthalten.

wird $x$ durch $\alpha$ nicht gesetzt.

Es sei $F$ eine Formel und $\alpha$ eine Belegung. Falls $\mathrm{Var}(\alpha) = \mathrm{Var}(F)$ gilt, also die Belegung $\alpha$ genau für die in $F$ vorkommenden Variablen definiert ist, nennen wir $\alpha$ zu $F$ *passend*. Weiter heißt $\alpha$ *total* für die Formel $F$, falls $\mathrm{Var}(F) \subset \mathrm{Var}(\alpha)$ gilt. Um zu betonen, dass eine Belegung $\alpha$ nicht total für eine Formel $F$ ist, nennen wir $\alpha$ *partiell* für $F$.

Ist $\alpha$ eine totale Belegung für $F$, so bezeichnen wir mit $F\alpha \in \{\mathtt{0}, \mathtt{1}\}$ den *Wahrheitswert* von $F$ unter der Belegung $\alpha$ und sprechen davon, dass wir $\alpha$ auf $F$ *angewandt* haben. Diesen Wahrheitswert definieren wir für eine totale Belegung $\alpha$ erneut induktiv, in Anlehnung an [Sch2013], [Sch2018b, Definition 2.6]:

1. Es ist $0\alpha = \mathtt{0}$ und $1\alpha = \mathtt{1}$ nach Fußnote 3.
2. Ist $F = x$ für ein $x \in \mathcal{V}$, dann ist $F\alpha = \mathtt{1}$, falls $\alpha(x) = \mathtt{1}$ ist, andernfalls ist $F\alpha = \mathtt{0}$.
3. Ist $F = \neg G$, dann ist $F\alpha = \mathtt{1}$, falls nicht $G\alpha = \mathtt{1}$ (also $G\alpha = \mathtt{0}$) gilt, andernfalls ist $F\alpha = \mathtt{0}$.
4. Ist $F = (G_1 \vee G_2)$, dann ist $F\alpha = \mathtt{1}$, falls $G_1\alpha = \mathtt{1}$ oder $G_2\alpha = \mathtt{1}$ gilt, andernfalls ist $F\alpha = \mathtt{0}$.
5. Ist $F = (G_1 \wedge G_2)$, dann ist $F\alpha = \mathtt{1}$, falls $G_1\alpha = \mathtt{1}$ und $G_2\alpha = \mathtt{1}$ gilt, andernfalls ist $F\alpha = \mathtt{0}$.

Ist $\alpha$ partiell für $F$, vereinbaren wir, dass nach den jeweiligen symbolischen Ersetzungen der entsprechenden Variablen aus $\mathrm{Var}(\alpha)$ in $F$ durch die Konstanten 0 (falls die Variable durch $\alpha$ auf $\mathtt{0}$ gesetzt wird) und 1 (andernfalls) einfache Vereinfachungen durchgeführt werden:

1. $(G \vee 0)$ bzw. $(0 \vee G)$ wird vereinfacht zu $G$,
2. $(G \vee 1)$ bzw. $(1 \vee G)$ wird vereinfacht zu 1,
3. $(G \wedge 0)$ bzw. $(0 \wedge G)$ wird vereinfacht zu 0,
4. $(G \wedge 1)$ bzw. $(1 \wedge G)$ wird vereinfacht zu $G$,
5. $\neg\neg G$ wird vereinfacht zu $G$,
6. $\neg 0$ wird vereinfacht zu 1,
7. $\neg 1$ wird vereinfacht zu 0.

Mit $F{\restriction_\alpha}$ bezeichnen wir die im obigen Sinne definierte *eingeschränkte Formel* und nennen $\alpha$ eine *Einschränkung* von $F$ (wir weisen an dieser Stelle bereits darauf hin, dass wir diesen Begriff in Definition 6.0.10 neu fassen werden). Für eine alternative Definition verweisen wir bspw. auf [Rac2007, S. 11].

Wir sagen, dass eine Formel $F$

- *erfüllbar* ist, falls es eine passende Belegung $\alpha$ gibt, die $F$ *erfüllt*, also $F\alpha = \mathtt{1}$ gilt. In diesem Fall heißt $\alpha$ *erfüllende Belegung* oder *Modell* für $F$. Wir wollen dies auch mit $\alpha \vDash F$ notieren.

- *gültig*, *tautologisch* oder eine *Tautologie* ist, falls alle passenden Belegungen $F$ erfüllen. Wir schreiben hierfür suggestiv $\vDash F$, da die genaue Belegung irrelevant ist.
- *falsifizierbar*, falls es eine passende Belegung $\alpha$ gibt, die $F$ *falsifiziert*, also $F\alpha = \mathsf{0}$ gilt.
- *unerfüllbar*, *widersprüchlich* oder *widerspruchsvoll*, falls alle passenden Belegungen die Formel $F$ falsifizieren.

Anstatt von einer Implikation $\Big((G_1 \wedge \cdots \wedge G_n) \to H\Big)$ zu sagen, dass sie eine Tautologie ist, werden wir auch sagen, dass $H$ eine *Folgerung* (oder *semantische Implikation*) von $(G_1 \wedge \cdots \wedge G_n)$ ist und dies mit $(G_1 \wedge \cdots \wedge G_n) \vDash H$ notieren. Ist $\Gamma = \{G_1, \ldots, G_n\}$ eine Menge von Formeln, so schreiben wir entsprechend $\Gamma \vDash H$, falls jede Wahrheitsbelegung, die alle $G \in \Gamma$ erfüllt, auch $H$ erfüllt. Sind sowohl $\Gamma = \{G_1, \ldots, G_n\}$ als auch $\Omega = \{H_1, \ldots, H_m\}$ Mengen von Formeln, so dehnen wir obige Notation aus und schreiben $\Gamma \vDash \Omega$, falls jede Wahrheitsbelegung, die alle $G \in \Gamma$ erfüllt, auch alle $H \in \Omega$ erfüllt.

Wir nennen zwei Formeln $F$ und $G$ (*semantisch*) *äquivalent*, in Zeichen $F \equiv G$, falls $F\alpha = G\alpha$ für alle Belegungen $\alpha$ mit der Eigenschaft $\mathrm{Var}(\alpha) = \mathrm{Var}(F) \cup \mathrm{Var}(G)$ gilt (d. h. die Belegung $\alpha$ ist total sowohl für $F$ als auch für $G$).

### Literale

Eine Formel $\ell$ heißt *Literal*, falls es eine Variable $x \in \mathcal{V}$ gibt, sodass $\ell = x$ oder $\ell = \neg x$ ist. D. h. ein Literal bezeichnet entweder eine Variable $x$ oder die entsprechende negierte Variable $\neg x =: \overline{x}$. Wir vereinbaren zudem die Notationen $x^1 := x$ und $x^0 := \overline{x}$. Es ist außerdem nützlich $\overline{\overline{x}} := x$ zu setzen. Entsprechend bezeichnen wir eine Variable $x$ als *positives Literal* und eine negierte Variable $\neg x$ als *negatives Literal*. Für ein Literal $\ell$ vereinbaren wir die Schreibweise

$$\overline{\ell} := \begin{cases} \neg x, & \text{falls } \ell = x \\ x, & \text{falls } \ell = \neg x. \end{cases}$$

### Konjunktive und Disjunktive Normalform sowie Klauselschreibweise

Eine Boolesche Formel $F$ liegt in *konjunktiver Normalform* (engl. *conjunctive normal form*, bzw. kurz CNF) vor, in Zeichen $F \in \mathrm{CNF}$, falls $F$ geschrieben werden kann als

$$F = C_1 \wedge C_2 \wedge \cdots \wedge C_m,$$

wobei die Teilformeln $C_i$ als *Klauseln* bezeichnet werden und *Disjunktionen* von Literalen sind, d. h.

$$C_i = (u_{i,1} \vee u_{i,2} \vee \cdots \vee u_{i,k_i}),$$

wobei die $u_{i,j}$ Literale sind. Kurz: Eine Formel in CNF ist eine *Konjunktion* von Klauseln, die wiederum Disjunktionen von Literalen sind:

$$F = (u_{1,1} \vee u_{1,2} \vee \cdots \vee u_{1,k_1}) \wedge \cdots \wedge (u_{m,1} \vee u_{m,2} \vee \cdots \vee u_{m,k_m}). \tag{2.1}$$

Analog dazu liegt eine Formel $F$ in *disjunktiver Normalform* (engl. *disjunctive normal form*, bzw. kurz DNF) vor, in Zeichen $F \in \text{DNF}$, falls $F$ als Disjunktion von *Konjunktionstermen* (d. h. Formeln, die ausschließlich durch die konjunktive Verknüpfung von Literalen gebildet werden) geschrieben werden kann.

Für die Anwendung der Resolution in Abschnitt 3.5 ist es nützlich, Formeln in konjunktiver Normalform als Mengen ihrer Klauseln darzustellen, wobei die Klauseln wiederum durch die Menge der in ihr enthaltenen Literale angeben werden. Für eine CNF-Formel $F$ wie in (2.1) werden wir daher auch

$$F = \Big\{\{u_{1,1}, u_{1,2}, \ldots, u_{1,k_1}\}, \ldots, \{u_{m,1}, u_{m,2}, \ldots, u_{m,k_m}\}\Big\} \tag{2.2}$$

schreiben. Man beachte, dass es sich hierbei streng genommen um Notationsmissbrauch handelt: Aufgrund der Nützlichkeit und Gängigkeit dieser Konvention wollen wir über diesen Missbrauch hinwegsehen und sowohl die Formel $F$ als auch die ihr zugeordnete Klauselmenge mit dem selben Symbol $F$ bezeichnen, obwohl natürlich keine eindeutige Abbildung zwischen den beiden Darstellungen besteht. Wir gehen wie [Sch2013] davon aus, dass Vereinfachungen, die sich aus Anwendung der Assoziativität, Kommutativität oder Idempotenz ergeben durch diese Mengennotation automatisch bereitgestellt werden – insbesondere „verschwinden“ doppelte Literalvorkommen durch die Mengenschreibweise. Ebenso werden wir „klauselspezifische“ Begriffe wie Äquivalenz oder Erfüllbarkeit für die entsprechend assoziierten Klauselmengen verwenden.

Ebenso sei erwähnt, dass gemäß unserer Definition in Fußnote 3 auf Seite 10 die Konstanten 0 und 1 CNF-Formeln sind.

Wir sagen, dass eine Klausel $C_1$ eine andere Klausel $C_2$ *subsummiert*, falls $C_1 \subset C_2$ gilt.

### Maße für Formeln bzw. Klauseln und die Codierung von Formeln

Die *Weite* $\text{Wi}(C)$ *einer Klausel* $C = (u_1 \vee \ldots \vee u_k)$ ist die Anzahl, der in ihr enthaltenen Literale, in Zeichen $\text{Wi}(C) := |\operatorname{Var}(C)|$. Ist $F$ gegeben wie in (2.1) und gibt es ein $k \in \mathbb{N}$, sodass $\text{Wi}(C_i) \leq k$ für alle $i = 1, \ldots, m$ gilt, so sagen wir, dass $F$ in $k$-CNF vorliegt und schreiben $F \in k\text{-CNF}$. Ist $F$ eine Menge von Klauseln, so bezeichne die *Weite* $\text{Wi}(F)$ *der Klauselmenge* $F$ die maximale Weite der Klauseln in $F$, in Zeichen $\text{Wi}(F) := \max_{C \in F} \text{Wi}(C)$. Entsprechend sei die *Weite* $\text{Wi}(\pi)$ *einer Klauselfolge* $\pi = (C_1, \ldots, C_t)$ de-

finiert als $\mathrm{Wi}(\pi) := \max_{i\in[t]} \mathrm{Wi}(C_i)$.

Die *Größe* $\mathrm{size}(F)$ einer Formel $F$ definieren wir als die Gesamtanzahl an Literalvorkommen in $F$, wobei mehrfach auftretende Literale auch mehrfach gezählt werden. Ist $F \in \mathrm{CNF}$, gilt also $\mathrm{size}(F) := \sum_{C\in F} \mathrm{Wi}(C)$.

Wir wählen für den Rest der Arbeit stillschweigend ein abzählbar unendliches Alphabet $\Sigma_{\mathrm{PROP}}$ so, dass $\mathrm{PROP} \subset \Sigma^*_{\mathrm{PROP}}$ gilt. Es ist z. B. möglich, $\Sigma_{\mathrm{PROP}} = \mathcal{V} \cup \{\neg, \vee, \wedge, 0, 1, (, )\}$ zu wählen. Für weitere Details hierzu sei der Leser auf [Sch2018b, Definition 2.3 & 2.4], [Dal2013] oder andere Standardbücher über Logik verwiesen.

Mit $|F|$ bezeichnen wir (angelehnt an die Notation $|w|$ für die Länge eines Wortes $w$) die *Länge des Strings, der die Formel $F$ codiert*, kurz die *Formellänge* von $F$. Z. B. lässt sich diese über unsere induktive Definition von PROP über einem geeigneten Alphabet $\Sigma_{\mathrm{PROP}}$ wie folgt definieren:

1. $|0| = |1| = |x| = 1$ für alle $x \in \mathcal{V}$,
2. $|\neg F| = 1 + |F|$,
3. $|(F \circ G)| = |F| + 1 + |G|$ für $\circ \in \{\wedge, \vee\}$.

Die Bezeichnung $|F|$ ist in vielen modernen Arbeiten über Beweiskomplexität üblich – wir weisen jedoch wie [Nor2001, Remark 2.20] auf die große Verwechslungsgefahr hin: Andere Autoren bezeichnen mit $|F|$ die Formelgröße von $F$ (wofür wir $\mathrm{size}(F)$ verwenden) oder die Anzahl der Klauseln in $F$ (angelehnt an die Mengennotation einer Formel wie in (2.2)).

Ebenso werden wir später für eine Folge $\pi = (C_1, \ldots, C_t)$ von Klauseln mit $|\pi|$ die Länge des Strings bezeichnen, der $\pi$ codiert. Dies ist nicht zu verwechseln mit der Länge $\mathrm{Le}(\pi)$ der Folge, die wir in Unterabschnitt 4.1 einführen werden.

### Boolesche Funktionen

Eine *Boolesche Funktion* ist eine Funktion $f\colon \{0,1\}^n \to \{0,1\}$, wobei $n \in \mathbb{N}_0$ die *Arität*[4] (oder: *Stelligkeit*) von $f$ ist. Jede Boolesche Funktion der Arität $n$ kann als aussagenlogische Formel in $n$ Variablen notiert werden. Der Operator Var sei wie bei Booleschen Funktionen definiert.

### Turingmaschinen

Eine *Turingmaschine* $M$ ist ein 7-Tupel $M = (Z, \Sigma, \Gamma, \delta, z_0, \square, E)$, wobei

- $Z \neq \varnothing$ die nicht-leere endliche *Zustandsmenge*,
- $\Sigma \neq \varnothing$ das nicht-leere endliche *Eingabealphabet*,
- $\Gamma \supset \Sigma$ das nicht-leere, $\Sigma$ enthaltende, endliche *Arbeitsalphabet*,

[4] In diesem Zusammenhang betrachten wir das leere kartesische Produkt $\{0,1\}^0$ als die Menge, die nur das leere Wort enthält. Eine Boolesche Funktion der Arität 0 fassen wir als konstante Abbildung $\{\varepsilon\} \ni \varepsilon \mapsto b \in \{0,1\}$ auf. Wir fassen Konjunktion und Disjunktion als Funktionen der Arität 2 auf, Komplementbildung als Funktion der Arität 1 und die Konstanten 0 und 1 als Funktionen der Arität 0.

- $z_0 \in Z$ der *Anfangszustand*,
- $\square \in \Gamma \setminus \Sigma$ das *Blank-Symbol*,
- $\varnothing \neq E \subset Z$ die nicht-leere *Menge der Endzustände*,
- und $\delta\colon (Z \setminus E) \times \Gamma \to \mathfrak{P}\left(Z \times \Gamma \times \{\mathtt{L}, \mathtt{R}, \mathtt{N}\}\right)$ die *Übergangsfunktion* ist.

Dabei bedeute die Notation $(z', b, d) \in \delta(z, a)$, dass die Turingmaschine, wenn sie sich im Zustand $z$ befindet und unter dem *Schreib-Lese-Kopf* das Zeichen $a$ steht im nächsten *Rechenschritt* in den Zustand $z'$ *übergehen* kann, auf die *Bandzelle* von $a$ das Symbol $b$ schreiben kann und danach ihren Schreib-Lese-Kopf in die Richtung $d \in \{\mathtt{L}, \mathtt{R}, \mathtt{N}\}$ bewegen kann, wobei `L` für eine *Bewegung* nach Links, `R` für eine nach Rechts und `N` für eine neutrale Bewegung – also das Stehenbleiben des Schreib-Lese-Kopfes – steht. Für eine genaue Einführung in Turingmaschinen, die über lediglich notationstechnische Details hinausgeht (insbesondere für Leser, die mit den kursiven Begriffen des letzten Satzes nicht vertraut sind), empfehlen wir z. B. die Bücher [HMU2011, Kapitel 8] oder [Pap1994, Kapitel 2]. Wir gehen insb. davon aus, dass Leser mit dem Begriff der *von einer Turingmaschine $M$ akzeptierten Sprache* $\mathrm{T}(M)$ vertraut sind.

Die Maschine $M$ heißt *deterministisch*, falls $|\delta(q, a)| \leq 1$ für alle $q \in Z \setminus E$ und alle $a \in \Gamma$ gilt, andernfalls heißt sie *nicht-deterministisch*.

*Mehrband-Turingmaschinen* sind Turingmaschinen mit $k \geq 2$ Bändern (1 Eingabeband, auf dem nicht geschrieben werden kann, und $k-1$ Arbeitsbänder) und $k$ Schreib-Lese-Köpfen. Man definiert sie durch Anpassung der Übergangsfunktion zu $\delta\colon (Z \setminus E) \times \Gamma^k \to \mathfrak{P}\left(Z \times (\Gamma \times \{\mathtt{L}, \mathtt{R}, \mathtt{N}\})^k\right)$. *Determinismus* und *Nicht-Determinismus* werden ähnlich wie oben definiert.

### Komplexitätsklassen

Die Funktion $\mathrm{time}_M\colon \Sigma^* \to \mathbb{N}_0$ ordne einem Wort $x \in \Sigma^*$ die Anzahl der Rechenschritte der deterministischen Turingmaschine $M$ bei Eingabe $x$ zu.

Die Funktion $\mathrm{space}_M\colon \Sigma^* \to \mathbb{N}_0$ ordne einem Wort $x \in \Sigma^*$ die Anzahl der Arbeitsbandzellen zu, die von der deterministischen Mehrband-Turingmaschine $M$ bei Eingabe von $x$ (zu irgendeinem Zeitpunkt während der Berechnung) benutzt werden.

Für eine nicht-deterministische Turingmaschine [resp. Mehrband-Turingmaschine] $M$ sei $\mathrm{ntime}_M(x)$ [resp. $\mathrm{nspace}_M(x)$] definiert als die minimale Länge eines akzeptierenden Berechnungspfades von $M$ auf $x$ [resp. als die minimale Anzahl der Arbeitsbandzellen über alle Berechnungspfade, die von $M$ bei Eingabe $x$ zu irgendeinem Zeitpunkt während der Berechnung benutzt worden sind], falls $x \in \mathrm{T}(M)$ ist und $\mathrm{ntime}_M(x) := 0$ [resp. $\mathrm{nspace}_M(x) := 0$] andernfalls.

Es sei $T\colon \mathbb{N} \to \mathbb{N}$ eine Funktion. Mit $\mathsf{DTIME}(T(n))$ [resp. $\mathsf{NTIME}(T(n))$] bezeichnen wir die Klasse aller Sprachen $L$, für die es eine deterministische [resp. nicht-deterministische]

Mehrband-Turingmaschine $M$ mit $L = \mathrm{T}(M)$ sowie $\mathrm{time}_M(x) \leq T(|x|)$ [resp. $\mathrm{ntime}_M(x) \leq T(|x|)$] für alle Eingaben $x \in \Sigma^*$ gibt.

Die Klasse $\mathsf{DSPACE}(T(n))$ [resp. $\mathsf{NSPACE}(T(n))$] besteht aus den Sprachen $L$, für die es eine deterministische [resp. nicht-deterministische] Mehrband-Turingmaschine $M$ mit $L = \mathrm{T}(M)$ und $\mathrm{space}_M(x) \leq T(|x|)$ [resp. $\mathrm{nspace}_M(x) \leq T(|x|)$] für alle $x \in \Sigma^*$ gibt.

Die Komplexitätsklasse $\mathsf{P}$ bezeichne die Klasse, die alle Entscheidungsprobleme enthält, die in Polynomialzeit durch eine deterministische Turingmaschine entschieden werden können, präziser: Eine Sprache $L$ ist in $\mathsf{P}$ genau dann, wenn eine deterministische Turingmaschine $M$ und ein Polynom $p$ existiert, sodass für alle Eingaben $x$ die Abschätzung $\mathrm{time}_M(x) \leq p(|x|)$ und folgende Äquivalenz gilt:

$$x \in L \iff M \text{ akzeptiert die Eingabe } x.$$

Äquivalent ist $\mathsf{P} = \bigcup_{k\in\mathbb{N}} \mathsf{DTIME}(n^k)$.

Die Komplexitätsklasse $\mathsf{NP}$ bezeichne die Klasse, die alle Entscheidungsprobleme enthält, die in Polynomialzeit durch eine nicht-deterministische Turingmaschine entschieden werden können. Für eine präzise Definition verweisen wir auf Definition B.1.1. Es ist $\mathsf{P} \subset \mathsf{NP} = \bigcup_{k\in\mathbb{N}} \mathsf{NTIME}(n^k) \subset \bigcup_{k\in\mathbb{N}} \mathsf{DTIME}(2^{n^k})$ (siehe bspw. [AB2009, Claim 2.3, Theorem 2.6]).

Weiter bezeichnen wir mit $\mathsf{EXP}$ die Komplexitätsklasse der Entscheidungsprobleme, die von einer deterministischen Turingmaschine in durch $\mathcal{O}\left(2^{p(n)}\right)$ beschränkter Zeit entschieden werden können, wobei $p$ ein beliebiges Polynom in der Eingabelänge $n$ ist. Es ist $\mathsf{EXP} = \bigcup_{k\in\mathbb{N}} \mathsf{DTIME}(2^{n^k})$ sowie $\mathsf{P} \subsetneq \mathsf{EXP}$ (siehe bspw. [Tor2017, § 3.3]).

Entsprechend bezeichnen wir mit $\mathsf{NEXP}$ die Komplexitätsklasse der Entscheidungsprobleme, die von einer nicht-deterministischen Turingmaschine in durch $\mathcal{O}\left(2^{p(n)}\right)$ beschränkter Zeit entschieden werden können, wobei $p$ ein beliebiges Polynom in der Eingabelänge $n$ ist.

Eine *richtige Komplexitätsfunktion* ist eine Funktion $f\colon \mathbb{N} \to \mathbb{N}$ mit $f(n) \leq f(n+1)$ für alle $n \in \mathbb{N}$, für die eine Mehrband-Turingmaschine existiert, die für jede Eingabe $x$ das Wort $0^{f(|x|)}$ in Zeit $\mathcal{O}(|x| + f(|x|))$ und Platz $\mathcal{O}(f(|x|))$ ausgibt.

Ist $\mathsf{C}$ eine Komplexitätsklasse, so definieren wir $\mathsf{co\text{-}C} := \{L : \overline{L} \in \mathsf{C}\}$. Gilt für alle Sprachen $L \in \mathsf{C}$ auch $\overline{L} \in \mathsf{C}$, so heißt $\mathsf{C}$ *komplementabgeschlossen* oder *abgeschlossen unter Komplementbildung*.

Für Begriffe wie *Reduktion* und *Vollständigkeit* verweisen wir auf Anhang B. Für mögliche Beziehungen zwischen $\mathsf{P}$, $\mathsf{NP}$ und $\mathsf{co\text{-}NP}$ verweisen wir auf Abbildung B.2.

### Wichtige Sprachen und Entscheidungsprobleme

Das *Erfüllbarkeitsproblem der Aussagenlogik* (engl. *satisfiability*) ist die algorithmische Aufgabenstellung zu entscheiden, ob eine gegebene aussagenlogische Formel erfüllbar ist. Wir

definieren passend dazu die Sprache SAT := $\{F \in \text{PROP} : F \text{ ist erfüllbar}\}$. An dieser Stelle sei an den Satz von COOK [Coo1971] (und LEVIN [Lev1973]) erinnert: Die Sprache SAT ist NP-vollständig (d. h. SAT ist nach Theorem B.2.5 (ii) nicht in P, außer es gilt P = NP). Für einen modernen Beweis zur NP-Vollständigkeit von SAT siehe z. B. [AB2009, § 2.3.4] oder [ST2013]. Algorithmen, die entscheiden, ob eine Formeln $F$ in SAT ist, nennen wir SAT-*Solver*.

Es sei CNF-SAT := CNF ∩ SAT, sowie $k$-SAT := $k$-CNF-SAT := $k$-CNF ∩ SAT.

Weiterhin setzen wir UNSAT := $\{F \in \text{PROP} : F \text{ ist unerfüllbar}\}$.

Die Sprache TAUT := $\{F \in \text{PROP} : F \text{ ist tautologisch}\}$ sei die Menge aller aussagenlogischer Tautologien. Sie ist co-NP-vollständig nach Korollar B.2.8.

Da wir PROP $\subset \Sigma^*_{\text{PROP}}$ für ein passendes Alphabet $\Sigma_{\text{PROP}}$ auf Seite 14 angenommen haben, sind TAUT und alle oben genannten Sprachen ebenfalls Teilmengen von $\Sigma^*_{\text{PROP}}$.

### Graphen

Ein (*ungerichteter*[5]) *Graph* ist ein Tupel $G = (V, E)$, wobei $V$ eine endliche, nicht-leere Menge ist und

$$E \subset \Big\{\{u, v\} : u, v \in V \text{ und } u \neq v\Big\} \tag{2.3}$$

eine zu $V$ disjunkte Menge ist.

Ein *unendlicher ungerichteter Graph* ist ein Tupel $G = (V, E)$, wobei $V$ eine abzählbare, nicht-leere Menge ist und $E$ eine Menge wie in (2.3) sowie zu $V$ disjunkt ist.

Die Elemente von $V$ heißen *Knoten* von $G$, die Elemente von $E$ *Kanten* von $G$. Oft schreiben wir $uv$ für eine Kante $\{u, v\}$.

Man beachte, dass wir durch obige Definition in ungerichteten Graphen keine Mehrfachkanten oder Schleifen zulassen (wir weisen jedoch auf die Einschränkung dieses Sachverhaltes ab Definition & Konvention 6.0.6 hin).

Ist ein Graph $G$ gegeben, so bezeichnen wir die Menge seiner Knoten mit $V(G)$ und die Menge seiner Kanten mit $E(G)$.

Zwei Knoten $u, v$ sind *benachbart*, wenn $uv \in E$ ist. Die Kante $uv$ ist dann *inzident* mit dem Knoten $u$ und mit dem Knoten $v$, in Zeichen $uv \sim u$ und $uv \sim v$, während $u$ und $v$ die *Endknoten* der Kante $uv$ sind.

Ist $v$ ein Knoten des Graphen $G$, so bezeichnen wir mit $\deg_G(v)$ die Anzahl der Kanten, die mit dem Knoten $v$ inzident sind. Wir nennen $\deg_G(v)$ den *Grad* des Knotens $v$ im Graphen $G$ und schreiben auch kurz $\deg(v)$, wenn klar ist, über welchen Graphen wir sprechen.

[5] Falls nicht gegenteilig behauptet wird, dass ein Graph gerichtet ist, verstehen wir unter dem Begriff *Graph* einen *ungerichteten* Graph.

Der *Maximalgrad* $\text{maxdeg}(G)$ eines Graphen $G$ ist definiert durch

$$\text{maxdeg}(G) := \max\{\deg_G(v) \,:\, v \in V(G)\}.$$

Analog ist der *Minimalgrad* durch $\text{mindeg}(G) := \min\{\deg_G(v) \,:\, v \in V(G)\}$ festgelegt.

Ein *Schnitt* (engl. *cut*) $C = (S, T)$ eines Graphen $G = (V, E)$ ist eine Partition seiner Knotenmenge $V$ in zwei disjunkte Teilmengen $S$ und $T = V \setminus S$. Das *Cut-Set* (dtsch. missverständlich *Schnittmenge*) $\text{CS}(C)$ eines Schnittes $C = (S, T)$ ist die Menge der Kanten, die den Schnitt „kreuzen“, in Zeichen $\text{CS}(C) := \{uv \in E \,:\, u \in S, v \in T\}$. Wir wollen auch $\text{CS}(S, T)$ für diese Menge schreiben, um Klammern zu sparen.

Ein Graph $H = (U, F)$ ist ein *Teilgraph*[6] des Graphen $G = (V, E)$, wenn $U \subset V$ und $F \subset E$ ist. Ein Teilgraph $H = (U, F)$ von $G = (V, E)$ ist *induziert*, falls er jede Kante $e \in E$ des Graphen $G$ enthält, deren beiden Endknoten in $U$ enthalten sind. Ist $V' \subset V$, so heißt der induzierte Teilgraph $H = (V', F)$ *der von $V'$ induzierte Teilgraph*, in Zeichen $G[V']$.

Ein *gerichteter Graph* (siehe Fußnote 5 auf Seite 17) ist ein Tupel $G = (V, E)$ mit einer endlichen, nicht-leeren Menge $V$, den *Knoten*, und einer Menge von *Kanten*

$$E \subset V \times V \setminus \{(v, v) \,:\, v \in V\},$$

die zu $V$ disjunkt ist.

Ist $e = (u, v)$ eine Kante, so nennen wir $u$ *Anfangsknoten* und $v$ *Endknoten* von $e$. Wir wollen in diesem Sinne $u$ auch einen *Vorgänger* von $v$ nennen, bzw. $v$ einen *Nachfolger* von $u$.

Eine *topologische Sortierung* eines gerichteten Graphen ist eine Totalordnung $\leq$ seiner Knoten (für den Begriff Totalordnung verweisen wir auf Fußnote 47 auf Seite 163), sodass für alle gerichteten Kanten $(u, v)$ gilt, dass $u \leq v$ ist.

Ein *Weg* oder *Pfad*[7] $P = v_1, v_2, \ldots, v_k$ eines (gerichteten oder ungerichteten) Graphen $G = (V, E)$ ist ein Teilgraph von $G$ mit der Knotenmenge $\{v_1, v_2, \ldots, v_k\} \subset V$, sodass $(v_i, v_{i+1}) \in E$ (bzw. $\{v_i, v_{i+1}\} \in E$) für $i = 1, \ldots, k-1$ ist. Ein Weg $P = v_1, v_2, \ldots, v_k$ heißt *einfach*, falls die zugrundeliegende Knotenfolge $(v_1, v_2, \ldots, v_k)$ aus paarweise verschiedenen Knoten besteht.

Ein *Zyklus* $Z$ im Graphen $G$ ist ein einfacher Weg $Z = v_1, v_2, \ldots v_{k-1}, v_k$ des Graphen $G$, bei dem $v_1 = v_k$ gilt (und falls $G$ ungerichtet ist, zusätzlich $k \geq 3$ gilt). Ein Graph mit mindestens einem Zyklus heißt *zyklisch*. Graphen ohne Zyklen heißen *azyklisch*.

[6] Man beachte, dass ein Teilgraph $H = (U, F)$ per Definition selbst ein Graph sein muss. D. h. die Knotenmenge $U$ muss alle Endknoten der Kanten aus $F$ enthalten, kann aber weitere Knoten aus $V$ enthalten.

[7] Diese Begriffe werden in der Literatur teilweise sehr unterschiedlich definiert.

Wir sagen ein ungerichteter Graph $G$ sei *zusammenhängend*, wenn es einen Weg zwischen je zwei Knoten von $G$ gibt. Ein gerichteter Graph $G$ heißt *stark zusammenhängend*, falls es einen Weg (unter Berücksichtigung der Kantenrichtungen) zwischen je zwei Knoten von $G$ gibt. Einen maximalen zusammenhängenden Teilgraphen eines beliebigen ungerichteten Graphen nennen wir *Zusammenhangskomponente* des Graphen. Ein nichtzusammenhängender Graph zerfällt in seine Zusammenhangskomponenten. Jeder Knoten und jede Kante des Graphen gehört zu genau einer Zusammenhangskomponente.

Ein *Baum* ist ein ungerichteter, zusammenhängender, azyklischer Graph. Die Knoten eines Baumes vom Grad 1 werden *Blätter* genannt, die Knoten vom Grad größer als 1 heißen *innere Knoten*. Indem wir genau einen Knoten eines Baumes besonders auszeichnen und *Wurzel* nennen, können alle anderen Knoten des Baumes entsprechend ihrer *Entfernung* (der Anzahl der Kanten zwischen dem Knoten und der Wurzel) als *Söhne*, *Enkel*, *Großenkel* etc. bezeichnet werden. Wir sprechen dann von einem *gewurzelten* Baum. Ein *gewurzelter Binärbaum* ist ein gewurzelter Baum, bei dem jeder Knoten höchstens zwei Söhne haben kann.

### Multimengen und Multigraphen

Gewöhnliche Mengen enthalten paarweise verschiedene Elemente. *Multimengen* erlauben das mehrfache endliche Vorkommen der jeweiligen Elemente (vergleichbar mit Tupeln/Listen), berücksichtigen jedoch (genauso wie Mengen, aber im Gegensatz zu Tupeln/Listen) nicht die Reihenfolge der Elemente. Angelehnt an [CPRS2001] betrachten wir eine Multimenge $M$ über einer Grundmenge $A$ als Abbildung $\mathrm{count}_M \colon A \to \mathbb{N}_0$, welche die Anzahl $\mathrm{count}_M(x)$ der Vorkommen eines Elementes $x \in A$ in der Multimenge $M$ angibt.

Ist bspw. $M$ eine Multimenge über $A = \{a, b, c\}$ mit $\mathrm{count}_M(a) = 2$, $\mathrm{count}_M(b) = 4$ und $\mathrm{count}_M(c) = 0$, so notieren wir (um die Notationen einfach zu halten) diese Multimenge suggestiv mit $M = \{\!\{a, a, b, b, b, b\}\!\}$ (wobei die Reihenfolge der Elemente keine Rolle spielt).

Ein *ungerichteter Multigraph* ist ein Tupel $G = (V, E)$, wobei $V$ eine endliche, nicht-leere Menge ist und $E$ eine Multimenge über $\Big\{\{u, v\} \,:\, u, v \in V \text{ und } u \neq v\Big\}$, ähnlich zu (2.3).

### Wahrscheinlichkeitstheorie

Es sei $\Omega$ eine endliche oder abzählbare Menge (die *Grundmenge*). Ein *Ereignis* ist eine Teilmenge von $\Omega$. Eine Funktion $\mathrm{Prob} \colon \mathfrak{P}(\Omega) \to [0, 1]$ heißt *Wahrscheinlichkeitsmaß* auf $\Omega$, falls $\mathrm{Prob}[\Omega] = 1$ (*Normiertheit*) und $\mathrm{Prob}\left[\biguplus_{k=1}^{\infty} A_k\right] = \sum_{k=1}^{\infty} \mathrm{Prob}[A_k]$ für beliebige paarweise disjunkte Ereignisse $A_1, A_2, \cdots \subset \Omega$ gilt ($\sigma$-*Additivität*). Das Tupel $(\Omega, \mathrm{Prob})$ heißt *diskreter Wahrscheinlichkeitsraum*. Unmittelbar aus obiger Definition ergibt sich $\mathrm{Prob}[\varnothing] = 0$; $\mathrm{Prob}[\Omega \setminus A] = 1 - \mathrm{Prob}[A]$ für alle $A \subset \Omega$; sowie $\mathrm{Prob}[A] \leq \mathrm{Prob}[B]$ für beliebige Ereignisse $A \subset B$. Häufig werden wir die $\sigma$-*Subadditivität* (im Englischen oft auch als *union bound*

bezeichnet) des Wahrscheinlichkeitsmaßes benutzen:

$$\text{Prob}\left[\bigcup_{k=1}^{\infty} A_k\right] \leq \sum_{k=1}^{\infty} \text{Prob}[A_k] \qquad \text{für eine beliebige Folge von Ereignissen } (A_k)_{k\in\mathbb{N}} \subset \Omega.$$

**Sonstiges**

Für $x \in \mathbb{R}$ definieren wir die *untere Gaußklammer* $\lfloor x \rfloor$ als die größte ganze Zahl kleinergleich $x$, in Zeichen $\lfloor x \rfloor := \max\{k \in \mathbb{Z} : k \leq x\}$. Entsprechend sei die *obere Gaußklammer* definiert durch $\lceil x \rceil := \min\{k \in \mathbb{Z} : k \geq x\}$.

Mit $\text{id}_M$ bezeichnen wir die *Identität* auf der Menge $M$, d. h. es ist $\text{id}_M(x) := x$ für alle $x \in M$.

Als Reihenfolge für die Komposition von Abbildungen $f\colon A \to B$ und $g\colon B \to C$ (mit beliebigen Mengen $A, B, C$) vereinbaren wir:

$$\begin{aligned} g \circ f\colon A &\longrightarrow C \\ x &\longmapsto \big(g \circ f\big)(x) := g\Big(f(x)\Big). \end{aligned}$$

Eine „Notiz“ ist eine besonders *wichtige* Bemerkung, ein „Hilfslemma“ ist von niedrigem Interesse und hat den Stellenwert einer Nebenüberlegung oder Nebenrechnung. Mit „Intuition“ kennzeichnen wir Bemerkungen, die helfen, die intuitive Idee hinter den Beweisen oder Definitionen besser zu verstehen. Diese Bemerkungen sind bewusst etwas lockerer und unformaler gehalten.

Die zu beweisende Richtung in Äquivalenzbeweisen kennzeichnen wir mit „$\longrightarrow$“, respektive „$\longleftarrow$“; die in Beweisen zu Mengengleichheiten mit „$\subset$“, respektive „$\supset$“.

Das Ende eines Beweises kennzeichnen wir mit einem großen $\square$, dem offenen „Halmos' thombstone“ (bzw. $\boxdot$ für das Ende des Hauptbeweises) und weisen auf die Verwechslungsgefahr von $\square$ mit dem kleiner gesetzten Symbol für die leere Klausel $\square$ aus Definition 3.5.1 (iii) und dem kleineren Blank-Symbol $\square$ von Turingmaschinen hin.

# Teil I.

# Einführung in die Beweiskomplexität und Tradeoffs

Wir geben in Kapitel 3 eine präzise Einführung in die Beweiskomplexität mitsamt einer Motivation für das Studium derselben. Anschließend stellen wir einige Beweissysteme beispielhaft dar und sprechen insbesondere das Beweissystem der Resolution an, das im Rest der Arbeit von zentraler Bedeutung sein wird. In Kapitel 4 diskutieren wir die Motivation für Tradeoffs, klären diesen Begriff und geben einige Beispiele bekannter Resultate, um das in Teil II dargestellte und bewiesene Tradeoff-Resultat im Beweissystem der Resolution für superlinearen Platz besser einordnen zu können.

# KAPITEL 3

# Einführung in die Beweiskomplexität

In diesem Kapitel wollen wir eine Einführung in das Themengebiet der Beweiskomplexität geben. Dabei sprechen wir platzbedingt lediglich einige der zentralen Konzepte des Gebietes an. Für weitere Details verweisen wir auf die Bücher [Bea2004], [Bus1998], [Kra2018] oder die Survey Paper [BP2001], [Seg2007], [Urq1995].

## Inhalt des Kapitels



Die nächsten zwei Abschnitte motivieren das Forschungsgebiet heuristisch und geben einen sanften Einstieg in typische Fragestellungen. Wir beginnen mit der formalen Behandlung ab Abschnitt 3.3.

## 3.1 Heuristische Betrachtung des Begriffs „Beweis“

Ausgehend von [Bus1998] wollen wir zunächst heuristisch klären, was wir unter einem „Beweis“ verstehen – dieser Begriff spielt eine zentrale Rolle in der Mathematik und Informatik

und ist die Methode, mit der wir die Wahrheit oder Falschheit von mathematischen Aussagen begründen. Der Begriff „Beweis“ umfasst dabei zwei Facetten:

1. Gewöhnlich verstehen wir unter einem Beweis eine **soziale Konvention**, mit der Mathematiker einander von der Richtigkeit eines Theorems überzeugen. Der Beweis selbst ist hierbei in einer natürlichen Sprache verfasst, gegebenenfalls unterstützt von Symbolen oder Zeichnungen. Als korrekt sehen wir einen Beweis in der Regel an, wenn ein Experte des jeweiligen Forschungsgebietes von seiner Richtigkeit überzeugt werden kann. Beweise in Büchern, Vorlesungen oder veröffentlichten Artikeln fallen in diese Kategorie.

   Offensichtlich ist obige Beschreibung keine präzise Definition eines Beweises, da seine Akzeptanz von sozialen Faktoren abhängt und sich gegebenenfalls auch über die Zeit hinweg verändern kann[8].

2. Ebenso kann ein Beweis als **String von Symbolen** angesehen werden (der strengen, präzise formulierten Regeln unterliegt), der einen mathematischen Satz (welcher wiederum als String vorliegt) beweist. Wir sprechen auch von *formalen Beweisen*. Diese treten vor allem in der Logik auf, um zu zeigen, dass Formeln tautologisch sind.

   Im Forschungsgebiet der Beweiskomplexität werden hauptsächlich solche formalen Beweise betrachtet. Auf den eher philosophischen Aspekt aus Punkt 1 werden wir in der vorliegenden Arbeit nicht näher eingehen – obgleich er nach Buss auch Gegenstand der Beweiskomplexität ist.

Wir möchten anmerken, dass diese zwei Facetten eng miteinander verwandt sind. Zunächst sollte jeder formale Beweis auch als sozialer Beweis dienen können (auch wenn das Übersetzen mühevoll und der Beweis selbst unintuitiv sein kann), sofern wir der Validität des Beweissystems glauben, in der er ursprünglich verfasst war. Zweitens sind die Ansprüche an soziale Beweise mitunter sehr hoch, sodass es in der Theorie[9] möglich sein sollte, einen formalen Beweis aus einem gegebenen sozialen Beweis zu formulieren – dieser implizite Anspruch ist einer der Gründe dafür, warum wir soziale Beweise überhaupt als richtig erachten. Zugleich dient er dazu, gewisse Standards an soziale Beweise zu stellen. Ein gemeinsamer Anspruch an Beweise in beiden Auffassungen ist die Möglichkeit, den Beweis effizient zu

[8] Beispielhaft sei z. B. der Beweis des Handschlag-Lemmas aus der Graphentheorie in der Version von Leonhard Euler in seinem Paper [Eul1741] über das Königsberger Brückenproblem und die moderne Formulierung bzw. der moderne Beweis, den wir in Definition & Lemma 5.0.1 führen, genannt.

[9] Für viele der wichtigsten mathematischen Theoreme wurden computergestützte Beweise erstellt. Beispielsweise bildet der Satz von Hahn-Banach *den* Grundpfeiler der Funktionalanalysis, einem Teilgebiet der Analysis, das weitreichende Konsequenzen in verschiedenste Teilgebiete der Mathematik hat. Aufgrund der Wichtigkeit des Satzes wurde dieser schon sehr früh mithilfe von Computern formal bewiesen. Beispielhaft verweisen wir für eine aktuelle Arbeit mit gut lesbarem formalisiertem Beweis auf [BW2000].

überprüfen, was wir zu Beginn des Abschnittes 3.3 für die Definition von Beweissystemen wieder aufgreifen werden.

Trotz der Tatsache, dass Beweise eines der wichtigsten Konzepte der Mathematik sind, wurde ein erster Versuch, sie im Sinne von Punkt 2 der obigen Aufzählung zu formalisieren erst von Gottlob Frege in seiner Begriffsschrift [Fre1879] im späten 19. Jahrhundert unternommen.

## 3.2 Motivation und typische Fragestellungen

Ähnlich wie [Kra2005, Kra2018] möchten wir die Beweiskomplexität wie folgt motivieren:

In vielen Teilgebieten der Mathematik finden sich Aussagen, dass endliche Objekte mit gewissen Eigenschaften (die „schnell“ überprüft werden können) nicht existieren können: Sei es die nicht-Lösbarkeit von partiellen Differentialgleichungen, die nicht-Existenz bestimmter algebraischer Objekte, oder selbst das nicht-Vorkommen kombinatorischer Muster, die weit in den Alltag reichen: *Es ist z. B. unmöglich $n+1$ Tauben auf $n$ Taubenschläge aufzuteilen, wenn jede Taube einen Schlag für sich haben soll* (siehe Formel 4.2.1 für ein konkretes Beispiel einer solchen unerfüllbaren Formel für den Rest dieses Kapitels). Ein besonders wichtiges Beispiel ist es, für eine gegebene aussagenlogische Formel die Unerfüllbarkeit nachzuweisen, bzw. für ihre Negation zu zeigen, dass sie eine Tautologie ist. In der Tat ist dies ein besonders wichtiges Beispiel, nicht nur aufgrund der Tatsache, dass sich Menschen mit dieser Frage aus der Logik seit über zweieinhalb Jahrtausenden beschäftigen ([Kra2018], [BHMW2009, Kapitel 1 & Fig. 1.2]), sondern da sich z. B. die oben angesprochenen kombinatorischen Prinzipien leicht in aussagenlogische Formeln übersetzen lassen.

Das Forschungsgebiet der *Beweiskomplexität*, wovon sich erste Resultate in den 1960er-Jahren finden, befasst sich daher vor allem damit, zu untersuchen, wie „komplex“ es ist, von einer gegebenen aussagenlogischen Formel zu zeigen, dass sie eine Tautologie ist. Typische Fragestellungen des Forschungsgebietes, angelehnt an [BP2001, Nor2012], sind:

- Welche (logischen) Aussagen haben „kurze“ *formale* Beweise? Wie lange sind Beweise bestimmter Aussagen? Vor allem: Wie lang ist der kürzeste Beweis der Aussage?
- Aus algorithmischer Sicht ist interessant: Wie kann man solche Beweise finden? Ist das Finden eines Beweises überhaupt automatisierbar? Unterscheidet sich dies in den verschiedenen formalen „Beweissystemen“? Gelingt dies für alle Tautologien? Ist der „automatisch gefundene“ Beweis der optimale? Um wie viel ist er ggfs. länger als ein optimaler Beweis? Sind die zuvor genannten Algorithmen überhaupt effizient?

- Welche verschiedenen Systeme gibt es, Beweise zu notieren? Wie formalisiert man diese? Wie lassen sich diese vergleichen? Was sind natürliche Begriffe und Definitionen, womit sich verschiedene Systeme bezüglich ihrer „Effizienz“ vergleichen lassen? In welchem Sinne ist „Effizienz“ zu verstehen (Länge des Beweises, Speicherverbrauch, strukturelle Komplexität des Beweises)? Welche Systeme sind wie „mächtig“?
- Wie hängen diese Begriffe untereinander und mit anderen Feldern der Informatik (z. B. der Komplexitätstheorie) zusammen.

In den folgenden Abschnitten möchten wir einen groben Abriss über die Beweiskomplexität darstellen. Aufgrund der Größe des Gebietes beschränken wir uns jedoch nur auf die relevantesten Begriffe und Untergebiete – insbesondere wird es uns nicht möglich sein, alle weitreichenden Verflechtungen mit Gebieten wie der Logik, Kombinatorik, Algebra, Schaltkreis-Komplexität und weiteren anzusprechen. Wir sprechen jedoch einen Teil der oben aufgegriffenen typischen Fragestellungen des Gebietes an und untermauern diese mit präzisen Definitionen, sodass die Anführungszeichen-Schreibweise von oben nicht mehr notwendig sein wird.

Dabei beleuchten wir vor allem: Was versteht man unter einem Beweissystem? Was unter einem Aussagenlogischen? Was ist die Komplexität eines Beweissystems, wie hängt dies mit der Komplexitätstheorie und tiefgreifenden Fragen wie dem $\mathsf{NP} \stackrel{?}{=} \mathsf{co\text{-}NP}$-Problem zusammen? Dabei gehen wir vor allem auf das Hauptziel der Beweiskomplexität ein: Zu zeigen, dass es kein aussagenlogisches Beweissystem gibt, dass „kurze“ Beweise für alle Tautologien akzeptiert. Hieraus folgt sofort $\mathsf{NP} \neq \mathsf{co\text{-}NP}$, was die weitreichende Konsequenz $\mathsf{P} \neq \mathsf{NP}$ hätte (vgl. auch Abbildung B.2). Cook und Reckhow waren die Ersten, die vorschlugen, dieses Problem systematisch anzugehen, indem untere Schranken für immer mächtigere Beweissysteme gezeigt werden. Wie [BP2001, BW2001] erwähnen wir – Abschnitt 1.1 aufgreifend –, dass solche Schranken aber noch von weitergehendem Interesse sind, da die meisten Theorem-Beweiser letztlich auf einem Algorithmus basieren, der Beweissysteme benutzt. Schranken für ein Beweissystem erlauben also Aussagen über die Komplexität aller Theorem-Beweiser zu gewinnen, die auf diesem Beweissystem basieren. Beispielhaft sei die Relevanz des vorhergehenden Satzes daran verdeutlicht, dass untere Schranken von aussagenlogischen Beweissystemen bereits genutzt wurden, um von ganzen Klassen an sat-Solvern zu zeigen, dass diese nicht polynomielle Laufzeit haben können (vgl. Abschnitte 4.2 & 4.3) – diese Praxisrelevanz ist ein zweiter wichtiger Grund für das Studium der Beweiskomplexität. Weiter untersuchen wir in den folgenden Abschnitten, wie sich Beweissysteme untereinander vergleichen lassen und welche Fragen dies aufwirft. Auch sind wir daran interessiert, was typische Beweissysteme sind.

## 3.3 Beweissysteme und deren Eigenschaften

Als Motivation für die Definition von Beweissystemen betrachten wir die Komplexitätsklasse NP. Gemäß der Suchproblem-Definition B.1.2 kann NP durch folgende einfache Eigenschaft charakterisiert werden: Eine Sprache $L$ ist genau dann in NP, wenn es für alle Wörter $x \in L$ einen „kurzen“ Beweis (bzgl. der Länge von $x$) für den Fakt $x \in L$ gibt, der zudem in Polynomialzeit überprüft werden kann.

Betrachten wir zum Beispiel das Erfüllbarkeitsproblem der Aussagenlogik, SAT $\in$ NP, bedeutet dies, dass es für alle Formeln in SAT einen kurzen Beweis für die Zugehörigkeit zur Sprache gibt – eine erfüllende Belegung.

Für Formeln, die nicht in SAT liegen, stellt sich die Frage wie ein Beweis der Zugehörigkeit zur Sprache UNSAT aussehen könnte. Wir werden im Folgenden einige Möglichkeiten kennen lernen. Eine zweite Frage ist: Sind diese Beweise immer „kurz“? Falls ja, hätten wir NP = co-NP (vgl. Theorem 3.3.9) – dies motiviert Beweissysteme in ihrer vollen Allgemeinheit zu studieren.

Die folgenden Ausführungen, ebenso wie obige Einleitung orientieren sich an den Darstellungen in [Bea2004, BP2001, Gal2009, Kra2018, Nor2008, Nor2016a].

Die Definition eines Beweissystems ist so gewählt, dass ein Beweis für die Wahrheit einer Behauptung lediglich verifiziert werden muss, also vollständiger Beleg für die Wahrheit dieser Behauptung ist – insbesondere keine weitere „Kreativität“ verlangt wird, wie [Kra2018, § 1.5] es ausdrückt. Eine natürliche Eigenschaft, die wir von einem Beweissystem fordern, ist die *Korrektheit* (d. h. es ist nicht möglich falsche Aussagen zu beweisen) und *Vollständigkeit* (d. h. alle wahren Aussagen können bewiesen werden). Schließlich fordern wir, dass wir einen Beweis „effizient“ prüfen können. Wir formalisieren diese Gedanken in der folgenden Definition.

**Definition 3.3.1** (Moderne Definition angepasst aus [CR1974a])**.** Es sei $L$ eine Sprache über einem endlichen Alphabet $\Sigma$. Ein *Beweissystem* (engl. *proof system*) für die Sprache $L$ ist eine Relation $S \subset \Sigma^* \times \Sigma^*$ mit den drei Eigenschaften

1. für alle $(x, \pi) \in S$ gilt $x \in L$ (*Korrektheit*),
2. für alle $x \in L$ existiert ein $\pi \in \Sigma^*$ mit $(x, \pi) \in S$ (*Vollständigkeit*),
3. es existiert ein deterministischer Algorithmus, der in Zeit $(|x| + |\pi|)^{\mathcal{O}(1)}$ entscheidet, ob $(x, \pi) \in S$ ist (*Polynomialzeit-Verifizierbarkeit*).

Jedes solche $\pi$ aus Eigenschaft 2 nennen wir *$S$-Beweis* von $x$. Wir werden auch von einem

Beweis von $x$ im Beweissystem $S$ sprechen. Ist klar, über welches Beweissystem wir reden, schreiben wir zukünftig auch knapp Beweis von $x$.

Etwas nützlicher und anschaulicher ist es, ein Beweissystem $S$ für eine Sprache $L$ über einem endlichen Alphabet $\Sigma$ als deterministischen „Verifikations"-Algorithmus $V(x, \pi)$ anzusehen, der als Eingabe das Wort $x$ und den (vermeintlichen) Beweis $\pi$ erhält und in Polynomialzeit in $x$ und $\pi$ läuft, sodass für alle Wörter $x \in \Sigma^*$ gilt:

$$x \in L \iff \text{es existiert ein String } \pi, \text{ sodass } V \text{ die Eingabe } (x, \pi) \text{ akzeptiert.}$$

Wir werden daher zukünftig in den meisten Fällen ein Beweissystem mit diesem Polynomialzeit-Algorithmus identifizieren, über den wir als Beweisverifizierer denken. Wir schreiben $V(x, \pi) = 1$, falls $V$ die Eingabe $(x, \pi)$ akzeptiert, und $V(x, \pi) = 0$, falls $V$ sie verwirft.

**Bemerkung 3.3.2.** Es sei angemerkt, dass ein Beweis $\pi$ sehr groß im Vergleich zum Wort $x$ sein kann, welches er beweist (siehe z. B. Theorem 4.2.2). Wir fordern lediglich, dass die Beweis*verifikation* in polynomieller Zeit in $|x| + |\pi|$ erreicht werden kann.

Besonders wichtig sind Beweissysteme für die Sprache TAUT der aussagenlogischen Tautologien, die wir in der folgenden Definition einführen wollen. Seit[10] STEPHEN COOKs Artikel [Coo1971] über NP-Vollständigkeit bzw. spätestens durch die Millennium-Probleme [CJW2006], insbesondere dem $\mathsf{P} \stackrel{?}{=} \mathsf{NP}$-Problem [Coo2006], ist die Frage „*Ist es effizient möglich zu entscheiden, ob eine beliebige gegebene aussagenlogische Formel eine Tautologie ist?*" von essentieller Wichtigkeit; denn $\text{TAUT} \in \mathsf{P}$ gilt genau dann, wenn $\mathsf{P} = \mathsf{NP}$ ist (siehe z. B. [Coo1971] bzw. Anhang B). Wir verweisen bereits hier auf Theorem 3.3.9.

Neben dieser Tragweite für die Theorie, erwähnen wir erneut, wie in Abschnitt 1.1, die Bedeutung für die Anwendung: Alle bekannten Algorithmen für TAUT benötigen exponentiell viel Zeit im worst case (trotz der enormen Fortschritte der letzten Jahre, u. a. in [BS1997, MS1999, MMZ+2001, ES2004, AS2009, Bie2010]); werden aber in zahlreichen Anwendungen, wie Hardware-Verifikation (genauer *bounded model checking* (*BMC*), siehe

[10] Nicht unerwähnt lassen wollen wir JOHN NASHs Brief [Nas1955] an die NSA, in dem er einige Konzepte der Komplexitätstheorie und Kryptographie vorausahnte, sowie KURT GÖDELs Brief [Göd1956] an JOHN VON NEUMANN kurz vor dessen Tod, indem er die Frage aufwarf, ob *theorem-proving* für Aussagenlogik (von dem man heute weiß, dass es co-NP-vollständig ist) in quadratischer oder gar linearer Zeit möglich ist und erläuterte, dass als Konsequenz die Entdeckung mathematischer Beweise effizient automatisiert werden könnte. Für eine Diskussion von GÖDELs Briefs im Zusammenhang mit dem $\mathsf{P} \stackrel{?}{=} \mathsf{NP}$-Problem verweisen wir auf die Survey Paper [Har1993], [Aar2016] (in welchem die Signifikanz des obigen Problems gut erörtert wird), [Lip2010], [Wig2011] sowie [Sip1992] (wo sich ein Abdruck des Originalbriefes findet).

u. a. [ST2013, § 1.4] und [BHMW2009, Kapiel 14]), bei Software-Tests ([BHMW2009, Kapiel 16]), im sog. software package management (vgl. z. B. [ST2013, § 1.4]), bei Künstlicher Intelligenz ([BHMW2009, Kapitel 15]), Kryptologie (siehe [ST2013, § 1.4] und [Weg1987, Kapitel 3]), in der Bioinformatik, im Gebiet Operations Research der angewandten Mathematik (siehe z. B. [Bru2014, Bru2015]), oder sogar für Schienenverkehrssignale benötigt, um [Nor2016a, Nor2016b] zu zitieren. Es konnten sogar offene gruppentheoretische Probleme der abstrakten Algebra mit SAT-Solvern gelöst werden (siehe [ZH1994]). Dennoch kennen wir Beispiele von Formeln mit sehr wenigen Variablen (z. B. $\mathrm{PHP}_n^{n+1}$ aus Formel 4.2.1), die selbst für modernste SAT-Solver ein Problem darstellen ([Nor2015]).

**Definition 3.3.3.** Ein *aussagenlogisches Beweissystem* (engl. *propositional proof system*, kurz: *pps*) ist ein Beweissystem für die Sprache $L = \mathrm{TAUT}$, der Menge der aussagenlogischen Tautologien.

Wir bemerken, dass ein aussagenlogisches Beweissystem also ein Polynomialzeit-Algorithmus $V$ ist, sodass für alle Formeln $F$ gilt: $F$ ist eine Tautologie genau dann, wenn es einen Beweis $\pi$ gibt, sodass $V$ den Input $(F, \pi)$ akzeptiert.

**Bemerkung 3.3.4.** Wegen der Äquivalenz $F \in \mathrm{UNSAT} \Leftrightarrow \neg F \in \mathrm{TAUT}$ aus Proposition A.0.1 können wir ein aussagenlogisches Beweissystem alternativ auch als Beweissystem für die Sprache UNSAT charakterisieren.

Oftmals konzentriert man sich auch auf aussagenlogische Beweissysteme für Formeln in Konjunktiver Normalform. Dies ist essentiell dasselbe Problem, da es mit der Tseitin-Transformation C.0.3 möglich ist, jede Formel in eine sat-äquivalente CNF-Formel zu transformieren, die lediglich linear größer als die Ausgangsformel ist.

**Notiz 3.3.5** (Klassische Definition eines Beweissystems)**.** Aussagenlogische Beweissysteme wurden von Cook und Reckhow in [CR1974a] eingeführt, weshalb sie heute gelegentlich auch als *Cook-Reckhow-Systeme* bezeichnet werden. Dabei wurde ein Cook-Reckhow-System klassischerweise als eine Polynomialzeit-berechenbare surjektive Funktion $f\colon \Sigma^* \to \mathrm{TAUT} \subset \Sigma^*$ definiert. Die Interpretation dieser Definition: Jeder String aus $\Sigma^*$ wird als potentieller Beweis angesehen und auf diejenige Tautologie abgebildet, die er beweist. Maßgebend für diese Definitionsidee war laut [Rec1975, § 3.3.1] die „Rate und Verifiziere“-Charakterisierung B.1.2 der Klasse NP: Ist $f(\pi) = x$, so kann das Finden eines solchen $\pi$ intuitiv als „Raten des Beweises“ und die Polynomial-Zeit Berechnung von $f(\pi)$ als „Verifikation“ des Faktes $x \in \mathrm{TAUT}$ aufgefasst werden.

Im Fall $L = \mathrm{TAUT}$ stimmt Cooks und Reckhows Definition mit Definition 3.3.1 überein.

*Beweisskizze.* „$\longrightarrow$“: Wir definieren die Relation $S$ über

$$(F, \pi) \in S \iff f(\pi) = F.$$

„$\longleftarrow$“: Wir definieren

$$f(F\#\pi) = \begin{cases} F, & \text{falls } (F, \pi) \in S \\ \texttt{True}, & \text{sonst.} \end{cases}$$

□

Eine natürliche Frage ist, wie groß ein Beweis $\pi$ in einem Beweissystem $V$ für ein gegebenes Wort $x$ ist, etwas genauer, wie groß $|\pi|$ als Funktion in $|x|$ ist (was Einfluss auf die Effizienz von $V$ hat). Hierfür definieren wir die Komplexität eines Beweissystems. Damit lassen sich insbesondere Vergleiche zwischen verschiedenen Beweissystemen durchführen.

**Definition 3.3.6.** Die *Komplexität* $\mathrm{comp}_V$ eines Beweissystems $V$ für die Sprache $L$ ist die Funktion $f\colon \mathbb{N} \to \mathbb{N}$ definiert durch

$$f(n) := \max_{\substack{x \in L \\ |x|=n}} \min_{\substack{\text{Beweise } \pi\colon \\ V(x,\pi)=1}} |\pi| = \max_{\substack{x \in L \\ |x|=n}} \mathrm{comp}_V(x), \tag{3.1}$$

wobei wir

$$\mathrm{comp}_V(x) := \min_{\substack{\text{Beweise } \pi\colon \\ V(x,\pi)=1}} |\pi| \tag{3.2}$$

als die Größe des kleinsten $V$-Beweises von $x$ zusätzlich definiert haben. Das Beweissystem $V$ heißt *polynomiell-beschränkt* (oder *p-beschränkt*, historisch *super* [Bus1999]), falls $f = \mathrm{comp}_V$ nach oben durch ein Polynom $p$ in $n$ beschränkt werden kann. Dann ist $\mathrm{comp}_V(x) \leq p(|x|)$ für alle $x \in L$. Mit anderen Worten: $V$ ist polynomiell-beschränkt, falls es ein Polynom $p$ gibt, sodass jedes $x \in L$ einen $V$-Beweis $\pi$ der Größe $|\pi| \leq p(|x|)$ besitzt.

Wir betrachten polynomiell-beschränkte Beweissysteme als effizient.

**Beispiel 3.3.7** ([Nor2016a])**.** Das aussagenlogische Beweissystem Wahrheitstafeln TT, das wir in Abschnitt 3.4.1 vorstellen werden, hat exponentielle Komplexität $\mathrm{comp}_{\mathtt{TT}}$.

Eines der bekanntesten Resultate aus dem Bereich der Beweiskomplexität stammt aus dem bahnbrechenden Paper [CR1974a] von Cook und Reckhow bzw. aus Reckhows Dissertation [Rec1975, Lemma 3.3.2.a]. Es schlägt eine Brücke zwischen Beweiskomplexität

und Komplexitätstheorie, indem es die Komplexitätsklasse NP als die Menge derjenigen Sprachen identifiziert, die ein polynomiell-beschränktes Beweissystem besitzen.

**Satz 3.3.8** ([CR1974a])**.** *Eine Sprache $L$ besitzt genau dann ein polynomiell-beschränktes Beweissystem, wenn $L \in \mathsf{NP}$ ist.*

*Beweis.* Dies ist unmittelbar klar aus der Suchproblemdefinition B.1.2 der Komplexitätsklasse NP. Angelehnt an [Pap1994, Proposition 9.1] und [Rec1975, Lemma 3.3.2.a] geben wir dennoch einen Beweis dieses wichtigen Satzes, der die Äquivalenz der Suchproblemdefinition mit der konventionellen Definition B.1.1 von NP etabliert:

„$\longrightarrow$“: Angenommen, die Sprache $L$ besitzt ein polynomiell-beschränktes Beweissystem $S \subset \Sigma^* \times \Sigma^*$ bzgl. des Polynoms $p$ im Sinne der Definitionen 3.3.1 und 3.3.6. Es kann eine nicht-deterministische Turingmaschine $N$ konstruiert werden, die bei Eingabe $x$ nicht-deterministisch einen String $y$ der Länge $\leq p(|x|)$ auf ihr Band schreibt und dann überprüft, ob $(x, y) \in S$ ist. Falls dies der Fall ist, akzeptiert $N$ die Eingabe $x$, andernfalls verwirft $N$ die Eingabe $x$.

Unmittelbar aus den Punkten 1 und 2 der Definition 3.3.1 ist klar, dass solch ein String $y$ genau dann existieren muss, falls $x \in L$ ist.

Weiterhin ist es nach Punkt 3 der Definition 3.3.1 möglich in Zeit

$$\leq (|x| + |y|)^{\mathcal{O}(1)} \leq (|x| + p(|x|))^{\mathcal{O}(1)} = \text{poly}(|x|)$$

zu entscheiden, ob $(x, y) \in S$ ist. Gemäß der Sprachakzeptanz-Definition B.1.1 ist $L \in \mathsf{NP}$.

„$\longleftarrow$“: Angenommen es gilt $L \in \mathsf{NP}$. Dann existiert gemäß Sprachakzeptanz-Definition B.1.1 eine nicht-deterministische Turingmaschine $M$ und ein Polynom $p$, sodass es zu jedem $x \in L$ einen akzeptierenden Berechnungspfad von $M$ bei Eingabe $x$ mit Länge höchstens $p(|x|)$ gibt. Dieser lässt sich in einem String $c_M^{\text{acc}}(x) \in \{0,1\}^*$ der Länge $\mathcal{O}(p(|x|))$ codieren.[11] Es bezeichne $C_M^{\text{acc}}(x)$ die Menge der akzeptierenden

[11] Falls nötig, nehmen wir o. B. d. A. an, dass jeder Zustand in $M = (Z, \Sigma, \Gamma, \delta, z_0, \square, E)$ höchstens zwei Nachfolgezustände haben kann. Ggfs. muss die Maschine dazu modifiziert werden, was die Laufzeit jedoch nur um einen konstanten Faktor $c$ ändert; respektive muss die Codierung des Pfades angepasst werden: Gilt für die ursprüngliche Maschine $|\delta(q, a)| \leq d$ für alle $q \in Z \setminus E$ und alle $a \in \Gamma$, so gibt es höchstens $d^{p(|x|)}$ Berechnungspfade. Die Konstante kann dann als $c = \log_2(d)$ gewählt werden, sodass $2^{cp(|x|)} = d^{p(|x|)}$ gilt, d. h. statt einer Stringlänge von $p(|x|)$ benötigen wir eine Länge von $cp(|x|)$, um alle Berechnungspfade einzeln zu adressieren. Gemäß dieser Argumentation können wir annehmen, dass $c_M(x) \in \{0,1\}^*$ ist.

Berechnungspfade von $M$ bei Eingabe $x$. Die Relation des polynomiell-beschränkten Beweissystems kann dann über

$$S_L := \Big\{(x, c_M(x)) \,:\, x \in L \text{ und } c_M(x) \in C_M^{\text{acc}}(x)\Big\}$$

definiert werden. Sie erfüllt offensichtlich die Eigenschaft 1 und 2 aus Definition 3.3.1.

Um die Polynomialzeit-Verifizierbarkeit der Relation $S_L$ nachzuweisen, betrachten wir die deterministische Turingmaschine $M_{\text{det}}$, die bei Eingabe $x\#c_M(x)$ die Maschine $M$ mit Eingabe $x$ simuliert und dabei dem durch $c_M(x)$ vorgegebenen Berechnungspfad folgt. Falls $M$ dabei in einem akzeptierenden Zustand endet, wird „$(x, c_M(x)) \in S_L$“ von $M_{\text{det}}$ zurückgegeben, andernfalls (oder falls $c_M(x)$ keine Codierung eines Berechnungspfades ist) „$(x, c_M(x)) \notin S_L$“. Dabei ist es möglich, $M_{\text{det}}$ so zu konstruieren, dass die Laufzeit von $M_{\text{det}}$ bei Eingabe $x\#c_M(x)$ abgeschätzt werden kann durch

$$\text{time}_{M_{\text{det}}}\Big(x\#c_M(x)\Big) \leq \text{poly}(|c_M(x)|) \leq \text{poly}(\mathcal{O}(p(|x|))) \leq \text{poly}(|x|),$$

was den Nachweis der Eigenschaft 3 aus Definition 3.3.1 abschließt. □

Das Beweisen von Tautologien ist für sich allein betrachtet ein wichtiges und interessantes Problem in Anwendungen. Das folgende Theorem macht jedoch deutlich, warum Beweiskomplexität auch aus theoretischer Sicht interessant ist, da sich die Frage nach der Komplementabgeschlossenheit von NP umformulieren lässt.

**Theorem 3.3.9** (Cook-Reckhow-Theorem [CR1974a])**.** *Es existiert ein polynomiell-beschränktes aussagenlogisches Beweissystem genau dann, wenn* NP = co-NP *ist.*

*Beweis.* „$\longrightarrow$“: Angenommen, es gibt ein polynomial-beschränktes aussagenlogisches Beweissystem. Dann ist TAUT $\in$ NP nach Satz 3.3.8. Da jedoch TAUT gemäß Korollar B.2.8 vollständig für co-NP ist, folgt mit Theorem B.2.6 (ii) sofort NP = co-NP.

„$\longleftarrow$“: Wegen NP = co-NP ist TAUT $\in$ NP, d. h. TAUT hat ein polynomiell-beschränktes Beweissystem nach Satz 3.3.8. Gemäß Definition 3.3.3 ist dies ein polynomiell-beschränktes aussagenlogisches Beweissystem. □

Mit anderen Worten: UNSAT bzw. TAUT haben polynomiell-beschränkte Beweissysteme genau dann, wenn NP = co-NP ist. Bzw. umkehrt: Es existieren polynomial-beschränkte Beweissysteme für TAUT und UNSAT (d. h. aussagenlogische Beweissysteme) genau dann, wenn eine (und damit beide) Sprachen zu NP gehören.

Allgemeiner als Theorem 3.3.9 lässt sich festhalten, dass NP genau dann komplementabgeschlossen ist, wenn es für eine (und damit für alle) NP-vollständigen Sprachen $L$ ein polynomiell-beschränktes Beweissystem für $\overline{L}$ gibt. Die natürliche Wahl hierfür ist jedoch die Suche nach polynomiell-beschränkten Beweissystemen für TAUT, da wir die vielen Beweissysteme für TAUT, die Logiker im Laufe der Jahre formuliert haben, untersuchen können. Dies führt zu der Version der Aussage, wie in Theorem 3.3.9 formuliert.

Gleichzeitig erhalten wir direkt die Hauptfragestellung der Beweiskomplexität, die eine Möglichkeit bietet, das $\mathsf{NP} \stackrel{?}{=} \mathsf{co\text{-}NP}$-Problem zu lösen:

**Problem 3.3.10** (Das Hauptproblem der Beweiskomplexität)**.** Gibt es ein polynomiell-beschränktes aussagenlogisches Beweissystem? Gilt also $\mathsf{NP} = \mathsf{co\text{-}NP}$?

**Korollar 3.3.11.** *Falls alle aussagenlogischen Beweissysteme superpolynomielle Komplexität haben (es also kein polynomiell-beschränktes aussagenlogisches Beweissystem gibt), dann ist* $\mathsf{P} \neq \mathsf{NP}$.

*Beweis.* Haben alle aussagenlogischen Beweissysteme superpolynomielle Komplexität, so folgt mit Theorem 3.3.9 direkt $\mathsf{NP} \neq \mathsf{co\text{-}NP}$, d. h. NP ist unter Komplementbildung nicht abgeschlossen. Als deterministische Komplexitätsklasse ist P jedoch unter Komplementbildung abgeschlossen, also muss $\mathsf{P} \neq \mathsf{NP}$ gelten. Siehe auch Abbildung B.2. □

Die folgenden Diskussionen sind angelehnt an die Darstellungen in [Bon2018, BP2001, Gal2009, Hof2007, Rac2007].

Um zu beweisen, dass ein Beweissystem $V$ nicht polynomiell-beschränkt ist, genügt es nach (3.1) eine Folge $(F_n)_{n\in\mathbb{N}}$ von Formeln (ggfs. auch nicht-konstruktiv) zu finden, sodass $|F_n| \leq p(n)$ für ein Polynom $p$ und alle $n \in \mathbb{N}$ gilt, aber

$$\mathrm{comp}_V(F_n) = \min_{\substack{\text{Beweise } \pi\text{:} \\ V(F_n,\pi)=1}} |\pi| \geq f(n)$$

für eine superpolynomielle Funktion $f$ ist (d. h. die minimale Länge von Beweisen von $F_n$ wächst im Beweissystem $V$ superpolynomiell bezüglich $|F_n|$). Diese Funktion $f$ nennt man *superpolynomielle untere Schranke* für $V$. Hilfreich hierfür ist es die „Codierung“ von gut studierten kombinatorischen Problemen zu verwenden: Wir verweisen auf Abschnitt 4.2, insbesondere auf $\mathrm{PHP}_n^{n+1}$ aus Formel 4.2.1, für Beispiele hierzu.

Motiviert durch das Hauptproblem der Beweiskomplexität 3.3.10 wurde bereits eine Vielzahl an aussagenlogischen Beweissystemen untersucht und gezeigt, dass sie nicht polynomiell-beschränkt sind bzw. superpolynomielle untere Schranken haben. Einen Überblick

bietet das Survey Paper [BP2001]. Hierdurch kann man diese Systeme von weiteren Betrachtungen mit dem Ziel der Beantwortung der obigen Frage ausschließen.

Ebenso ist es nützlich, Resultate zu erhalten, die aussagen, dass ein Beweissystem polynomiell-beschränkt ist genau dann, wenn ein anderes System dies ist. Dies erlaubt, den Fokus auf *eines* der Systeme einzuengen.

Um zu zeigen, dass es kein polynomiell-beschränktes aussagenlogisches Beweissystem gibt, werden zudem Versuche unternommen, untere Schranken für immer „stärkere“ Beweissysteme zu zeigen, in der Hoffnung, dass irgendwann genügend Techniken und Methoden gesammelt wurden, um superpolynomielle untere Schranken für *alle* Beweissysteme zu zeigen und damit die Antwort $\mathsf{NP} \neq \mathsf{co\text{-}NP}$ durch das Cook-Reckhow Theorem 3.3.9 zu erhalten – diese Unternehmungen nennt man *Cooks Programm*. Einen guten Überblick über die bisherigen Anstrengungen und Ergebnisse sowie offenen Probleme diesbezüglich liefert das Survey Paper [Bus2012].

Um Resultate der oben angesprochenen Form zu erhalten und Beweissysteme, genauer deren „Effizienz“, zu vergleichen, sind die Konzepte im Folgenden nützlich. Sie dienen zudem dem Versuch der Beantwortung natürlicher Fragen, die sich unmittelbar an die Definition der Komplexität eines Beweissystems stellen:

- Wie effizient sind bekannte Beweissysteme?
- Lassen sich Beweissysteme hinsichtlich ihrer Komplexität untereinander vergleichen?
- Gibt es optimale Beweissysteme (bis auf ein Polynom)?

Die folgenden Definitionen können leicht auf allgemeine Beweissysteme verallgemeinert werden. Jedoch fokussieren wir uns motiviert durch das Hauptproblem der Beweiskomplexität 3.3.10 ab hier auf aussagenlogische Beweissysteme.

**Definition 3.3.12.** (i) Ein aussagenlogisches Beweissystem $V_1$ *p-simuliert* ein aussagenlogisches Beweissystem $V_2$, wofür wir $V_2 \preccurlyeq_p V_1$ schreiben wollen[12], falls es eine in polynomial-Zeit berechenbare Funktion $f$ gibt, sodass gilt:

$$V_2(F,\pi) = 1 \iff V_1(F, f(\pi)) = 1 \quad \text{für alle } F \in \textsc{taut} \text{ und alle Beweise } \pi. \tag{3.3}$$

[12]Der Autor weist an dieser Stelle auf die Analogie der Schreibweise $V_2 \preccurlyeq_p V_1$ für den Sachverhalt (3.3) und der Schreibweise für Polynomialzeitreduktionen zwischen formalen Sprachen hin: Eine Sprache $L_1 \subset \Sigma^*$ heißt *polynomiell many-one-reduzierbar* auf eine Sprache $L_2 \subset \Gamma^*$, in Zeichen $L_1 \preccurlyeq_m^P L_2$, falls es eine in polynomieller Zeit berechenbare Stringfunktion $f\colon \Sigma^* \to \Gamma^*$ gibt, sodass für alle $x \in \Sigma^*$ die Äquivalenz

$$x \in L_1 \iff f(x) \in L_2$$

gilt. Vergleiche dazu auch Definition B.2.2 im Anhang B.

Beame und Pitassi bezeichneten $V_2 \preccurlyeq_p V_1$ bzw. die Notation $V_2 \leq_p V_1$ in [BP2001] noch als nicht-Standard. Der Autor der vorliegenden Arbeit ist jedoch der Meinung, dass sich diese Notation seit 1998 verbreitet und durchgesetzt hat.

(ii) $P$-Simuliert $V_1$ das Beweissystem $V_2$ *nicht*, so nennen wir $V_2$ *separiert* von $V_1$ und schreiben $V_2 \not\preccurlyeq_p V_1$.

Wegen (3.2) folgt aus (3.3) direkt die Existenz eines Polynoms $p$, sodass die Ungleichung

$$\mathrm{comp}_{V_1}(F) \leq p\Big(\mathrm{comp}_{V_2}(F)\Big) \quad \text{für alle } F \in \textsc{taut} \tag{3.4}$$

gilt.[13] $P$-Simuliert also $V_1$ das System $V_2$ und ist $V_1$ nicht polynomiell-beschränkt, so kann $V_2$ auch nicht polynomiell-beschränkt sein (siehe z. B. [Gal2009]). Ebenso ergibt sich hieraus eine Beweismethode, um $V_2 \not\preccurlyeq_p V_1$ zu zeigen: Man zeigt dies durch Konstruktion einer Familie $(F_n)_{n\in\mathbb{N}}$ von unerfüllbaren Formeln, sodass der kleinste Beweis von $F_n$ in $V_1$ für alle $n \in \mathbb{N}$ die Größe $f\Big(\mathrm{comp}_{V_2}(F_n)\Big)$ hat, wobei $f$ eine superpolynomielle Funktion ist (vgl. [Rac2007, S. 12]).

Motiviert durch (3.4) betrachten wir zwei Beweissysteme als „gleich mächtig“, wenn ihre Komplexität jeweils nur um einen polynomiellen Faktor verschieden ist. Wir konkretisieren dies in der nachfolgenden Definition.

**Definition 3.3.13.**

(i) Zwei Beweissysteme $V_1$ und $V_2$ heißen *polynomiell äquivalent*, oder *$p$-äquivalent*, wenn jedes Beweissystem das andere $p$-simuliert. Wir schreiben $V_1 \equiv_p V_2$.

(ii) Gilt $V_2 \preccurlyeq_p V_1$ und ist $V_1$ separiert von $V_2$ (d. h. gilt $V_1 \not\preccurlyeq_p V_2$), dann schreiben wir $V_2 \prec_p V_1$ und nennen $V_1$ *mächtiger als* $V_2$.

(iii) Falls weder $V_1 \preccurlyeq_p V_2$ noch $V_2 \preccurlyeq_p V_1$ gilt, so heißen $V_1$ und $V_2$ *unvergleichbar*.[14]

Offensichtlich sind zwei $p$-äquivalente Beweissysteme entweder beide polynomiell-beschränkt, oder keines ist polynomiell-beschränkt.

In der Vergangenheit wurden zahlreiche Resultate erzielt, welche die $p$-Simulation verschiedener Beweissysteme untersucht haben. Als Beispiel führen wir an, dass alle Frege-Systeme $p$-äquivalent sind (siehe Satz & Beispiel 3.4.10; [CR1974a]). In Bemerkung 3.5.8 werden wir anmerken, dass Frege-Systeme mächtiger als Resolution sind. Siehe ebenso Bemerkung & Beispiel 3.5.17, sowie Notiz 3.5.18 für eine Trennung verschiedener Resolutionsarten. Da dies aber keinen Fokus dieser Arbeit darstellt, halten wir uns an dieser Stelle kurz.

Es ist zudem leicht nachzuweisen, dass das Konzept der $p$-Simulation eine Quasiordnung $\preccurlyeq_p$ auf der Menge $\mathfrak{M}$ der aussagenlogischen Beweissysteme definiert (vgl. Beispiel &

[13] Gilt die Ungleichung (3.4), wissen wir aber nicht, ob eine Reduktionsabbildung $f$, sodass (3.3) gilt, existiert, spricht man auch von *schwacher $p$-Simulation*.

[14] Man beachte, dass $\preccurlyeq_p$ keine Totalordnung ist (siehe hierzu Fußnote 47 auf Seite 163) und dieser Fall eintreten kann.

Satz D.1.6), ebenso, wie $\equiv_p$ eine Äquivalenzrelation auf $\mathfrak{M}$ definiert. Nach Faktorisierung durch die Äquivalenzrelation $\equiv_p$ erhalten wir die kanonische Partialordnung auf $\mathfrak{M}/_{\equiv_p}$. Ausführliche Details zu diesen Begriffen und dem genauen Vorgehen haben wir im Anhang D, insbesondere in Hauptbeispiel D.3.6, bereit gestellt.

Eine natürliche Frage ist, ob es ein (oder mehrere) maximale/s Element/e bzgl. dieser Ordnung gibt (vgl. Definition D.2.1, Notiz D.2.2 und Beispiel D.2.3). Ein solches maximales Element bezüglich dieser Partialordnung nennen wir – angelehnt an die Definition von Krajíček und Pudlák in [KP1989] – *p-optimales* aussagenlogisches Beweissystem (und vernachlässigen dabei wie in Notiz D.3.7 erläutert die Unterscheidung zwischen Äquivalenzklassen und Repräsentanten). M. a. W. heißt ein aussagenlogisches Beweissystem $p$-optimal, falls es alle aussagenlogischen Beweissysteme simuliert (siehe z. B. [Bus1999]). Historisch heißt ein $p$-optimales aussagenlogisches Beweissystem auch *super-duper*. Die Existenz eines $p$-optimalen aussagenlogischen Beweissystems ist ungeklärt:

**Problem 3.3.14** (Das *Optimalitätsproblem* nach [KP1989])**.** Gibt es $p$-optimale aussagenlogische Beweissysteme?

In Hinblick auf das Hauptproblem der Beweiskomplexität 3.3.10 wäre die Existenz eines $p$-optimalen aussagenlogischen Beweissystems äußerst nützlich: Um NP von co-NP zu separieren, würde es genügen, zu zeigen, dass ebendieses System nicht $p$-beschränkt ist. Jedoch konnten in [KMT2003] Resultate erzielt werden, die Hinweise darauf geben, dass es keine $p$-optimalen aussagenlogischen Beweissysteme gibt.

Wir schließen diesen Abschnitt mit einem weiteren Resultat ab, das Beweiskomplexität mit Komplexitätstheorie verbindet. Es liefert eine hinreichende Bedingung für die Existenz eines p-optimalen aussagenlogischen Beweissystems.

**Theorem 3.3.15** (Theorem 2.1. (ii) in [KP1989])**.** *Ist* EXP = NEXP*, dann gibt es ein p-optimales aussagenlogisches Beweissystem.*

Leider gewährt uns dieses Resultat keine weiteren Erkenntnisse, da die obige Komplexitätsklassen-Gleichheit als unwahrscheinlich gilt ([Nor2008, S. 38]). Nicht unerwähnt lassen, wollen wir jedoch die Erfolge im Verschärfen der hinreichenden Bedingung EXP = NEXP im Paper [KMT2003], das zudem weitere komplexitätstheoretische Konsequenzen aus der Existenz von $p$-optimalen aussagenlogischen Beweissystemen folgert.

Zum Abschluss unserer Einführung in die Beweiskomplexität sprechen wir den Themenkomplex *Automatisierbarkeit* (engl. *automatizability*) an, da die bisherigen Definitionen keine Auskunft darüber erlauben, wie schwer es ist, einen Beweis zu *finden*: Kurze Beweise mögen existieren, es könnte jedoch sein, dass sie schwer zu finden sind. Unsere Darstel-

lungen im Folgenden sind eine Adaption von [Bea2004, § 3.1], [Gal2009, S. 31], [Nor2008, Definition 4.8] und [Pit2002].

**Problem 3.3.16** (Das Kürzester-Beweis-Finden-Problem, [Pit2002])**.** Es sei $V$ ein aussagenlogisches Beweissystem. Das *Kürzester-Beweis-Finden-Problem* für $V$ ist folgende Problemstellung: Gegeben eine Boolesche Formel $F \in \text{TAUT}$, finde den kürzesten $V$-Beweis für den Fakt $F \in \text{TAUT}$.

Entsprechend dem obigen Problem sagen wir, dass $A_V$ ein *Beweis-Suche-Algorithmus* für das aussagenlogische Beweissystem $V$ ist, falls $A_V$ ein deterministischer Algorithmus ist, dessen Eingabe eine Formel $F$ ist und der als Ausgabe einen Beweis $\pi$ für $F$ im System $V$ liefert (also einen String $\pi$, sodass $V(F, \pi) = 1$ ist), falls $F \in \text{TAUT}$ ist, bzw. andernfalls „`falsifizierbar`“ ausgibt.

Das Problem an obiger Aufgabenstellung ist jedoch, dass versucht wird, den absolut kürzesten Beweis algorithmisch finden. Dieses Problem ist aber für fast alle Beweissysteme NP-schwer ([Pit2002]). Wir nehmen also eine Relaxation der Definition vor und versuchen statt des kürzesten Beweises einen Beweis zu finden, dessen Länge eine gute Approximation der kürzesten Länge ist.

**Definition 3.3.17** (Automatisierbarkeit, [Bea2004])**.** Es sei $V$ ein aussagenlogisches Beweissystem und $f\colon \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ eine Funktion.

(i) Das Beweissystem $V$ heißt $f(n, S)$-*automatisierbar*, falls es einen Beweis-Suche-Algorithmus $A_V$ gibt, der bei Eingabe einer Tautologie $F \in \text{TAUT}$ einen $V$-Beweis von $F$ in Zeit höchstens $f(n, S)$ ausgibt, wobei $n$ die Länge von $F$ und $S$ die Länge des kürzesten $V$-Beweises von $F$ bezeichne.

(ii) Wir nennen das Beweissystem $V$ *automatisierbar*, falls es $f(n, S)$-automatisierbar für eine Funktion $f(n, S) = \text{poly}(n) \cdot \text{poly}(S)$ ist.

(iii) Wir nennen das Beweissystem $V$ *quasi-automatisierbar*, falls es $f(n, S)$-automatisierbar für eine Funktion $f(n, S) = n^{c_1} \cdot \exp(\log^{c_2} S)$ und Konstanten $c_1$ und $c_2$ ist.

Wie Nordström in [Nor2008, S. 39] merken wir an, dass Automatisierbarkeit in einem gewissen Sinn „das Beste ist, auf das man hoffen kann“: Gibt es keine kurzen Beweise im Beweissystem $V$, kann auch von keinem Algorithmus $A_V$ erwartet werden, einen Beweis schnell zu finden; gibt es jedoch eine obere Schranke für die Laufzeit des *besten* theoretisch möglichen Beweis-Suche-Algorithmus, so wollen wir, dass $A_V$ bezüglich dieser oberen Schranken eine gute Performance hat.

Wir verdeutlichen die Begriffe aus der obigen Definition anhand einiger Beispiele und ausgewählter tieferer Resultate.

**Beispiel & Satz 3.3.18.** (i) Es existiert ein polynomiell-beschränktes und automatisierbares aussagenlogisches Beweissystem genau dann, wenn $\mathsf{P} = \mathsf{NP}$ ist.

(ii) Das Beweissystem TT der Wahrheitstafeln (siehe Abschnitt 3.4.1) ist automatisierbar; jedoch sind die Beweise von exponentieller Größe, was das Resultat trivial macht.

(iii) Gewöhnliche Resolution (vgl. Abschnitt 3.5) ist $2^{\mathcal{O}(\sqrt{n \log S} \log n)}$-automatisierbar (siehe [BW2001, Bea2004]); baumartige Resolution (siehe Definition 3.5.15) ist $S^{\mathcal{O}(\log n)}$-automatisierbar (siehe [CEI1996, BP2001, Bea2004, BW2001]).

(iv) Resolution ist nicht automatisierbar unter plausiblen komplexitätstheoretischen Annahmen an die Klasse $\mathsf{W}[\mathsf{P}]$ (siehe hierzu [AR2008]).

Bedauerlicherweise gibt es nach [Nor2008, S. 40] *vermutlich* einen *Tradeoff* für Beweissysteme: Ist ein Beweissystem „hinreichend stark“, so ist es nicht automatisierbar. Für Hinweise in diese Richtung (unter plausiblen kryptographischen Annahmen) verweisen wir auf [BDG$^+$2004] bzw. [Nor2008, Corollary 4.13].

## 3.4 Erste Beispiele für Beweissysteme

Motiviert von einer der zentralsten Probleme der Aussagenlogik – nützliche Methoden und Algorithmen zum Erkennen von Tautologien finden – stellen wir im folgenden eine kleine Auswahl an Beweissystemen für diese Fragestellung vor. Logiker haben in den vergangenen Jahrzehnten eine Vielzahl von Beweissystemen vorgeschlagen. Allen gemeinsam ist, dass sie einfache Regeln beinhalten, um Beweise für gegebene Formeln zu konstruieren.

Wir weisen darauf hin, dass das Beweissystem der Resolution von übergeordneter Wichtigkeit für die vorliegende Arbeit ist und sich deshalb im separaten Abschnitt 3.5 findet.

### 3.4.1 Wahrheitstafeln

Einer der einfachsten und zugleich naïvsten Wege, um zu zeigen, dass eine aussagenlogische Formel eine gewünschte Funktion berechnet (oder spezieller eine Tautologie ist), ist das Angeben einer vollständig ausgefüllten *Wahrheitstafel* (engl. *truth table*). An diesem aussagenlogischen Beweissystem TT lassen sich jedoch mehrere Kritikpunkte festhalten:

1. Der Test, ob die Wahrheitstafel korrekt ausgefüllt ist, kann in polynomieller Zeit relativ zur Zeilenanzahl der Tabelle (also relativ zur Beweisgröße) geschehen. Was ist jedoch der Sinn einer vollständig ausgefüllten Tabelle, wenn dies vom Beweisverifizierer auch selbst gemacht werden könnte? In diesem Sinne, wäre nach [BP2001, § 2] die Formel selbst schon der Beweis.

2. Um zu verifizieren, dass eine Formel mit $n$ verschiedenen Variablen eine Tautologie ist, ist es notwendig alle $2^n$ Zeilen der Wahrheitstafel auf den Eintrag „`True`“ zu überprüfen. Die Beweisgröße $|\pi|$ ist exponentiell in der Länge der Formel (also von exponentieller Komplexität gemäß Definition 3.3.6), sodass das Beweissystem für praktische Anwendungen aufgrund des exorbitant hohen Berechnungsaufwandes nutzlos ist.
3. Die Aussage aus Punkt 2 gilt selbst, wenn die gegebene Formel eine „strukturell einfache“ Tautologie ist, wie bspw. $(x_1 \vee \overline{x_1}) \wedge (x_2 \vee \overline{x_2}) \wedge \cdots \wedge (x_n \vee \overline{x_n})$ oder $G \wedge \neg G$ für eine beliebig komplizierte Formel $G$. Siehe hierzu [Ben2009a, § 1.2].
4. Ein „Beweis“ in Form einer ausgefüllten Wahrheitstafel liefert zudem keinerlei Intuition bzw. Einblick, weshalb die Formel eine Tautologie ist oder weshalb sie dies nicht ist. Reckhow stellte 1975 in seiner Dissertation [Rec1975] heraus, dass die Regeln innerhalb eines Beweissystems zur Konstruktion eines Beweises eines Theorems (einer Formel) einfacher sein sollten, als das Verstehen des Theorems. Ein Beweis stellt somit einen konstruktiven Weg dar, um die Korrektheit eines Theorems zu verstehen.

### 3.4.2 SAT-Algorithmen als Beweissysteme für TAUT

In diesem Unterabschnitt wollen wir kurz die nahe Verbindung zwischen SAT-Algorithmen und Beweissystemen aufzeigen. Wir orientieren uns in unserer Darstellung an [Hof2007].

Wird ein SAT-Algorithmus auf eine unerfüllbare Formeln $F$ angewandt, kann dies wegen Proposition A.0.1 als Beweis für die Tautologie $\neg F$ angesehen werden, sodass wir aus dem Algorithmus ein Beweissystem gewinnen können. Vermöge der Tseitin-Transformation aus Definition & Theorem C.0.3 gilt diese Aussage auch für CNF-SAT-Algorithmen.

**Proposition 3.4.1.** *Ist $A$ ein Algorithmus für* SAT *(oder* CNF-SAT*), der Laufzeit $f(|F|)$ bei Eingabe einer Formel $F$ hat (wobei $f$ eine monoton nicht-fallende Funktion ist), dann gibt es ein aussagenlogisches Beweissystem $S_A$ und eine Konstante $c > 0$, sodass gilt:*

$$\mathrm{comp}_{S_A}(F) \leq f\Big(c|F|\Big) \quad \textit{für alle } F \in \mathrm{TAUT}.$$

*Beweis.* Es sei $f$ eine monoton nicht-fallende Funktion. Wir notieren die Beweise im Folgenden immer über dem Alphabet $\Sigma := \{0\}$.

(I) Es sei zunächst $A$ ein SAT-Algorithmus mit Laufzeit $f(|F|)$ bei Eingabe einer Formel $F$. Wir definieren die Relation $S_A \subset \Sigma^*_{\mathrm{PROP}} \times \Sigma^* \subset \Sigma^*_{\mathrm{PROP}} \times \Sigma^*_{\mathrm{PROP}}$ durch

$$(F, \pi) \in S_A \iff \begin{array}{l} A(\neg F) \text{ terminiert nach maximal } |\pi| \text{ Schritten} \\ \text{und liefert den Output „}\neg F \notin \mathrm{SAT}\text{“}. \end{array}$$

Man beachte, dass $F$ nach Proposition A.0.1 und den Voraussetzungen genau dann eine Tautologie ist, wenn $\left(F, 0^{f(|F|)}\right) \in S_A$ ist, weshalb die Bedingungen 1 und 2 aus Definition 3.3.1 erfüllt sind. Ob der Algorithmus in höchstens $|\pi|$ Schritten terminiert und welchen Output er liefert, kann offensichtlich in Zeit $\mathcal{O}(|\pi|)$ entschieden werden, wodurch Bedingung 3 der Definition 3.3.1 erfüllt ist. Gemäß (3.2) ist schließlich

$$\mathrm{comp}_{S_A}(F) = \min_{\substack{\text{Beweise } \pi\text{:} \\ (F,\pi)\in S_A}} |\pi| \leq \left|0^{f(|F|)}\right| = f(|F|) \quad \text{für alle } F \in \textsc{taut}.$$

(II) Ist hingegen $A$ ein CNF-SAT-Algorithmus mit Laufzeit $f(|F|)$ bei Eingabe einer Formel $F$, so setzen wir $G := \mathrm{Tseitin}(\neg F)$, wobei wir damit die Tseitin-Transformation von $\neg F$ aus Definition & Theorem C.0.3 bezeichnen, und definieren die Relation $S_A \subset \Sigma^*_{\mathrm{PROP}} \times \Sigma^* \subset \Sigma^*_{\mathrm{PROP}} \times \Sigma^*_{\mathrm{PROP}}$ durch

$$(F,\pi) \in S_A \iff \begin{array}{l} A(G) \text{ terminiert nach maximal } |\pi| \text{ Schritten} \\ \text{und liefert den Output „}G \notin \textsc{sat}\text{“}. \end{array}$$

Nach Definition & Theorem C.0.3 gibt es eine *universelle* (d. h. von $F$ unabhängige) Konstante $c > 0$, sodass $|G| = c \cdot |F|$ gilt. Es ist also $F \in \textsc{taut}$ genau dann, wenn $\left(F, 0^{f(c\cdot|F|)}\right) \in S_A$ ist. Wie in (I) zeigt man sofort, dass $S_A$ in linearer Zeit in der zweiten Komponente entscheidbar ist. Ebenso zeigt man wie in (I), dass $\mathrm{comp}_{S_A}(F) \leq f\left(c \cdot |F|\right)$ für alle $F \in \textsc{taut}$ ist. □

Wendet man Proposition 3.4.1 auf den trivialen SAT-Algorithmus (der nacheinander alle $2^n$ möglichen Belegungen prüft, was in Linearzeit möglich ist [Sch2018b, Lemma 2.63]) an, folgt unmittelbar folgendes Korollar.

**Korollar 3.4.2.** *Es gibt ein aussagenlogisches Beweissystem $S$ mit* $\mathrm{comp}_S(F) = \mathcal{O}(|F|2^n)$ *für alle $F \in \textsc{taut}$ in $n$ Variablen.*

Dieser oben aufgeführte Zusammenhang zwischen SAT-Solvern und aussagenlogischen Beweissystemen kann deutlich weiter ausgebaut und ausgenutzt werden: So ist es z. B. möglich aus unteren Schranken für die Komplexität von Beweissystemen auf untere Schranken ganzer Klassen von SAT-Algorithmen zu schließen, was in der Praxis große Relevanz hat.

Ebenso lassen sich Vergleiche zwischen Klassen von SAT-Algorithmen ziehen, indem die entsprechenden korrespondierenden Beweissysteme unter $p$-Simulation verglichen werden.

Wir studieren den Zusammenhang zwischen dem Beweissystem der Resolution und `DPLL`-SAT-Solvern genauer in Abschnitt 4.3, um die Gedanken des Abschnitts 1.1 aufzugreifen.

### 3.4.3 Frege-Systeme

Keine Einführung in die Beweiskomplexität wäre komplett, ohne Frege-Systeme zu erwähnen, da sie eine der wichtigsten Klassen von Beweissystemen bilden. Ursprünglich basieren sie auf Ideen von Gottlob Freges Begriffsschrift [Fre1879] und wurden zu seinen Ehren von Cook und Reckhow so bezeichnet, als diese die $p$-Äquivalenz aller Frege-Systeme bewiesen (siehe Satz & Beispiel 3.4.10).

Wir stellen zunächst die grundlegenden Ideen eines Frege-Systems dar, bevor wir diese formal konkretisieren. Unsere Darstellungen beruhen auf [Rec1975, Bus1998, Hof2007, Nor2008, Sch2016, Sch2018a].

Ein Frege-System besteht aus *Axiomen*, einfachen Formeln, die sofort als Tautologien identifiziert werden können. Mit *Schlussregeln*[15] lassen sich weitere Formeln aus den Axiomen und zuvor abgeleiteten Formeln ableiten. Die Regeln sind dergestalt, dass die Gültigkeit der ursprünglichen Formel auch die Gültigkeit der abgeleiteten Formel impliziert.

Prinzipiell sind viele verschiedene Wahlen für die Menge an Axiomen denkbar. Man verfolgt jedoch das Ziel, dass die Menge groß genug ist, um jede Tautologie aus den Axiomen ableiten zu können. Das System ist dann *vollständig*. Zudem sollte es „leicht“ (z. B. zumindest in polynomieller Zeit) prüfbar sein, ob eine Schlussregel angewendet werden kann.

Ein Beweis für den Fakt $F \in \text{TAUT}$ ist schließlich eine Ableitung von $F$ aus den Axiomen gemäß den Regeln des Systems.

Wir formalisieren diese grundlegenden Ideen im Folgenden.

**Definition 3.4.3.** Es seien $F_1(x_1, \ldots, x_n), \ldots, F_m(x_1, \ldots, x_n)$ und $G(x_1, \ldots, x_n)$ aussagenlogische Formeln in den Variablen $x_1, \ldots, x_n$. Eine *Schlussregel*[15] (auch *Inferenzregel*) ist ein Tupel

$$S := \Big( \{F_1, \ldots, F_m\}, G \Big) \tag{3.5}$$

so, dass

$$(F_1 \wedge \cdots \wedge F_m) \vDash G \tag{3.6}$$

[15] Genauer handelt es sich bei diesen Regeln um *Regelschemata*. Eine Regel kann als unendliche Menge von Regeln betrachtet werden, da (wie wir auf Seite 42 erklären werden) eine Regel angewandt wird, indem beliebige Formeln für die in der Regel enthaltenen Variablen substituiert werden. Wir verweisen auch auf Beispiel 3.4.4 (a): Mit der Schlussregel des Modus Ponens lässt sich aus den Formeln $\phi_1$ und $\phi_1 \to \phi_2$ die Formel $\phi_2$ folgern – dabei ist es egal, ob z. B. $\phi_1$ nur ein Literal oder eine komplexere Formel ist.

gilt[16]. Wir notieren eine solche Schlussregel für gewöhnlich als

$$S = \frac{F_1, \ldots, F_m}{G}.$$

Enthalten die Formeln in der Menge $\Gamma := \{F_1, \ldots, F_m\}$ sowie die Formel $G$ nur logische Operatoren aus einer Menge $\kappa$, so nennen wir $S$ eine *Schlussregel in* $\kappa$. Ist $\Gamma = \varnothing$, liegt eine Schlussregel $S = (\varnothing, G)$ ohne Voraussetzungen vor, was wir für gewöhnlich mit $\frac{}{G}$ notieren und *Axiom*[17] nennen.

Eine Schlussregel $S$ wie aus (3.5) wird angewandt, indem beliebige Formeln für die Variablen $x_1, \ldots, x_n$ substituiert werden[18], d. h. wir wenden $S$ auf beliebige aussagenlogische Formeln $\varphi_1, \ldots, \varphi_n$ an und erhalten die semantische Folgerung

$$\Big(F_1(\varphi_1, \ldots, \varphi_n) \wedge \cdots \wedge F_m(\varphi_1, \ldots, \varphi_n)\Big) \vDash G(\varphi_1, \ldots, \varphi_n). \tag{3.7}$$

Man beachte, dass (3.7) aus (3.6) mit Hilfe des „Substitutionslemmas der Aussagenlogik“ folgt: Ist $F(x_1, \ldots, x_n)$ eine Tautologie, so ist auch $F(\varphi_1, \ldots, \varphi_n)$ für gegebene aussagenlogische Formeln $\varphi_1, \ldots, \varphi_n$ eine Tautologie.

**Beispiel 3.4.4.** Wir stellen in diesem Beispiel einige Schlussregeln bzw. Axiome vor.

(a) Die Schlussregel $S$ des *Modus Ponens* hat die Form

$$S := \Big(\{A, A \to B\}, B\Big) \quad \text{bzw.} \quad \frac{A, A \to B}{B}.$$

Man beachte, dass die semantische Folgerung $A \wedge (A \to B) \vDash B$ gilt. Sind also $\varphi_1$ und $\varphi_2$ beliebige aussagenlogische Formeln, so können wir $\varphi_2$ aus $\varphi_1$ und $\varphi_1 \to \varphi_2$ folgern. Vergleiche auch Fußnote 15.

(b) Die Schlussregel $S$ des *Modus Tollens* kann geschrieben werden als $\frac{\neg B, A \to B}{\neg A}$. Man beachte die enge Verwandtschaft zum Modus Ponens: Es ist $(A \to B) \equiv (\neg B \to \neg A)$, d. h. wendet man den Modus Ponens auf $\neg B$ und $\neg B \to \neg A$ an, erhält man $\neg A$.

[16] Die Forderung (3.6) nennt man auch *Korrektheit* der Schlussregel. Wir verlangen in der vorliegenden Arbeit, dass jede Schlussregel korrekt ist. Einige Autoren, so zum Beispiel [Rec1975], fordern dies a priori nicht, verlangen aber später – in den Definitionen 3.4.5 eines Schlussregel-Systems und 3.4.7 eines Frege-Systems –, dass die verwendeten Regeln korrekt sind.

[17] Reden wir im folgenden von Schlussregeln, wollen wir darunter *auch* Axiome verstehen. In [Bus1999, § 2.1] werden Axiome auch Axiomschemata genannt. Siehe hierzu Fußnote 15.

[18] Wir haben diese Definition bewusst „locker“ formuliert, da in der vorliegenden Arbeit die genauen Details keine Rolle spielen werden und wir uns in diesem Kapitel, das Beispiele von Beweissystemen vorstellt, mit den groben Details begnügen. Dem an weiteren Details interessierten Leser sei die Dissertation [Rec1975, § 4.2.1] nahe gelegt.

(c) Der *Satz vom ausgeschlossenen Dritten* (lat. *tertium non datur*), $\frac{}{A \vee \neg A}$, ist ein gebräuchliches Axiom der Aussagenlogik, das auf ARISTOTELES *De Interpretatione* zurückgeht.

**Definition 3.4.5.** Ein *Schlussregel-System* (engl. *inference system*) ist ein Tupel $\mathfrak{S} := (\kappa, \mathcal{S})$ mit einer endlichen und vollständigen[19] Menge $\kappa$ von logischen Operatoren (engl. *connectives*) und einer endlichen Menge $\mathcal{S}$ an Schlussregeln in $\kappa$.

Wenn nötig, verwenden wir auch die Schreibweise $\mathfrak{S} := (\kappa, \mathcal{S}, \mathcal{A})$ für ein Schlussregel-System, um darauf hinzuweisen, dass die Schlussregelmenge $\mathcal{S}$ des Systems $\mathfrak{S}$ disjunkt zerlegt werden kann in die Menge der Axiome $\mathcal{A}$ und die der restlichen Schlussregeln $\mathcal{S} \setminus \mathcal{A}$.

Um schließlich definieren zu können, was ein Frege-System ist, wollen wir rekursiv definieren, wann sich in einem Schlussregel-System $\mathfrak{S}$ eine Formel $G$ aus einer gegebenen Menge von Formeln $\Gamma$ ableiten lässt, was wir mit $\Gamma \vdash_{\mathfrak{S}} G$ notieren wollen.

**Definition 3.4.6.** Es sei $\mathfrak{S} := (\kappa, \mathcal{S}, \mathcal{A})$ ein Schlussregel-System. Weiter sei $\Gamma$ eine Menge von Formeln und $G$ eine Formel.

(i) Alle Axiome, also alle Formeln $\phi \in \mathcal{A}$, sind aus $\Gamma$ in $\mathfrak{S}$ *ableitbar*. Wir schreiben $\Gamma \vdash_{\mathfrak{S}} \phi$. Da wir diese Eigenschaft unabhängig von der gegebenen Menge an Formeln $\Gamma$ fordern, schreiben wir auch $\vdash_{\mathfrak{S}} \phi$.

(ii) Alle Formeln $\psi \in \Gamma$ sind aus $\Gamma$ in $\mathfrak{S}$ ableitbar. Wir schreiben $\Gamma \vdash_{\mathfrak{S}} \psi$.

(iii) Gilt $\Gamma \vdash_{\mathfrak{S}} \phi_i$ für $i = 1, \ldots, \ell$ und folgt die Formel $\phi$ nach Anwendung einer Schlussregel $S \in \mathcal{S}$ aus $\phi_1, \ldots, \phi_\ell$, dann gilt auch $\Gamma \vdash_{\mathfrak{S}} \phi$.

Da wir per Definition durch (3.6) vorausgesetzt gesetzt haben, dass alle Schlussregeln *korrekt* (engl. *sound*) sind, folgt aus der syntaktischen Ableitbarkeit $\Gamma \vdash_{\mathfrak{S}} F$ leicht[20] die semantische Folgerung $\Gamma \vDash F$. Diese Eigenschaft nennt man *Korrektheit* des Schlussregel-Systems $\mathfrak{S}$.

Wir nennen ein Schlussregel-System $\mathfrak{S}$ über $\kappa$ *vollständig*, falls $\vdash_{\mathfrak{S}} F$ für jede Tautologie $F$ über den Operatoren $\kappa$ gilt. Folgt aus $\Gamma \vDash F$ zudem $\Gamma \vdash_{\mathfrak{S}} F$, heißt das System $\mathfrak{S}$

[19] Dabei ist *vollständig* im Sinne einer *vollständigen Basis* zu verstehen: Mittels den logischen Operatoren in $\kappa$ lassen sich alle aussagenlogischen Formeln darstellen.

[20] Der Beweis ist eine einfache Induktion über die Länge eines Beweises (vgl. nachfolgende Definition 3.4.8 bzw. den nachfolgenden Beweis zur Korrektheit des Resolutionskalküls 3.5.5 (a)). Im Beweis ist zu benutzen, dass Schlussregeln semantische Folgerungen bewahren: Hat man bereits die semantischen Folgerungen

$$\Gamma \vDash F_1(x_1, \ldots, x_n), \ldots, \Gamma \vDash F_m(x_1, \ldots, x_n)$$

gezeigt, dann folgt durch Anwendung der Schlussregel $S = (\{F_1, \ldots, F_m\}, G)$ auch $\Gamma \vDash G(x_1, \ldots, x_n)$. Für Beweisdetails verweisen wir auf [Sch2016, Satz 7.1].

(bzw. die darin enthaltene Schlussregel-Menge $\mathcal{S}$) *implikationsvollständig*.

Mit Einführung dieser Begriffe können wir nun ein Frege-System definieren.

**Definition 3.4.7.** Ein *Frege-System* ist ein Schlussregel-System $\mathfrak{F} = (\kappa, \mathcal{S})$ mit einer implikationsvollständigen Schlussregel Menge $\mathcal{S}$ und passendem $\kappa$.

Wir konzentrieren uns ab hier auf Frege-Systeme. Die folgende Definition 3.4.8 kann jedoch auch für Schlussregel-Systeme analog formuliert werden: Um $\Gamma \vdash_{\mathfrak{F}} \phi$ nachzuweisen, gibt man einen *Beweis* $(F_1, \ldots, F_n)$ an.

**Definition 3.4.8.** Ein *Beweis* im Frege-System $\mathfrak{F} = (\kappa, \mathcal{S}, \mathcal{A})$ für die Ableitung $\Gamma \vdash_{\mathfrak{F}} F$ ist eine (endliche) Folge $\pi = (F_1, \ldots, F_t)$ von Formeln über den Operatoren $\kappa$ mit den Eigenschaften

(i) die letzte Formel der Folge ist $F$, d. h. es gilt $F_t = F$,

(ii) für alle $i = 1, \ldots, t$ gilt:

  (a) entweder ist $F_i$ ein Axiom, d. h. es gilt $F_i \in \mathcal{A}$, oder

  (b) $F_i$ gehört zur Formelmenge $\Gamma$, oder

  (c) es gibt Indizes $1 \leq i_1 < \cdots < i_\ell < i$, sodass $F_i$ nach Anwendung einer Schlussregel $S \in \mathcal{S}$ aus $F_{i_1}, \ldots, F_{i_\ell}$ folgt.

  Die Formeln $F_i$ (für $i = 1, \ldots, t$) werden auch *Linien* des Beweises genannt.

**Beispiel 3.4.9** (Frege's propositional calculus)**.** Der allgemeine Konsens ist, dass Freges System in seiner Begriffsschrift [Fre1879] das erste Frege-System ist. Darum wollen wir dieses in diesem Beispiel vorstellen. Wir betrachten die Menge der sechs Axiome

$$\begin{aligned}
\mathcal{A} := \Big\{ & A_1 := \big(\varnothing, A \to (B \to A)\big), \\
& A_2 := \big(\varnothing, (A \to (B \to C)) \to ((A \to B) \to (A \to C))\big), \\
& A_3 := \big(\varnothing, (A \to (B \to C)) \to (B \to (A \to C))\big), \\
& A_4 := \big(\varnothing, (A \to B) \to (\neg B \to \neg A)\big), \\
& A_5 := \big(\varnothing, \neg\neg A \to A\big), \\
& A_6 := \big(\varnothing, A \to \neg\neg A\big) \Big\}
\end{aligned}$$

zusammen mit der Schlussregel des Modus Ponens aus Beispiel 3.4.4 (a),

$$S_{\mathrm{MP}} := \frac{A, A \to B}{B}$$

und setzen $\mathcal{S} := \mathcal{A} \cup \{S_{\mathrm{MP}}\}$. Schließlich wählen wir $\kappa = \{\neg, \wedge, \vee, \to\}$ und setzen $\mathfrak{F} = (\kappa, \mathcal{S})$. Exemplarisch wollen wir den Beweis $\pi = (F_1, \ldots, F_7)$ für die Ableitung

$$\{A \to B, B \to C\} \vdash_{\mathfrak{F}} A \to C \tag{3.8}$$

in Tabelle 3.1 führen.

Tabelle 3.1.: Beweis für die Ableitung (3.8) im Frege-System $\mathfrak{F}$ aus Beispiel 3.4.9. Wir haben in der dritten Spalte Begründungen angegeben.

| $i$ | **Formel $F_i$ im Beweis $\pi$** | **Begründung** |
|---|---|---|
| 1 | $B \to C$ | via Definition 3.4.8 (ii) (b) |
| 2 | $(B \to C) \to (A \to (B \to C))$ | $A_1$ mit Substitution |
| 3 | $A \to (B \to C)$ | $S_{\mathrm{MP}}$ auf $F_1$ und $F_2$ |
| 4 | $(A \to (B \to C)) \to ((A \to B) \to (A \to C))$ | $A_2$ |
| 5 | $(A \to B) \to (A \to C)$ | $S_{\mathrm{MP}}$ auf $F_3$ und $F_4$ |
| 6 | $A \to B$ | via Definition 3.4.8 (ii) (b) |
| 7 | $A \to C$ | $S_{\mathrm{MP}}$ auf $F_6$ und $F_5$ |

Es finden sich zahlreiche weitere Frege-Systeme wie in Beispiel 3.4.9 in der Literatur. Meistens sind diese jedoch leicht unterschiedlich. Der folgende Satz erklärt, warum diese Unterschiede in Wahrheit vernachlässigbar sind.

**Satz & Beispiel 3.4.10** ([CR1974a], [CR1979])**.** *Alle / je zwei Frege-Systeme sind p-äquivalent, d. h. die Länge der Binär-Codierungen von Beweisen in zwei Frege-Systemen unterscheiden sich höchstens um ein Polynom. Ebenso gilt: Sind $\mathfrak{F}_1$ und $\mathfrak{F}_2$ zwei Frege-Systeme und bezeichne* $\mathrm{Le}_{\mathfrak{F}_i}(\Gamma \vdash F)$ *die minimale* Länge *eines Beweises (definiert als die Anzahl der Linien im Beweis) für die Ableitung $\Gamma \vdash_{\mathfrak{F}_i} F$ ($i = 1, 2$), dann gibt es Polynome $p$ und $q$ mit*

$$\mathrm{Le}_{\mathfrak{F}_1}(\Gamma \vdash F) \leq p\Big(\mathrm{Le}_{\mathfrak{F}_2}(\Gamma \vdash F)\Big) \leq q\Big(\mathrm{Le}_{\mathfrak{F}_1}(\Gamma \vdash F)\Big).$$

Wie [Bus1999] und [Nor2008, S. 39] merken wir an, dass jedoch keine starken unteren Schranken in Hinblick auf Hauptproblem 3.3.10 bzw. Korollar 3.3.11 für Frege-Systeme bekannt sind (insb. ist $\mathrm{PHP}_n^m$ aus Formel 4.2.1 für alle $m > n$ „leicht“ für Frege-Systeme, d. h. die Formel besitzt Frege-Beweise, die polynomiell in ihrer Formellänge sind [Bus1987]): Die stärkste bekannte untere Schranke für Frege-Systeme ist von quadratischer Größe ([Bus1999, Theorem 13 (a)]). Superpolynomielle untere Schranken konnten bisher erst für

Untersysteme von Frege-Systemen, z. B. *Frege Systeme mit beschränkter Tiefe*[21], gezeigt werden [Ajt1994, PBI1993, KPW1995] (die Einschränkungen erfolgen dabei ähnlich wie in der Schaltkreiskomplexität, siehe z. B. [Pud2008, § 5], [Bon2018, § 1.1.1]): Diese Systeme benötigen für die Widerlegung der Formel $\mathrm{PHP}^n_{n-1}$ mindestens $\Omega((2^n)^{1/6^d})$ Symbole ([Bus1999, Theorem 17]). Dies führt zu folgendem bedeutenden Problem in der Beweiskomplexität[22], das z. B. Buss *explizit* ausformuliert in [Bus1998] erwähnt:

**Offenes Problem 3.4.11.** Sind Frege-Systeme polynomiell-beschränkt?

**Vermutung 3.4.12** ([Bon2018])**.** Zufalls $k$-CNF-Formeln (siehe Abschnitt 4.2, Beispiel 3) *könnten* schwer für Frege-Systeme sein.

Wir werden hierauf jedoch nicht weiter eingehen und verweisen abschließend auf die Survey [Bus1995] für einen Überblick an bekannten Resultaten.

### 3.4.4 Ausgewählte weitere Beispiele

Abschließend für Abschnitt 3.4 möchten wir an dieser Stelle erwähnen, dass obige Beispiele nur einen kleinen Teil der möglichen aussagenlogischen Beweissysteme darstellen (wir haben nur *Logik-basierte* Systeme vorgestellt). Einige weitere erwähnenswerte und interessante (aber für unsere Arbeit nicht direkt relevante) Beweissysteme basieren u. a. auf algebraischer oder geometrischer Beweisführung und sind:

- *Sequenzenkalkül* (oder seltener, vor allem im Englischen, *Gentzen-Kalkül* zu Ehren Gerhard Gentzens Arbeit [Gen1935]) – eine besonders lesbare Einführung (für den Spezialfall der Schnitt-freien Gentzen Systeme) findet sich in [Urq1995, § 6]. Ebenfalls empfehlenswert ist [Bus1998, § 1.2] und [BP2001, § 2.1].
- *Model Elimination Systeme* – wir verweisen auf die Originalarbeiten [Lov1969] und [Lov1978] von Loveland.
- *Nullstellensatz Beweissysteme* – für eine knappe Beschreibung verweisen wir auf [Bea2004, § 1.2.7] bzw. [BP2001, § 3.2].
- *Polynom-Kalkül* (engl. *Polynomial Calculus*, PC) – beruhend auf algebraischen Ideen und Gröbner-Basis-Algorithmen. Klauseln werden dabei als multilineare Polynome

[21] Hierbei werden Formeln über der Basis $\{\wedge, \vee, \neg\}$ betrachtet. Die *Tiefe* einer Formel ist die maximale Anzahl an Alternierungen der logischen Operatoren in ihr. Ein *Tiefe-d Frege-Beweis* ist ein Frege-Beweis, bei dem alle Formeln in der Beweisfolge eine Tiefe von höchstens $d$ haben ([Nor2008, Definition 4.11]).

[22] Das Problem wurde *implizit* bereits 1974 von Cook und Reckhow durch [CR1974a, Figure 1] aufgeworfen.

über einem (typischerweise endlichen) Körper $\mathbb{F}$ interpretiert, siehe z. B. [Bon2018, § 1.1.2]. Wir merken an, dass das Tradeoff-Resultat 7.0.1, das in [BNT2013, Bec2017] gezeigt wurde, auch im Beweissystem Polynomial Calculus Resolution (`PCR`) gilt.

- *Schnittebenenverfahren* (engl. *Cutting Planes*, `CP`) – basierend auf den Ideen von Chvátal und Gomory aus [Chv1973] und [Gom1958], die eine Brücke zwischen ganzzahliger linearer Programmierung und linearer Programmierung durch LP-Relaxation schlugen. Eine detaillierte Darstellung dieser Querverbindung findet sich z. B. in [Bru2015, § 10 Cutting Planes] oder in vielen Standard-Büchern zur Optimierung / Operations Research. Für eine heuristische Beschreibung des Verfahrens empfehlen wir [BP2001, § 3.3], eine ausführliche Darstellung findet sich bspw. in [KL1994, § 6.2] und [Bus1999, Part III], eine graphische Illustration in [Juk2012, Fig. 19.1].

Für den Zusammenhang zwischen vielen dieser Systeme verweisen wir auf [Rec1975, Abb. 1.5.i]. In [Nor2013] finden sich Beispiele für die „Codierung“ von Klauseln in diesen Systemen und entsprechende Komplexitätsmaße, wie wir sie für Resolution in Abschnitt 4.1 vorstellen werden.

## 3.5 Das Beweissystem der Resolution

Wir stellen in diesem Abschnitt eines der wichtigsten aussagenlogischen Beweissysteme vor: Das der Resolution. Das große Forschungsinteresse an der Resolution ist vor allem darauf zurückzuführen, dass nur eine Schlussregel benutzt wird. Speziell die Verwendung der Resolution im Zusammenhang mit sat-Algorithmen ab den 1960er-Jahren machte dieses Beweissystem interessant für die Praxis. Bis heute stellt Resolution die Basis vieler modernster sat-Solver dar.

Im Laufe der Zeit wurden zahlreiche Weiterentwicklungen und Strategien der Resolution untersucht, wie bspw. die *Unit Preference Strategy* [WCR1964], `DPLL`-*artige Algorithmen* wie in [DLL1962], *P-Resolution*, *N-Resolution*, *lineare Resolution* [Lov1970] u. v. m. (eine sehr gute Übersicht bietet [ST2013, § 2.4]). Das ursprüngliche Resolutionsprinzip geht jedoch auf Robinson [Rob1965] zurück (der die Widerlegungsvollständigkeit und Korrektheit der Resolution zeigte) und basiert wiederum auf früheren Arbeiten u. a. von Prawitz [Pra1960]. Als der Resolution vorausgegangene Methode sei an dieser Stelle der `Davis-Putnam`-Algorithmus aus [DP1960] erwähnt. Wie [KL1994] wollen wir nicht unerwähnt lassen, dass sich ansatzweise Ideen bereits in den sehr frühen Arbeiten [Löw1908, Löw1910, Löw1913], [Qui1955], sowie Blakes [Bla1937] (der ebenfalls häufig als Urheber der Resolution attribuiert wird) finden. Für tiefergehende historische Bemerkungen verweisen wir auf [GN2013, § 4.12].

### 3.5.1 Definition und Eigenschaften des Resolutionskalküls

Man beachte beim Lesen des Abschnittes 3.5, dass Resolution ein Widerlegungsverfahren ist: Statt also die Allgemeingültigkeit einer Formel zu zeigen, folgert der Resolutionskalkül einen Widerspruch aus der negierten ursprünglichen Formel (wir verweisen für eine Diskussion hierzu auf Seite 51).

Unsere Darstellung im Folgenden orientiert sich an [KL1994, Sch2013, ST2013]. Wir geben in diesem Abschnitt maximal Beweisskizzen der Resultate, da diese allgemein bekannt sind.

**Definition 3.5.1.** Es seien $C_1$ und $C_2$ Klauseln einer aussagenlogischen Formel in konjunktiver Normalform.

(i) $C_1$ und $C_2$ heißen *resolvierbar*, falls es ein Literal $u$ mit $u \in C_1$ und $\overline{u} \in C_2$ gibt.

(ii) In diesem Fall kann eine Klausel $C_3$ definiert werden, die *Resolvente* (auch: der *Resolvent*) von $C_1$ und $C_2$, die durch

$$C_3 := (C_1 \setminus \{u\}) \cup (C_2 \setminus \{\overline{u}\})$$

definiert ist. Die Klauseln $C_1$ und $C_2$ heißen *Elternklauseln* von $C_3$.

(iii) Die Schlussregel

$$\frac{C_1 \quad , \quad C_2}{(C_1 \setminus \{u\}) \cup (C_2 \setminus \{\overline{u}\})} \tag{3.9}$$

bezeichnen wir als *Resolutionsregel*.

(iv) Den *leeren Disjunktionsterm* $\varnothing$ (die Klausel mit keinen Literalen), bzw. die *leere Klausel*, die Resolvente von $C_1 = \{u\}$ und $C_2 = \{\overline{u}\}$, bezeichnen wir mit $\square$. Die leere Klausel wird unter jeder Belegung als falsch interpretiert, ist also unerfüllbar.

Wir bezeichnen (3.9) auch als *Resolution nach u* bzw. sprechen davon, dass *nach u (in einem Schritt) resolviert* wurde und schreiben dafür symbolisch $\{C_1, C_2\} \vdash^1_{\mathfrak{R}} C_3$. Graphisch hat sich die Darstellung als Baum wie in Abbildung 3.1 etabliert (ein größeres Beispiel findet sich in Abbildung 3.2 bei dem allerdings kein Baum, sondern ein gerichteter azyklischer Graph vorliegt).

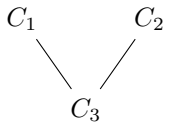



Abbildung 3.1.: Graphische Repräsentation von $\{C_1, C_2\} \vdash^1_{\mathfrak{R}} C_3$ als Baum.

Man beachte insbesondere: Falls kein solches Literal $u$ wie in Definition 3.5.1 (i) existiert mit $u \in C_1$ und $\overline{u} \in C_2$, so sind $C_1$ und $C_2$ nicht resolvierbar, es gibt insbesondere keine Resolvente der beiden Klauseln. Des Weiteren darf immer nur genau ein Literal pro Schritt resolviert werden. Falls die zwei zu resolvierenden Klauseln mehr als ein Paar komplementärer Literale enthalten, kann die Resolutionsregel zwar unabhängig für jedes solche Paar angewandt werden, die Resolvente ist jedoch immer eine Tautologie. Dennoch ist je nach vorliegenden Klauseln die Bildung von verschiedenen Resolventen möglich.

Obige Definition rechtfertigt auch vom *Resolutionskalkül* zu reden, da die Schlussregel (3.9) eine Vorschrift zur Transformation von Klauselmengen darstellt.

**Bemerkung 3.5.2.** Der Modus Ponens aus Beispiel 3.4.4 (a), $\frac{A, A \to B}{B}$, ist äquivalent zu $\frac{A, \neg A \vee B}{B}$ und kann daher als Spezialfall der Resolution betrachtet werden.

Ähnlich wie in Definition 3.4.6 wollen wir im Folgenden erklären, wie wir eine Resolutionsableitung $F \vdash_{\mathfrak{R}} C$ verstehen wollen.

**Definition 3.5.3.** Es sei $F$ eine Klauselmenge und $C$ eine Klausel. Wir definieren den Begriff einer *Resolutionsableitung* (engl. *resolution derivation*) $F \vdash_{\mathfrak{R}} C$ rekursiv durch:

(i) *Basisregel:* Für jede Klausel $C \in F$ gilt $F \vdash_{\mathfrak{R}} C$.

(ii) *Rekursive Regel:* Gilt $F \vdash_{\mathfrak{R}} C_1 \cup \{u\}$ sowie $F \vdash_{\mathfrak{R}} C_2 \cup \{\overline{u}\}$, dann gilt auch $F \vdash_{\mathfrak{R}} C_1 \cup C_2$.

Entsprechend definieren wir eine *Resolutionswiderlegung* (engl. *resolution refutation*) als Ableitung der leeren Klausel aus einer gegebenen Klauselmenge. Existiert eine solche Resolutionswiderlegung für $F$, notieren wir dies mit $F \vdash_{\mathfrak{R}} \square$ und sprechen davon, dass es *möglich ist, die leere Klausel aus $F$ abzuleiten.*

Analog zu Definition 3.4.8 klären wir, was wir unter einem Resolutionsbeweis verstehen wollen.

**Definition 3.5.4.** Es sei $F$ eine Klauselmenge und $C$ eine Klausel. Ein *Resolutionsbeweis* für die Ableitung $F \vdash_{\mathfrak{R}} C$ ist eine Folge $(C_1, \ldots, C_t)$ von Klauseln mit den Eigenschaften

(i) die letzte Klausel der Folge ist $C$, d. h. es gilt $C_t = C$,

(ii) für alle $i = 1, \ldots, t$ gilt:

  (a) entweder ist $C_i$ eine der Klauseln der Klauselmenge $F$ (ein *Axiom*),

  (b) oder $C_i$ ist ein Resolvent zweier Klauseln $C_j$ und $C_\ell$, die zuvor in der Folge aufgeführt wurden, d. h. es gibt Indizes $1 \leq j < \ell < i$, sodass $C_i$ aus $C_j$ und $C_\ell$ nach Anwendung der Resolutionsregel (3.9) folgt, in Zeichen $\{C_j, C_\ell\} \vdash^{1}_{\mathfrak{R}} C_i$.

Die Klauseln $C_i$ (für $i = 1, \ldots, t$) werden auch *Linien* des Beweises genannt.

Wie [Nor2008] begehen wir im Folgenden den bewussten Notationsmissbrauch und bezeichnen sowohl eine Resolutionsableitung $\pi\colon F \vdash_{\mathfrak{R}} C$, als auch ihren zugehörigen Beweis $\pi = (C_1, \ldots, C_t)$ mit dem gleichen Symbol $\pi$, um die Zugehörigkeit von Beweis zu Ableitung zu unterstreichen. Insbesondere in Kapitel 4 werden wir die Begriffe Ableitung und Beweis teilweise synonym verwenden.

An dieser Stelle ist zu erwähnen, dass die Resolvente nicht äquivalent zu den Ursprungsklauseln ist; vielmehr sind die Ausgangsklauseln nur dann gleichzeitig erfüllbar, wenn die Resolvente erfüllbar ist. Die Erfüllbarkeit der Resolvente ist also eine notwendige Bedingung für die gleichzeitige Erfüllbarkeit der Ursprungsklauseln – hierin liegt der Nutzen der Resolution: Gelingt es die leere Klausel aus einer Formel abzuleiten, können wir schließen, dass die Formel unerfüllbar ist. Mehr noch: Ist die Ausgangsformel unerfüllbar, so gelingt uns das Ableiten der leeren Klausel auch. Wir halten dies formal im folgenden Theorem fest.

**Theorem 3.5.5** (Eigenschaften des Resolutionskalküls)**.**

*(a) Der Resolutionskalkül ist* korrekt*, d.h existiert eine (syntaktische) Resolutionsableitung $\pi\colon F \vdash_{\mathfrak{R}} C$, dann gilt auch die semantische Ableitung $F \vDash C$. M. a. W. Der Resolvent zweier Klauseln ist eine logische Konsequenz ihrer Konjunktion.*

*Insbesondere ergibt sich: Gilt $F \vdash_{\mathfrak{R}} \square$, so ist $F \in$ UNSAT.*

*(b) Der Resolutionskalkül ist* nicht vollständig*, d. h. es gibt Formeln $F \in$ CNF und Klauseln $C$, sodass $F \vDash C$, aber nicht $F \vdash_{\mathfrak{R}} C$ gilt.*

*(c) Der Resolutionskalkül ist* widerlegungsvollständig*, d. h. gilt $F \notin$ SAT (m. a. W. $F$ ist widersprüchlich), kann immer $F \vdash_{\mathfrak{R}} \square$ gefolgert werden.*

Beweise dieser Aussagen findet man in den meisten Standardlehrbüchern über Logik. Wir begnügen uns daher mit einer groben Beweisskizze.

*Beweisskizze.* Wir orientieren uns an [KL1994, Satz 4.1.4].

(a) Es würde genügen nachzuweisen, dass die Resolutionsregel (3.9) korrekt ist, d. h. dass aus $\{C_1, C_2\} \vdash^{1}_{\mathfrak{R}} (C_1 \setminus \{u\}) \cup (C_2 \setminus \{\overline{u}\})$ auch $C_1 \wedge C_2 \vDash (C_1 \setminus \{u\}) \cup (C_2 \setminus \{\overline{u}\})$ folgt. Der Rest der Aussage folgt dann sofort durch vollständige Induktion über die Anzahl der Resolutionsschritte in $F \vdash_{\mathfrak{R}} C$. Siehe [KL1994, Satz 4.1.4].

Um die „Insbesondere“-Aussage zu zeigen, nehmen wir an, um einen Widerspruch zu erhalten, dass $F \vdash_{\mathfrak{R}} \square$ aber auch $F \in$ SAT gilt. D. h. es gibt eine Belegung $\alpha$, die $F$ erfüllt. Wegen des ersten Teils der Aussage erfüllt $\alpha$ aber auch die leere Klausel $\square$, was absurd ist!

(b) Wir wählen zum Beispiel $F = (x_1)$ und $C = (x_1 \vee x_2)$. Dann gilt $F \vDash C$, aber $C$ lässt sich nicht aus $F$ mit Resolution herleiten, in Zeichen $F \nvdash_{\mathfrak{R}} C$.
(c) Siehe z. B. [KL1994, Satz 4.1.4], [Ben2009a, Lemma 2.13] oder [Sch2013, S. 12]. □

Wir merken schließlich an, dass insbesondere $F$ genau dann unerfüllbar ist, wenn es eine Resolutionswiderlegung von $F$ gibt. Im Sinne von aussagenlogischen Beweissystemen halten wir fest:

$$F \in \text{TAUT} \iff \neg F \vdash_{\mathfrak{R}} \square.$$

Man beachte, dass $\neg F$ hierzu in CNF vorliegen muss. Falls $F$ in DNF vorliegt, erhält man direkt durch die De Morganschen Regeln eine CNF-Darstellung von $\neg F$. Andernfalls ist es immer möglich, diese Umformung „manuell“ durchzuführen, was die Formel jedoch exponentiell vergrößern kann (siehe Theorem C.0.2), sodass es sich generell anbietet, die Formel $\neg F$ durch die Tseitin-Transformation C.0.3 in eine sat-äquivalente Klauselmenge $\Gamma_{\neg F}$ umzuformen, die linear in der Formellänge von $\neg F$ ist.

Wir halten dies im nachfolgenden Korollar, angelehnt an [Nor2001, S. 19] und [Nor2016a, S. 22], fest.[23]

**Korollar 3.5.6.** *$S_{\mathfrak{R}} := \{(F, \pi) : \pi$ ist eine Resolutionswiderlegung von $F\}$ ist ein Beweissystem für* UNSAT *nach Definition 3.3.1. Vermöge Proposition A.0.1 sowie obiger Argumentation erhalten wir hieraus auch ein Beweissystem für* TAUT*.*

*Dank der Tseitin-Transformation C.0.3 kann jedes korrekte und vollständige Beweissystem, das Widerlegungen für Formeln in* CNF *produziert, als aussagenlogisches Beweissystem benutzt werden.*

**Bemerkung 3.5.7.** Vermöge $\mathfrak{R} := (\{\wedge, \vee, \neg\}, \mathcal{S}_{\mathfrak{R}}, \mathcal{A}_{\mathfrak{R}})$ mit der leeren Axiommenge $\mathcal{A}_{\mathfrak{R}} = \varnothing$ und der Resolutionsregel (3.9) als einzige Schlussregel in $\mathcal{S}_{\mathfrak{R}}$ wäre es möglich, den Resolutionskalkül als Schlussregel-System gemäß 3.4.5 aufzufassen – der allgemeine Konsens in der Literatur ist jedoch, dies *nicht* zu tun, da Resolution nur auf Klauseln operiert, nicht jedoch auf allgemeinen Formeln über $\{\wedge, \vee, \neg\}$ (siehe [Bon2018, S. 4] und [Nor2008, S. 40] für einen „Ausweg“).

Ungeachtet dieses technischen Details wäre $\mathfrak{R}$, wie oben definiert, kein Frege-System: Die Resolutionsregel (3.9) ist nicht implikationsvollständig (vgl. Theorem 3.5.5 (b)), was eine wichtige Forderung in Definition 3.4.7 war.

[23] Wegen der im Korollar angesprochenen Dualität nennt NORDSTRÖM die Terminologie der Beweiskomplexität „slightly schizophrenic“: Unerfüllbare Formeln werden in der Literatur teilweise als Tautologien bezeichnet (so z. B. die in Kapitel 5 besprochenen Tseitin-Formeln). Von diesem Gebrauch nehmen wir Abstand, verwenden aber die Begriffe *Beweis* und *Widerlegung* teilweise synonym.

**Bemerkung 3.5.8.** In Hinblick auf Bemerkung 3.5.7 beachte man außerdem folgendes theoretische Resultat [Tse1983, Urq1987]: Es gibt eine Folge $(F_n)_{n\in\mathbb{N}}$ von tautologischen Formeln, die $\mathcal{O}(n)$ Klauseln konstanter Weite besitzen, sodass jede Formel $\neg F_n$ eine polynomiell-große $n^{\mathcal{O}(1)}$ Frege-Widerlegung besitzt, aber jede Resolutionswiderlegung Länge $2^{\Omega(n)}$ hat. Für eine moderne Beweisadaption verweisen wir auf [CK2002, Theorem 5.4.9].

Ebenso erinnern wir an das auf Seite 45f. angesprochene Resultat von [Bus1987]. Zusammen mit Theorem 4.2.2 folgt, dass Resolution Frege-Systeme nicht $p$-simulieren kann.

Wir schwächen das im zweiten Absatz der Bemerkung erwähnte „Dilemma“ durch Einführung der *weakening rule* (auch *subsumption rule*),

$$\frac{C}{C \vee D}$$

in Lees Theorem[24] 3.5.9, das auch als Subsumptions-Theorem bekannt ist, ab.

**Theorem 3.5.9** (Lees Theorem)**.** *Es sei $F \in$ CNF eine Formel und $C$ eine Klausel. Es gilt $F \vDash C$ genau dann, wenn $C$ entweder eine tautologische Klausel ist, oder $F \vdash_{\mathfrak{R}} C'$ für eine Klausel $C' \subset C$ gilt (was wir auch mit $C'$* subsummiert *$C$ ausdrücken).*

Man beachte, dass dies eine Verschärfung der Widerlegungsvollständigkeit 3.5.5 (c) ist: Ist $F$ unerfüllbar, so impliziert $F$ jede Klausel. Die einzige Klausel, die jedoch jede Klausel subsummiert, ist die leere Klausel, welche also gemäß dem obigen Theorem mittels Resolution aus $F$ hergeleitet werden kann.

*Beweis von Theorem 3.5.9.* Die erste Formulierung und der erste Beweis dieses Fakts findet sich in Char-Tung Lees Dissertation [Lee1967]. Aufgrund der heute von der Arbeit abweichenden Terminologie, sowie Ungenauigkeiten im Beweis verweisen wir auf einen einfachen Beweis in [KL1994, Lemma 4.1.5]. □

**Korollar 3.5.10.** *Es sei $F \in$ CNF eine Formel und $C$ eine Klausel, sodass $F \vDash C$ gilt. Dann gibt es eine Resolutionsableitung von $F \vdash_{\mathfrak{R}} C$ (die möglicherweise die weakening rule benutzt).*

[24] Viele Autoren nennen dies die *Implikationsvollständigkeit der Resolution*. Wir nehmen jedoch von diesem Begriff Abstand aufgrund der in Bemerkung 3.5.7 dargestellten Tatsache, dass die Resolutionsregel (3.9) nicht vollständig ist. Die Verwechslungsgefahr dieser Begriffe erscheint uns zu hoch.

Wir merken ebenfalls an, dass dieses Resultat in vielen Standardbücher nicht erwähnt wird. Dies gilt selbst für Lees eigenes Buch [CL2014]. Mögliche Gründe (u. a. Ungenauigkeiten im Originalbeweis [Lee1967] des Resultats) werden in [Fer2000] genannt.

Zudem ist es nicht schwer, das folgende Theorem zu zeigen, welches zeigt, dass das Hinzufügen der weakening rule das Beweissystem der Resolution nicht stärker macht.

**Theorem 3.5.11** (Siehe z. B. [Bon2018])**.** *Die beiden Beweissysteme* `Resolution` *und* `Resolution mit weakening rule` *sind p-äquivalent. Für jede Resolutionswiderlegung $\pi$ von $F$ mit weakening rule gibt es eine Widerlegung $\pi'$ von $F$ ohne weakening (eine* weakening-freie Widerlegung*), sodass die Anzahl der Resolventen und Axiome in $\pi$ gleich derer in $\pi'$ ist. Wenn $\pi$ baumartig (resp. regulär)*[25] *ist, dann ist es möglich auch $\pi'$ baumartig (resp. regulär) zu konstruieren und die eben genannte Bedingung zu erfüllen.*

Im Sinne der Komplexitätsmaße, die wir in Abschnitt 4.1 einführen werden, bedeutet dies (bzw. kann zusätzlich gezeigt werden), dass weakening Schritte aus einer Resolutionswiderlegung problemlos eliminiert werden, ohne die Länge, Weite oder den Platz der Widerlegung zu vergrößern (siehe [Nor2001, Proposition 2.30], [Nor2008, S. 44]).

In dieser Hinsicht wird deutlich, dass weakening Schritte lediglich ein Werkzeug darstellen, um Beweise einfacher zu formulieren – wir werden sie im Beweis von Proposition 7.1.11 verwenden.

*Beweis von Theorem 3.5.11.* Einfache Beweise dieser bekannten Tatsache findet man z. B. in [Rac2007, Theorem 1.10], [BHJ2008, Proposition 2 & 3] oder (in allgemeinerer Form aus der man aber leicht den Beweis für obiges Theorem erhält) in [Sze2005, Lemma 1 & 9]. □

### 3.5.2 Repräsentation eines Resolutionsbeweises

Bisher haben wir Beweise $\pi$ gemäß Definition 3.5.4 als Folge von Klauseln betrachtet (man spricht auch davon, dass Resolution ein *sequentielles Beweissystem* ist[26]). Wir wollen diese Art der Beweise (zur besseren Unterscheidung) auch *folgenartig* nennen. Wie bereits in Abbildung 3.1 angedeutet, bietet es sich an, einen Resolutionsbeweis graphisch zu repräsentieren. Wir erörtern in den folgenden Definitionen, angelehnt an [Bon2018, Definition 2.1], wie diese *graphenartigen*[27] Beweise definiert sind.

[25] Wir führen diese Begriffe im auf dieser Seite beginnenden Abschnitt 3.5.2 ein. Wir haben uns dennoch dafür entschieden das Theorem direkt in seiner Vollständigkeit zu formulieren.

[26] Frege-Systeme sind auch sequentiell. Siehe [Ben2009a, Definition 1.3] für eine formale Definition.

[27] Es ist anzumerken, dass wir i. A. nicht mehr wie in Abbildung 3.1 einen Baum erhalten: Klauseln, die als Elternklauseln in mehrere Resolutionsschritte eingehen, wollen wir aus Gründen der Übersichtlichkeit nicht zwingend mehrfach im Baum aufnehmen. Wir erhalten also keinen Baum mehr, sondern einen gerichteten azyklischen Graphen (dag). Um sowohl die baumartige Struktur, als auch die dag-artige Struktur zu erfassen, haben wir uns für das Wort *graphenartig* entschieden.

**Definition 3.5.12.** Es sei $\pi = (C_1, \ldots, C_t)$ ein Resolutionsbeweis für die Resolutionsableitung $F \vdash_{\mathfrak{R}} A$. Eine[28] Funktion $\sigma\colon [t] \to \binom{[t]}{2} \cup \{\star\}$ ist eine *Zeugenfunktion* für den Fakt, dass $\pi$ eine Resolutionsableitung von $A$ aus $F$ ist, falls für alle $i \in [t]$ die folgenden zwei Bedingungen gelten:

(i) Falls $\sigma(i) = \{j, \ell\} \in \binom{[t]}{2}$ ist, dann gilt $j < i$, $\ell < i$ und $\frac{C_j, C_\ell}{C_i}$ ist eine korrekte Anwendung der Resolutionsregel (3.9).

(ii) Falls $\sigma(i) = \star$ ist, dann ist $C_i \in F$ ein Axiom.

**Definition 3.5.13.** Es sei $\pi = (C_1, \ldots, C_t)$ ein Resolutionsbeweis für die Resolutionsableitung $F \vdash_{\mathfrak{R}} A$, sowie $\sigma$ eine Zeugenfunktion für diesen Fakt. Der *beschriftete* (engl. *labeled*), gerichtete azyklische Graph (engl. *directed acyclic graph*, kurz *dag*) $G_\pi^{(\sigma)}$ mit Knotenmenge $V(G_\pi^{(\sigma)}) = \{v_1, \ldots, v_t\}$, Kantenmenge $E(G_\pi^{(\sigma)}) = \{(v_k, v_i) \,:\, k, i \in [t] \text{ und } k \in \sigma(i)\}$ und Beschriftung der Knoten durch $\mathrm{label}(v_i) = C_i$ für alle $i \in [t]$ heißt *Beweisgraph von $\pi$ bzgl. $\sigma$*.

Man beachte, dass es Knoten geben kann, die durch dasselbe Label beschriftet werden. Dies, zusammen mit Fußnote 28, erklärt die notwendigen Formalitäten der obigen zwei Definitionen.

Es ist üblich die Richtungen der Kanten in einer graphischen Repräsentation eines Beweisgraphen nicht einzuzeichnen, sondern Vorgängerknoten in der Graphik „höher“ zu stellen als Nachfolger (vgl. dies mit Darstellung von Hasse-Diagrammen in Abbildung D.1). Ebenso verwendet man direkt das Label der Knoten.

**Beispiel 3.5.14.** Wir betrachten die unerfüllbare Formel

$$F = \Big\{\{\neg x, \neg y\}, \{y, \neg x, \neg z, \neg w\}, \{x, \neg w\}, \{z, \neg w\}, \{w\}\Big\}$$

zusammen mit dem Beweis für die Unerfüllbarkeit

$$\pi = \Big(\{\neg x, \neg y\}, \{y, \neg x, \neg z, \neg w\}, \{x, \neg w\}, \{z, \neg w\}, \{w\}, \\ \{y, \neg x, \neg z\}, \{x\}, \{z\}, \{y, \neg z\}, \{y\}, \{\neg x\}, \square\Big),$$

für den ein möglicher Beweisgraph in Abbildung 3.2 gegeben ist.

[28] Zu einem gegebenen folgenartigen Beweis kann es i. A. mehrere Zeugenfunktionen geben: Als Beispiel sei die unerfüllbare Formel $F = \{\{x, z\}, \{x, \overline{z}\}, \{x, y\}, \{\overline{y}\}, \{\overline{x}\}\}$ sowie der folgenartige Resolutionsbeweis $\pi = (\{x, z\}, \{x, \overline{z}\}, \{x, y\}, \{\overline{y}\}, \{\overline{x}\}, \{x\}, \square)$ gegeben. Es ist nicht eindeutig, ob die sechste Klausel, $\{x\}$, des Beweises durch Resolution der Klauseln $\{x, z\}$ und $\{x, \overline{z}\}$ entstand, oder durch Resolution der Klauseln $\{x, y\}$ und $\{\overline{y}\}$. Folglich hat $\pi$ mindestens zwei verschiedene Zeugenfunktionen.

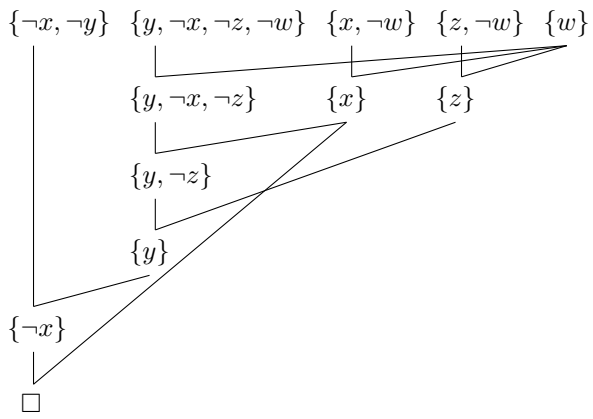



Abbildung 3.2.: Ein Beweisgraph $G_\pi$ zur Resolutionswiderlegung $\pi$ der Formel $F$ aus Beispiel 3.5.14. Hier ist $G_\pi$ in der Tat ein gerichteter azyklischer Graph und kein Baum. Um einen Baum zu erhalten, müssen wir die Klauseln $\{x\}$ bzw. $\{w\}$ erneut ableiten bzw. mehrfach als Axiom zur Verfügung stellen – was sich allerdings u. a. auf die Größe des Beweises auswirkt.

Viele Modifikationen der Resolution, wie z. B. die von Tseitin [Tse1968] eingeführte baumartige Resolution, die wir im folgenden definieren, hängen – wie in Abbildung 3.2 angedeutet – von der Wahl des zugrundeliegenden Graphen (bzw. von der Zeugenfunktion $\sigma$) ab und können nicht direkt für folgenartige Beweise ohne gegebene Graphenstruktur definiert werden. Beispielsweise ist es vielmehr NP-vollständig zu entscheiden, ob ein folgenartiger Resolutionsbeweis einen *regulären* graphenartigen Beweis hat (d. h. eine Zeugenfunktion zu finden, sodass der korrespondierende Beweisgraph keinen gerichteten Weg besitzt, bei dem nach derselben Variable mehr als einmal im Verlauf des Pfades resolviert wurde). Vergleiche hierzu [BH2008].

**Definition 3.5.15.** Eine Resolutionsableitung $\pi$ ist *baumartig* (engl. *tree-like*), falls eine Zeugenfunktion $\sigma$ existiert, sodass der Beweisgraph $G_\pi^{(\sigma)}$ ein Baum ist (d. h. jede abgeleitete Klausel kann höchstens in einem Schritt als Elternklausel benutzt werden).

Um eine Resolutionsableitung in eine baumartige Resolutionsableitung umzuwandeln, kann es notwendig sein, „zeit-gestempelte“ Klauseln, wie [Nor2008, S. 43] es nennt, anzulegen. Vergleiche hierzu die Bildunterschrift zu Abbildung 3.2.

**Theorem 3.5.16** (Korrektheit und Widerlegungsvollständigkeit von baumartiger Resolution)**.** *Baumartige Resolution ist korrekt und widerlegungsvollständig.*

*Beweis.* Siehe z. B. den Beweis für die Widerlegungsvollständigkeit der gewöhnlichen Resolution (Theorem 3.5.5) in [ST2013, S. 42]. Der dort konstruierte Resolutionsbeweis ist baumartig. □

Es sei angemerkt, dass das Beweissystem der Resolution mit folgenartigen Beweisen und jenes mit graphenartigen Beweisen $p$-äquivalent sind: Ein folgenartiger Beweis kann offensichtlich durch einen Polynomialzeit-Algorithmus in einen graphenartigen Beweis umgewandelt werden und umgekehrt (eine korrespondierende Folge kann z. B. durch eine breadth-first Traversierung gefunden werden, wie in [Pit2002] erörtert). Wir weisen jedoch auf Fußnote 28 hin: Zu einem gegebenen folgenartigen Beweis kann es i. A. mehrere graphenartige Beweise geben.

**Bemerkung & Beispiel 3.5.17.** Gibt es eine baumartige Resolutionswiderlegung für eine unerfüllbare Formel $F$ mit $t$ Klauseln im Beweis, so gibt es auch eine reguläre Resolutionswiderlegung für $F$ mit höchstens $t$ Klauseln im Beweis. Es gilt sogar: Reguläre Resolution $p$-simuliert baumartige Resolution, in Zeichen

$$\texttt{baumartige Resolution} \preccurlyeq_p \texttt{reguläre Resolution}.$$

*Beweis.* Ein Beweis findet sich z. B. in [Rac2007, Theorem 1.9], wo URQUHARTs Artikel [Urq1995, Lemma 5.1] adaptiert wurde. □

**Notiz 3.5.18** (Adaptiert aus [Juk2012, Nor2013, Bon2018])**.** In [AJPU2007] konnte gezeigt werden, dass eine Familie $(F_n)_{n\in\mathbb{N}}$ von unerfüllbaren Formeln existiert, deren Widerlegung mit regulärer Resolution exponentielle Länge benötigt, die aber (*gewöhnliche* bzw. *uneingeschränkte*) Resolutionswiderlegungen von polynomieller Länge besitzt. Präziser: Bezeichnet $R$ die kürzeste Länge aller regulärer Resolutionswiderlegungen von $F_n$ und $G$ die kürzeste Länge aller (gewöhnlicher) Resolutionswiderlegungen von $F_n$, so gilt $R = 2^{\Omega(\sqrt[3]{G})}$. URQUHART [Urq2011] erhielt später durch Nutzung einer anderen Formelfamilie eine größere *Separation*: $\log_2 R = \Omega\left(\frac{G}{\text{polylog}(G)}\right)$.

Eine ähnliche Separation erhält man zwischen baumartiger und regulärer Resolution. Wir verweisen hierfür auf [Rac2007, § 3.1.1, § 3.5], basierend auf [BIW2004].

Wir haben also

$$\texttt{baumartige Resolution} \prec_p \texttt{reguläre Resolution} \prec_p \texttt{gewöhnliche Resolution},$$

wobei die jeweils links von $\prec_p$ stehenden Beweissysteme *in diesem Fall* exponentiell schwächer sind, als die jeweils rechts stehenden. Insbesondere gilt also z. B.: Baumartige Resolution kann gewöhnliche Resolution nicht $p$-simulieren. Für weitere Resultate zur $p$-Simulation verschiedener Resolutionseinschränkungen verweisen wir auf [BP2007], sowie [Rac2007, § 3].

# KAPITEL 4

# Komplexitätsmaße und Tradeoffs

Wir konzentrieren uns ab hier ausschließlich auf das Beweissystem der Resolution. Um die Komplexität von Resolutionsbeweisen zu studieren, wurden in der Vergangenheit zahlreiche *Komplexitätsmaße* (auch: *hardness measures* [Bon2018]) eingeführt. Wir stellen in diesem Kapitel zwei der wichtigsten hiervon vor, illustrieren beispielhaft untere und obere Schranken, deren Bedeutung wir in einem Anwendungskapitel über SAT-Solving verdeutlichen, um schließlich über Tradeoffs zu sprechen.

## Inhalt des Kapitels



## 4.1 Zwei Komplexitätsmaße für Resolution: Länge und Platz

Im Folgenden werden wir die zwei wichtigen Komplexitätsmaße Länge und Platz für das Beweissystem der Resolution einführen. Wir studieren ebenfalls das Hilfsmaß Weite.

**Definitionsversuch 4.1.1** (Konventionelle Definition – Erster Versuch)**.** Die *Länge* (engl. *length*) $\mathrm{Le}(\pi)$ einer Resolutionswiderlegung $\pi = (C_1, \ldots, C_t)$ ist die Anzahl $t$ der Klauseln im Beweis $\pi$.

**Bemerkung 4.1.1.** Neben der oben definierten Länge (engl. *length*) eines Beweises definiert man häufig auch seine *Größe* (engl. *size*) als die Gesamtanzahl der Literale im Beweis. Gemäß [Nor2016a, S. 30] wird in der Praxis der lineare Faktor zwischen den beiden Definitionen ignoriert und die beiden Begriffe synonym verwendet.

**Beispiel 4.1.2.** Die Resolutionswiderlegung $\pi$ in Beispiel 3.5.14 hat Länge 12 und Weite 4 gemäß dem Definitionsversuch 4.1.1 und der Definition von Weite in den Präliminarien.

Im Folgenden wollen wir das Komplexitätsmaß *Platz* (genauer: des *Klauselplatzes*) einführen, das berücksichtigt, dass nicht notwendigerweise alle gebildeten Resolventen in einer Resolutionswiderlegung dauerhaft gespeichert werden müssen, da sie im weiteren Verlauf nicht mehr als Elternklauseln auftreten. Die Untersuchung des Platzes für aussagenlogische Beweise wurde zum ersten Mal[29] durch Armin Haken während des Workshops „Complexity Lower Bounds“ im Fields Institute in Toronto im Jahre 1998 angesprochen ([Bon2018]). Ursprünglich wurde dieses Komplexitätsmaß für Resolution von Kleine Büning und Lettman in [KL1994, § 4.3.13] eingeführt und aufbauend durch Esteban und Torán in [ET2001] in die heute (für sublinearen Platz) verwendete Form gebracht. Später wurde es in [ABRW2002] für andere Beweissysteme generalisiert und ist informell gesprochen die maximale Anzahl an Klauseln, die im Verlauf der Widerlegung *gleichzeitig* gespeichert werden müssen[30].

**Intuition 4.1.3** (Das Tafelmodell)**.** Neben der informellen Erklärung von oben findet sich in der Literatur in zahlreichen Quellen (u. a. [BN2011, Nor2013, Bon2018]) das Tafelmodell (engl. *blackboard modell*). Man stellt sich dabei vor, dass der Beweis durch einen Dozenten (dem *Prover*) vor einer Klasse von Studenten (dem *Verifier*) vorgetragen wird.[31] Der Dozent versucht dabei, um die Klasse von der Unerfüllbarkeit einer gegebenen CNF-Formel zu überzeugen, nacheinander Klauseln an der Tafel zu notieren und Resolventen aus den sich an der Tafel befindlichen Klauseln zu bilden. Dabei darf er Klauseln wegwischen. Bei vorgegebener Schriftgröße stellt sich die Frage, wie groß seine Tafel sein muss. In [BN2011] wird erläutert:

> Roughly speaking, the space of a proof corresponds to the minimal size of a blackboard needed to give a self-contained presentation of the proof, where the correctness of each step is verifiable from what is currently on the blackboard.

Die im Folgenden eingeführten Klauselkonfigurationen $\mathfrak{M}_i$ entsprechen dabei Momentaufnahmen der Tafel zum Zeitpunkt $i$.

[29] Bonacina merkt in [Bon2018, S. 10] an, dass sich zuvor nur ein Paper, [Koz1977], findet, dass den Platz von Beweisen untersucht. Dieses enthielt jedoch keine aussagenlogischen Operatoren. Siehe hierzu auch die Bemerkungen in [Nor2013, Fußnote 2].

[30] Der Speicherverbrauch zu einem Zeitpunkt $t$ ist dabei die Anzahl der Klauseln aus den Zeitpunkten $\leq t$, die in Schritten zu Zeitpunkten $\geq t$ gebraucht werden.

[31] Die Begriffe Prover und Verifier sind dabei der Terminologie von *interaktiven Beweissystemen* entlehnt, die z. B. in der Definition der Komplexitätsklasse IP benötigt werden. Siehe u. a. [BM1988, GMR1989].

Um Platz elegant und gleichzeitig formal definieren zu können und später über Zeit-Platz Tradeoffs sprechen zu können, ist es vorteilhaft, motiviert durch Intuition 4.1.3, Beweise von Resolutionsableitungen geringfügig anders zu beschreiben als bisher. Gleichzeitig geben wir unseren Versuch der Definition der Länge in 4.1.1 hiermit auf. Unsere Definition stellt eine Adaption von [ET2001, ABRW2002, Nor2008, Bon2018] dar.

**Definition 4.1.4** (Resolutionsableitungen mit Klauselkonfigurationen)**.**

(i) Eine *Klauselkonfiguration* (engl. auch: *memory configuration*) $\mathfrak{M}$ ist eine Menge von Klauseln (also eine CNF-Formel).

(ii) Eine Folge $\pi = (\mathfrak{M}_1, \ldots, \mathfrak{M}_t)$ von Klauselkonfigurationen ist eine *Resolutionsableitung* aus einer CNF-Formel $F$, falls gilt:
   (a) $\mathfrak{M}_1 = \varnothing$,
   (b) für alle $i = 2, \ldots, t$ entsteht $\mathfrak{M}_i$ aus $\mathfrak{M}_{i-1}$ durch die Anwendung einer der folgenden drei Regeln:
      - *Axiom-Download*: $\mathfrak{M}_i := \mathfrak{M}_{i-1} \cup \{C\}$ für eine Klausel $C \in F$,
      - *Löschung*: $\mathfrak{M}_i := \mathfrak{M}_{i-1} \setminus \widehat{\mathfrak{M}}$ für ein $\widehat{\mathfrak{M}} \subset \mathfrak{M}_{i-1}$,
      - *Resolvieren*: $\mathfrak{M}_i := \mathfrak{M}_{i-1} \cup \{D\}$, wobei $D$ die Resolvente zweier Klauseln $C_1, C_2 \in \mathfrak{M}_{i-1}$ ist.

(iii) Eine *Resolutionsableitung* $\pi \colon F \vdash_{\mathfrak{R}} A$ *einer Klausel* $A$ aus der CNF-Formel $F$ ist eine Ableitung $(\mathfrak{M}_1, \ldots, \mathfrak{M}_t)$, wie in Punkt (ii), mit $A \in \mathfrak{M}_t$. Eine *Resolutionswiderlegung* von $F$ ist eine Resolutionsableitung $\pi \colon F \vdash_{\mathfrak{R}} \square$.

**Bemerkung 4.1.5.** Die Betrachtung von sublinearem Platz macht es nötig, Ursprungsklauseln der Formel $F$ (sog. *Axiome*) in späteren Schritten per „Axiom-Download“ ggfs. zur Verfügung zu stellen. Dies geschieht in Analogie zur Unterscheidung bei Turingmaschinen in *read-only*-Eingabeband sowie Arbeitsbänder ([ET2001]), z. B. um die Komplexitätsklasse $\mathsf{L}$ (die Menge der Entscheidungsprobleme, die mit einer deterministischen Turingmaschine mit *logarithmischem Platzverbrauch* gelöst werden können) zu studieren.

Wir erinnern an die Definitionen der Präliminarien, angepasst auf Klauselkonfigurationen und erweitern diese in der folgenden Definition. Die Nicht-Berücksichtigung von trivialen Klauseln im Platz von $\mathfrak{M}$ ist entlehnt von [Bon2018, Gleichung (3.1)] und wird sich im weiteren Verlauf, insb. in Beispiel 4.1.10 und Lemma 7.1.3, als nützlich erweisen.

**Definition 4.1.6** (Weite und Platz einer Klauselkonfiguration)**.**

(i) Die *Weite einer Klausel* $C$ ist die Anzahl der in ihr enthaltenen Literale, in Zeichen $\mathrm{Wi}(C) = |\operatorname{Var}(C)|$. Die *Weite einer Klauselkonfiguration* $\mathfrak{M}$ ist $\mathrm{Wi}(\mathfrak{M}) := \max_{C \in \mathfrak{M}} \mathrm{Wi}(C)$.

(ii) Der *Klauselplatz* $\mathrm{Sp}(\mathfrak{M})$ *einer Klauselkonfiguration* $\mathfrak{M}$ ist $|\{C \in \mathfrak{M} : C \neq \square\}|$, d. h. die Anzahl der nicht-trivialen Klauseln in $\mathfrak{M}$.

**Definition 4.1.7** (Länge, Weite und Platz einer Resolutionsableitung)**.** Es sei im Folgenden $\pi = (\mathfrak{M}_1, \ldots, \mathfrak{M}_t)$ eine Resolutionsableitung.

(i) Die *Länge* $\mathrm{Le}(\pi)$ von $\pi$ ist die Anzahl der Axiom-Download- und Resolvierungs-Schritte im Verlauf der Ableitung.

(ii) Die *Weite* von $\pi$ ist $\mathrm{Wi}(\pi) := \max_{i\in[t]} \mathrm{Wi}(\mathfrak{M}_i)$.

(iii) Der *Klauselplatz* von $\pi$ ist $\mathrm{Sp}(\pi) := \max_{i\in[t]} \mathrm{Sp}(\mathfrak{M}_i)$.

Ähnlich wie in Satz & Beispiel 3.4.10 treffen wir die folgende Definition.

**Definition 4.1.8.** Es sei $F$ eine CNF-Formel und $A$ eine Klausel (evtl. die leere Klausel). Dann sei

$$\mathrm{Le}(F \vdash_{\mathfrak{R}} A) := \min_{\pi\colon F\vdash_{\mathfrak{R}} A} \mathrm{Le}(\pi),$$

wobei das Minimum über alle Resolutionsableitungen von $A$ aus $F$ zu nehmen ist. Die Bezeichner $\mathrm{Wi}(F \vdash_{\mathfrak{R}} A)$ und $\mathrm{Sp}(F \vdash_{\mathfrak{R}} A)$ seien komplett analog definiert.

**Konvention 4.1.9.** Neben Klauselplatz ist es ebenfalls möglich, *Variablenplatz* oder *Totalen Platz*[32] zu definieren (siehe z. B. [Nor2008, Nor2013, Bon2018]). Da wir nur eine Platz-Art studieren werden, werden wir künftig kurz *Platz* statt *Klauselplatz* schreiben.

Wir illustrieren die obigen Definitionen mit einem sehr leichten Beispiel, auf das wir im Beweis unseres Hauptresultates in Lemma 7.1.3 zurückkommen werden.

**Beispiel 4.1.10** (Korrektur von Theorem 3.1 in [Bon2018])**.** Der zur Resolutionswiderlegung der Formel $F = \{x, \neg x\}$ benötigte Platz ist 2: Wähle dazu

$$\pi = (\mathfrak{M}_1, \mathfrak{M}_2, \mathfrak{M}_3, \mathfrak{M}_4) = \Big(\varnothing, \big\{\{x\}\big\}, \big\{\{x\}, \{\neg x\}\big\}, \big\{\{x\}, \{\neg x\}, \square\big\}\Big).$$

Es ist $\mathrm{Sp}(\mathfrak{M}_1) = 0$, $\mathrm{Sp}(\mathfrak{M}_2) = 1$, $\mathrm{Sp}(\mathfrak{M}_3) = 2$ und $\mathrm{Sp}(\mathfrak{M}_4) = 2$. Weiter ist $\mathrm{Le}(\pi) = 3$.

**Bemerkung 4.1.11.** Unter der neuen Beschreibung einer Resolutionsableitung ist eine baumartige Resolution eine Ableitung, bei der eine Klausel unmittelbar nach ihrer Benut-

[32] Es sei angemerkt, dass Klauselplatz den tatsächlichen Speicherverbrauch eines SAT-Solvers *unterschätzt*, da das Speichern einer Klausel mehr als konstant viel Platz benötigt. Totaler Platz zählt daher die Gesamtanzahl der Literale im Speicher (mit Wiederholungen).

zung in einem Resolutionsschritt gelöscht werden muss.

**Bemerkung 4.1.12.** Wir führen ein weiteres Komplexitätsmaß, das der *Tiefe*, in Definition 7.1.2 ein.

## 4.2 Untere und obere Schranken für Komplexitätsmaße

Die ersten unteren Schranken für die Größe von Resolutionswiderlegungen konnten in den Papern [Tse1983, Hak1985] gezeigt werden. Durch das darauffolgende Studium von Komplexitätsmaßen wurden untere Schranken in einer Vielzahl von Artikeln untersucht (was den hauptsächlichen Grund darstellt, diese Maße in erster Linie zu definieren). Wir verweisen auf die Survey Paper [BP2001, Raz2001, Seg2007, Pud2008, Pit2011, Nor2015]. Neben dem theoretischen Interesse an unteren Schranken liegt das heutige Interesse an diesen Schranken vor allem in der Verbindung mit SAT-Solvern begründet, die wir in Abschnitt 4.3 beleuchten werden.

In diesem Abschnitt stellen wir einige ausgewählte theoretische Resultate vor. Wir orientieren uns im Folgenden eng an der Darstellung in [Nor2015].

Für die folgenden Diskussionen bezeichne $N$ die Größe der entsprechenden diskutierten Formel. Für $k$-CNF-Formeln ist $N$ also die Anzahl der Klauseln bis auf einen konstanten linearen Faktor.

Man beachte, dass jede unerfüllbare CNF-Formel der Größe $N$ durch Resolution in Länge $\exp(\mathcal{O}(N))$ widerlegt werden kann (z. B. existiert für jede unerfüllbare CNF-Formel $F$ in $n$ Variablen eine baumartige Resolutionswiderlegung $\pi$ mit $\mathrm{Le}(\pi) \leq 2^{n+1} - 1$, was Lemma 7.1.3 inspiriert hat). Siehe z. B. [ST2013, S. 42] für einen Beweis der Widerlegungsvollständigkeit des Resolutionskalküls, der diese Aussage mitliefert. Übereinstimmende untere Schranken von $\exp(\Omega(N))$ sind für spezielle Formeln bekannt.

#### Beispiel 1: Das Taubenschlag-Prinzip

In einer Booleschen Formel wollen wir den formalen Fakt ausdrücken, dass es keine totale injektive Funktion von $\{1, \ldots, m\}$ nach $\{1, \ldots, n\}$ gibt, falls $m > n$ ist bzw. den Fakt, dass eine totale Funktion $f\colon \{1, \ldots, m\} \to \{1, \ldots, n\}$ notwendigerweise zwei Elemente aus ihrem Definitionsbereich $\mathrm{Dom}(f)$ auf dasselbe Element in ihrem Bild $\mathrm{Bi}(f)$ abbildet, falls $m > n$ ist.

Dies ist das berühmte sog. *Taubenschlag-Prinzip* (engl. *pigeon hole principle*), welches zudem ein klassisches Beispiel für die „Codierung“ eines kombinatorischen Prinzips als Boolesche Formel ist.

**Formel 4.2.1** (Das Pigeonhole Principle PHP)**.** Für $n \in \mathbb{N}$ und $m > n$ benutzen wir die Variable $p_{i,j}$, um „Taube $i$ sitzt in Taubenschlag $j$“ auszudrücken und definieren $\mathrm{PHP}_n^m$ als die Menge der folgenden Klauseln: Für alle $i = 1, \ldots, m$ drückt die Klausel

$$(p_{i,1} \vee p_{i,2} \vee \cdots \vee p_{i,n})$$

den Fakt aus, dass „Taube $i$ im Taubenschlag 1 oder 2 oder … oder $n$ platziert werden muss“ (insgesamt bekommt also jede Taube einen Taubenschlag; *Totalitätsaxiome*); für alle $j = 1, \ldots, n$ drücken die $\binom{m}{2}$ Klauseln

$$(\overline{p_{1,j}} \vee \overline{p_{2,j}}), (\overline{p_{1,j}} \vee \overline{p_{3,j}}), \ldots, (\overline{p_{m-1,j}} \vee \overline{p_{m,j}})$$

den Fakt aus, dass „in Taubenschlag $j$ höchstens eine Taube platziert werden kann“ (*Injektivitätsaxiome*) ([ST2013, Bon2018]). Nach obigen Bemerkungen ist die Formel $\mathrm{PHP}_n^m$ für $m > n$ unerfüllbar (siehe auch [Juk2012, S. 501] für einen *formalen* Beweis).

Das in Hakens bahnbrechendem Resultat studierte Beispiel ist $\mathrm{PHP}_n^{n+1}$. Diese Menge enthält $n \cdot (n+1) = \mathcal{O}(n^2)$ Variablen und hat $(n+1) + n \cdot \binom{n+1}{2} = \mathcal{O}(n^3)$ Klauseln. Sie war die erste Formel, von der gezeigt wurde, dass sie keine Resolutionswiderlegungen polynomieller Länge besitzen kann.

**Theorem 4.2.2** ([Hak1985])**.** *Jede Resolutionswiderlegung von* $\mathrm{PHP}_n^{n+1}$ *benötigt Länge mindestens* $2^{n/20} = \exp(\Omega(n))$.

*Bezeichne $k$ die Anzahl der Variablen und $m$ die Anzahl der Klauseln in* $\mathrm{PHP}_n^{n+1}$*, so ist die Länge jeder Resolutionswiderlegung von* $\mathrm{PHP}_n^{n+1}$ *also mindestens* $2^{\Omega(\sqrt{k})}$ *respektive* $2^{\Omega(\sqrt[3]{m})}$.

**Korollar 4.2.3.** *Resolution ist nicht p-beschränkt.*

*Beweisidee von Theorem 4.2.2.* Der Originalbeweis findet sich in [Hak1985]. Wir verweisen weiterhin auf [ST2013, § 2.5] für eine Adaption des deutlich vereinfachten Beweises von Beame und Pitassi [BP1996]. Wir erläutern grob die grundlegende Idee dieses Beweises, da sich gewisse Parallelen zum Beweis von Haupttheorem 7.0.1 (iii) bzw. Theorem 7.3.27 feststellen lassen. Nach Definition einer passenden Festlegung des Begriffes einer „weiten“ Klausel funktioniert der Widerspruchsbeweis im Wesentlichen in zwei Schritten:

(i) Man zeigt: Angenommen es gibt eine kurze Resolutionswiderlegung von $\mathrm{PHP}_n^{n+1}$, so kann durch eine passend iterativ konstruierte Einschränkung diese Widerlegung so eingeschränkt werden, dass man eine Widerlegung von $\mathrm{PHP}_m^{m+1}$ erhält, die keine

weite Klausel enthält (mit einem $m$, das geschickt gewählt ist, sodass es zwar kleiner als das ursprüngliche $n$ ist, aber immer noch „groß genug“).

(ii) Um einen Widerspruch zu erhalten zeigt man ein sog. *Wide Clause Lemma*: Ist $m$ „groß genug“, so muss jede Resolutionswiderlegung von $\mathrm{PHP}_m^{m+1}$ mindestens eine „weite“ Klausel beinhalten. Widerspruch! D. h. $\mathrm{PHP}_n^{n+1}$ kann keine kurzen Resolutionswiderlegungen besitzen.

Ein wesentlicher Schritt im Beweis des Wide Clause Lemmas ist die Definition eines passenden Komplexitätsmaßes für Klauseln (siehe z. B. [Juk2012, § 18.5]) – eine Idee, die wir in Abschnitt 7.3.2 für unseren Beweis aufgreifen werden. □

**Bemerkung 4.2.4.** Selbst bei Erweiterung der Axiome von $\mathrm{PHP}_n^{n+1}$ um sogenannte *onto-Axiome* zu onto-$\mathrm{PHP}_n^{n+1}$, wie „jedes Loch $j$ bekommt eine Taube“, d. h. Konjunktion von $\mathrm{PHP}_n^{n+1}$ mit den Klauseln

$$p_{1,j} \vee p_{2,j} \vee \cdots \vee p_{n+1,j} \quad \text{für alle } j = 1, \ldots, n,$$

bzw. bei Erweiterung mit „Funktionalität“ zu f-$\mathrm{PHP}_n^{n+1}$, wie „keine Taube $i$ bekommt zwei Löcher $j \neq j'$“,

$$\neg p_{i,j} \vee \neg p_{i,j'} \quad \text{für alle } i = 1, \ldots, n+1, \quad j = 1, \ldots, n, \quad j \neq j',$$

(siehe [Nor2015, Bon2018] für Details zu diesen Formeln), benötigt jede Resolutionswiderlegung Länge $\exp(\Omega(n))$. Ebenso konnte in [Raz2003, Raz2004a, Raz2004b] gezeigt werden, dass selbst für beliebig viele Tauben $m$, wodurch die Behauptung $\mathrm{PHP}_n^m$ noch „absurder“ wird (der Beweis also *intuitiv* kürzer sein könnte, weshalb auch vom *weak* pigeonhole principle gesprochen wird), jede Resolutionswiderlegung dieser Formel (oder entsprechenden Erweiterungen wie onto-f-PHP) Länge $\exp(\Omega(n^\delta))$ für ein $\delta > 0$ benötigt. Nordström liefert in [Nor2015] eine intuitive Interpretation dieser Resultate:

> What this means, intuitively, is that the resolution proof system really cannot count—even faced with the preposterous claim that infinitely many pigeons can be mapped in a one-to-one fashion into a finite number $n$ of holes, resolution cannot find a short proof to refute this claim.

#### Beispiel 2: Tseitin-Formeln

Wie wir in Theorem 4.2.2 gesehen haben, ist Hakens untere Schranke „lediglich“ von der Ordnung $\exp(\Omega(\sqrt[3]{N}))$ in der Größe $N = \Theta(n^3)$ der Formel (was in [Nor2013] auch als *weakly exponential* bezeichnet wird). Die erste *echte* exponentielle untere Schranke $\exp(\Omega(N))$

konnte mittels Tseitin-Formeln etabliert werden. Wir verweisen auf Kapitel 5 für eine saubere Einführung dieser unerfüllbaren Formeln, denen ein *ungerade markierter Graph* zugrunde liegt.

URQUHART zeigte in [Urq1987, Theorem 5.7], dass Tseitin-Formeln Resolutionslänge $\exp(\Omega(N))$ benötigen, falls der zugrunde liegende Graph *hochzusammenhängend* (engl. *well-connected*), ein sog. *Expander-Graph*[33] ist. In diesem Zusammenhang ist SCHÖNINGs Beweisvereinfachung und -verbesserung aus [Sch1997] ebenfalls erwähnenswert. Für weitere Resultate verweisen wir auf Bemerkung 5.0.8. URQUHARTs Resultat liefert also erneut: Resolution ist nicht $p$-beschränkt gemäß der Diskussion auf Seite 33.

Wie zu Beispiel 1 liefert [Nor2015] eine intuitive Interpretation dieser Resultate:

> Intuitively, this shows that resolution also is not able to count mod 2 efficiently.

**Beispiel 3: Zufallsformeln**

Um SAT-Solver zu testen, werden häufig Zufallsformeln als Benchmark-Test erstellt. Wir nehmen an, dass wir eine Formel mit $n$ Variablen (wir erlauben es, dass $n$ wächst) und $m$ Klauseln mit jeweils exakt $k$ Literalen ($k$ betrachten wir als fixierte Konstante) generieren wollen. Es bezeichne $\gamma := \frac{m}{n}$ das Verhältnis von Klauselanzahl zu Variablenanzahl. Wir schreiben $F \sim \mathcal{F}_k^{n,\gamma}$, falls die Formel $F$ durch einen probabilistischen Algorithmus „erstellt“ wurde, sodass sie $m = \gamma n$ Klauseln über $n$ Variablen hat, die unabhängig unter Gleichverteilung aus der Menge aller $2^k\binom{n}{k}$ nicht-trivialen Klauseln mit exakt $k$ Literalen gewählt wurden (d. h. Klauseln mit $k$ paarweise verschiedenen, nicht-komplementären Literalen). Siehe [ST2013, § 7.1] für einen einfachen Pseudo-Code eines solchen Algorithmus. Ist $F \sim \mathcal{F}_k^{n,\gamma}$, so ist offensichtlich $F \in k$-CNF.

[33] Ein ungerichteter Graph $G = (V, E)$ ist ein $(d, k)$-*Kanten-Expander*, falls $G$ beschränkten Grad $d = \mathcal{O}(1)$ hat und für alle Knotenmengen $S \subset V$ der Größe $|S| \leq |V|/2$ die Ungleichung $|\delta_G(S)| := |\{uv \in E : u \in S \text{ und } v \in V \setminus S\}| \geq k|S|$ gilt. D. h. ein konstanter Bruchteil der mit $S$ inzidenten Kanten verlässt die Menge $S$ obwohl der Graph $G$ nur vergleichsweise wenige Kanten (linear in der Anzahl der Knoten) besitzt. In Abschnitt 7.3.1 werden wir uns ausführlich mit solchen *isoperimetrischen Ungleichungen* beschäftigen.

Ist nun $d \in \mathbb{N}$ und $k > 0$ fixiert, sowie $(G_n)_{n\in\mathbb{N}_{\geq 3}}$ eine Familie von $n$-knotigen, $(d, k)$-Kanten-Expandern, sowie $(\chi)_{n\in\mathbb{N}_{n\geq 3}}$ eine beliebige Familie von ungeraden Markierungen des Graphen $G_n$, so benötigt die Formelfamilie $(\mathcal{T}(G_n, \chi_n))_{n\in\mathbb{N}_{\geq 3}}$, die wir in Kapitel 5 einführen werden, Resolutionswiderlegungen der Länge $\exp(\Omega(n))$. Da die Formelgröße $\Theta(n)$ ist, ist dies eine echt exponentielle untere Schranke. Die genauen Konstanten hängen von $d$ und $k$ ab.

Es sei angemerkt, dass obige Aussage *nicht* nichtssagend ist, d. h. solche Expander-Graphen existieren. Elegant wurde dies in [Bol1988] gezeigt: Ist $d \in \mathbb{N}_{\geq 3}$, dann gibt es eine universelle Konstante $k > 0$, sodass ein Zufalls $d$-regulärer Graph (ein Graph bei dem alle Knoten Grad $d$ haben) asymptotisch fast sicher ein $(d, k)$-Kanten-Expander ist. Dabei tritt eine Familie von Ereignissen $(E_n)_{n\in\mathbb{N}}$ *asymptotisch fast sicher* ein, falls $\lim_{n\to\infty} \text{Prob}[E_n] = 1$ gilt. NORDSTRÖM merkt in [Nor2016a] an, dass explizite Konstruktionen von Familien von Expander-Graphen “highly nontrival” sind – wir werden dies in Abschnitt 7.3.1 sehen, wo wir eine solche explizite Konstruktion mit erweiterten Zusatzforderungen unter großem Aufwand vornehmen.

Wir weisen an dieser Stelle auf die *Schwellenwert-Eigenschaft* des Parameters $\gamma$ hin: Für jedes $k \in \mathbb{N}_{\geq 2}$ existiert eine Konstante $\gamma_k$ mit $\gamma_2 < \gamma_3 < \gamma_4 < \ldots$, sodass die Wahrscheinlichkeit $\mathrm{Prob}_{F \sim \mathcal{F}_k^{n,\gamma}}[F \in \text{SAT}]$ nahe bei 1 liegt, falls $\gamma < \gamma_k$ ist, und nahe bei 0, falls $\gamma > \gamma_k$ gilt. Für einen kurzen Überblick über die *Stability Threshold Conjecture* und bisher erzielte Resultate zu dieser Thematik verweisen wir auf [BHMW2009, Kapitel 8], [CK2002, Kapitel 4]. Weiter verweisen wir auf [CM1997] und [ST2013, § 7.1] für eine Diskussion zur erhöhten Laufzeit von DPLL-Algorithmen um den jeweiligen Schwellenwert (experimentelle Untersuchen, wie [KS1994, S. 1298] und [CM1997], legen $\gamma_3 \approx 4{,}2$ nahe). Wir begnügen uns an dieser Stelle mit der Zitation zweier gefeierter Resultate.

**Theorem 4.2.5** (Adaptiert aus [FP1983])**.** *Falls $\gamma \geq 2^k \ln 2$ gilt, so ist*

$$\lim_{n \to \infty} \operatorname*{Prob}_{F \sim \mathcal{F}_k^{n,\gamma}} \Big[ F \in \text{UNSAT} \Big] = 1.$$

*Beweis.* Siehe z. B. [ST2013, S. 122f.] für einen modernen, kurzen Beweis. □

**Theorem 4.2.6** (Adaptiert aus [CS1988])**.** *Für jede Wahl von natürlichen Zahlen $\gamma$ und $k$, sodass $k \geq 3$ und $\gamma \geq 2^k \ln 2$ ist, existiert ein $\varepsilon > 0$, sodass gilt:*

$$\lim_{n \to \infty} \operatorname*{Prob}_{F \sim \mathcal{F}_k^{n,\gamma}} \Big[ F \in \text{UNSAT} \text{ und } \mathrm{Le}(F \vdash_{\mathfrak{R}} \square) \geq (1+\varepsilon)^n \Big] = 1.$$

Die beste untere Schranke für 3-SAT konnte weiter verbessert werden, sodass wir folgendes anschauliches Resultat erhalten (siehe z. B. [CK2002, Theorem 4.3.1], [Nor2015]): Für $\gamma \gtrsim 4{,}51$ ist eine Zufallsformel $F \sim \mathcal{F}_k^{n,\gamma}$ asymptotisch fast sicher unerfüllbar und benötigt Resolutionslänge mindestens $\exp(\Omega(N))$ in der Länge $N$ der Formel zur Widerlegung.

### Beispiel 4: Platz

Wie in der Einleitung in Abschnitt 1.1 erwähnt, konnte in [ET2001] gezeigt werden, dass eine unerfüllbare CNF-Formel der Größe $N$ immer durch Resolution widerlegt werden kann ohne Platz mehr als $N + \mathcal{O}(1)$ zu benutzen. Diese Widerlegung kann jedoch exponentiell lang sein. Passende untere Schranken konnten für PHP-Formeln (jede Resolutionswiderlegung von $\mathrm{PHP}_n^m$ benötigt Klauselplatz mindestens $n$, unabhängig von der Anzahl der Tauben $m$) und Tseitin-Formeln [ET2001, ABRW2002] sowie $k$-CNF Zufallsformeln [BG2003] gezeigt werden.

Für weiterführende Untersuchungen zwischen Platz und Weite empfehlen wir [Nor2015, § 2.3]. Weitere Schranken bzgl. Platz finden sich z. B. in [Ben2009a, § 7].

**Beispiel 5: Weite**

Neben den trivialen Abschätzungen $\mathrm{Wi}(F \vdash_{\mathfrak{R}} \square) \leq |\operatorname{Var}(F)| \leq \operatorname{size}(F)$ ist es auch möglich einfache obere Schranken an die Länge von Resolutionswiderlegungen aus oberen Schranken an die Weite von Resolutionswiderlegungen zu gewinnen: Ist $F$ eine unerfüllbare CNF-Formel in $n$ Variablen für die $\mathrm{Wi}(F \vdash_{\mathfrak{R}} \square) \leq w$ gilt, so folgt sofort $\mathrm{Le}(F \vdash_{\mathfrak{R}} \square) \leq \mathcal{O}(n^w)$, da die Anzahl an unterschiedlichen Klauseln in $n$ Variablen mit Weite höchstens $w$ maximal $(3n)^w$ betragen kann. Trotz der Einfachheit des Argumentes ist diese Abschätzung essentiell scharf: Es gibt unerfüllbare $k$-CNF-Formeln, die in Weite $w$ widerlegbar sind und Resolutionslänge $n^{\Omega(w)}$ brauchen, wie in [ALN2016] gezeigt wurde.

In ihrem bahnbrechenden Paper [BW2001] zeigten Ben-Sasson und Wigderson, dass ausreichend starke untere Schranken an die Weite starke untere Schranken an die Länge implizieren[34]: Für eine unerfüllbare $k$-CNF Formel mit Größe $N$ gilt

$$\mathrm{Le}(F \vdash_{\mathfrak{R}} \square) \geq \exp\left(\Omega\left(\frac{(\mathrm{Wi}(F \vdash_{\mathfrak{R}} \square) - \mathrm{Wi}(F))^2}{N}\right)\right). \tag{4.1}$$

Dieses Resultat ist aus einer Vielzahl von Gründen interessant: Ist es etwa möglich, zu zeigen, dass eine Formel Weite $\omega(\sqrt{N \log N})$ zur Resolutionswiderlegung benötigt, impliziert (4.1) eine echt-exponentielle untere Schranke $\exp(\Omega(N))$ an die Länge der Widerlegung. Interessanterweise können, wie z. B. in [Ben2009a, § 3.2] und [Nor2015, § 2.2] angemerkt wurde, alle unteren Schranken an die Resolutionslänge in den vorangegangenen Beispielen 1–3 auf diese Art gezeigt werden (auch wenn die ursprünglichen Beweise der Resultate teils andere Methoden verwendet haben).

## 4.3 Zusammenhang der Komplexitätsmaße mit SAT-Solvern

Um den in der Überschrift angedeuteten Zusammenhang zu erläutern, folgen wir in der Darstellung der folgenden Unterabschnitte [Bon2018, CM1997, ST2013, Nor2015]. Diese Diskussionen werden insbesondere die Praxisrelevanz des Tradeoff-Resultates 7.0.1 etablieren und zeigen, dass die unteren Schranken des vorigen Kapitels algorithmische Gegenstücke besitzen[35].

Im Folgenden bezeichne eine *Unit-Klausel* $\{u\}$ eine Klausel mit nur einem Literal. Ein *pures Literal* ist ein Literal, das in einer Formel nur negiert oder nur positiv vorkommt.

[34] Wir verwenden im Folgenden eine geringfügig modifizierte Version des Resultates nach [Nor2015].

[35] Dies stellt einen wesentlichen Antrieb der aktuellen Forschung in der Beweiskomplexität dar, wie in [Bon2018, § 1.1.4] dargelegt wurde.

### Die Davis-Putnam-Prozedur

Da Resolution, wie wir in Theorem 3.5.5 (c) festgehalten haben, widerlegungsvollständig ist, ist eine denkbar einfache Prozedur um die Erfüllbarkeit einer Formel zu testen, alle möglichen Resolventen dieser zu generieren und zu untersuchen, ob die leere Klausel enthalten ist (siehe [Sch2018b, S. 88]). Davis und Putnam nutzten diese Idee in [DP1960] für folgenden Algorithmus 1: Von einer gegebenen Formel $F$ werden in einem Basisschritt alle Resolventen, die durch Resolvieren nach einer Variable $x$ möglich sind, erzeugt. Alle Elternklauseln dieser Resolventen werden anschließend eliminiert, sodass man eine Klauselmenge ohne die Variable $x$ erhält. Etwas formaler: Bezeichne $F_x \subset F$ die Klauseln von $F$, welche die Variable $x$ oder ihr Komplement $\overline{x}$ enthalten. Weiter sei $\mathrm{Res}_x(F)$ die Menge der Resolventen, die aus $F_x$ durch Resolution nach $x$ entstehen können. Offensichtlich enthält keine Klausel in $\mathrm{Res}_x(F)$ mehr die Variable $x$. Schließlich wird $F$ durch die Menge $(F \cup \mathrm{Res}_x(F)) \setminus F_x$ ersetzt. Es ist leicht zu zeigen, dass $F \in$ SAT ist, genau dann, wenn $(F \cup \mathrm{Res}_x(F)) \setminus F_x \in$ SAT ist (siehe z. B. [ST2013, S. 49]).

**Algorithmus 1 :** Der Davis-Putnam Resolutions-Algorithmus

**Input :** Eine Klauselmenge $F$
**Output :** *sat*, falls $F \in$ SAT; *unsat* andernfalls
**Resolutions-Algorithmus** `DavisPutnam(`$F$`)`
  **if** $\square \in F$ **then return** *unsat*
  **if** $F = \varnothing$ **then return** *sat*
  **while** *F enthält eine Unit-Klausel* $\{u\}$ **do** $F := F\restriction_{\{u=1\}}$
  Wähle eine Variable $x \in \mathrm{Var}(F)$
  **return** `DavisPutnam(`$(F \cup \mathrm{Res}_x(F)) \setminus F_x$`)`

### Die Weiterentwicklung durch Davis, Logemann and Loveland

Jeder Basisschritt des Algorithmus 1 generiert also ein Unterproblem des ursprünglichen Problems mit einer Variable weniger, aber evtl. quadratisch mehr Klauseln. Es ist außerdem nicht schwierig zu zeigen, dass die `DavisPutnam`-Prozedur i. A. exponentiell viele Klauseln generiert.

Da dies in der Praxis eine unhandliche Zahl von Klauseln darstellt, entwickelten Davis, Logemann und Loveland den Algorithmus in [DLL1962] durch folgende Idee weiter: Durch Selektion einer Variable betrachtet man statt einem großen – unhandlichen – Unterproblem zwei kleine Unterprobleme (je eins für eine mögliche Belegung der Variable). Der entstehende Algorithmus ist eine *backtracking depth-first* Suche durch alle (partiellen) Belegungen zusammen mit einer Unit-Klausel und Puren-Literal-Heuristik, wie in Algorithmus 2 dargestellt.

**Algorithmus 2 :** Grundstruktur eines DPLL-Algorithmus

**Input :** Eine Klauselmenge $F$
**Output :** ***sat***, falls $F \in$ SAT; ***unsat*** andernfalls
**Basis-Backtracking-Algorithmus** DPLL($F$)
  **if** $\square \in F$ **then return** ***unsat***
  **if** $F = \varnothing$ **then return** ***sat***
  **if** *$F$ enthält eine Unit-Klausel* $\{u\}$ **then return** DPLL($F\restriction_{\{u=1\}}$)
  **if** *$F$ enthält ein pures Literal* $u$ **then return** DPLL($F\restriction_{\{u=1\}}$)
(∗) Wähle mit einer geeigneten Strategie eine Variable $x \in \mathrm{Var}(F)$
  **if** DPLL($F\restriction_{\{x=0\}}$) **then return** ***sat***
  **return** DPLL($F\restriction_{\{x=1\}}$)

Die Heuristik in (∗) sollte so gewählt sein, dass möglichst viele Verzweigungsschritte des Algorithmus gespart werden (wozu auch die Unit-Klausel und Pure-Literale-Heuristik verwendet werden; siehe [ST2013, S. 73]). Für eine Darstellung von verschiedenen Heuristiken verweisen wir lediglich auf [CM1997, § 3f.], [ST2013, S. 74], [Nor2015, § 3]. Für einen Beispiellauf eines DPLL-Algorithmus siehe bspw. [Sch2018b, S. 89–93].

**Theorem 4.3.1** (Korrespondenz zwischen DPLL und Resolution)**.** *Es sei $F$ eine unerfüllbare* CNF*-Formel und $r$ die minimale Anzahl an rekursiven Aufrufen von* DPLL($F$) *(bis* ***unsat*** *ausgegeben wird) bei beliebiger Heuristik in* (∗)*. Dann gibt es eine baumartige Resolutionswiderlegung von $F$ mit Länge höchstens $r$. D. h. es gilt* $\mathrm{Le}(F \vdash_{\text{baumartige } \mathfrak{R}} \square) \leq \mathrm{time}_{\mathtt{DPLL}}(F)$.

*Umgekehrt gilt ebenso: Gibt es eine baumartige Resolutionswiderlegung der Länge $r$ für eine unerfüllbare* CNF*-Formel $F$, so benötigt* DPLL($F$) *mit der korrespondierenden Heuristik für* (∗) *höchstens $r$ rekursive Aufrufe bis zur Ausgabe von* ***unsat****.*

*Beweis.* Siehe bspw. [ST2013, § 4], [Ben2009a, Claim 15] oder [Pit2002]. □

Folgendes einfaches Korollar liefert die Praxisrelevanz des Tradeoff-Resultates.

**Korollar 4.3.2.** *Eine untere Schranke für die minimale Länge einer (baumartigen) Resolutionswiderlegung einer Formel $F$ impliziert eine untere Schranke für die Laufzeit des* DPLL*-Algorithmus, unabhängig der verwendeten Heuristik in* (∗)*. Folglich ist dies damit auch eine untere Schranke für alle* DPLL*-artigen Algorithmen.*

*Im Sinne der Beweiskomplexität sind* baumartige Resolution *und die* DPLL*-Prozedur p-äquivalent.*

Selbst nach über 50 Jahren, stellt die DPLL-Prozedur immer noch die Basis der effizientesten SAT-Solver (wie bspw. Chaff, zChaff, GRASP oder MiniSAT, siehe auch den nächsten

Unterabschnitt) dar. Eine theoretische *obere* Schranke für DPLL bei 3-SAT ergibt sich durch eine einfache Rekursionsanalyse zu $\mathcal{O}(1{,}913^n) = \mathcal{O}(2^{0{,}936n})$ (siehe bspw. [ST2013, S. 72f.]), was bedeutend besser als der naïve SAT-Algorithmus mit $\mathcal{O}(2^n)$ ist.

**Conflict Driven Clause Learning und die Bedeutung von Tradeoffs**

Ein „Problem“ an Algorithmus 2 ist, dass Information, die über die Formel während des Suchprozesses innerhalb eines rekursiven Aufrufes „gelernt“ wurde, nicht gespeichert wird und später, wenn der Algorithmus zum Stadium dieses Aufrufes zurückkehrt, verloren geht. So wird evtl. dieselbe Arbeit in mehreren Teilbäumen des Algorithmus dupliziert. Durch die Arbeiten [BS1997, MS1999, MMZ+2001] wurde die Idee populär, dass informell gesprochen, eine *Konflikt-Analyse* durchgeführt wird, wenn ein ***unsat*** erreicht wird und der *Grund* hierfür in Form einer neuen Klausel $C$, die zur ursprünglichen Formel $F$ hinzugefügt wird, „gelernt“ wird. Danach muss der Solver so lange Variablen „freigeben“ (evtl. auch aus vorherigen Entscheidungen), bis die neue Klausel $C$ nicht mehr falsifiziert ist. Algorithmen dieser Art werden kurz CDCL-Algorithmen genannt (wobei es auch hier zahlreiche Heuristiken für das Lernen von Klauseln gibt). Für weiterführenden Details zu diesen Heuristiken, *Implikationsgraphen*, *Non-Chronological Backtracking* und *Restarts* sowie den Zusammenhang zur Beweiskomplexität verweisen wir auf [BKS2004, Hof2007, Jär2008, BHMW2009, ST2013, Nor2015].

Ähnlich wie in Theorem 4.3.1 kann festgehalten werden, dass untere Schranken an die Resolutionslänge untere Schranken an die Laufzeit von CDCL-Solvern implizieren, da man *im Prinzip*[36] einen Resolutionsbeweis aus einem Lauf eines CDCL-Algorithmus extrahieren kann (vgl. [PD2011] und [Nor2015, S. 4]). Ähnliche Feststellungen lassen sich auch über den Platz einer Resolutionswiderlegung und Speicherverbrauch von (optimalen) CDCL-Solvern treffen (siehe bspw. [Bon2018, § 1.1.4], [JMNŽ2012]). Ein Zeit-Platz Tradeoff-Resultat ist daher deshalb so interessant, da CDCL-Solver sowohl Laufzeit als auch Speicherplatz aggressiv minimieren wollen. Wir zitieren [JMNŽ2012, Nor2013]:

> The main bottleneck for CDCL solvers—apart from the obvious exponential worst case behavior—is the amount of memory used. In practice, it is completely infeasible to store all clauses learned during a CDCL run, and one therefore needs to design a highly selective and efficient clause caching scheme that learns and keeps the clauses needed for the CDCL solver to finish fast. Thus, understanding time and memory requirements for clause learning algorithms, and how these resources are related to each other, is a question of great practical importance.

[36] Wir ignorieren Details wie *Preprocessing*. Für eine saubere Diskussion zur Beschreibung von CDCL als Beweissystem verweisen wir auf den kürzlich erschienen Konferenzbeitrag [EJL+2016]. Siehe auch [Ben2009a, Claim 17]. Eine genaue Diskussion liegt leider außerhalb des Rahmens dieser Arbeit.

## 4.4 Beispiele für Tradeoffs

### Motivation: Ein Tradeoff außerhalb der Beweiskomplexität

Die zwei fundamentalsten Ressourcen für die Berechnung eines Problems sind Zeit und Speicherplatz. Viele grundlegende Resultate der Komplexitätstheorie adressieren das Zusammenspiel dieser Ressourcen. Oft ist es bei der Lösung von Problemen möglich, den Speicherbedarf dramatisch zu reduzieren – bei anderen, häufig auch sehr simplen Problemen ist dies unmöglich.

Zur Hinführung zum Thema *Tradeoffs* wollen wir einen solchen einzugehenden *Kompromiss* an einem einfachen Beispiel aus der Komplexitätstheorie illustrieren.

**Beispiel & Satz 4.4.1** (Adaption aus [Tor2017])**.** Für ein endliches Wort $x \in \{0,1\}^+$ bezeichne $x^{\mathrm{R}}$ das rückwärts geschriebene Wort $x$, das sog. *Reverse* des Wortes, in Zeichen $(x_1 x_2 \dots x_{n-1} x_n)^{\mathrm{R}} = x_n x_{n-1} \dots x_2 x_1$. Es bezeichne weiter $L_{\mathrm{PAL}} := \left\{ x \in \{0,1\}^+ \,:\, x = x^{\mathrm{R}} \right\}$ die *Sprache der Palindrome*. Mit der Bezeichnung $n := |x|$ gilt dann:

(i) Es gibt eine 2-Band Turingmaschine, die zum Entscheiden von $L_{\mathrm{PAL}}$ Zeit $3n = \mathcal{O}(n)$ und Speicherbedarf $\mathcal{O}(n)$ benötigt.

(ii) Ebenfalls gibt es eine 2-Band Turingmaschine, die zum Entscheiden von $L_{\mathrm{PAL}}$ lediglich $\mathcal{O}(\log n)$ Speicherplatz, dafür aber Zeit $\mathcal{O}(n^2)$ benötigt.

(iii) Ist $M$ eine beliebige Mehrband-Turingmaschine mit $\mathrm{T}(M) = L_{\mathrm{PAL}}$ und bezeichne $T\colon \mathbb{N} \to \mathbb{N}$ die Zeitkomplexität und $S\colon \mathbb{N} \to \mathbb{N}$ die Platzkomplexität dieser Maschine, so gilt der *Zeit-Platz Tradeoff*

$$T(n) \cdot S(n) = \Omega(n^2).$$

*Beweisskizze.* (i) Die Maschine kopiert die Eingabe auf das zweite Band, was $n$ Schritte und $n$ Speicherzellen kostet. Der Lesekopf des ersten Bandes wird zurück auf das erste Zeichen bewegt, was $n$ Schritte kostet. Der Lesekopf des ersten Bandes läuft nun von links nach rechts; der Schreib-Lese-Kopf des zweiten Bandes steht auf dem letzten Zeichen und läuft von rechts nach links. Die Maschine vergleicht so nacheinander die Zeichen auf den Bändern, was erneut $n$ Schritte kostet.

(ii) Auf Band 2 befindet sich ein Zähler, der die Position des Lesekopfes auf Band 1 mit $\mathcal{O}(\log n)$ Zellen speichern kann. Mit dem Zähler ist es möglich, folgenden Algorithmus für eine 1-Band Turingmaschine zu simulieren[37]: Der Kopf läuft vom ersten zum letzten Zeichen der Eingabe, dann vom zweiten zum vorletzten, usw., und vergleicht

[37] Zusätzlich kann mit Methoden wie im Schritt (iii) gezeigt werden, dass jede 1-Band Turingmaschine, die $L_{\mathrm{PAL}}$ akzeptiert, Zeit $\Omega(|x|^2)$ benötigt.

dabei die jeweiligen Zeichen auf dem Band (und ersetzt das jeweils gelesene Symbol durch ein „Gelesen"-Symbol, das den Kopf zur „Umkehr" bewegt); dies kostet insgesamt $\mathcal{O}(n^2)$ Schritte.

(iii) Der Widerspruchsbeweis nutzt *nicht-komprimierbare Wörter*, *Kolmogorov-Komplexitäten* und *Crossing Sequences*. Wir verweisen auf [Tor2017, Satz 2.3.3]. □

**Tradeoffs in der Beweiskomplexität**

In Abschnitt 4.1 hatten wir einige *Komplexitätsmaße* für das Beweissystem der Resolution kennengelernt: Länge, Weite und Platz. Es sei angemerkt, dass diese nur die wichtigsten und für unsere Arbeit relevanten Komplexitätsmaße sind. Für weitere siehe z. B. [Nor2008]. Ähnliche Definitionen sind ebenso leicht für andere Beweissysteme zu treffen, siehe z. B. [ABLM2008, JMNŽ2012, BNT2013, BK2014].

In Abschnitt 4.2 haben wir Beispiele für Formeln gesehen, die *maximal hart* für die Komplexitätsmaße Länge und Platz im Beweissystem der Resolution sind. Was aber passiert bei Formeln, die z. B. leicht für diese beiden Komplexitätsmaße sind? Ist es möglich beide Maße gleichzeitig zu optimieren? Oder führt die Minimierung eines Maßes zu einem steilen Anstieg, einem *Blow-up*, des anderen Maßes? Wir sprechen davon, dass es einen *Tradeoff* im Beweissystem der Resolution *zwischen zwei Komplexitätsmaßen* gibt, falls diese nicht beide simultan für die gleiche Resolutionswiderlegung optimiert werden können.

Das erste solche Resultat findet sich in [Ben2009b]: Dort wurde gezeigt, dass es unerfüllbare $k$-CNF-Formeln[38] $F_n$ der Größe $\Theta(n)$ gibt, die sowohl in konstantem Platz $\mathrm{Sp}(F_n \vdash_{\mathfrak{R}} \square) = \mathcal{O}(1)$ als auch in konstanter Weite $\mathrm{Wi}(F_n \vdash_{\mathfrak{R}} \square) = \mathcal{O}(1)$ widerlegt werden können, dass aber jede Resolutionswiderlegung $\pi_n$, die versucht beide Komplexitätsmaße simultan zu optimieren, nie besser sein kann als $\mathrm{Sp}(\pi_n) \cdot \mathrm{Wi}(\pi_n) = \Omega(n/\log n)$.

Für einen Überblick über bekannte Tradeoff-Resultate verweisen wir auf den Survey-Artikel [Nor2013]. Tradeoffs zwischen Länge und Platz im Beweissystem der Resolution wurden u. a. in [BN2008, BN2011, BNT2013, NH2013, Nor2013, BBI2016] gezeigt. Den Resultaten gemeinsam ist, dass die studierten Formeln für gewöhnlich kurze Beweise besitzen und Beweise, die wenig Platz verbrauchen. Aber beides kann nicht simultan erreicht werden.

In [Nor2013] bzw. [Lau2017b] wurde darauf hingewiesen, dass alle Tradeoff-Resultate mit Pebblings leider jedoch lediglich in das *lineare Platz Regime* einzuordnen sind: Wird unlimitiert viel Platz zur Verfügung gestellt, um eine auf Pebblings aufbauende Formel zu widerlegen, dann reicht bereits linearer Platz aus. Ein Resultat dieser Richtung wurde bspw.

[38] Die Konstruktion der Formeln erfolgte dabei mithilfe von sog. *Pebblings*. Wir verzichten in dieser Arbeit vollständig auf dieses Hilfswerkzeug. Aufgrund der Wichtigkeit in der Beweiskomplexität verweisen wir den interessierten Leser jedoch auf das Survey Paper [Nor2013] für eine ausführliche Einführung.

in [BN2011] gezeigt: Es gibt Pebbling Formeln mit Resolutionsbeweisen in linearer Länge $\mathcal{O}(n)$ sowie linearem Platz $\mathcal{O}(n)$, wobei jede Einschränkung des Platzes auf $\mathcal{O}(n/\log n)$ einen exponentiellen Blow-up der Länge auf $\exp(n^{\Omega(1)})$ nach sich zieht.

In [BBI2016] wurde schließlich der erste Tradeoff für *superlinearen Platz* gezeigt – wir verweisen insbesondere auf das hierdurch gelöste Problem 7.0.2. Der nun folgende Teil II ist dem Beweis dieses Resultates gewidmet.

# Teil II.

# Zeit-Platz Tradeoffs im Beweissystem der Resolution für superlinearen Platz

Wir beginnen diesen Teil mit zwei Kapiteln, die die notwendigen Grundlagen liefern, um unser Tradeoff-Resultat zu formulieren und direkt zu beweisen: In Kapitel 5 führen wir die für unsere Konstruktion wichtigen Tseitin-Formeln ein und beschäftigen uns alsdann in Kapitel 6 mit den für die Konstruktion der Formeln im Tradeoff-Resultat benötigten Gittergraphen, Substitutionsformeln und Zufallseinschränkungen. In Kapitel 7 formulieren und beweisen wir schließlich das Haupttheorem der vorliegenden Arbeit.

# KAPITEL 5

## Tseitin-Formeln

Zu Beginn dieses Kapitels zitieren wir zur Motivation einen Satz aus der Graphentheorie, der von Leonhard Euler in seiner Arbeit [Eul1741] über das Königsberger Brückenproblem – das die Graphentheorie begründet hat – bewiesen wurde.

**Definition & Lemma 5.0.1** (Handschlag-Lemma[39]/Grad-Summen-Formel)**.** *Es sei $G = (V, E)$ ein ungerichteter Graph[40]. Der* Gesamtgrad *(engl.* total degree*)* $\deg(G)$ *von $G$ ist stets gerade, präziser gilt*

$$\deg(G) := \sum_{v \in V} \deg(v) = 2|E|.$$

*Insbesondere ist die Anzahl der Knoten ungeraden Grades in $G$ stets gerade.*

*Beweis.* Wir beweisen die zwei Behauptungen separat.

(I) Wir formalisieren die Beweisskizze aus [Vol2013, Satz 1.1] durch die Beweismethode des *doppelten Abzählens* aus der abzählenden Kombinatorik: Wir betrachten dazu die Anzahl der *Inzidenzpaare* $(v, e)$, d. h. derjenigen Tupel, bei denen $e$ eine zum Knoten $v$ inzidente Kante ist.

Ein Knoten $v$ gehört zu $\deg(v)$ solchen Paaren, da $\deg(v)$ gerade als die Anzahl der zu $v$ inzidenten Kanten definiert war. Folglich ist die Anzahl der Inzidenzpaare gerade $\sum_{v \in V} \deg(v)$.

Jedoch gehört auch jede Kante im Graphen zu genau zwei Inzidenzpaaren – eines für jeden ihrer Endknoten. Daher kann die Anzahl der Inzidenzpaare ebenfalls durch $2|E|$ angegeben werden.

[39] Der in der Graphentheorie häufig verwendete Name *Handschlag-Lemma* stammt von dem anschaulichen Beispiel einer Party, auf der einige der anwesenden Personen sich mit Handschlag begrüßen. Das Lemma besagt in diesem Fall, dass die Anzahl an Partygästen, die einer ungeraden Zahl von Gästen die Hand gegeben haben, gerade ist.

[40] Wir erinnern an die Präliminarien: Graphen haben in dieser Arbeit keine Schlingen oder Mehrfachkanten (außer wir erwähnen dies *explizit* – siehe hierzu die nachfolgende Definition & Konvention 6.0.6), sind also *schlichte* Graphen.

(II) Eine Summe natürlicher Zahlen ist genau dann gerade, wenn sie eine gerade Anzahl an ungeraden Summanden enthält. Insbesondere beeinflusst die Anzahl gerader Summanden die Parität der Summe nicht. Wegen

$$2|E| \overset{\text{(I)}}{=} \sum_{v \in V} \deg(v) = \sum_{\substack{v \in V: \\ \deg(v) \text{ gerade}}} \deg(v) + \sum_{\substack{v \in V: \\ \deg(v) \text{ ungerade}}} \deg(v)$$

liefert dies die Behauptung. □

Wie TSEITIN in seiner Arbeit [Tse1968] wollen wir diesen kombinatorischen Fakt als Formel codieren. Unsere Darstellungen im Folgenden sind eine Kombination aus [Sch1997, Tor1999, BW2001, BBI2016].

Es sei $G = (V, E)$ ab hier ein zusammenhängender Graph. Wir nennen eine Funktion $\chi\colon V \to \{0, 1\}$ eine *Markierung* (engl. teilw. *charge*) des Graphen $G$. Eine solche Markierung heißt *ungerade*, falls

$$\sum_{v \in V} \chi(v) \equiv 1 \pmod 2$$

gilt und *gerade* andernfalls. In beiden Fällen nennen wir das Tupel $(G, \chi)$ *markierter Graph*.

Zum Codieren des oben angesprochenen kombinatorischen Faktes assoziieren wir mit jeder Kante $e \in E$ eine eindeutige Boolesche Variable $x_e$. Weiter fixieren wir einen festen Knoten $v \in V$ und betrachten über $\mathbb{F}_2$ die Paritätsbedingung

$$\bigoplus_{e \sim v} x_e \equiv \chi(v). \tag{5.1}$$

Für alle $v \in V$ setzen wir

$$F_{v,\chi(v)} := \begin{cases} x_{e_1(v)} \oplus \cdots \oplus x_{e_d(v)}, & \text{falls } \chi(v) = 1 \\ \overline{x_{e_1(v)} \oplus \cdots \oplus x_{e_d(v)}}, & \text{falls } \chi(v) = 0, \end{cases} \tag{5.2}$$

wobei wir hier mit $x_{e_1(v)}, \ldots, x_{e_d(v)}$ die Variablen bezeichnen, die mit den zum Knoten $v$ inzidenten Kanten $e_1(v), \ldots, e_d(v)$ assoziiert sind und $d := \deg(v)$ ist. Wir setzen schließlich

$$\mathcal{T}(G, \chi) := \bigwedge_{v \in V} F_{v,\chi(v)}.$$

Da wir im Folgenden an der CNF-Repräsentation von $\mathcal{T}(G, \chi)$ interessiert sind, beachten

wir, dass

$$\begin{aligned} F_{v,\chi(v)} &\equiv \mathrm{PARITY}_{v,\chi(v)} \\ &:= \bigwedge \left\{ \bigvee_{e \sim v} x_e^{a(e)} \,:\, a(e) \in \{0,1\}, \text{ sodass } \bigoplus_{e \sim v} \Big(a(e) \oplus 1\Big) \not\equiv \chi(v) \right\}, \end{aligned} \tag{5.3}$$

eine CNF-Darstellung der Paritätsbedingung (5.1) ist, sodass

$$\mathcal{T}(G,\chi) = \bigwedge_{v \in V} \mathrm{PARITY}_{v,\chi(v)} \tag{5.4}$$

eine CNF-Darstellung von $\mathcal{T}(G,\chi)$ ist, welche wir *Tseitin-Formel* nennen wollen. Sind $G$ und $\chi$ aus dem Kontext klar, werden wir im Zukunft auch kurz $\mathcal{T}$ schreiben.

Die Tseitin-Formel „behauptet“ also – als „Codierung“ der Paritätsbedingungen (5.1) für alle $v \in V$ – dass es möglich ist, jeder Kante des Graphen einen Wert `0` bzw. `1` zuzuordnen, sodass für jeden Knoten die Summe der Werte, die den zu diesem Knoten inzidenten Kanten zugeordnet sind, modulo 2 mit dem Wert der Markierung des Knotens übereinstimmt.

Wir illustrieren die obige Definition an zwei Beispielen.

**Beispiel 5.0.2.** Wir betrachten wie [Lau2017a] den „Dreiecksgraphen“

$$G = (\{u,v,w\}, \{uv, vw, wu\})$$

zusammen mit der ungeraden Markierung $\chi(u) = 1$, $\chi(v) = \chi(w) = 0$. Die Tseitin-Formel $\mathcal{T}(G,\chi)$ „behauptet“, dass das folgende Kongruenzsystem erfüllbar ist:

$$\begin{aligned} &\text{Paritätsbedingung zum Knoten } u: && x_{uv} + x_{wu} \equiv 1 \pmod 2 \\ &\text{Paritätsbedingung zum Knoten } v: && x_{uv} + x_{vw} \equiv 0 \pmod 2 \\ &\text{Paritätsbedingung zum Knoten } w: && x_{vw} + x_{wu} \equiv 0 \pmod 2. \end{aligned}$$

Dies ist offensichtlich nicht möglich. Die korrespondierende Tseitin-Formel ergibt sich zu

$$\begin{aligned} \mathcal{T}(G,\chi) =& (x_{uv} \vee x_{wu}) \wedge (\overline{x_{uv}} \vee \overline{x_{wu}}) \\ &\wedge (x_{uv} \vee \overline{x_{vw}}) \wedge (\overline{x_{uv}} \vee x_{vw}) \\ &\wedge (x_{vw} \vee \overline{x_{wu}}) \wedge (\overline{x_{vw}} \vee x_{wu}). \end{aligned}$$

**Beispiel 5.0.3.** Wir betrachten den in Abbildung 5.1 skizzierten Graphen $G = (V, E)$ mit den Knoten $V = \{u, v, w, x, y, z\}$ und Kanten $E = \{e_1, \ldots, e_7\}$. Die Markierung $\chi$ sei gegeben durch $\chi(u) = 1$, sowie $\chi(v) = \chi(w) = \chi(x) = \chi(y) = \chi(z) = 0$.

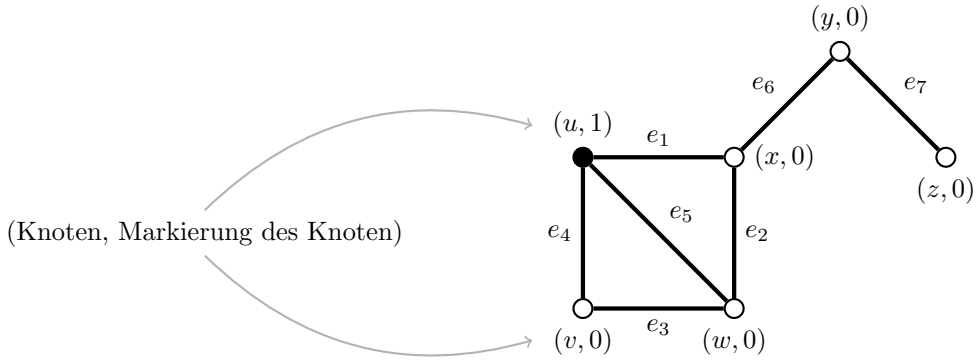



Abbildung 5.1.: Der Graph $G$ mit Markierung $\chi$ aus Beispiel 5.0.3. Die eingezeichneten Tupel $(v_0, \chi(v_0))$ liefern sowohl die Knotenbeschriftung als auch die Knotenmarkierung. Schwarz markierte Knoten symbolisieren eine Markierung mit 1, weiße eine mit 0.

Wir erhalten

$$F_{u,1} = x_{e_1} \oplus x_{e_4} \oplus x_{e_5}, \qquad F_{x,0} = \overline{x_{e_1} \oplus x_{e_2} \oplus x_{e_6}},$$
$$F_{v,0} = \overline{x_{e_3} \oplus x_{e_4}}, \qquad F_{y,0} = \overline{x_{e_6} \oplus x_{e_7}},$$
$$F_{w,0} = \overline{x_{e_2} \oplus x_{e_3} \oplus x_{e_5}}, \qquad F_{z,0} = \overline{x_{e_7}},$$

also

$$\begin{aligned}\mathcal{T}(G, \chi) &= F_{u,1} \wedge F_{v,0} \wedge F_{w,0} \wedge F_{x,0} \wedge F_{y,0} \wedge F_{z,0} \\ &= (x_{e_1} \oplus x_{e_4} \oplus x_{e_5}) \wedge (\overline{x_{e_3} \oplus x_{e_4}}) \wedge (\overline{x_{e_2} \oplus x_{e_3} \oplus x_{e_5}}) \\ &\quad \wedge (\overline{x_{e_1} \oplus x_{e_2} \oplus x_{e_6}}) \wedge (\overline{x_{e_6} \oplus x_{e_7}}) \wedge \overline{x_{e_7}}.\end{aligned}$$

Um eine Darstellung von $\mathcal{T}(F, \chi)$ in CNF zu erhalten, berechnen wir beispielhaft $F_{u,1} = \text{PARITY}_{u,\chi(u)} = \text{PARITY}_{u,1}$: Wegen $\{e \in E : e \sim u\} = \{e_1, e_4, e_5\}$ ist

$$\text{PARITY}_{u,1} = \bigwedge \left\{ \bigvee_{e \in \{e_1, e_4, e_5\}} x_e^{a(e)} : \bigoplus_{e \in \{e_1, e_4, e_5\}} \big(a(e) \oplus 1\big) \not\equiv 1 \right\} \tag{5.5}$$

$$= (x_{e_1}^1 \vee x_{e_4}^1 \vee x_{e_5}^1) \wedge (x_{e_1}^1 \vee x_{e_4}^0 \vee x_{e_5}^0) \wedge (x_{e_1}^0 \vee x_{e_4}^1 \vee x_{e_5}^0) \wedge (x_{e_1}^0 \vee x_{e_4}^0 \vee x_{e_5}^1)$$

$$= (x_{e_1} \vee x_{e_4} \vee x_{e_5}) \wedge (x_{e_1} \vee \overline{x_{e_4}} \vee \overline{x_{e_5}}) \wedge (\overline{x_{e_1}} \vee x_{e_4} \vee \overline{x_{e_5}}) \wedge (\overline{x_{e_1}} \vee \overline{x_{e_4}} \vee x_{e_5}). \tag{5.6}$$

Die Formel in (5.6) ist genau die CNF-Darstellung, die man auch aus der Wahrheitstafel der Booleschen Funktion $F_{u,1} = x_{e_1} \oplus x_{e_4} \oplus x_{e_5}$ durch Konjunktion von Maxtermen erhalten hätte.

**Beobachtung 5.0.4** ([Urq1987])**.** Anschaulich erhält man die obige Formel (5.5) in Beispiel 5.0.3 als Konjunktion aller Klauseln, die genau die Literale $\{x_e : e \sim u\}$ enthalten, sodass die Parität der Anzahl der konjugierten Literale in jeder Klausel verschieden von der Markierung $\chi(u)$ des Knoten $u$ ist.

Vermöge dieser anschaulichen Beobachtung ergibt sich folgende Notiz:

**Notiz 5.0.5** (Adaption aus [BW2001, GTT2018])**.** Hat der Graph $G$ Maximalgrad $d := \text{maxdeg}(G)$, so können wir jede der Formeln $F_{v,\chi(v)}$ (für $v \in V(G)$) durch die Vorschrift in (5.3) als $d$-CNF-Formel mit höchstens $2^{d-1}$ Klauseln schreiben. Folglich ist $\mathcal{T}(G,\chi)$ eine $d$-CNF-Formel mit höchstens $|V(G)| \cdot 2^{d-1}$ Klauseln über höchstens $\frac{nd}{2}$ Variablen.

Analoge Abschätzungen nach unten lassen sich durch Verwendung des Minimalgrades des Graphen erhalten.

Tseitin-Formeln stellen daher eine Möglichkeit dar, Graphen in aussagenlogische Formeln zu überführen und Eigenschaften des Graphen an die Formel zu „vererben“ – eine Eigenschaft, die wir im späteren Beweis des Haupttheorems 7.0.1, speziell in Proposition 7.1.11 ausnutzen werden, um obere Schranken für die Länge und den Platz von Resolutionswiderlegungen von Tseitin-Formeln zu zeigen. Die hierfür notwendige Unerfüllbarkeit von Tseitin-Formeln zeigen wir im nachfolgenden Theorem.

Um die Notationen an einigen Stellen im Folgenden einfacher zu halten, vereinbaren wir (wie in der Literatur üblich), dass wir sowohl eine Kante $e$ im Graph, als auch die zu ihr assoziierte Variabel $x_e$ mit dem Symbol $e$ bezeichnen und von Belegungen der Kanten reden können.

**Theorem 5.0.6.** *Ist $G = (V, E)$ ein zusammenhängender Graph, so ist die Tseitin-Formel $\mathcal{T}(G,\chi)$ unerfüllbar genau dann, wenn $\chi$ eine ungerade Markierung des Graphen $G$ ist.*

*Beweis.* Wir zeigen die Implikationen separat und folgen dabei den Beweisen von [Urq1987, Lemma 4.1], sowie [Tor1999], [Juk2012, Lemma 18.16] und [Hås2017, Lemma 2.1].

„$\longleftarrow$“: Es sei $\chi$ eine ungerade Markierung von $G$. Um einen Widerspruch zu erhalten, nehmen wir an, dass es eine erfüllende Belegung $\alpha\colon \{x_e : e \in E\} \to \{0,1\}$ für die Formel $\mathcal{T}(G,\chi)$ gibt. Insbesondere erfüllt $\alpha$ jede Teilformel $F_v$ (für $v \in V$) von $\mathcal{T}(G,\chi)$. We-

gen (5.2) gilt daher für alle Knoten $v \in V$, dass die Anzahl der zu $v$ inzidenten Kanten, die durch $\alpha$ die Belegung `1` erhalten, die gleiche Parität wie $\chi(v)$ hat. Folglich gilt

$$\sum_{v \in V} \sum_{\substack{u \in V: \\ uv \sim v}} \alpha(x_{uv}) \equiv \sum_{v \in V} \chi(v) \equiv 1 \pmod 2, \tag{5.7}$$

wobei wir hierbei die Belegung `1` mit der reellen Zahl 1 und `0` mit 0 identifiziert haben. Jedoch ist die linke Seite dieser Kongruenz gerade, da über jede Kante zweimal summiert wird (vgl. den Beweisschritt (I) des Handschlag-Lemmas 5.0.1), ein Widerspruch.

„$\longrightarrow$“: Wir zeigen statt dieser Aussage die Kontraposition: Ist $\chi$ eine gerade Markierung, so lässt sich eine erfüllende Belegung für die Formel $\mathcal{T}(G, \chi)$ konstruieren.

Ist $v \in V$ ein Knoten, so bezeichne für die folgende Konstruktion $P(v)$ die Paritätsbedingung (5.1) zum Knoten $v$, also $\bigoplus_{e \sim v} x_e \equiv \chi(v)$.

Ist nun $x_q$ die zu einer Kante $q = \{y, z\} \in E$ assoziierte Variable, so beobachten wir, dass wir äquivalente Darstellungen der Paritätsbedingungen $P(y)$ und $P(z)$ erhalten, indem wir in diesen beiden Gleichungen $x_q$ durch $1 - x_q$ ersetzen und $\chi(y)$ und $\chi(z)$ durch $1 - \chi(y)$ und $1 - \chi(z)$. Auf Graphen- bzw. Formelebene korrespondiert diese Aktion mit dem Austausch der Variable $x_q$ durch $\overline{x_q}$ und der Abänderung der Markierung $\chi$ an den Knoten $y$ und $z$ wie oben beschrieben (siehe Abbildung 5.2). Wir wollen künftig kurz hierfür sagen, dass die *Markierung von $y$ zu $z$ übertragen* wird[41].

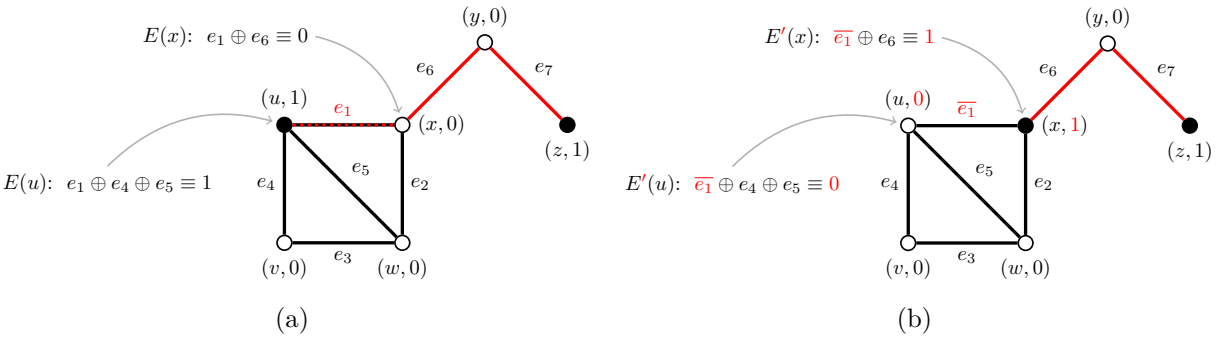



Abbildung 5.2.: Übertragung der Markierung von $u$ nach $x$ im (ladungsmodifizierten) Graph aus Beispiel 5.0.3 mit gerader Markierung. (a) Der Graph *vor* der Übertragung. In rot der Pfad zwischen den Knoten mit Markierung 1. Wir übertragen die Ladung entlang der rot-schwarz gestrichelten Kante. (b) Der Graph *nach* der Übertragung. Man beachte die rot markierten Veränderungen, insbesondere in den Paritätsbedingungen $P'(\cdot)$.

[41] Man denke dabei an wandernde *Ladungen* im Graphen, entsprechend der teilw. verwendeten Bezeichnung *charge* für eine Markierung.

Gibt es nun im Graphen zwei Knoten $y$ und $z$ mit $\chi(y) = \chi(z) = 1$, so gibt es auch einen Pfad $y = v_1, \ldots, v_n = z$ zwischen $y$ und $z$ (siehe erneut Abbildung 5.2). Falls wir also sukzessive die Markierung von $v_1$ auf $v_2$ ... auf $v_n$ übertragen, erhalten wir eine sat-äquivalente Tseitin-Formel (siehe für diesen Begriff Anhang C), der eine Graphen-Markierung mit 2 weniger Knoten mit ungerader Markierung als zu Beginn zugrunde liegt. Da die Anzahl der Knoten mit ungerader Markierung zu Beginn gerade war, lassen sich sukzessive solche Knotenpaarungen finden: Nach einer endlichen Anzahl an Wiederholungen des obigen Prozesses erhalten wir also für alle Konten $v \in V(G)$ Paritätsbedingungen $\bigoplus_{e \sim v} x_e \equiv 0$. Eine erfüllende Belegung hierzu erhalten wir, indem wir alle Literale auf `0` setzen. □

**Hinweis 5.0.7.** Im Folgenden werden wir nur daran interessiert sein, ob eine Markierung ungerade oder gerade ist. Alle ungeraden Markierungen induzieren aus Sicht der Resolution äquivalente Formeln.

**Historische Bemerkung 5.0.8** (Adaption und Erweiterung aus [Juk2012, GTT2018])**.** Tseitin-Formeln haben eine lange und erfolgreiche Geschichte in der Beweiskomplexität. Häufig wurden sie als schwere Formeln für Resolution benutzt (vergleiche die Diskussionen in Abschnitt 4.2) und dienen heute oft als Benchmark-Instanzen für SAT-Solver. Wir merken an, dass Resolutionswiderlegungen dieser Formeln große Weite haben müssen (siehe z. B. [Juk2012, § 18.7]). Beispielhaft führen wir an dieser Stelle einige Resultate an, die mit Tseitin-Formeln erzielt werden konnten:

- Die erste superpolynomielle untere Schranke für die Länge von (baumartigen und regulären) Resolutionswiderlegungen konnte in [Tse1968, Theorem 9] für Tseitin-Formeln erzielt werden. Wegen Notiz 3.5.18 ist HAKENs Resultat aus Theorem 4.2.2 jedoch stärker. Später konnten in [Urq1987, Sch1997] exponentielle untere Schranken für die Resolutionslänge von Tseitin-Formeln über *Expander-Graphen* gezeigt werden (für Expander-Graphen sei auf Fußnote 33 auf Seite 64 verwiesen).
- Des Weiteren konnten mit Tseitin-Formeln untere Schranken für die Komplexitätsmaße Weite in [BW2001] und Platz in [ET2001] gezeigt werden.
- In [BBI2016, BNT2013, Bec2017] konnten schließlich Zeit-Platz Tradeoffs mithilfe von Tseitin-Formeln gezeigt werden.

# KAPITEL 6

## Gittergraphen und Substitutionsformeln

**Definition 6.0.1** (Gittergraph)**.** Es seien $n, \ell \in \mathbb{N}$ natürliche Zahlen. Der *Gittergraph* $G_{n\times\ell}$ ist der Graph mit Knotenmenge

$$V(G_{n\times\ell}) = \{v_{i,j} \,:\, 1,\ldots,n,\, j = 1,\ldots,\ell\}$$

und der Kantenmenge $E(G_{n\times\ell})$, bestehend aus der Menge der *horizontalen*[42]*Kanten*

$$\Big\{\big\{v_{i,j}, v_{i,j+1}\big\} \,:\, i = 1,\ldots,n,\, j = 1,\ldots,\ell-1\Big\}$$

vereinigt mit der Menge der *vertikalen*[42] *Kanten*

$$\Big\{\big\{v_{i,j}, v_{i+1,j}\big\} \,:\, i = 1,\ldots,n-1,\, j = 1,\ldots,\ell\Big\}.$$

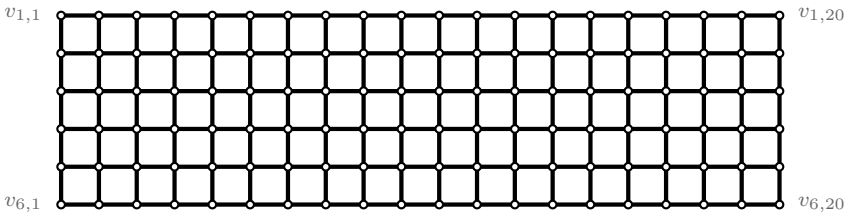



Abbildung 6.1.: Der Gittergraph $G_{6\times20}$ mit Beschriftung der „Eck“-Knoten.

Wir halten an dieser Stelle elementare Eigenschaften des Gittergraphen $G_{n\times\ell}$ fest, die uns im Beweis des Haupttheorems 7.0.1 von großem Nutzen sein werden.

[42]Wir wählen die „Ausrichtung der Nummerierung“ in Analogie zu der Nummerierung von Einträgen in Matrizen. Siehe Abbildung 6.1.

**Lemma 6.0.2.** *Es gelten die folgenden Aussagen:*

*(i)* $|V(G_{n\times\ell})| = n\ell$.

*(ii)* $|E(G_{n\times\ell})| = n(\ell-1)+(n-1)\ell = 2n\ell - n - \ell$. *Dies entspricht der Variablenanzahl in der zugehörigen Tseitin-Formel.*

*(iii)* $\operatorname{maxdeg}(G_{n\times\ell}) = 4$.

*(iv)* $\operatorname{mindeg}(G_{n\times\ell}) = 2$.

*Beweis.* (i), (iii), (iv) Diese Aussagen sind trivial (vgl. ggfs. Abbildung 6.1).

(ii) $G_{n\times\ell}$ hat $n(\ell-1)$ horizontale Kanten und $(n-1)\ell$ vertikale Kanten, folglich ist $|E(G_{n\times\ell})| = n(\ell-1)+(n-1)\ell = 2n\ell - n - \ell$. □

Für die Umsetzung der in [BNT2013] vorgeschlagenen Beweisvereinfachung des Papers [BBI2016] verwenden wir die Technik der Substitutionsformeln, insbesondere der XORification, die bereits erfolgreich in [Ben2009b] und [BN2011] eingesetzt wurde, um unerfüllbare Formeln „härter“ zum Widerlegen für Beweissysteme zu machen und dadurch z. B. untere Platzschranken zu erhalten.

Wir beschränken uns hauptsächlich auf den für uns relevanten Spezialfall der XORification, um die Notationen kurz und übersichtlich zu halten.

Die Idee der XORification [oder allgemeiner der Substitutionsformeln] einer Formel $F$ ist es, die Funktion $x' \oplus_2 x''$ [eine Boolesche Funktion $f_n$ der Arität $n$] für jede in der Formel $F$ auftauchende Variable $x$ zu substituieren und die entstehende Formel in konjunktive Normalform zu entwickeln, wobei $x'$ und $x''$ Variablen sind, die bisher nicht in der Formel $F$ auftraten.

Wir präzisieren diesen Gedanken durch folgende Definition, welche die Definitionen aus [Nor2013, § 2.4] und [Bon2018, Definition 8.2] auf unseren Spezialfall anpasst.

**Definition 6.0.3** (XORification)**.** Ist $F$ eine Boolesche Formel über der Variablenmenge $\operatorname{Var}(F) = \{x_1, \ldots, x_n\}$, so ist die XORification $F[\oplus_2]$ von $F$ eine CNF-Formel über den Variablen $\operatorname{Var}(F[\oplus_2]) = \{x_i', x_i'' : i = 1, \ldots, n\}$ (mit der Eigenschaft $\operatorname{Var}(F) \cap \operatorname{Var}(F[\oplus_2]) = \varnothing$), die aus $F$ entsteht, indem für alle $i = 1, \ldots, n$ alle Vorkommen der Variable $x_i$ durch die Formel $x_i' \oplus_2 x_i''$ ersetzt werden und die resultierende Formel anschließend in konjunktive Normalform entwickelt wird.

**Beispiel 6.0.4.** Wir veranschaulichen die XORification an einigen Beispielen, die im weiteren Verlauf eine Rolle spielen werden.

(i) Ist $x$ ein positives Literal, so ist die XORification von $x$ die Formel

$$x[\oplus_2] = (x' \vee x'') \wedge (\overline{x'} \vee \overline{x''}).$$

(ii) Unter Benutzung der De Morganschen Regeln ist

$$\overline{y' \oplus y''} = \overline{(y' \wedge \overline{y''}) \vee (\overline{y'} \wedge y'')} = (\overline{y'} \vee y'') \wedge (y' \vee \overline{y''}),$$

also ist die XORification eines negativen Literal $\overline{y}$ die Formel

$$\overline{y}[\oplus_2] = (\overline{y'} \vee y'') \wedge (y' \vee \overline{y''}).$$

(iii) Ist beispielsweise $C = x \vee \overline{y}$, so erhält man nach einer leichten Rechnung unter hauptsächlicher Ausnutzung der Distributivgesetze

$$\begin{aligned} C[\oplus_2] = (x \vee \overline{y})[\oplus_2] &= x[\oplus_2] \vee \overline{y}[\oplus_2] \\ &= \quad (x' \vee x'' \vee \overline{y'} \vee y'') \wedge (\overline{x'} \vee \overline{x''} \vee \overline{y'} \vee y'') \\ &\quad \wedge (x' \vee x'' \vee y' \vee \overline{y''}) \wedge (\overline{x'} \vee \overline{x''} \vee y' \vee \overline{y''}). \end{aligned}$$

**Bemerkung 6.0.5** ([Ben2009b])**.** Ist die ursprüngliche CNF-Formel $F$ nicht zu weit, dann vergrößert die XORification die Formel nicht zu viel, konkreter: Ist $F$ eine $k$-CNF-Formel mit $m$ Klauseln und $n$ Variablen, dann ist $F[\oplus_2]$ eine $2k$-CNF-Formel mit höchstens $2^k \cdot m$ Klauseln und $2n$ Variablen.

Des Weiteren erhält XORification die Unerfüllbarkeit einer Formel, d. h. $F$ ist unerfüllbar genau dann, wenn $F[\oplus_2]$ unerfüllbar ist.

**Definition & Konvention 6.0.6.** Ist $G = (V, E)$ mit $E = \{e_1, e_2, \ldots, e_k\}$, so bezeichnen wir mit $G^{\oplus_2}$ den Multigraphen mit Knotenmenge $V$ und Kanten(multi)menge

$$E^{\oplus_2} := \{\!\{e_1, e_1, e_2, e_2, \ldots, e_k, e_k\}\!\}.$$

Fortan schreiben wir statt *Multigraph* auch *Graph mit Doppelkanten* und lassen (falls dies wie oben durch das Superskript $\oplus_2$ notiert ist) auch Doppelkanten in Graphen zu.

Bezeichnen wir für $i = 1, \ldots, k$ eine der Doppelkanten in $G^{\oplus_2}$ mit $e_i'$ und die andere mit $e_i''$, so erhalten wir unmittelbar die folgende Proposition.

**Proposition 6.0.7.** *Es ist* $\mathcal{T}(G, \chi)[\oplus_2] = \mathcal{T}(G^{\oplus_2}, \chi)$.

**Beispiel 6.0.8.** Wir betrachten den sehr einfachen Graphen $G = (\{u, v\}, \{e\})$ und die Markierung $\chi$ gegeben durch $\chi(u) = 1$, $\chi(v) = 0$, wie in Abbildung 6.2 dargestellt.

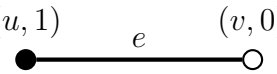



Abbildung 6.2.: Der Graph $G$ mit Markierung $\chi$ aus Beispiel 6.0.8. Die eingezeichneten Tupel $(v_0, \chi(v_0))$ liefern sowohl die Knotenbeschriftung als auch die Knotenmarkierung.

Die entsprechende Tseitin-Formel ergibt sich zu

$$\mathcal{T}(G, \chi) = e \wedge \overline{e}. \tag{6.1}$$

Unmittelbar aus Beispiel 6.0.4 (i) & (ii) ergibt sich

$$\mathcal{T}(G, \chi)[\oplus_2] = (e' \vee e'') \wedge (\overline{e'} \vee \overline{e''}) \wedge (\overline{e'} \vee e'') \wedge (e' \vee \overline{e''}).$$

Andererseits ist $G^{\oplus_2} = (\{u, v\}, \{\!\{e', e''\}\!\})$, wie in Abbildung 6.3 dargestellt.

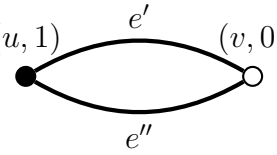



Abbildung 6.3.: Der Graph $G^{\oplus_2}$ mit Markierung $\chi$ aus Beispiel 6.0.8.

Für diesen Graphen erhalten wir

$$\begin{aligned} \mathrm{PARITY}_{u,1} &= (e' \vee e'') \wedge (\overline{e'} \vee \overline{e''}), \\ \mathrm{PARITY}_{v,0} &= (\overline{e'} \vee e'') \wedge (e' \vee \overline{e''}), \end{aligned}$$

also ist in der Tat $\mathcal{T}(G, \chi)[\oplus_2] = \mathcal{T}(G^{\oplus_2}, \chi)$.

Wir betrachten abschließend einige einfache Eigenschaften des Gittergraphen mit Doppelkanten, $G^{\oplus_2}_{n\times\ell}$, der von zentraler Wichtigkeit für den Beweis unseres Haupttheorems 7.0.1 sein wird (vgl. Abbildung 6.4).

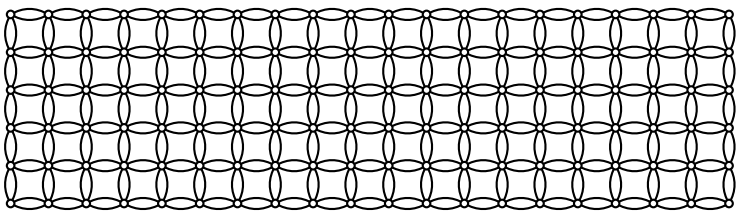


Abbildung 6.4.: Der Gittergraph $G^{\oplus_2}_{6\times 20}$.

**Lemma 6.0.9.** *Es gelten die folgenden Aussagen:*

*(i)* $|V(G^{\oplus_2}_{n\times\ell})| = n\ell$.
*(ii)* $|E(G^{\oplus_2}_{n\times\ell})| = 4n\ell - 2n - 2\ell$.
*(iii)* $\operatorname{maxdeg}(G^{\oplus_2}_{n\times\ell}) = 8$.
*(iv)* $\operatorname{mindeg}(G^{\oplus_2}_{n\times\ell}) = 4$.

Für unsere folgenden Diskussionen und insbesondere für den späteren Beweis des Tradeoff-Resultates in Haupttheorem 7.0.1 ist es nützlich, den Begriff *Einschränkung* (speziell von Substitutionsformeln) wie in [Bec2017, Definition 4.26] geringfügig auszudehnen und zu modifizieren. Wir erhalten dadurch eine signifikante Vereinfachung und klarere Formulierungen als in [BNT2013, § 3.1] und formalisieren Ideen aus [Ben2009b, Theorem 4.2].

Wir weisen an dieser Stelle wie [Bec2017] darauf hin, dass die nachfolgende Definition einer *Einschränkung im erweiterten Sinne* nicht Standard ist, insofern als dass sie nicht nur eine partielle Belegung von Variablen darstellt, sondern eine gewöhnliche Einschränkung mit einer Substitution kombiniert, da wir erlauben werden, Variablen anderen Variablen zuzuordnen. Bereits an dieser Stelle weisen wir jedoch darauf hin, dass die in Bemerkung 6.0.11 und Notiz 6.0.14 dargestellten Eigenschaften gewöhnlicher Einschränkungen erhalten werden (bzw. wie oben bereits angedeutet sogar verbessert werden).

**Definition 6.0.10** (Einschränkungen im erweiterten Sinne)**.** Eine *Einschränkung im erweiterten Sinne* ist eine Abbildung

$$\rho\colon \operatorname{Dom}(\rho) \to \{0,1\} \cup \mathcal{V},$$

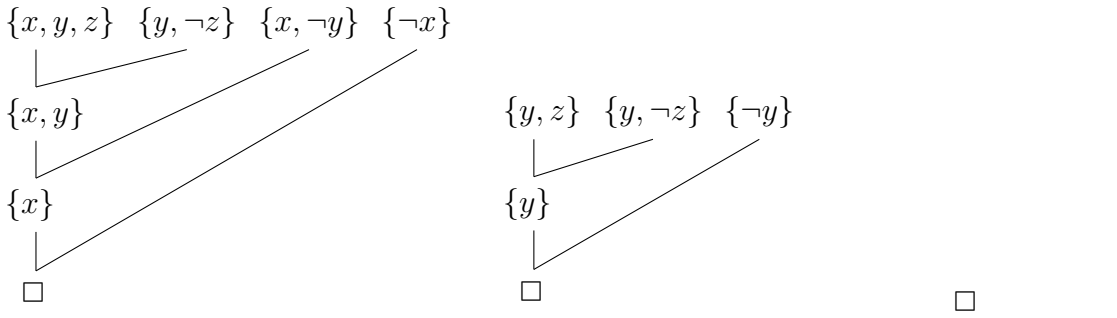
von einer endlichen Teilmenge $\operatorname{Dom}(\rho) \subset \mathcal{V}$ der Booleschen Variablen auf die Konstanten $0, 1$ und ggfs. andere Variablen aus $\mathcal{V}$.

Angelehnt an unsere bisherige Notation aus den Präliminarien in Kapitel 2 bezeichnen wir mit $\operatorname{Var}(\rho) := \operatorname{Dom}(\rho)$ den Definitionsbereich von $\rho$. Die *Einschränkung einer Formel $F$ im erweiterten Sinne* bezeichnen wir zu einer gegebenen Einschränkung $\rho$ im erweiterten Sinne erneut mit $F\restriction_\rho$ und definieren sie als diejenige Formel, die sich durch die Ersetzung der jeweiligen Variablen aus $\operatorname{Dom}(\rho)$ durch ihre Bilder unter $\rho$ und lokalen Vereinfachungen wie im Abschnitt „Wahrheitsbelegungen, Einschränkungen und Logik“ des Kapitels 2 beschrieben, ergibt.

Die *Einschränkung von Mengen von Formeln* oder *Folgen von Formeln* (z. B. von Resolutionsbeweisen) *im erweiterten Sinne* definieren wir als die Menge bzw. Folge der jeweiligen im erweiterten Sinne eingeschränkten Formeln.

Ähnlich wie [ST2013, Resolution Restricion Lemma (S. 40)] formulieren wir die folgende Bemerkung.

**Bemerkung 6.0.11.** Ist $\pi$ eine Resolutionswiderlegung einer Formel $F$ und $\rho$ eine Einschränkung im erweiterten Sinne in den Variablen $\mathrm{Var}(F)$, so ist $\pi{\restriction_\rho}$ ein *induzierter Beweis* von $F{\restriction_\rho}$ (bzw. kann in einen solchen – unter eventueller Benutzung der weakening rule – transformiert werden, ohne die Länge, Weite oder den Platz von $\pi$ zu vergrößern). Gegebenenfalls ist es möglich, die Beweisstruktur weiter zu vereinfachen. Beispielhaft haben wir dies in Abbildung 6.5, adaptiert aus [Gal2009], illustriert. Der formale Beweis der Aussage ist eine einfache Induktion über die Ableitungsschritte von $\pi$; siehe [Nor2008, S. 45].




(a) Der Beweis $\pi$. (b) Der Beweis $\pi{\restriction_{\{x=0\}}}$. (c) Der Beweis $\pi{\restriction_{\{x=1\}}}$.

Abbildung 6.5.: In (a) eine Resolutionswiderlegung $\pi$ der unerfüllbaren Formel $F := \big\{\{x, y, z\}, \{y, \neg z\}, \{x, \neg y\}, \{\neg x\}\big\}$. In (b) der Beweis $\pi{\restriction_{\{x=0\}}}$ der eingeschränkten Formel $F{\restriction_{\{x=0\}}} = \big\{\{y, z\}, \{y, \neg z\}, \{\neg y\}, \{1\}\big\}$. In (c) schließlich der sehr kurze Beweis $\pi{\restriction_{\{x=1\}}} = (\square)$ der offensichtlich unerfüllbaren Formel $F{\restriction_{\{x=1\}}} = \big\{\{1\}, \{y, \neg z\}, \{1\}, \square\big\}$.

Ist $F$ eine beliebige Formel, so erlauben uns die eben definierten Einschränkungen im erweiterten Sinne eine elegante Wahrscheinlichkeitsverteilung von Zufallseinschränkungen über den Variablen aus $F[\oplus_2]$ im erweiterten Sinne zu definieren:

**Definition 6.0.12** (Zufallseinschränkungen im erweiterten Sinne)**.** Es sei $F$ eine beliebige Formel. Eine *Zufallseinschränkung $\rho$ im erweiterten Sinne* ist eine Einschränkung im erweiterten Sinne mit Definitionsbereich $\mathrm{Dom}(\rho) = \mathrm{Var}(F[\oplus_2])$, wobei unabhängig für jede Variable $x \in \mathrm{Var}(F)$ eine der Variablen $x'$ oder $x''$ mit Wahrscheinlichkeit $\frac{1}{2}$ gewählt wird und auf `0` oder `1` mit Wahrscheinlichkeit $\frac{1}{2}$ gesetzt wird; die andere Variable wird so auf $x$ oder $\overline{x}$ gesetzt, dass die Gleichung $\rho(x') \oplus \rho(x'') \equiv x$ gilt.

Ist $\rho$ gemäß dieser Beschreibung zufallsverteilt, notieren wir dies mit $\rho \sim \mathcal{Z}$.

Wir illustrieren Zufallseinschränkungen im erweiterten Sinne im nachfolgenden Beispiel und in den Beispielen 6.0.16 sowie 6.0.18.

**Beispiel 6.0.13.** Wir betrachten wie in Beispiel 6.0.4 (i) die Formel $F := x$ und ihre XORification $x[\oplus_2] = (x' \vee x'') \wedge (\overline{x'} \vee \overline{x''})$. In Tabelle 6.1 geben wir alle vier möglichen Zufallseinschränkungen im erweiterten Sinne an. Beispielhaft betrachten wir die Zufallseinschränkung bei der $\rho(x') = \mathtt{0}$ gewählt wurde. Es soll nun die Gleichung $\rho(x') \oplus \rho(x'') \equiv 0 \oplus \rho(x'') \equiv x$ gelten, weshalb $\rho(x'') = x$ zu wählen ist.

Tabelle 6.1.: Die vier Zufallseinschränkungen für $x[\oplus_2]$ und ihre Wirkung.

| $\boldsymbol{\rho(x')}$ | $\boldsymbol{\rho(x'')}$ | $\boldsymbol{F[\oplus_2]\restriction_\rho = ((x' \vee x'') \wedge (\overline{x'} \vee \overline{x''}))\restriction_\rho}$ |
|---|---|---|
| $\mathtt{0}$ | $x$ | $(0 \vee x) \wedge (1 \vee \overline{x}) \equiv x \wedge 1 \equiv x = F$ |
| $\mathtt{1}$ | $\overline{x}$ | $(1 \vee \overline{x}) \wedge (0 \vee x) \equiv 1 \wedge x \equiv x = F$ |
| $x$ | $\mathtt{0}$ | $(x \vee 0) \wedge (\overline{x} \vee 1) \equiv x \wedge 1 \equiv x = F$ |
| $\overline{x}$ | $\mathtt{1}$ | $(\overline{x} \vee 1) \wedge (x \vee 0) \equiv 1 \wedge x \equiv x = F$ |

Wir beobachten, dass für alle vier möglichen Zufallseinschränkungen im erweiterten Sinne $F[\oplus_2]\restriction_\rho = F$ gilt.

Wir fassen die Beobachtung des vorangegangen Beispiels in folgender Notiz erneut auf.

**Notiz 6.0.14.** Ist $F$ eine Formel und $\rho$ eine Zufallseinschränkung im erweiterten Sinne, so gilt $F[\oplus_2]\restriction_\rho = F$.

Zudem halten wir als Ergänzung zu Notiz 6.0.14 fest, wie sich eine Zufallseinschränkung von Tseitin-Formeln im erweiterten Sinne aus graphentheoretischer Sicht bemerkbar macht (was einen weiteren Grund für unsere Wahl in Definition 6.0.12 darstellt):

**Intuition 6.0.15.** Ist $G$ ein zusammenhängender Graph und $\chi$ eine ungerade Markierung, sowie $\rho$ eine Zufallseinschränkung im erweiterten Sinne, so entspricht $\mathcal{T}(G, \chi)[\oplus_2]\restriction_\rho = \mathcal{T}(G^{\oplus_2}, \chi)\restriction_\rho$ der Tseitin-Formel, die sich durch Löschen jeweils einer der Doppelkanten des Graphen $G^{\oplus_2}$ und Umbenennung der anderen Kanten, sodass wir den Graphen $G$ zurückerhalten, ergibt; also $\mathcal{T}(G, \chi)$. Dies folgt unmittelbar aus Notiz 6.0.14.

Wir illustrieren dies an einem Beispiel auf der nächsten Seite.

**Beispiel 6.0.16.** Wir betrachten wie in Beispiel 6.0.8 die Tseitin-Formel $\mathcal{T}(G,\chi) = e \wedge \overline{e}$ zum Graphen und der Markierung aus Abbildung 6.2. In diesem Beispiel hatten wir gesehen, dass $\mathcal{T}(G,\chi)[\oplus_2] = \mathcal{T}(G^{\oplus_2},\chi) = (e' \vee e'') \wedge (\overline{e'} \vee \overline{e''}) \wedge (\overline{e'} \vee e'') \wedge (e' \vee \overline{e''})$ ist, wobei der Graph mit Doppelkanten in Abbildung 6.3 zu sehen war. Wir modifizieren Tabelle 6.1 aus Beispiel 6.0.13 in Tabelle 6.2, um die Wirkung aller vier möglichen Zufallseinschränkungen im erweiterten Sinne auf $\mathcal{T}(G^{\oplus_2},\chi)$ abzudecken. In der Tat ergibt sich $\mathcal{T}(G^{\oplus_2},\chi)\restriction_\rho = \mathcal{T}(G,\chi) = e \wedge \overline{e}$ in allen Fällen.

Beispielhaft betrachten wir die Zufallseinschränkung $\rho = \{e' = \mathtt{0},\, e'' = e\}$ aus der ersten Zeile der Tabelle 6.2: Die Anwendung von $\rho$ korrespondiert mit dem Löschen der Kante $e'$ und der Umbenennung der Kante $e''$ in $e$.

Tabelle 6.2.: Durch Zufallseinschränkungen im erweiterten Sinne erhalten wir aus der Tseitin-Formel zu einem Doppelgraphen die Tseitin-Formel zum Graphen mit Einfachkanten.

| $\rho(e')$ | $\rho(e'')$ | $\mathcal{T}(G^{\oplus_2},\chi)\restriction_\rho$ |
|---|---|---|
| $\mathtt{0}$ | $e$ | $(0 \vee e) \wedge (1 \vee \overline{e}) \wedge (1 \vee e) \wedge (0 \vee \overline{e}) \equiv e \wedge 1 \wedge 1 \wedge \overline{e} \equiv e \wedge \overline{e}$ |
| $\mathtt{1}$ | $\overline{e}$ | $(1 \vee \overline{e}) \wedge (0 \vee e) \wedge (0 \vee \overline{e}) \wedge (1 \vee e) \equiv 1 \wedge e \wedge \overline{e} \wedge 1 \equiv e \wedge \overline{e}$ |
| $e$ | $\mathtt{0}$ | $(e \vee 0) \wedge (\overline{e} \vee 1) \wedge (\overline{e} \vee 0) \wedge (e \vee 1) \equiv e \wedge 1 \wedge \overline{e} \wedge 1 \equiv e \wedge \overline{e}$ |
| $\overline{e}$ | $\mathtt{1}$ | $(\overline{e} \vee 1) \wedge (e \vee 0) \wedge (e \vee 1) \wedge (\overline{e} \vee 0) \equiv 1 \wedge e \wedge 1 \wedge \overline{e} \equiv e \wedge \overline{e}$ |

Der Vorteil an oben definierten Zufallseinschränkungen im erweiterten Sinne besteht zum einen in der klareren Formulierung in Notiz 6.0.14 im Vergleich zu [BNT2013, § 3.1]; zum anderen darin, dass eine solche Zufallseinschränkung Variablen auf die Konstanten $\mathtt{0}$ oder $\mathtt{1}$ mit konstanter Wahrscheinlichkeit abbildet und dadurch weite Klauseln mit hoher Wahrscheinlichkeit eliminiert. Diese Beobachtung formalisieren wir in nachfolgendem Satz, der uns im weiteren Verlauf der Arbeit einen wertvollen Dienst erweisen wird.

Der Satz und sein Beweis sind inspiriert durch die Ideen in [Ben2009b, Theorem 4.2], die 2001 aus persönlicher Kommunikation zwischen Eli Ben-Sasson, Michael Alekhnovich und Alexander Razborov entstanden sind. Wir folgen [Bec2017, Lemma 4.28].

**Satz 6.0.17** (Zufallseinschränkungen im erweiterten Sinn eliminieren weite Klauseln mit hoher Wahrscheinlichkeit)**.** *Es sei $F$ eine beliebige Formel und $C$ eine beliebige Klausel über* $\mathrm{Var}(F[\oplus_2])$*. Dann gilt für alle $K \in \mathbb{N}_0$ die Abschätzung*

$$\operatorname*{Prob}_{\rho\sim\mathcal{Z}}\left[\left|\mathrm{Var}(C\restriction_\rho)\right| \geq K\right] \leq \left(\frac{3}{4}\right)^K,$$

*wobei $\rho$ eine Zufallseinschränkung im erweiterten Sinne unter der in Definition 6.0.12 erörterten Wahrscheinlichkeitsverteilung $\mathcal{Z}$ ist.*

*Beweis.* Für $K = 0$ ist die Aussage trivial. Für $K \in \mathbb{N}$ zeigen wir den Satz durch eine einfache Fallunterscheidung:

Fall 1: Angenommen es ist $\left|\mathrm{Var}(C\restriction_\rho)\right| < K$ für alle zu betrachtenden Zufallseinschränkungen $\rho$ aus $\mathcal{Z}$. Dann ist $\operatorname*{Prob}_{\rho\sim\mathcal{Z}}\left[\left|\mathrm{Var}(C\restriction_\rho)\right| \geq K\right] = 0$, weshalb die Aussage trivialerweise gilt.

Fall 2: Angenommen Fall 1 tritt nicht ein, d. h. es gibt zumindest eine Zufallseinschränkung $\rho_*$ im erweiterten Sinne, sodass $\left|\mathrm{Var}(C\restriction_{\rho_*})\right| \geq K$ gilt. Es gibt also mindestens $K$ Variablen in der ursprünglichen Klausel $C$, welche durch eine Zufallsbelegung $\rho$ unabhängig belegt werden, auf die Konstante `0` mit Wahrscheinlichkeit mindestens $\frac{1}{4}$ und auf die Konstante `1` mit Wahrscheinlichkeit mindestens $\frac{1}{4}$. Weiter gibt es für jede der mindestens $K$ Variablen der ursprünglichen Klausel $C$ einen spezifischen Wert unter $\rho$, sodass $C\restriction_\rho = \texttt{True}$ und $\mathrm{Var}(C\restriction_\rho) = \varnothing$ gilt. Durch Ausnutzung der statistischen Unabhängigkeit aus Definition 6.0.12 erhalten wir

$$\operatorname*{Prob}_{\rho\sim\mathcal{Z}}\left[\left|\mathrm{Var}(C\restriction_\rho)\right| \geq K\right] \leq \operatorname*{Prob}_{\rho\sim\mathcal{Z}}\left[\mathrm{Var}(C\restriction_\rho) \neq \varnothing\right] \leq \left(\frac{3}{4}\right)^K. \qquad \square$$

**Beispiel 6.0.18.** Es sei $F$ eine beliebige Formel in den Variablen $\{e, f\}$. Es ist also $\mathrm{Var}(F[\oplus_2]) = \{e', e'', f', f''\}$. Wir betrachten die Klausel $C = (e' \vee e'' \vee \overline{f'})$. Die möglichen 16 Wirkungen der Zufallseinschränkungen im erweiterten Sinne auf $C$ zusammen mit der sich hieraus jeweils ergebenden Variablenanzahl haben wir in Tabelle 6.3 auf der nächsten Seite angegeben (man beachte, dass die Spalte $\rho(f'')$ lediglich der Vollständigkeit dient, weshalb wir sie eingegraut haben).

Wir beobachten dank der Tabelle:

$$\begin{aligned}
K = 0: &\quad \operatorname*{Prob}_{\rho\sim\mathcal{Z}}\left[\left|\mathrm{Var}(C\restriction_\rho)\right| \geq 0\right] = \frac{16}{16} = 1 \leq \left(\frac{3}{4}\right)^0 = 1,\\
K = 1: &\quad \operatorname*{Prob}_{\rho\sim\mathcal{Z}}\left[\left|\mathrm{Var}(C\restriction_\rho)\right| \geq 1\right] = \frac{6}{16} \leq \left(\frac{3}{4}\right)^1 = \frac{3}{4} = \frac{12}{16},\\
K = 2: &\quad \operatorname*{Prob}_{\rho\sim\mathcal{Z}}\left[\left|\mathrm{Var}(C\restriction_\rho)\right| \geq 2\right] = \frac{4}{16} \leq \left(\frac{3}{4}\right)^2 = \frac{9}{16},\\
K \geq 3: &\quad \operatorname*{Prob}_{\rho\sim\mathcal{Z}}\left[\left|\mathrm{Var}(C\restriction_\rho)\right| \geq 3\right] = 0 \leq \left(\frac{3}{4}\right)^3.
\end{aligned}$$

Beispielhaft haben wir also die Aussage von Satz 6.0.17 verifiziert.

Tabelle 6.3.: Die 16 möglichen Wirkungen von Zufallseinschränkungen im erweiterten Sinne auf die Klausel $C = (e' \vee e'' \vee \overline{f'})$.

| $\boldsymbol{\rho(e')}$ | $\boldsymbol{\rho(e'')}$ | $\boldsymbol{\rho(f')}$ | $\rho(f'')$ | $\boldsymbol{C\restriction_\rho}$ | $\boldsymbol{\lvert \mathbf{Var}\,(C\restriction_\rho)\rvert}$ |
|---|---|---|---|---|---|
| 0 | $e$ | 0 | $f$ | $0 \vee e \vee 1 \equiv 1$ | 0 |
| 1 | $\overline{e}$ | 0 | $f$ | $1 \vee \overline{e} \vee 1 \equiv 1$ | 0 |
| $e$ | 0 | 0 | $f$ | $e \vee 0 \vee 1 \equiv 1$ | 0 |
| $\overline{e}$ | 1 | 0 | $f$ | $\overline{e} \vee 1 \vee 1 \equiv 1$ | 0 |
| 0 | $e$ | 1 | $\overline{f}$ | $0 \vee e \vee 0 \equiv e$ | 1 |
| 1 | $\overline{e}$ | 1 | $\overline{f}$ | $1 \vee \overline{e} \vee 0 \equiv 1$ | 0 |
| $e$ | 0 | 1 | $\overline{f}$ | $e \vee 0 \vee 0 \equiv e$ | 1 |
| $\overline{e}$ | 1 | 1 | $\overline{f}$ | $\overline{e} \vee 1 \vee 0 \equiv 1$ | 0 |
| 0 | $e$ | $f$ | 0 | $0 \vee e \vee \overline{f} \equiv e \vee \overline{f}$ | 2 |
| 1 | $\overline{e}$ | $f$ | 0 | $1 \vee \overline{e} \vee \overline{f} \equiv 1$ | 0 |
| $e$ | 0 | $f$ | 0 | $e \vee 0 \vee \overline{f} \equiv e \vee \overline{f}$ | 2 |
| $\overline{e}$ | 1 | $f$ | 0 | $\overline{e} \vee 1 \vee \overline{f} \equiv 1$ | 0 |
| 0 | $e$ | $\overline{f}$ | 1 | $0 \vee e \vee f \equiv e \vee f$ | 2 |
| 1 | $\overline{e}$ | $\overline{f}$ | 1 | $1 \vee \overline{e} \vee f \equiv 1$ | 0 |
| $e$ | 0 | $\overline{f}$ | 1 | $e \vee 0 \vee f \equiv e \vee f$ | 2 |
| $\overline{e}$ | 1 | $\overline{f}$ | 1 | $\overline{e} \vee 1 \vee f \equiv 1$ | 0 |

# KAPITEL 7

## Das Zeit-Platz Tradeoff-Resultat

Wir haben in den letzten zwei Kapiteln genügend Grundlagen gelegt, um unser Tradeoff-Resultat im folgenden Haupttheorem 7.0.1 verständlich formulieren zu können. Unser Theorem modifiziert und verbessert dabei das Tradeoff-Resultat für das Beweissystem des *Polynomkalküls mit Resolution* (engl. *Polynomial Calculus Resolution*, PCR) aus [BNT2013, Theorem 4] aufbauend auf den Arbeiten [BBI2016] und [Bec2017]. Für eine genaue Beschreibung unseres Beitrags und der Modifikationen verweisen wir auf Abschnitt 1.2. Den Beweis der einzelnen Punkte des Haupttheorems teilen wir auf die Abschnitte 7.1 und 7.3 auf, um die Möglichkeit zu haben, unterstützende Definitionen und Beispiele darzustellen.

### Inhalt des Kapitels

Wir formulieren nun unser Haupttheorem – das Zeit-Platz Tradeoff-Theorem für superlinearen Platz.

**Haupttheorem 7.0.1** (Zeit-Platz Tradeoff für superlinearen Platz)**.** *Es existiert eine* explizit konstruierbare *Familie* $(F_{n,\ell})_{n\in\mathbb{N}}$ *von unerfüllbaren* 8-CNF*-Formeln für jedes* $\ell \in \mathbb{N}$ *mit* $8n^3 \leq \ell \leq 2^n$ *der Größe* $\mathrm{size}(F_{n,\ell}) = \Theta(n\ell)$ *in* $\Theta(n\ell)$ *Variablen mit den folgenden drei Eigenschaften:*

*(i) Die Formel* $F_{n,\ell}$ *besitzt eine Resolutionswiderlegung* $\pi'_{n,\ell}$ *in kurzer Länge*

$$\mathrm{Le}(\pi'_{n,\ell}) \leq 1024 \cdot n\ell \cdot 2^{2n} = \mathcal{O}(\ell^{\mathcal{O}(1)} 2^{\mathcal{O}(n)})$$

*und simultanem Platz*

$$\mathrm{Sp}(\pi'_{n,\ell}) \leq 4 \cdot 2^{2n} + 128n\ell + 9 = \mathcal{O}(2^{\mathcal{O}(n)} + \ell^{\mathcal{O}(1)}).$$

*(ii) Die Formel* $F_{n,\ell}$ *besitzt ebenfalls eine Resolutionswiderlegung* $\pi''_{n,\ell}$ *in kleinem Platz*

$$\mathrm{Sp}(\pi''_{n,\ell}) = \mathcal{O}(n \log \ell)$$

*und simultaner Länge*

$$\mathrm{Le}(\pi''_{n,\ell}) = 2^{\mathcal{O}(n \log \ell)}.$$

*(iii) Für jede Resolutionswiderlegung* $\pi_{n,\ell}$ *von* $F_{n,\ell}$ *gilt für* $n \geq 5$ *der Zeit-Platz Tradeoff*

$$\mathrm{Le}(\pi_{n,\ell}) \geq \left( \frac{2^{\Omega(n)}}{\mathrm{Sp}(\pi_{n,\ell})} \right)^{\Omega\left(\frac{\log_2 \log_2 n}{\log_2 \log_2 \log_2 n}\right)}.$$

*Dabei bezeichnen wir obige Familie von Formeln als* explizit konstruierbar*, falls es einen Polynomialzeit-Algorithmus gibt, der bei Eingabe* $1^n$ *das* $n$*-te Mitglied der Familie (für eine gegeben Funktion* $\ell(n)$*) als Ausgabe liefert.*

Wir verweisen auf Intuition 7.3.1 für eine anschauliche (wenn auch stark vereinfachende) Erklärung dieses Resultates. Für ein ähnliches Tradeoff-Resultat für reguläre Resolution verweisen wir auf [BBI2016, Theorem 30]. Wir halten wie in Abschnitt 1.1 fest, dass damit folgendes Problem 7.0.2 gelöst ist: Es muss *zwingend* ein starker Tradeoff stattfinden.

**Problem 7.0.2** (Gelöst in [BBI2016] – Ursprünglich gestellt in [Nor2013] als Open Problem 17)**.** Es sei $F$ eine beliebige unerfüllbare $k$-CNF-Formel mit $n$ Klauseln. Angenommen es gilt $L := \mathrm{Le}(F \vdash_{\mathfrak{R}} \square) = \mathrm{poly}(n)$. Impliziert dies, dass es eine Resolutionswiderlegung

$\pi\colon F \vdash_{\mathfrak{R}} \square$ in Platz $\mathrm{Sp}(\pi) = \mathcal{O}(n)$ und Länge $\mathrm{Le}(\pi) = \mathrm{poly}(L)$ gibt? Oder gibt es Formeln, die für alle Resolutionswiderlegungen $\pi$ von $F$ einen Tradeoff der beiden Komplexitätsmaße aufweisen, selbst wenn $\mathrm{Sp}(\pi) \geq \mathrm{size}(F)$ (d. h. *superlinearer Platz*) erlaubt ist?

Wir beweisen das Hauptresultat im Folgenden in vier Teilen: Konstruktion der Formeln, gefolgt von den Beweisen zu den Aussagen (i)–(iii).

*Konstruktion der Formeln.* Für gegebenes $n \in \mathbb{N}$ und $\ell \in \mathbb{N}$ mit $8n^3 \leq \ell \leq 2^n$ definieren wir $F_{n,\ell}$ als die Tseitin-Formel zum $(n \times \ell)$-Gittergraphen und einer ungeraden Markierung $\chi$ mit anschließender XORification, in Zeichen

$$F_{n,\ell} := \mathcal{T}(G_{n\times\ell}, \chi)[\oplus_2].$$

Wir bemerken, dass dies nach Proposition 6.0.7 äquivalent dazu ist, die Formel $F_{n,\ell}$ über die Tseitin-Formel zum $(n \times \ell)$-Gittergraphen mit Doppelkanten, $G^{\oplus_2}_{n\times\ell}$, zu definieren. Offensichtlich ist

$$\mathrm{Wi}(C) \leq \mathrm{maxdeg}(G^{\oplus_2}_{n\times\ell}) = 8 \quad \text{für alle } C \in F_{n,\ell}, \tag{7.1}$$

wie sofort aus Lemma 6.0.9 (iii) zusammen mit Notiz 5.0.5 folgt. Ebenso folgt erneut mit Lemma 6.0.9 (i) und Notiz 5.0.5, dass es höchstens

$$|V(G^{\oplus_2}_{n\times\ell})| \cdot 2^{\mathrm{maxdeg}(G^{\oplus_2}_{n\times\ell})-1} = n\ell \cdot 2^7 \quad \text{Klauseln} \tag{7.2}$$

in $F_{n,\ell}$ gibt. Gemäß der Definition der Größe einer Formel aus den Präliminarien erhalten wir aus den Gleichungen (7.1) und (7.2) sofort

$$\mathrm{size}(F_{n,\ell}) = \sum_{C\in F_{n,\ell}} \mathrm{Wi}(C) \leq 8 \cdot 2^7 \cdot n\ell.$$

Analog mit Lemma 6.0.9 (iv) ist

$$\mathrm{Wi}(C) \geq \mathrm{mindeg}(G^{\oplus_2}_{n\times\ell}) = 4 \quad \text{für alle } C \in F_{n,\ell},$$

und es gibt mindestens

$$|V(G^{\oplus_2}_{n\times\ell})| \cdot 2^{\mathrm{mindeg}(G^{\oplus_2}_{n\times\ell})-1} = n\ell \cdot 2^3 \quad \text{Klauseln}$$

in $F_{n,\ell}$, sodass wir

$$\mathrm{size}(F_{n,\ell}) = \sum_{C\in F_{n,\ell}} \mathrm{Wi}(C) \geq 4 \cdot 2^3 \cdot n\ell$$

erhalten. Folglich ist $\text{size}(F_{n,\ell}) = \Theta(n\ell)$. Aus Lemma 6.0.9 (ii) folgt, dass $F_{n,\ell}$ eine Formel in $\Theta(n\ell)$ Variablen ist. □

*Beweis der oberen Schranken in (i).* Wir zeigen diese obere Schranken in einem ausführlichen Beweis im Abschnitt 7.1 in Theorem 7.1.13. □

*Beweis der oberen Schranken in (ii).* Wir zeigen diese in Abschnitt 7.2. □

*Beweis des Tradeoff-Resultates in (iii).* Wir zeigen diesen Hauptbeweis gesondert in Abschnitt 7.3 in mehreren Schritten. Diese kulminieren im Spezialfall von Theorem 7.3.28, welches das Tradeoff-Resultat darstellt. □

## 7.1 Beweis des Haupttheorems 7.0.1 (i)

Wir legen an dieser Stelle den groben Plan des Abschnitts dar: Zu Beginn betrachten wir einige Eigenschaften gewurzelter Bäume bzw. baumartiger Resolutionsbeweise. Wir fahren fort mit der Definition der Grapheigenschaft *Schnittweite* und untersuchen diese an unserem Gittergraph mit Doppelkanten $G^{\oplus_2}_{n\times\ell}$. Schließlich führen wir die bis dahin gemachten Untersuchungen in Proposition 7.1.11 zusammen, die es ermöglicht, Resolutions-Ressourcen zur Widerlegung einer Tseitin-Formel mithilfe von Eigenschaften des ihr zugrundeliegenden Graphen abzuschätzen. Der Abschnitt gipfelt in Theorem 7.1.13, das den ersten Teil unseres Haupttheorems – den *high space upper bound* ([Bec2017, S. 46]) – vermöge der gerade erwähnten Proposition zeigt.

**Definition 7.1.1.** Die *Höhe* (engl. *height*) $\text{Ht}(T)$ eines gewurzelten Baumes $T$ ist die Anzahl der Kanten auf dem längsten Pfad zwischen seiner Wurzel und einem Blatt des Baumes (wobei wir festlegen, dass ein gewurzelter Baum, der nur aus einem Knoten besteht, Höhe 0 hat).

Durch die graphenartige Repräsentation eines Resolutionsbeweises, wie in Abschnitt 3.5.2 erörtert, zusammen mit obiger Definition erhalten wir das Komplexitätsmaß der *Tiefe* eines Resolutionsbeweises.

**Definition 7.1.2** (Adaption aus [GTT2018])**.** Die *Tiefe* (engl. *depth*) $\text{De}(\pi)$ eines (graphenartigen) Resolutionsbeweises $\pi$ ist die Höhe seines Beweisgraphen, in Zeichen $\text{De}(\pi) = \text{Ht}(G_\pi)$.

Inspiriert von Esteban und Toráns Paper [ET2001], in welchem gezeigt wurde, dass für jede unerfüllbare CNF-Formel $F$ in $n$ Variablen eine (baumartige) Resolutionswiderlegung $\pi$ mit $\text{Le}(\pi) \leq 2^{n+1} - 1$ und $\text{Sp}(\pi) \leq n + 1$ existiert, beweisen wir folgendes Lemma

(ohne jedoch auf sog. *Pebblings*, die wir bereits in Fußnote 38 respektive auf Seite 71 erwähnt haben, zurückgreifen zu müssen). Der folgende Beweis orientiert sich an [Bon2018, Theorem 3.1], [BBI2016, Observation 55] und [ET2001, Theorem 2.1].

**Lemma 7.1.3.** *Jeder baumartige Resolutionsbeweis $\pi$ mit Tiefe $\mathrm{De}(\pi) = r$ hat eine Länge von höchstens $2^{r+1} - 1$ und benötigt höchstens Platz $r + 1$.*

*Beweis.* (I) Der zum baumartigen Resolutionsbeweis assoziierte Beweisgraph ist ein gewurzelter Binärbaum mit Höhe $r$ (wobei wir die letzte Klausel des Resolutionsbeweises als Wurzel des Baumes wählen). Maximal können wir also eine Wurzel, zwei Söhne der Wurzel, vier Enkel der Wurzel, 8 Großenkel der Wurzel, usw. haben. Die Gesamtzahl der Knoten lässt sich also durch die geometrische Reihe nach oben abschätzen:

$$1 + 2 + 4 + 8 + \cdots + 2^r = \sum_{k=0}^{r} 2^k = \frac{2^{r+1} - 1}{2 - 1} = 2^{r+1} - 1.$$

Alternativ zeigt man die Behauptung leicht induktiv.

(II) Wir zeigen die Behauptung durch vollständige Induktion nach der Tiefe $r$. Im Induktionsanfang, $r = 1$, besteht der Beweisgraph aus der leeren Klausel und ihrer zweier Elternklauseln. Dass eine solche Resolutionswiderlegung Platz 2 benötigt, haben wir bereits in Beispiel 4.1.10 gezeigt.

Wir nehmen also an, dass die Induktionshypothese für eine feste Tiefe $r \in \mathbb{N}$ gelte. Für den Induktionsschritt betrachten wir daher einen Beweis $\pi$ mit Tiefe $r+1$. Angenommen, $C$ ist die letzte Klausel im Beweis $\pi$ (d. h. die Wurzel des Baumes $G_\pi$) und $A$ und $B$ sind die Elternklauseln dieser letzten Klausel. Dann sind die Beweise $\pi_A = (\mathfrak{M}_1, \ldots, \mathfrak{M}_\ell)$ und $\pi_B = (\mathfrak{M}_{\ell+1}, \ldots, \mathfrak{M}_t)$, die jeweils $A$ bzw. $B$ herleiten, von Tiefe $r$, benutzen also höchstens Platz $r+1$ nach Induktionshypothese. Wir leiten die Klausel $C$ wie folgt her: Zunächst leiten wir die Klausel $A$ wie in $\pi_A$ unter Benutzung von Platz maximal $r+1$ her; $A$ ist die letzte für weitere Herleitungen benötigte Klausel dieses Teilbeweises, also löschen wir alle anderen Klauseln dieses Teilbeweises; mit lediglich $A$ im Speicher leiten wir $B$ unter Benutzung von Zusatzplatz höchstens $r+1$ her; schließlich löschen wir bis auf $A$ und $B$ alles aus dem Speicher und resolvieren abschließend $A$ mit $B$, wodurch wir $C$ erhalten. Diese gesamte Herleitung benötigt maximal Platz $r + 2$:

$$\pi = \Big(\mathfrak{M}_1, \ldots, \mathfrak{M}_\ell, \{A\}, \{A\} \cup \mathfrak{M}_{\ell+1}, \ldots, \{A\} \cup \mathfrak{M}_t, \{A, B\}, \{A, B, C\}\Big).$$

Nach Definition 4.1.4 ist also $\mathrm{Sp}(\pi) \leq r + 2$, was zu zeigen war. □

Wir fahren fort mit der Definition der *Schnittweite* eines Graphen und betrachten diese an einem Beispiel.

**Definition 7.1.4** (Adaption aus [KS1993])**.** Es sei $G = (V, E)$ ein Graph.

(i) Eine *Knotennummerierung* des Graphen $G$ ist eine Bijektion $\sigma\colon V \to \{1, \ldots, |V|\}$.

(ii) Die *Schnittweite* (engl. *cut width*) $\mathrm{cw}_\sigma(G)$ des Graphen $G$ *bzgl. einer Knotennummerierung* $\sigma$ ist definiert als

$$\mathrm{cw}_\sigma(G) := \max_{1 \le t < |V|} \mathrm{cw}_\sigma(t) := \max_{1 \le t < |V|} \left|\{uv \in E \,:\, \sigma(u) \le t < \sigma(v)\}\right|.$$

(iii) Die *Schnittweite* $\mathrm{cw}(G)$ *des Graphen* $G$ ist das Minimum der Schnittweiten über alle Knotennummerierungen $\sigma$, in Zeichen

$$\mathrm{cw}(G) := \min_\sigma \mathrm{cw}_\sigma(G).$$

Vermöge einer Knotennummerierung wird eine natürliche *Knotensortierung*, gegeben durch $(\sigma^{-1}(1), \ldots, \sigma^{-1}(|V|))$, induziert. Wir werden daher die Begriffe austauschbar verwenden.

**Bemerkung 7.1.5** (Korrigiert aus [BBI2016])**.** Die Schnittweite eines Graphen $G$ ist somit die kleinste natürliche Zahl $W$, sodass es eine Knotensortierung $(v_1, \ldots, v_n)$ gibt, sodass für alle $1 \le t < n$ nicht mehr als $W$ Kanten den Schnitt $(\{v_1, \ldots, v_t\}, \{v_{t+1}, \ldots, v_n\})$ kreuzen.

**Beispiel 7.1.6** ([MPDP2010])**.** Wir betrachten den linken Graph $G = (V, E)$ aus Abbildung 7.1. Auf der rechten Seite der Abbildung betrachten wir eine feste Knotennummerierung $\sigma$ des Graphen und visualisieren diese durch eine lineare Anordnung der Knoten von $G$. Die Schnittweiten $\mathrm{cw}_\sigma(t)$ bezüglich der Knotennummern $t$ sind durch die grauen Strich-Punkt-Linien mit zugehörigen Werten angedeutet. Beispielsweise ist

$$\begin{aligned}\mathrm{cw}_\sigma(2) &= |\{uv \in E \,:\, \sigma(u) \le 2 < \sigma(v)\}| \\ &= |\{CB, AB, AE, AD\}| = 4.\end{aligned}$$

Wir sehen mit Hilfe der Abbildung 7.1, dass $\mathrm{cw}_\sigma(G) = 5$ ist. Hieraus folgt die obere Abschätzung $\mathrm{cw}(G) \le 5$.

Wie die Knotensortierung $(C, B, A, D, E, F)$ zeigt, ist diese Abschätzung jedoch nicht optimal, sondern lässt sich zu $\mathrm{cw}(G) \le 3$ verbessern (siehe hierzu Abbildung 7.4 auf Seite 103). Es ist nicht schwer zu sehen, dass in der Tat $\mathrm{cw}(G) = 3$ ist.

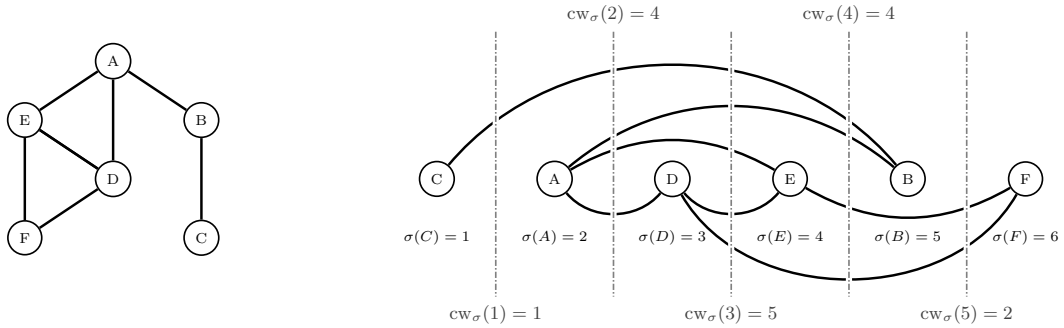



Abbildung 7.1.: Ein Graph $G$ zusammen mit einer festen Knotennummerierung $\sigma$ und Schnittweiten $\mathrm{cw}_\sigma(t)$.

**Lemma 7.1.7.** *Es sei $n \in \mathbb{N}$ und $\ell \in \mathbb{N}$ mit $8n^3 \leq \ell \leq 2^n$. Dann gelten die folgenden zwei Aussagen:*

*(i) Der Gittergraph $G_{n\times\ell}$ hat Schnittweite $\mathrm{cw}(G_{n\times\ell}) \leq n+1$.*

*(ii) Der Gittergraph mit Doppelkanten $G^{\oplus 2}_{n\times\ell}$ hat Schnittweite $\mathrm{cw}(G^{\oplus 2}_{n\times\ell}) \leq 2(n+1)$.*

*Beweis.* (i) Wir betrachten die Knotennummerierung $\sigma\colon V(G_{n\times\ell}) \to \{1,\dots,n\ell\}$ aus Abbildung 7.2 gegeben durch

$$\sigma(v_{i,j}) := (j-1)n + i \quad \text{für } i = 1,\dots,n,\ j = 1,\dots,\ell, \tag{7.3}$$

die $n \ll \ell$ für alle $n \in \mathbb{N}$ ausnutzt: Deswegen ist

$$\mathrm{cw}_\sigma(t) \in \{2,\dots,n+1\} \quad \text{für alle } t = 1,\dots,n\ell-1.$$

Also ist $\mathrm{cw}_\sigma(G_{n\times\ell}) = n+1$, woraus direkt $\mathrm{cw}(G_{n\times\ell}) \leq n+1$ folgt. Vgl. erneut Abbildung 7.2 für eine graphische Unterstützung dieser einfachen Argumente.

(ii) Durch „Umwandeln“ von Einzelkanten in $G_{n\times\ell}$ zu Doppelkanten in $G^{\oplus 2}_{n\times\ell}$ kreuzen höchstens doppelt so viele Kanten wie zuvor jeden der zu betrachtenden Schnitte $(\{\sigma^{-1}(1),\dots,\sigma^{-1}(t)\},\{\sigma^{-1}(t+1),\dots,\sigma^{-1}(n\ell)\})$ für $t = 1,\dots,n\ell-1$, falls die gleiche Knotennummerierung $\sigma$ zugrunde liegt. Die Aussage folgt also dank Bemerkung 7.1.5 sofort aus (i). □

Bevor wir mit der Formulierung der in der Einleitung erwähnten Proposition beginnen, führen wir eine Notation ein, die uns den Beweis der Proposition erleichtern wird.

**Notation 7.1.8.** Ist $C = (x_1^{a(1)} \vee \dots \vee x_k^{a(k)})$ eine Klausel, so bezeichnen wir mit $\neg\dot{C}$ die *zur Klausel $\neg C$ assoziierte Belegung* $\{x_1 = \overline{a(1)},\dots,x_k = \overline{a(k)}\}$.

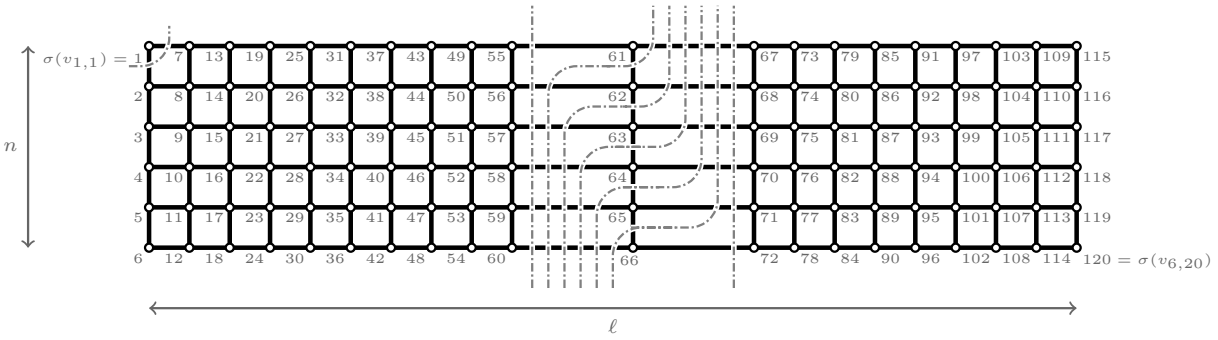



Abbildung 7.2.: Der Gittergraph $G_{6\times 20}$ zusammen mit der Knotennummerierung $\sigma$ aus (7.3). Mit Strich-Punkt-Linien sind die Schnittweiten $\mathrm{cw}_{G_{6\times 20}}(t)$ für $t = 1, 60, 61, 62, 63, 64, 65, 66$ eingezeichnet, die zwischen 2 und 7 variieren.

**Beispiel 7.1.9.** Ist

$$C = (x_1 \vee x_2 \vee \overline{x_3}),$$

so ist

$$\neg C = \overline{x_1} \wedge \overline{x_2} \wedge x_3,$$

also

$$\neg\dot{C} = \{x_1 = \mathtt{0},\, x_2 = \mathtt{0},\, x_3 = \mathtt{1}\}.$$

**Bemerkung 7.1.10.** Ist $C$ eine Klausel, so ist $C\restriction_{\neg\dot{C}} = \mathtt{False}$.

*Beweis.* Es ist $a(i)^{\overline{a(i)}} \in \{\mathtt{0}^1, \mathtt{1}^0\} = \{\mathtt{0}, \overline{\mathtt{1}}\} = \{\mathtt{0}\}$ für alle $i$. □

Wir formulieren nun – wie angekündigt – eine Proposition, die es uns erlaubt, die benötigten Resolutionsschritte und den benötigten Platzbedarf für die Widerlegung einer Tseitin-Formel nach oben mithilfe von Eigenschaften des ihr zugrundeliegenden Graphen abzuschätzen.

**Proposition 7.1.11.** *Es sei $G$ ein zusammenhängender Graph und $\chi$ eine ungerade Markierung dieses Graphen. Dann gibt es eine Resolutionswiderlegung $\pi'$ der Tseitin-Formel $\mathcal{T}(G,\chi)$ mit*

$$\mathrm{Le}(\pi') \leq (|V(G)| - 1) \cdot 2^{\mathrm{maxdeg}(G)+\mathrm{cw}(G)}$$

*und*

$$\mathrm{Sp}(\pi') \leq 2^{\mathrm{cw}(G)} + \mathrm{maxdeg}(G) + 1 + |\mathcal{A}|,$$

*wobei $\mathcal{A} := \{C \,:\, C \in \mathcal{T}(G,m)\}$ die Menge der Axiome ist.*

Unser Beweis ist inspiriert durch die Konstruktion in [Bus2006, § 5], mit der gezeigt

wurde, dass es Resolutionswiderlegungen der Negation des kombinatorischen Prinzips „zwei Pfade in einem $(d_0 \times n)$-Gittergraph, die diagonal gegenüberliegende Eck-Knoten verbinden, müssen sich in einem Knoten schneiden“ mit polynomieller Größe und konstanter Weite gibt, falls $d_0$ eine feste Konstante ist.

Wir adaptieren diese Methode, wie in [BBI2016, Lemma 56] (verbessern dabei aber einige Nachlässigkeiten in der Nummerierung der Mengen $\mathfrak{C}_i$ der Autoren sowie weitere Ungenauigkeiten bezüglich der verwendeten Notation – wodurch wir die Abschätzungen stark verbessern konnten), um obige Proposition zu zeigen.

Vor dem formalen (sehr langen) Beweis geben wir eine intuitive Erklärung der Beweisidee, basierend auf [Bec2017, Bec2018].

**Intuition 7.1.12** (Beweisidee von Proposition 7.1.11)**.** Unsere Hauptidee im Beweis wird sein, dass wir die Resolutionswiderlegung eine algebraische Widerlegung der Tseitin-Formeln nachahmen lassen: Addiert man alle Paritätsbedingungen (5.1) der Tseitin-Formel im Körper $\mathbb{F}_2$, so erhält man den Widerspruch $0 \stackrel{!}{=} 1$, wie wir im Beweis des Theorems 5.0.6 in Gleichung (5.7) gesehen haben. In dieser Sichtweise ist eine Tseitin-Formel ein System von inkonsistenten, linearen $\mathbb{F}_2$-Gleichungen.

Die algebraische Widerlegung basiert auf *dynamischer Programmierung*: Sie „verwaltet“ für die Widerlegung der Tseitin-Formel Klauseln, die zu den Paritätsbedingungen (7.4) der modulo 2 addierten Cut-Set-Kantenbelegungen des Graphen korrespondieren. Dann „bewegt“ sie diese Schnitte langsam von einem Ende des Graphen zum anderen Ende, wodurch wir in der Reihenfolge der durch den Zeugen der Schnittweite vorgegebenen Knotensortierung (also spaltenweise) die entsprechenden Paritätsgleichungen schrittweise nacheinander addieren und die „Zwischenresultate“ speichern.

Unser Beweis wird zeigen, dass Resolution fähig ist, diese eben beschriebene algebraische Widerlegung (mit einem gewissen Blow-up) zu simulieren: Anschaulich und vereinfachend gesprochen korrespondiert eine $\mathbb{F}_2$-Gleichung in $k$ Variablen zu $2^k$ Klauseln. Resolution kann eine Addition von zwei Gleichungen mit je $k$ Variablen in $2^{\mathcal{O}(k)}$ Schritten simulieren.

*Formaler Beweis von Proposition 7.1.11.* Um die Notationen übersichtlich zu halten definieren wir $n := |V(G)|$, $d := \operatorname{maxdeg}(G)$ und $W := \operatorname{cw}(G)$. Wir konstruieren im Folgenden eine Folge von $n$ *Klausel-Konfigurationsmengen* $\mathfrak{C}_i$, sodass die folgenden vier Eigenschaften gelten:

1. $\mathfrak{C}_1 \subset \mathcal{A}$,
2. $|\mathfrak{C}_i| \leq 2^{W-1}$ für alle $i = 1, \ldots, n$,
3. $\mathfrak{C}_n = \{\square\}$,
4. für alle $i = 1, \ldots, n-1$ gilt: Sind die Klauseln der Konfigurationsmenge $\mathfrak{C}_i$ *und*

die Axiome aus $\mathcal{A}$ im Speicher, so können wir jede Klausel $C \in \mathfrak{C}_{i+1}$ mit höchstens $2^{d+1}$ Resolutionsschritten und höchstens $d+1$ zusätzlichen „Speicher-Zellen“ herleiten.

Dem ungeduldigen Leser empfehlen wir, Abbildung 7.6 für die Hauptidee des Beweises zu studieren. Wir fahren an dieser Stelle mit der Konstruktion der oben erwähnten Mengen fort. Hierzu sei $(v_1, \ldots, v_n)$ die Knotensortierung, die bezeugt, dass $\mathrm{cw}(G) = W$ ist. Wir definieren für $1 \leq i \leq n$ die Hilfs-Kantenmenge $E_i$ durch

$$E_i := \mathrm{CS}(\{v_1, \ldots, v_i\}, \{v_{i+1}, \ldots, v_n\}),$$

also als die Menge der (höchstens $W$) Kanten, die den Schnitt $(\{v_1, \ldots, v_i\}, \{v_{i+1}, \ldots, v_n\})$ kreuzen. Ähnlich wie in Kapitel 5 über Tseitin-Formeln (vgl. insb. Notiz 5.0.5) sei die Konfigurationsmenge $\mathfrak{C}_i$ definiert als die $2^{|E_i|-1}$ Klauseln, deren Konjunktion semantisch äquivalent zur Paritätsbedingung

$$\bigoplus_{e \in E_i} x_e \equiv \bigoplus_{1 \leq j \leq i} \chi(v_j) \tag{7.4}$$

ist. Wir zeigen zunächst, dass diese Definition die Bedingungen 1–4 in der Tat erfüllt:

„ad 1“: Es ist $E_1 = \mathrm{CS}(\{v_1\}, \{v_2, \ldots, v_n\}) = \{e \in E : e \sim v_1\}$, wie in Abbildung 7.3 skizziert. Die Paritätsbedingung (7.4) ergibt sich also zu

$$\bigoplus_{e \sim v_1} \equiv \chi(v_1).$$

Wie wir in Kapitel 5 gesehen haben, ist $\mathrm{PARITY}_{v_1, \chi(v_1)}$ die CNF-Darstellung dieser Bedingung. $\mathrm{PARITY}_{v_1, \chi(v_1)}$ ist aber natürlich eine Teilmenge der Axiommenge $\mathcal{A}$.

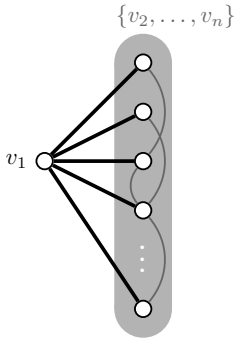



Abbildung 7.3.: Symbolische Skizze für den Schnitt $(\{v_1\}, \{v_2, \ldots, v_n\})$. Die Kanten, die den Schnitt kreuzen, sind dick in schwarz eingezeichnet. Eventuell vorhandene weitere Kanten zwischen den Knoten $v_2, \ldots, v_n$ dünn in grau.

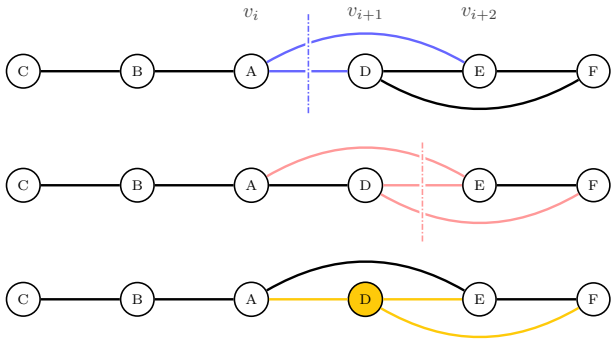



Abbildung 7.4.: Wir betrachten beispielhaft die in Beispiel 7.1.6 angesprochene Knotensortierung $(C, B, A, D, E, F)$ des Graphen aus Abbildung 7.1. In blau haben wir die Kanten aus $\mathrm{CS}(\{C, B, A\}, \{D, E, F\})$ und in rot die aus $\mathrm{CS}(\{C, B, A, D\}, \{E, F\})$ markiert. Man sieht, dass die symmetrische Differenz dieser beiden Mengen gerade die in gelb markierten Kanten sind. Dies sind diejenigen, die zum Knoten $D$ inzident sind.

„ad 2“: Zunächst halten wir fest, dass $|E_i| \leq W$ für alle $i = 1, \ldots, n$ nach Definition der Schnittweite $W$ ist. Weiter ist wegen $\bigoplus_{1 \leq j \leq i} \chi(v_j) \in \{0, 1\}$ die Paritätsbedingung (7.4) von vergleichbarer Form zur Paritätsbedingung (5.1). Folglich sind unsere Argumentationen aus Kapitel 5 (speziell die der Notiz 5.0.5) auch hier gültig und wir können schließen, dass (7.4) mit höchstens $2^{|E_i|-1} \leq 2^{W-1}$ Klauseln „codiert“ werden kann.

„ad 3“: Wegen $E_n = \mathrm{CS}(\{v_1, \ldots, v_n\}, \varnothing) = \{(u, v) \in E : u \in V, v \in \varnothing\} = \varnothing$ ist diese Bedingung offensichtlich.

„ad 4“: Wir zeigen diese Behauptung in zwei Schritten: Zunächst verifizieren wir sie für alle $1 \leq i < n - 1$ und schließlich für $i = n - 1$.

(I) Wir fixieren $1 \leq i < n - 1$ und beobachten zunächst, dass sich wegen

$$E_i = \mathrm{CS}(\{v_1, \ldots, v_i\}, \{v_{i+1}, v_{i+2} \ldots, v_n\})$$
$$E_{i+1} = \mathrm{CS}(\{v_1, \ldots, v_i, v_{i+1}\}, \{v_{i+2}, \ldots, v_n\})$$

die symmetrische Differenz von $E_i$ und $E_{i+1}$ zu

$$E_i \triangle E_{i+1} = \{e \in E : e \sim v_{i+1}\} \tag{7.5}$$

ergibt (für ein illustratives unterstützendes Beispiel hierzu siehe Abbildung 7.4). Schließlich definieren wir $A_{i+1} \subset \mathcal{A}$ als die Menge der Axiome, die zum Knoten $v_{i+1}$ assoziiert sind, d. h.

$$A_{i+1} := \mathrm{PARITY}_{v_{i+1}, \chi(v_{i+1})}. \tag{7.6}$$

In $\mathbb{F}_2$ können wir die zu $A_{i+1}$ und $\mathfrak{C}_i$ assoziierten Paritätsbedingungen addieren:

$$\begin{aligned} &\text{Gleichung assoziiert mit } A_{i+1}: && \bigoplus_{e\sim v_{i+1}} x_e \equiv \chi(v_{i+1}). \\ &\text{Gleichung assoziiert mit } \mathfrak{C}_i: && \bigoplus_{e\in E_i} x_e \equiv \bigoplus_{1\le j\le i} \chi(v_j). \qquad (7.7) \\ \hline &\oplus_2\text{-Summe in } \mathbb{F}_2: && \bigoplus_{e\in E_{i+1}} x_e \equiv \bigoplus_{1\le j\le i+1} \chi(v_j). \qquad (7.8) \end{aligned}$$

Dabei haben wir in (7.8) die einführende Beobachtung zur symmetrischen Differenz (7.5) benutzt (vergleiche auch das Venn-Diagramm in Abbildung 7.5 zur folgenden formalen Rechnung[43]):

$$\bigoplus_{e\sim v_{i+1}} x_e \oplus \bigoplus_{e\in E_i} x_e \equiv \bigoplus_{e\in E_i \triangle \{e\sim v_{i+1}\}} x_e \overset{(7.5)}{\equiv} \bigoplus_{e\in E_i\triangle E_i\triangle E_{i+1}} x_e \equiv \bigoplus_{e\in E_{i+1}} x_e.$$

Wir beobachten, dass das Ergebnis (7.8) gerade die zu $\mathfrak{C}_{i+1}$ assoziierte Paritäts-

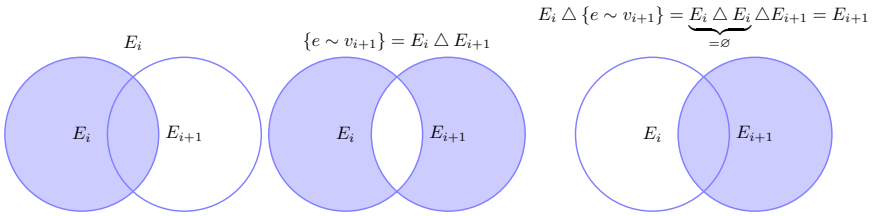



Abbildung 7.5.: Verdeutlichungen der formalen Rechnungen im Beweis durch unterstützende Venn-Diagramme.

bedingung ist. Wir folgern das für den Beweis entscheidende Resultat:

$$\boxed{\{A_{i+1},\ \mathfrak{C}_i\} \ \vDash\ \mathfrak{C}_{i+1}.} \qquad (7.9)$$

Dank Korollar 3.5.10 aus Lees Theorem ist klar, dass wir diesen „Schritt“ (7.9) von der Menge $\mathfrak{C}_i$ zur Menge $\mathfrak{C}_{i+1}$ (unter Benutzung der Axiome) durch Resolution erreichen können (wir zeigen dies weiter unten ohne Benutzung des Korollars).

[43] Wir erinnern hierzu daran, dass sich die symmetrische Differenz zweier Mengen $A$ und $B$ auch durch ein XOR zweier Prädikate ausdrücken lässt:

$$A \triangle B = \{x : (x \in A) \oplus_2 (x \in B)\}.$$

Unser Ziel wird es nun sein, zu zeigen, dass wir jede Klausel aus $\mathfrak{C}_{i+1}$ in Zeit höchstens $2^{d+1}$ und Platz maximal $d+1$ herleiten können, was beweisen würde, dass es möglich ist, obigen „Schritt“ (7.9) in Zeit $|\mathfrak{C}_{i+1}| \cdot 2^{d+1}$ und Arbeitsplatz $d+1$ (zusätzlich zum Platz, den wir brauchen, um (Teile der) Mengen $\mathfrak{C}_i$, $\mathfrak{C}_{i+1}$ und $\mathcal{A}$ im Speicher zu halten) zu gehen.

Dazu fixieren wir eine beliebige Klausel $C \in \mathfrak{C}_{i+1}$. Dank (7.9) gilt insbesondere

$$A_{i+1},\ \mathfrak{C}_i \ \vDash\ C,$$

mit Bemerkung 7.1.10 folglich

$$A_{i+1}{\restriction}_{\neg\dot{C}},\ \mathfrak{C}_i{\restriction}_{\neg\dot{C}} \vDash \mathsf{0}.$$

Dank der Widerlegungsvollständigkeit des Resolutionskalküls 3.5.5 (c) haben wir

$$A_{i+1}{\restriction}_{\neg\dot{C}},\ \mathfrak{C}_i{\restriction}_{\neg\dot{C}} \ \vdash_{\mathfrak{R}}\ \square. \tag{7.10}$$

Es ist bekannt, dass die Ableitung (7.10) in die Ableitung

$$A_{i+1},\ \mathfrak{C}_{i+1} \ \vdash_{\mathfrak{R}}\ C$$

umgewandelt werden kann, wobei wir den gleichen Beweisgraphen nutzen können (aber womöglich weakening Schritte einfügen müssen, die aber am Ende der Konstruktion mit Theorem 3.5.11 eliminiert werden können, ohne den berechneten Zeit- oder Platzaufwand zu verändern).

Da die Menge $\mathfrak{C}_{i+1}$ logisch äquivalent zur (für diese Menge angepassten) Paritätsbedingung (7.4), d. h. zu $\bigoplus_{e\in E_{i+1}} x_e \equiv \bigoplus_{1\leq j\leq i+1} \chi(v_j)$ ist, belegt $\neg\dot{C}$ für jedes $C \in \mathfrak{C}_{i+1}$ nach Beobachtung 5.0.4 alle zur Menge $E_{i+1}$ assoziierten Variablen. Aus den Gleichungen (7.5, 7.7, 7.8) sowie Abbildung 7.5 ergibt sich also $\mathrm{Var}(\mathfrak{C}_i{\restriction}_{\neg\dot{C}}) \subset \{x_e : e \sim v_{i+1}\}$. Eine Widerlegung wie (7.10) hat also höchstens $d$ verschiedene Variablen. Gemäß Theorem 3.5.16 existiert also eine *baumartige* Resolutionswiderlegung $\pi_{(\mathrm{I})}$ von (7.10). Diese kann wegen der gerade getätigten Argumentation maximale Tiefe $\mathrm{De}(\pi_{(\mathrm{I})}) \leq d$ haben. Wegen Lemma 7.1.3 (bzw. dem eingangs auf Seite 96 zitierten Resultat aus [ET2001]) benötigt die Widerlegung $\pi_{(\mathrm{I})}$ also höchstens $2^{d+1}$ Resolutionsschritte und $d+1$ Platz, was unsere Behauptung zeigt.

Da es nach Bedingung 2 maximal $2^{W-1}$ Klauseln in $\mathfrak{C}_{i+1}$ geben kann, können wir alle Klauseln in $\mathfrak{C}_{i+1}$ nacheinander herleiten, wobei wir für jede Klauselher-

leitung höchstens Zeit $2^{d+1}$ benötigen, den jeweils benötigten Platz von maximal $d+1$ aber wiederbenutzen können und nur $2^{W-1}$ zusätzlichen Speicher benötigen, um die neuen Klauseln zu speichern. Schließlich können wir alle Speicherzellen leeren, die benutzt wurden, um Klauseln der Menge $\mathfrak{C}_i$ zu speichern. Insgesamt haben wir nie mehr als $2 \cdot 2^{W-1} + d + 1 + |\mathcal{A}|$ Speicher benutzt (zweimal Platz, um die Klauseln der Mengen $\mathfrak{C}_i$ und $\mathfrak{C}_{i+1}$ zu speichern, zusätzlicher Arbeitsspeicher für den Übergang zur neuen Menge, plus Platz für die Axiome). Wir verweisen für eine Illustration dieser Hauptidee auf Abbildung 7.6.

(II) Schließlich zeigen wir Behauptung 4 auch für den Fall $i = n - 1$, d. h. wir zeigen, dass es möglich ist, aus $\mathfrak{C}_{n-1}$ und den Axiomen einen Widerspruch in Form der leeren Klausel herzuleiten (also $\mathfrak{C}_n$ herzuleiten) und dabei nur $2^{d+1}$ Resolutionsschritte und $d + 1$ Zusatzplatz zu benötigen.

Wir beobachten zunächst, dass

$$E_{n-1} = \mathrm{CS}(\{v_1, \ldots, v_{n-1}\}, \{v_n\}) = \{e \in E \,:\, e \sim v_n\}$$

ist (vgl. hierzu erneut Abbildung 7.3). Folglich ist $\mathfrak{C}_{n-1}$ gemäß (7.4) die Menge der Klauseln, deren Konjunktion semantisch äquivalent zur Paritätsbedingung

$$\bigoplus_{e \sim v_n} x_e \equiv \bigoplus_{1 \leq j \leq n-1} \chi(v_j) \tag{7.11}$$

ist. Gleichzeitig ist wegen (7.6) jedoch

$$A_n \equiv \mathrm{PARITY}_{v_n, \chi(v_n)},$$

was nach Kapitel 5 gerade die CNF-Darstellung der Paritätsbedingung

$$\bigoplus_{e \sim v_n} x_e \equiv \chi(v_n) \tag{7.12}$$

darstellt. Da nach Voraussetzung $\chi$ allerdings eine ungerade Markierung war, d. h.

$$\bigoplus_{1 \leq j \leq n} \chi(v_j) \equiv \bigoplus_{1 \leq j \leq n-1} \chi(v_j) \oplus \chi(v_n) \equiv 1$$

gilt, müssen $\bigoplus_{1 \leq j \leq n-1} \chi(v_j)$ und $\chi(v_n)$ von verschiedener Parität sein. Ergo sind wegen (7.11) und (7.12) die Klauselmengen $\mathfrak{C}_{n-1}$ und $A_n$ widersprüchlich.

Wie in Notiz 5.0.5 festgehalten, ist $A_n$ eine Formel in den höchstens $d$ Variablen $\{x_e \,:\, e \sim v_n\}$. Wie bereits im Beweisabschnitt „ad 2“ erörtert, ist auch $\mathfrak{C}_{n-1}$

eine Klauselmenge in höchstens eben diesen $d$ Variablen. Gemäß Theorem 3.5.16 existiert also eine *baumartige* Resolutionswiderlegung $\pi_{(\mathrm{II})}$ von $A_n \cup \mathfrak{C}_{n-1}$. Diese kann wegen der gerade getätigten Argumentation maximale Tiefe $\mathrm{De}(\pi_{(\mathrm{II})}) \leq d$ haben. Wegen Lemma 7.1.3 benötigt die Widerlegung $\pi_{(\mathrm{II})}$ also höchstens $2^{d+1}$ Resolutionsschritte und $d+1$ Platz, was unsere Behauptung zeigt.

Damit sind die Behauptungen 1 – 4 nachgewiesen.

Wir schließen den Beweis nach Nachweis der vier Behauptungen nun ab: Der Resolutionsbeweis benötigt $n-1$ Zwischenphasen wie (7.9), um von $\mathfrak{C}_1$ die Endmenge $\mathfrak{C}_n$ zu erreichen (vgl. hierzu auch Abbildung 7.6). Jede der Phasen kann ausgeführt werden in Zeit $\leq 2^{W-1}2^{d+1} = 2^{d+W}$. Zudem hat jede Phase nur $\leq 2 \cdot 2^{W-1} + d + 1$ Arbeitsplatz benötigt. Um die Axiome dauerhaft im Speicher zu behalten benötigen wir nach Notiz 5.0.5 Platz $|\mathcal{A}| = n2^{d-1}$.

Insgesamt benötigen wir also Zeit[44] höchstens $(n-1)2^{d+W}$ und Platz höchstens $2^W + d + 1 + |\mathcal{A}|$, wie behauptet. □

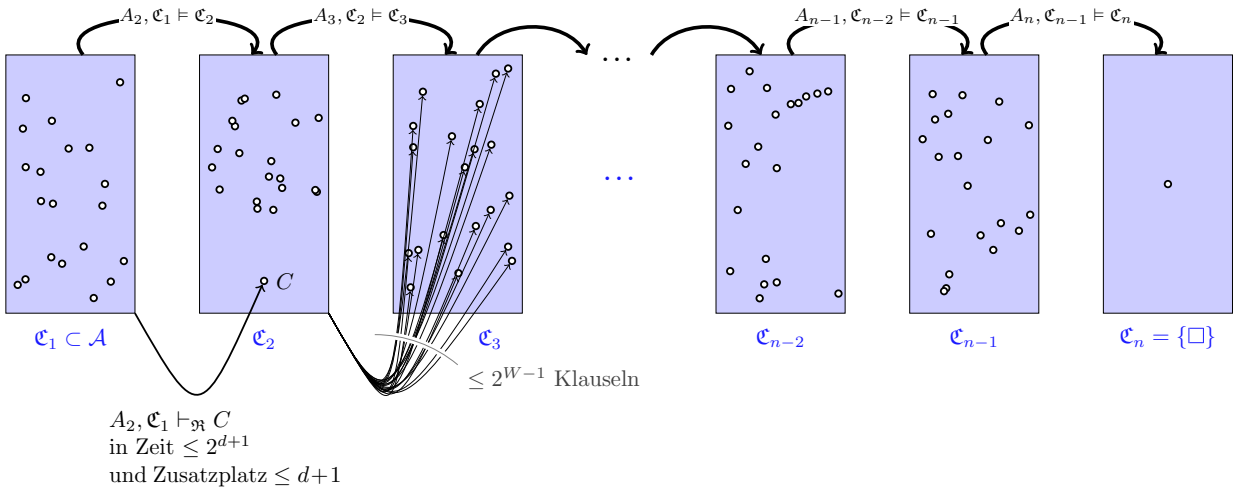



Abbildung 7.6.: Skizze der Hauptbeweisidee. Die blauen Rechtecke symbolisieren die $n$ Mengen $\mathfrak{C}_1, \ldots, \mathfrak{C}_n$, die Punkte in den Rechtecken symbolisieren die enthaltenen Klauseln. Exemplarisch haben wir die Ableitung einer Klausel $C \in \mathfrak{C}_2$ mitsamt des benötigten Zeit und Platzbedarfs skizziert. In jeder neuen Menge $\mathfrak{C}_i$ gibt es höchstens $2^{W-1}$ Klauseln, was wir durch das „Bündel“ an Pfeilen ebenfalls angedeutet haben. Man beachte im Beweis: $A_1 = \mathfrak{C}_1$.

Wir haben nun genug Fortschritte erzielt, um den ersten Teil unseres Hauptheorems beweisen zu können:

**Theorem 7.1.13** (Obere Schranke (i) des Hauptheorems 7.0.1)**.** *Die Familie* $(F_{n,\ell})_{n\in\mathbb{N}}$ *mit* $8n^3 \leq \ell \leq 2^n$ *aus Hauptheorem 7.0.1 besitzt eine Resolutionswiderlegung* $\pi'_{n,\ell}$ *in kurzer*

[44] Unser verbesserter Beweis liefert sogar, dass wir Zeit höchstens $(n-2)2^{d+W} + 2^{d+1}$ benötigen.

*Länge*

$$\mathrm{Le}(\pi'_{n,\ell}) \leq 1024 \cdot n\ell \cdot 2^{2n} = \mathcal{O}(\ell^{\mathcal{O}(1)} 2^{\mathcal{O}(n)})$$

*und simultanem Platz*

$$\mathrm{Sp}(\pi'_{n,\ell}) \leq 4 \cdot 2^{2n} + 128n\ell + 9 = \mathcal{O}(2^{\mathcal{O}(n)} + \ell^{\mathcal{O}(1)}).$$

*Beweis.* Nach Proposition 7.1.11 und der Konstruktion von $F_{n,\ell}$ über den Graph $G^{\oplus 2}_{n\times\ell}$ gilt unter Ausnutzung der Lemmata 6.0.9 und 7.1.7 (ii):

$$\begin{aligned}\mathrm{Le}(\pi'_{n,\ell}) &\leq (|V(G^{\oplus 2}_{n\times\ell})| - 1) \cdot 2^{\mathrm{maxdeg}(G^{\oplus 2}_{n\times\ell}) + \mathrm{cw}(G^{\oplus 2}_{n\times\ell})} \\ &\leq (n\ell - 1) \cdot 2^{8+2(n+1)} \\ &= (n\ell - 1) \cdot 2^{10} \cdot 2^{2n}.\end{aligned}$$

Unter großzügiger Abschätzung nach oben erhalten wir wie behauptet

$$\mathrm{Le}(\pi'_{n,\ell}) \leq 1024 \cdot n\ell \cdot 2^{2n}.$$

Wegen der Voraussetzung $n \ll \ell$ ist $1024 \cdot n\ell \cdot 2^{2n} \leq 1024 \cdot \ell^2 \cdot 2^{2n} = \mathcal{O}(\ell^{\mathcal{O}(1)} 2^{\mathcal{O}(n)})$.

Ebenso erhalten wir unmittelbar aus der Konstruktion der Resolutionswiderlegung $\pi'_{n,\ell}$ in Proposition 7.1.11 zusammen mit Notiz 5.0.5 sowie den Lemmata 6.0.9 und 7.1.7 (ii):

$$\begin{aligned}\mathrm{Sp}(\pi'_{n,\ell}) &\leq 2^{\mathrm{cw}(G^{\oplus 2}_{n\times\ell})} + \mathrm{maxdeg}(G^{\oplus 2}_{n\times\ell}) + 1 + |\{C \,:\, C \in F_{n,\ell}\}| \\ &\leq 2^{2(n+1)} + 8 + 1 + n\ell \cdot 2^7 \\ &= 4 \cdot 2^{2n} + 128n\ell + 9.\end{aligned}$$

Analog wie oben ergibt sich die Abschätzung in der $\mathcal{O}$-Notation hieraus. □

**Bemerkungen 7.1.14.**

- Durch eine leichte Überlegung ist es möglich zu zeigen, dass die Widerlegungen in Proposition 7.1.11 bzw. Theorem 7.1.13 durch reguläre Resolutionswiderlegungen durchgeführt werden können: In jeder Phase ist die Resolutionsableitung baumartig und jede Phase operiert auf disjunkten Mengen von Variablen. Vgl. [BBI2016, Bec2017].
- Es ist möglich, die Ideen in den Beweisen mithilfe des Konzeptes der *carving width* zu generalisieren. Hierfür verweisen wir auf die Hauptresultate des Papers [AR2011] bzw. auf [Bec2017, S. 48f.] für eine Zusammenstellung der wichtigsten Definitionen und Resultate.
- Unsere in diesem Abschnitt gezeigten Resultate zeigen zunächst lediglich, dass Reso-

lutionsbeweise mit den angesprochenen Ressourcenverbräuchen *existieren*. Dies liefert a priori noch keine *algorithmische* Methode, diese Beweise auch *tatsächlich* zu finden. In [Bec2017, Claim 3.64] wurde jedoch erörtert, dass der auf *Zweigbreite* (engl. *branch-width*) basierende Algorithmus [AR2011] von Alekhnovich und Razborov eine effiziente Implementierung der obigen Resultate darstellt, der mit den behaupteten Ressourcenverbräuchen bis auf Konstanten im Exponenten übereinstimmt. Der im Paper [ACL+2014] vorgestellte Algorithmus erreicht Ähnliches. Für weiter führende Diskussionen dieser Art verweisen wir auf [Bec2017, S. 50].

## 7.2 Beweis des Haupttheorems 7.0.1 (ii)

Wir haben im letzten Abschnitt dynamische Programmierung ausgenutzt, um kurze Resolutionswiderlegungen unter Ausnutzung von viel Platz zu erhalten (siehe Intuition 7.1.12). Wir benutzen nun das Paradigma *Divide and Conquer*, um eine Resolutionswiderlegung in kleinem Platz zu erhalten.

**Proposition 7.2.1.** *Es sei $G$ ein zusammenhängender Graph und $\chi$ eine ungerade Markierung dieses Graphen. Dann gibt es eine baumartige Resolutionswiderlegung $\pi''$ der Tseitin-Formel $\mathcal{T}(G, \chi)$ mit*

$$\mathrm{Sp}(\pi'') \leq \mathrm{cw}(G) \cdot \lceil \log_2(|V(G)|) \rceil + 1$$

*und*

$$\mathrm{Le}(\pi') \leq 2^{\mathrm{Sp}(\pi'')} - 1 \leq 2^{\mathrm{cw}(G) \cdot \lceil \log_2(|V(G)|) \rceil + 1} - 1.$$

Ähnlich wie in Intuition 7.1.12 zur dynamischen Programmierung lässt sich auch hier, wie oben angedeutet, eine Parallele zu einem Algorithmen-Paradigma festhalten.

**Intuition 7.2.2** (Beweisidee von Proposition 7.2.1)**.** Intuitiv denke man bei nachfolgendem Beweis an einen `DPLL`-Algorithmus, der in *Divide and Conquer*-Manier den Graphen wiederholt zweiteilt (auf der Suche nach der ungerade markierten Komponente). Nach jedem Schnitt wird genau ein Graphenteil „unerfüllbar“ sein. Nach „genügend“ Schnitten erhalten wir einen Widerspruch (isolieren also einen Knoten, der für einen Widerspruch sorgt). Siehe auch [Nor2016b].

Aufgrund der Ähnlichkeit zum vorigen Abschnitt und dem Fokus dieser Arbeit auf dem Beweis des Tradeoff-Resultates in Abschnitt 7.3, begnügen wir uns an dieser Stelle mit einer detaillierten Beweisskizze, angelehnt an [BBI2016, Lemma 57].

*Beweisskizze von Proposition 7.2.1.* Um auch in diesem Beweis die Notationen übersichtlich zu halten, definieren wir $n := |V(G)|$, $W := \mathrm{cw}(G)$ und $\mathcal{T} := \mathcal{T}(G, \chi)$.

Wir werden zeigen, dass, falls $n$ eine 2er-Potenz ist, die Formel $\mathcal{T}$ eine baumartige Resolutionswiderlegung $\pi''$ mit Tiefe $\mathrm{De}(\pi'') = W \log_2 n$ hat. Mittels Lemma 7.1.3 folgt hieraus die Behauptung.

Angenommen also es gilt $n = 2^p$ für ein $p \in \mathbb{N}$. Wir zeigen die Behauptung induktiv:

„$p = 1$“: In Abbildung 6.2 haben wir einen ungerade markierten Graphen $G$ mit 2 Knoten gesehen. Gemäß (6.1) ist eine korrespondierende Tseitin-Formel von der Bauart $\mathcal{T} = e \wedge \overline{e}$. Offensichtlich gilt $\mathrm{cw}(G) = 1$. In Beispiel 4.1.10 haben wir gesehen, wie eine solche Formel in Platz 2 und Länge 3 widerlegt werden kann. Dies liegt in den behaupteten Schranken von $1 \cdot \lceil \log_2(2) \rceil + 1 = 2$ für Platz und $2^2 - 1 = 3$ für Länge.

„$p - 1 \curvearrowright p$“: Angenommen, die Induktionshypothese gilt für Graphen der Größe $n/2$. Wie im Beweis von Proposition 7.1.11 sei $E_{n/2} := \mathrm{CS}(\{v_1, \ldots, v_{n/2}\}, \{v_{n/2+1}, v_{i+2} \ldots, v_n\})$. Es bezeichne $C$ eine beliebige Klausel, die alle zu $E_{n/2}$ assoziierten Variablen enthält. Dann kann $\mathcal{T}\restriction_{\neg \dot{C}}$ geschrieben werden als ein Paar disjunkter Tseitin-Formeln, jeweils eine von jeder Seite des Schnittes. Eine der korrespondierenden Teilgraphen mit $n/2$ Knoten ist gerade markiert, der andere ungerade markiert (weshalb die zugehörige Tseitin-Formel nach Theorem 5.0.6 unerfüllbar ist). Trivialerweise hat jeder der Teilgraphen eine Schnittweite von höchstens $W$. Nach Induktionshypothese hat also $\mathcal{T}\restriction_{\neg \dot{C}}$ eine baumartige Resolutionswiderlegung mit Tiefe höchstens $W \cdot \log_2(n/2) = W(\log_2 n - 1)$. Diese kann zu einer baumartigen Resolutionsableitung von $C$ aus $\mathcal{T}$ angehoben werden ohne die Tiefe der Ableitung zu vergrößern.

Man kann leicht zeigen, dass es eine baumartige Resolutionswiderlegung mit Tiefe $|E_{n/2}| \leq W$ gibt, welche die Menge aller solcher oben spezifizierten Klauseln $C$ als Axiommenge benutzt. In der erhaltenen Resolutionswiderlegung ersetzen wir alle Vorkommen dieser Axiomklauseln durch ihre baumartigen Resolutionsherleitungen aus $\mathcal{T}$ mit Tiefe höchstens $W(\log_2 n - 1)$. Folglich erhalten wir eine baumartige Resolutionswiderlegung von $\mathcal{T}$ mit Tiefe höchstens $W \log_2 n$. □

**Theorem 7.2.3** (Obere Schranke (ii) des Haupttheorems 7.0.1)**.** *Die Familie $(F_{n,\ell})_{n \in \mathbb{N}}$ mit $8n^3 \leq \ell \leq 2^n$ aus Haupttheorem 7.0.1 besitzt eine Resolutionswiderlegung $\pi''_{n,\ell}$ in kleinem Platz*

$$\mathrm{Sp}(\pi''_{n,\ell}) \leq 2(n+1)\lceil \log_2(n\ell) \rceil + 1 = \mathcal{O}(n \log \ell)$$

*und simultaner Länge*

$$\mathrm{Le}(\pi''_{n,\ell}) \leq 2^{\mathcal{O}(n \log \ell)} - 1 \leq 2^{\mathcal{O}(n \log \ell)}.$$

*Beweis.* Proposition 7.2.1 zusammen mit Lemma 7.1.7 liefert unter Ausnutzung von $n \leq 8n^3 \leq \ell$ durch eine einfache Rechnung die Behauptung. Aufgrund der Ähnlichkeit der Rechnung verweisen wir lediglich auf den Beweis von Theorem 7.1.13. □

Wie [BBI2016, BNT2013] weisen wir kurz auf folgenden Fakt hin: Der Parameter $n$ kann als *Baum-Weite* der Formel aufgefasst werden (siehe hierzu z. B. [ACL+2014, Definition 2.1]), sodass sich die obere Schranke auch aus [ACL+2014] durch einen Baum-Weiten-basierten SAT-Algorithmus ergibt. Ausführliche Diskussionen dazu finden sich in [BBI2016, S. 1642–1644].

## 7.3 Beweis des Hauptheorems 7.0.1 (iii) in fünf Schritten

Bevor wir mit dem Beweis des Hauptheorems 7.0.1 (iii) beginnen, setzen wir das bisher Erreichte der Punkte (i) und (ii) in der folgenden Intuition in Kontext.

**Intuition 7.3.1** (Stark vereinfacht)**.** Wir konnten bisher zeigen, dass eine Resolutionswiderlegung der Formel $F_{n,\ell}$ mit $\ell = 8n^3$ möglich ist

(i) in Zeit $\approx 2^n$ und Platz $\approx 2^n$,

(ii) oder in Zeit $\approx \ell^n \geq n^n \ggg 2^n$ und Platz $n \log \ell \leq \text{poly}(n)$.

Wir verweisen auf Abbildung 7.7. BECK nennt die Einsparungen von (i) nach (ii) in [Bec2018] auch *"Savitch-like" savings*, angelehnt an den Satz von Savitch [Sav1970]. Dieser besagt, dass ein von einer nicht-deterministischen Turingmaschine mit einer bestimmten Platzkomplexität lösbares Problem auf einer deterministischen Turingmaschine mit einer quadratisch höheren Platzschranke gelöst werden kann; formaler: Ist $s\colon \mathbb{N} \to \mathbb{N}$ eine richtige Komplexitätsfunktion mit $s(n) \geq \log(n)$, dann gilt $\mathsf{NSPACE}\big(s(n)\big) \subset \mathsf{DSPACE}\big(s(n)^2\big)$.

Punkt (iii) wird nun zeigen, dass ein quasipolynomieller Blow-up in der Länge/Zeit des Resolutionsbeweises *zwingend notwendig* ist, wenn weniger Platz als $\approx 2^n$ zur Verfügung steht bzw. der Platz entsprechend eingeschränkt wird.

BECK erläutert in seiner Dissertation [Bec2017, § 4.3] eine generelle Strategie für das Beweisen von Zeit-Platz Tradeoffs, die wir an dieser Stelle zitieren wollen:

> Generally speaking, time-space trade-off results are usually established by some variation of the following plan:
>
> 1. Formalize a notion of work or progress specific to the model and the problem.
> 2. Divide the time period of a hypothetical computation into a large number of equal-sized epochs.

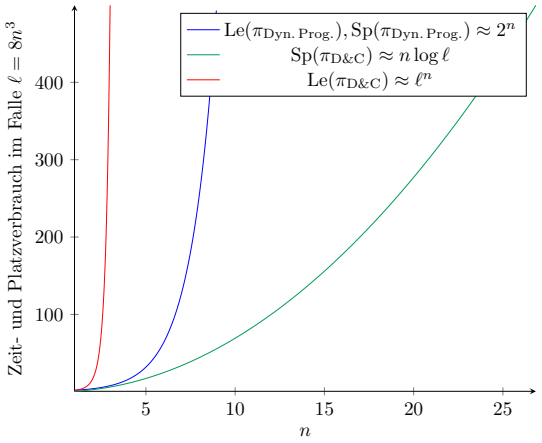



Abbildung 7.7.: Massiver Blow-up der Länge des Resolutionsbeweises von $F_{n,8n^3}$ – in rot – beim Übergang (und der damit verbundenen Platzreduktion) von dynamischer Programmierung – in blau – zu Divide and Conquer – in grün. Siehe hierzu Intuition 7.1.12 & 7.2.2). Man beachte, dass $\ell^n \geq n^n \geq n!$ ist.

3. Prove the following claims: (a) If the epochs are small, then no single epoch makes very much progress. (b) If the space is small, then not much progress can be carried over from one epoch to the next. (c) To solve the problem, the computation needs to make substantial progress summed over all epochs.
4. Conclude that if the computation is too short and uses too little space, then this leads to a contradiction.

Wir adaptieren diese grundlegende Beweis-Schablone, verfeinern aber insbesondere die Epochenunterteilung rekursiv, um dadurch ein qualitativ stärkeres Resultat zu erhalten.

**Gliederung des Beweises.** Für eine bessere Lesbarkeit des Beweises haben wir diesen in fünf Hauptschritte gegliedert: Wir leiten zunächst für unseren verwendeten Gittergraphen in Abschnitt 7.3.1 eine erweiterte isoperimetrische Ungleichung her. Dieser (wie bereits in Fußnote 33 auf Seite 64 angekündigt) durchaus nicht-triviale, technische Abschnitt sichert die Explizitheit des in den darauf folgenden Abschnitten 7.3.2–7.3.5 bewiesenen Tradeoff-Resultates: Dieses gilt nämlich für alle Graphen, die eine erweiterte isoperimetrische Eigenschaft besitzen. Der Nachweis dieser Eigenschaft für Gittergraphen liefert also ein konkretes Beispiel eines Graphen, der für den Tradeoff benutzt werden kann.

Punkt 1 der oben zitierten Beweisschablone erledigen wir in Abschnitt 7.3.2 durch Einführung eines Komplexitätsmaßes für Klauseln.

Mit diesem Maß können wir in Abschnitt 7.3.3 die in Punkt 2 angesprochene Unterteilung des Beweises in Epochen vornehmen.

In Abschnitt 7.3.4 etablieren wir eine Verbindung zwischen Isoperimetrie und Komplexitätsmaßen, die wir in Abschnitt 7.3.5 geschickt ausnutzen werden, um das Tradeoff-Resultat in einem Widerspruchsbeweis zu zeigen.

### 7.3.1 Eine isoperimetrische Ungleichung für Gittergraphen

Für die Formulierung einer isoperimetrischen Ungleichung[45] benötigen wir den Begriff des Randes eines Graphen, dem wir bereits in Fußnote 33 auf Seite 64 begegnet sind.

**Definition 7.3.2.** Es sei $G = (V, E)$ ein ungerichteter Graph und $S \subset V$ eine Teilmenge der Knoten des Graphen. Der *Rand* $\delta_G(S)$ von $S$ in $G$ ist die Menge der Kanten von $G$ mit genau einem Endknoten in $S$, in Zeichen

$$\delta_G(S) := \{uv \in E \,:\, u \in S \text{ und } v \in V \setminus S\}.$$

Wir nennen die Elemente von $\delta_G(S)$ *Randkanten* von $S$ (bzgl. $G$). Ist aus dem Zusammenhang klar, über welchen Graphen wir reden, schreiben wir lediglich $\delta(S)$.

Eine *isoperimetrische Ungleichung* für einen Graphen $G$ ist eine untere Schranke für $|\delta(S)|$, welche für alle Teilmengen $S \subset V(G)$ gilt, abhängig lediglich von der Kardinalität $|S|$; präziser sucht man also eine Funktion $F$, sodass die Ungleichung

$$|\delta(S)| \geq F(|S|) \quad \text{für alle } S \subset V(G) \tag{7.13}$$

gilt.

Ausgehend von dieser Idee, zeigen wir eine *erweiterte isoperimetrische Ungleichung* für Gittergraphen, die eine untere Schranke für die Kardinalität der Vereinigung von Rändern

[45] Isoperimetrische Ungleichungen bzw. Probleme (aus dem Griechischen für „*gleicher Umfang*“) sind Probleme aus der Geometrie, zu einem gegebenen Umfang die geometrische Figur mit größtmöglichem Flächeninhalt zu finden (verwandt mit dem *Problem der Dido*, der phönizischen Königin, die bei der Gründung der Stadt Karthago mit einer Kuhhaut ein Stück Land für ihr Volk umspannen durfte; siehe [Blå2005]). Der einfachste Spezialfall aus der Analysis, im Fall der zweidimensionalen Ebene $\mathbb{R}^2$ bringt die Länge $L$ einer geschlossenen Kurve $\gamma$ im $\mathbb{R}^2$ mit ihrem eingeschlossenen Flächeninhalt $A$ durch $4\pi A \leq L^2$ in Verbindung (wobei Gleichheit genau dann gilt, wenn $\gamma$ ein Kreis ist; für die bemerkenswerte Historie dieser Lösung und den großen Einfluss auf die Mathematik verweisen wir auf den Vortrag [Tap2009]). In der geometrischen Analysis werden solche Ungleichungen (unter Zuhilfenahme von Lebesgue- und Hausdorff-Maßen) auch in $n$ Dimensionen bewiesen (wobei die $n$-dimensionale Kugel auch hier im Wesentlichen das Optimum liefert). Diese haben (durch die implizierte Sobolev-Ungleichung) weitreichende Konsequenzen bis in den Bereich der partiellen Differentialgleichungen.

Angeregt durch diese Erfolge begann man auch Isoperimetrie bei Graphen zu studieren. Insbesondere für Expander-Graphen führt dies zu interessanten Resultaten u. a. auf dem Gebiet der Komplexitätstheorie und Codierungstheorie (wir verweisen für Details zu diesen Zusammenhängen auf das Survey Paper [HLW2006] sowie auf Abschnitt 4.2, Beispiel 2).

*stark wachsender* (engl. *superincreasing*) Mengen darstellt.

**Definition 7.3.3.** Es sei $G = (V, E)$ ein ungerichteter Graph. Des Weiteren seien $W, t_0 \in \mathbb{N}$ mit $t_0 \leq \frac{|V|}{4}$ und $r \in (1, \infty)$ Parameter.

(i) Eine Knotenmenge $S \subset V$ hat *mittlere Größe* (bzgl. $t_0$), falls für die Kardinalität der Menge $t_0 \leq |S| \leq \frac{|V|}{4}$ gilt.

(ii) Eine endliche, nicht-leere Folge $(S_i)_{i=1}^k \subset V$ von Knotenmengen heißt *r-stark wachsend*, falls

$$|S_{i+1}| \geq r \cdot |S_i| \quad \text{für alle } i = 1, \ldots, k-1$$

gilt.

(iii) Wir sagen, dass der Graph $G$ eine *erweiterte isoperimetrische Eigenschaft mit Parametern* $(W, t_0, r)$ erfüllt, falls jede endliche, nicht-leere Folge $(S_i)_{i=1}^k \subset V$ von Knotenmengen mittlerer Größe (bzgl. $t_0$), die $r$-stark wachsend ist, die *erweiterte isoperimetrische Ungleichung*

$$\left| \bigcup_{i=1}^{k} \delta(S_i) \right| \geq k \cdot W \tag{7.14}$$

erfüllt.

**Bemerkung 7.3.4.** Obige Definition rechtfertigt den Namen *erweiterte* isoperimetrische Ungleichung: Gilt sie mit Parametern $(W, t_0, r)$, so gilt insbesondere die isoperimetrische Ungleichung wie in (7.13) für Mengen mittlerer Größe (bzgl. $t_0$) mit $F \equiv W$.

Um eine erweiterte isoperimetrische Ungleichung für Gittergraphen zu zeigen, ist es hilfreich, als Vorarbeit zunächst Pfadgraphen zu betrachten.

**Definition 7.3.5.** Mit $G_{\mathbb{Z}}$ bezeichnen wir den *unendlichen ungerichteten Pfadgraphen* mit Knotenmenge $V(G_{\mathbb{Z}}) := \mathbb{Z}$ und Kantenmenge $E(G_{\mathbb{Z}}) := \Big\{\{i, i+1\} \,:\, i \in \mathbb{Z}\Big\}$, wie in Abbildung 7.8 skizziert.

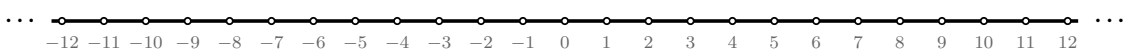



Abbildung 7.8.: Der Graph $G_{\mathbb{Z}}$.

Um die nachfolgenden Beweise von Lemma 7.3.7 und Proposition 7.3.9 weniger technisch und übersichtlicher zu gestalten, lagern wir zwei leicht zu zeigende Eigenschaften 2-stark wachsender Mengen in nachfolgendes Hilfslemma aus.

**Hilfslemma 7.3.6.** *Ist $k \in \mathbb{N}$ eine natürliche Zahl und $(S_i)_{i=1}^{k+1}$ eine Folge von $k+1$ endlichen, nicht-leeren Teilmengen der ganzen Zahlen $\mathbb{Z}$, die 2-stark wachsend ist, so gilt:*

*(i)* $\sum_{i=1}^{k} |S_i| < |S_{k+1}|$.

*(ii)* $\sum_{i=1}^{k} |S_i| < 2|S_k|$.

*Beweis.* Wir zeigen lediglich Aussage (ii), da (i) unter Benutzung der Abschätzung $2|S_k| \leq |S_{k+1}|$ sofort aus (ii) folgt (bzw. analog gezeigt werden kann). Gemäß der Definition von 2-stark wachsend gilt

$$|S_{i+1}| \geq 2 \cdot |S_i| \quad \text{für alle } i = 1, \ldots, k-1,$$

bzw. (wie man leicht induktiv verifiziert)

$$|S_i| \leq \frac{|S_{i+1}|}{2^1} \leq \frac{|S_{i+2}|}{2^2} \leq \cdots \leq \frac{|S_k|}{2^{k-i}} \quad \text{für alle } i = 1, \ldots, k.$$

Folglich ist mit der geometrischen Summe

$$\sum_{i=1}^{k} |S_i| \leq |S_k| \sum_{i=1}^{k} \frac{1}{2^{k-i}} = |S_k| \sum_{i=0}^{k-1} \left(\frac{1}{2}\right)^i = \left(2 - \left(\frac{1}{2}\right)^{k-1}\right) |S_k| < 2|S_k|. \qquad \square$$

Wir zeigen nun eine isoperimetrische Ungleichung für unendliche ungerichtete Pfadgraphen.

**Lemma 7.3.7.** *Es sei $k \in \mathbb{N}$ eine natürliche Zahl und $(S_i)_{i=1}^{k}$ eine Folge von $k$ endlichen, nicht-leeren Teilmengen der ganzen Zahlen $\mathbb{Z}$, die 2-stark wachsend ist. Dann ist*

$$\left| \bigcup_{i=1}^{k} \delta_{G_{\mathbb{Z}}}(S_i) \right| \geq k + c, \tag{7.15}$$

*wobei $c$ die Anzahl der Zusammenhangskomponenten des durch $\bigcup_{i=1}^{k} S_i$ in $G_{\mathbb{Z}}$ induzierten Teilgraphen $G_{\mathbb{Z}}[\bigcup_{i=1}^{k} S_i]$ bezeichne.*

Unser Beweis ist angelehnt an [Bec2017, Lemma 4.51], verbessert aber einige technische Fehler.

*Beweis.* Wir zeigen die Aussage über eine *Doppelinduktion* nach $k$ und $c$ in den Schritten (I)–(III), wie in Abbildung 7.9 skizziert. Sind $k, c \in \mathbb{N}$, so bezeichne dazu $A_{k,c}$ die zu be-

weisende Aussage „(7.15) gilt für alle Folgen $(S_i)_{i=1}^k$ von endlichen, nicht-leeren Teilmengen der ganzen Zahlen $\mathbb{Z}$, die 2-stark wachsend sind“.

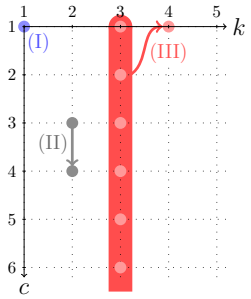



Abbildung 7.9.: Beweisidee der Doppelinduktion und Aufteilung in die drei Beweisschritte (I)–(III): (I) ist der Beweis der Aussage $A_{1,1}$. (II) ist die Induktionsschritt-Implikation $A_{k,c} \frown A_{k,c+1}$. (III) ist schließlich $(A_{k,c} \forall c) \frown A_{k+1,1}$.

(I) Ist $k = 1$ und $c = 1$, so besteht $G_{\mathbb{Z}}[S_1]$ nach Wahl von $c$ aus lediglich einer Zusammenhangskomponente, weshalb wir $S_1 = [a_1, b_1] \cap \mathbb{Z}$ für passende $a_1, b_1 \in \mathbb{Z}$ mit $a_1 \leq b_1$ schreiben können. Folglich haben wir $|\delta_{G_{\mathbb{Z}}}(S_1)| = \left|\left\{\{a_1 - 1, a_1\}, \{b_1, b_1 + 1\}\right\}\right| = 2$, also

$$\left|\bigcup_{i=1}^{1} \delta_{G_{\mathbb{Z}}}(S_i)\right| = |\delta_{G_{\mathbb{Z}}}(S_1)| = 2 \geq 1 + 1 = 2,$$

weshalb $A_{1,1}$ gilt. Vergleiche auch Abbildung 7.10.

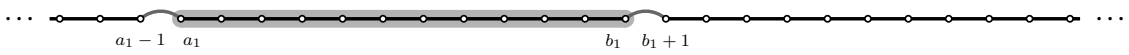



Abbildung 7.10.: Der Fall $k = c = 1$ mit der einzigen Zusammenhangskomponente $S_1 = [a_1, b_1] \cap \mathbb{Z}$ für passende $a_1, b_1 \in \mathbb{Z}$ mit $a_1 \leq b_1$. Die Randkanten aus $\delta_{G_{\mathbb{Z}}}(S_1)$ sind gebogen dargestellt.

(II) Angenommen $A_{k,c}$ gelte für ein Tupel $(k, c) \in \mathbb{N} \times \mathbb{N}$. Wir zeigen, dass dann $A_{k,c+1}$ gilt:

Dazu sei $S_1, \ldots, S_k$ eine 2-stark wachsende Folge von endlichen, nicht-leeren Teilmengen $S_i$ der ganzen Zahlen $\mathbb{Z}$, sodass es $c + 1$ Zusammenhangskomponenten in $G_{\mathbb{Z}}[\bigcup_{i=1}^k S_i]$ gibt. Da $c \in \mathbb{N}$ ist, handelt es sich also um mindestens 2 Komponenten. Daher existiert ein Knoten $z \in \mathbb{Z}$, der in keiner der Mengen $S_i$ enthalten ist und zwischen zwei Zusammenhangskomponenten liegt, sodass der Knoten $z - 1$ in einer der Mengen $S_i$ liegt. Dies erlaubt es uns, die *Kontraktionsfunktion* $f \colon \mathbb{Z} \to \mathbb{Z}$ durch

$$f(x) := \begin{cases} x, & \text{falls } x < z \\ x - 1, & \text{falls } x \geq z \end{cases}$$

zu definieren, welche die Knoten $z-1$ und $z$ auf einen einzigen Punkt abbildet, wie in Abbildung 7.11 skizziert. Wir unterscheiden zwischen zwei Fällen:

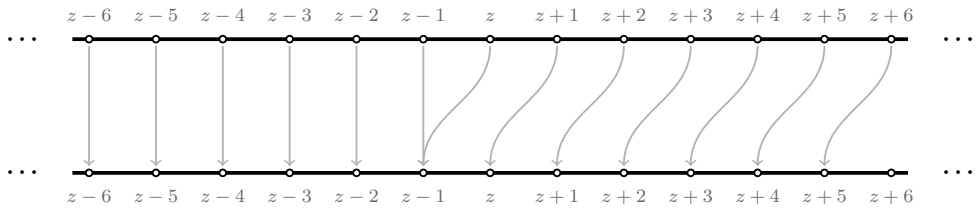



Abbildung 7.11.: Wirkung der Kontraktionsfunktion $f$.

Fall $z+1 \notin \bigcup_{i=1}^{k} S_i$: Durch die Fallvoraussetzungen sind die folgenden Größen offensichtlich $f$-invariant: Die Anzahl $k$ der Mengen $S_i$, die Kardinalitäten $|S_i|$, die Anzahl der Zusammenhangskomponenten in $G_{\mathbb{Z}}[\bigcup_{i=1}^{k} S_i]$ und die Kardinalität $\left|\bigcup_{i=1}^{k} \delta_{G_{\mathbb{Z}}}(S_i)\right|$. Also gilt die folgende Äquivalenz:

$$\text{(7.15) gilt für } S_1, \ldots, S_k \iff \text{(7.15) gilt für } f(S_1), \ldots, f(S_k).$$

Da es nach Wahl des Knotens $z$ zwischen zwei Komponenten einen Knoten $v^\star > z$ in $\bigcup_{i=1}^{k} S_i \subset \mathbb{Z}$ geben muss, erreichen wir nach einer endlichen Anzahl $m := v^\star - (z+1)$ von Anwendungen der Kontraktionsfunktion $f$:

$$\text{(7.15) gilt für } S_1, \ldots, S_k \iff \begin{array}{l}\text{(7.15) gilt für } f^m(S_1), \ldots, f^m(S_k) \text{ und} \\ z+1 \text{ liegt in einer der Mengen } f^m(S_i).\end{array}$$

Damit haben wir diesen Fall auf den folgenden Fall zurückgeführt (wir bezeichnen im folgenden Fall die eventuell verschobenen Mengen $f^m(S_1), \ldots, f^m(S_k)$ wieder mit den Bezeichnungen der ursprünglichen Mengen $S_1, \ldots, S_k$).

Fall $z+1 \in \bigcup_{i=1}^{k} S_i$: Wie eingangs im vorigen Fall halten wir fest, dass die Anzahl $k$ der Mengen $S_i$ und die Kardinalitäten $|S_i|$ weiterhin $f$-invariant sind, da $z \notin \bigcup_{i=1}^{k} S_i$ ist. Da jedoch eine Komponente $K$ existiert, die $z-1$ beinhaltet und eine andere Komponente $K'$ existiert, die $z+1$ beinhaltet, sind die Bilder von $K$ und $K'$ anschließend eine Komponente. Da keine anderen „Kollisionen“ von Komponenten auftreten können, wird die Anzahl der Komponenten durch Anwendung von $f$ um genau 1 reduziert (was es uns erlaubt $A_{k,c}$ anzuwenden). Weiterhin beinhaltet $\bigcup_{i=1}^{k} \delta_{G_{\mathbb{Z}}}(f(S_i))$ mindestens eine Randkante weniger als $\bigcup_{i=1}^{k} \delta_{G_{\mathbb{Z}}}(S_i)$, da die ehemaligen Randkanten $\{z-1, z\}$ und $\{z, z+1\}$ unter $f$ zu einer nicht-Randkante „verschmelzen“ (tatsächlich gibt es keine weiteren ver-

schwindenden Kanten). Dank dieser zwei Beobachtungen und $A_{k,c}$ erhalten wir:

$$\left|\bigcup_{i=1}^{k} \delta_{G_\mathbb{Z}}(S_i)\right| \geq \left|\bigcup_{i=1}^{k} \delta_{G_\mathbb{Z}}(f(S_i))\right| + 1 \overset{(A_{k,c})}{\geq} k + c + 1.$$

Folglich gilt $A_{k,c+1}$.

(III) Angenommen $A_{k,c}$ gelte für ein festes $k \in \mathbb{N}$ und alle $c \in \mathbb{N}$. Wir zeigen, dass dann $A_{k+1,1}$ gilt:

Hierzu wählen wir eine beliebige Folge von $k+1$ endlichen, nicht-leeren Teilmengen der ganzen Zahlen $\mathbb{Z}$, die 2-stark wachsend ist, und bezeichnen die Mengen der Folge mit $S_1, \ldots, S_{k+1}$. Mit $c' := c'(S_1, \ldots, S_k)$ bezeichnen wir im Folgenden die Anzahl der Zusammenhangskomponenten in $G_\mathbb{Z}[\bigcup_{i=1}^{k} S_i]$. Wir unterscheiden zwischen zwei Fällen:

Fall $c' = 1$: Da in diesem Fall $\bigcup_{i=1}^{k} S_i$ die einzige Komponente in $G_\mathbb{Z}$ ist, können wir den Abstand des „kleinsten“ Knotens $v_{\min} := \min \bigcup_{i=1}^{k} S_i$ zum „größten“ Knotens $v_{\max} := \max \bigcup_{i=1}^{k} S_i$ nach oben abschätzen durch

$$v_{\max} - v_{\min} < \left|\bigcup_{i=1}^{k} S_i\right| \leq \sum_{i=1}^{k} |S_i| < |S_{k+1}|,$$

wobei wir in der letzten Abschätzung Lemma 7.3.6 (i) benutzt haben. Also ist mindestens einer der beiden Punkte $\min S_{k+1}$ oder $\max S_{k+1}$ ein „Extrempunkt“ $v_{\text{extr}}$ der Menge $\bigcup_{i=1}^{k+1} S_i$, welcher nicht in $\bigcup_{i=1}^{k} S_i$ enthalten war (man beachte, dass wir hierzu angenommen haben, dass es nur eine Zusammenhangskomponente in $G_\mathbb{Z}[\bigcup_{i=1}^{k+1} S_i]$ gibt). Im Falle $v_{\text{extr}} = \min S_{k+1}$ ist die Kante $\{v_{\text{extr}} - 1, v_{\text{extr}}\}$ eine Randkante von $S_{k+1}$, aber keine von $\bigcup_{i=1}^{k} S_i$; im anderen Falle gilt die analoge Aussage für die Kante $\{v_{\text{extr}}, v_{\text{extr}} + 1\}$. Wir verweisen für eine Illustration auf Abbildung 7.12. In beiden Fällen kann jedoch

$$\left|\bigcup_{i=1}^{k+1} \delta_{G_\mathbb{Z}}(S_i)\right| > \left|\bigcup_{i=1}^{k} \delta_{G_\mathbb{Z}}(S_i)\right| \overset{(A_{k,1})}{\geq} k + 1$$

unter Benutzung von $A_{k,1}$ geschlossen werden. Also gilt $A_{k+1,1}$.

Fall $c' > 1$: Direkt aus $A_{k,c'}$ folgt

$$\left|\bigcup_{i=1}^{k} \delta_{G_\mathbb{Z}}(S_i)\right| \geq k + c' > k + 1.$$

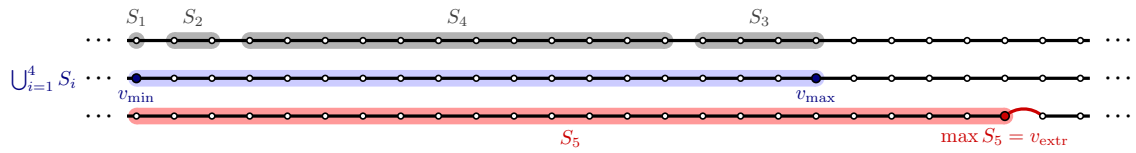



Abbildung 7.12.: Illustration des „worst-cases“ im Falle $c' = 1$ im Beweis von Lemma 7.3.7. In grau sind die Mengen der 2-stark wachsenden Folge $(S_1, \ldots, S_4)$ in $G_{\mathbb{Z}}$ eingezeichnet, welche die einzige vorhandene Komponente, $\bigcup_{i=1}^{4} S_i$, die in hellblau eingezeichnet ist, bilden. In rot wurde die Menge $S_5$ (die „optimal“ am linken Rand der hellblauen Menge ausgerichtet ist) eingezeichnet, die größer als die hellblaue Menge ist. Die Knoten $v_{\min}$, $v_{\max}$ und $v_{\text{extr}}$ wurden entsprechend beschriftet.

Trivialerweise gilt dann auch

$$\left|\bigcup_{i=1}^{k+1} \delta_{G_{\mathbb{Z}}}(S_i)\right| \geq \left|\bigcup_{i=1}^{k} \delta_{G_{\mathbb{Z}}}(S_i)\right| > k+1,$$

d. h. es ist $\left|\bigcup_{i=1}^{k+1} \delta_{G_{\mathbb{Z}}}(S_i)\right| \geq k+1+1$, weshalb auch in diesem Fall $A_{k+1,1}$ gilt.

Dies schließt die Doppelinduktion ab. □

Um den Beweis der erweiterten isoperimetrischen Eigenschaft für Gittergraphen in Proposition 7.3.9 einfacher zu gestalten, lagern wir die nachfolgende technische Abschätzung in ein Hilfslemma aus.

**Hilfslemma 7.3.8.** *Es seien $n \in \mathbb{N}$ und $\ell \in \mathbb{N}$ mit $8n^3 \leq \ell \leq 2^n$. In solch einem $n \times \ell$-Gittergraph $G_{n\times\ell}$ kann eine $\frac{7}{2}$-stark wachsende Folge höchstens $k < n$ Knotenmengen mittlerer Größe (bzgl. $t_0 = 2n^3$) haben.*

*Beweis.* Der Gittergraph kann höchstens $n \cdot 2^n$ Knoten enthalten. Im besten Fall (d. h. um ein möglichst großes $k$ zu erhalten) haben die Knotenmengen mittlerer Größe in der $\frac{7}{2}$-stark wachsenden Folge jeweils kleinstmögliche Kardinalität, d. h.

$$\begin{aligned}
|S_1| &= t_0 = 2n^3, \\
|S_2| &= \frac{7}{2} \cdot 2n^3, \\
|S_3| &= \left(\frac{7}{2}\right)^2 \cdot 2n^3, \\
&\vdots \\
|S_k| &= \left(\frac{7}{2}\right)^{k-1} \cdot 2n^3.
\end{aligned}$$

Da auch $S_k$ mittlere Größe haben soll, erhalten wir die notwendige Bedingung

$$\begin{aligned}
\left(\frac{7}{2}\right)^{k-1} \cdot 2n^3 &\overset{!}{\leq} \frac{n \cdot 2^n}{4} \\
\iff \qquad \left(\frac{7}{2}\right)^{k-1} &\leq \frac{n \cdot 2^n}{8n^3} = \frac{2^n}{8n^2} \\
\iff \qquad k - 1 &\leq \log_{7/2}\left(\frac{2^n}{8n^2}\right) \\
\iff \qquad k &\leq \log_{7/2}\left(\frac{2^n}{8n^2}\right) + 1.
\end{aligned}$$

Durch den Monotonietest aus der Analysis (siehe z. B. [Sch2011, Satz 5.3.12]) zeigen wir schließlich, dass $\log_{7/2}\left(\frac{2^n}{8n^2}\right) + 1 < n$ für alle $n \in \mathbb{N}$ ist: Die Hilfsfunktion $h\colon \mathbb{R}^+ \to \mathbb{R}$ gegeben durch

$$h(x) := \log_{7/2}\left(\frac{2^x}{8x^2}\right) + 1 - x = \frac{\log\left(\frac{2^{x-3}}{x^2}\right)}{\log(7/2)} + 1 - x$$

ist offensichtlich stetig und differenzierbar in $\mathbb{R}^+$ mit Ableitung

$$\begin{aligned}
h'(x) &= \frac{\frac{\underline{2^{x-3}(x^2 \log 2 - 2x)}}{x^4}}{\log(7/2)\frac{2^{x-3}}{x^2}} - 1 \\
&= \frac{\frac{\underline{x \log 2 - 2}}{x}}{\log(7/2)} - 1 \\
&= \frac{x \log 2 - 2 - x \log(7/2)}{x \log(7/2)}.
\end{aligned}$$

Es ist $x \log(7/2) > 0$ für alle $x \in \mathbb{R}^+$. Also ist $h'(x) < 0$, genau dann, wenn

$$-x \log(7/4) - 2 < 0 \tag{7.16}$$

ist. Da jedoch $-x \log(7/4) - 2$ die Gleichung einer Geraden mit $y$-Achsenabschnitt $-2$ und negativer Steigung $-\log(7/4)$ ist, gilt (7.16) für alle $x \in \mathbb{R}^+$, weshalb unsere Hilfsfunktion $h$ streng monoton fallend auf $\mathbb{R}^+$ ist. Zudem beachten wir $h(1) = \log_{7/2}\left(\frac{2}{8}\right) = -\frac{\log 4}{\log(7/2)} < 0$. Folglich ist $h(n) < 0$ für alle $n \in \mathbb{N}$, was wir zeigen wollten. □

Lemma 7.3.7 erlaubt uns nun eine erweiterte isoperimetrische Ungleichung für Gittergraphen zu beweisen. Wir formulieren und beweisen diese in ihrer vollen Allgemeinheit,

werden später aber für unsere konkrete Anwendung lediglich den Fall $r = 4$ benutzen.

**Proposition 7.3.9** (Isoperimetrische Ungleichung für Gittergraphen)**.** *Es seien $n \in \mathbb{N}$ und $\ell \in \mathbb{N}$ mit $8n^3 \leq \ell \leq 2^n$. Jeder solche $n \times \ell$-Gittergraph $G_{n\times\ell}$ erfüllt die erweiterte isoperimetrische Eigenschaft mit Parametern $(W = n, t_0 = 2n^3, r = \frac{7}{2} + \varepsilon)$ für alle $\varepsilon > 0$.*

**Intuition 7.3.10.**

- Anschaulich wollen wir durch eine erweiterte isoperimetrische Ungleichung also folgendes (für passende Parameter zu den Begriffen in Anführungszeichen) zeigen: Haben wir (beliebige) $k$ Knotenmengen „mittlerer Größe“ im Gittergraphen, die „wachsende“ Größe haben, dann haben wir mindestens $kn$ Kanten in der Vereinigung ihrer Ränder.
- Die Beweisidee, die wir im Folgenden benutzen werden, lässt sich wie in [Bec2018] beschreiben: Sind die Mengen nicht im Wesentlichen „Blöcke“ im Gittergraphen (siehe Abbildung 7.13), sind wir fertig, da die Mengen große Ränder haben. Falls alle Mengen im Wesentlichen Blockgestalt haben, reduzieren wir den Beweis auf eine Aussage im Pfadgraphen $G_{\mathbb{Z}}$ durch eine geschickte Projektion (siehe Abbildung 7.14) und zeigen, dass sich in jeder Zeile des Gittergraphen mindestens so viele Randkanten wie Mengen befinden.

Unser Beweis orientiert sich lose an der Beweisskizze in [Bec2017, Lemma 4.52].

*Formaler Beweis von Proposition 7.3.9.* Zunächst seien $\varepsilon > 0$ und $n \in \mathbb{N}$ beliebig. Weiter sei $\ell \in \mathbb{N}$ mit $8n^3 \leq \ell \leq 2^n$. Für diesen Beweis bezeichnen wir eine Spalte des Gittergraphen als *voll* bzgl. einer Knotenmenge $S$, falls alle $n$ Knoten der Spalte in $S$ enthalten sind; als *leer*, falls keine der Knoten der Spalte in $S$ enthalten sind; als *partiell* sonst. Entsprechend reden wir auch von vollen, leeren und partiellen Spalten von $S$.

Es seien nun $S_1, \ldots, S_k \subset V(G_{n\times\ell})$ Knotenmengen mittlerer Größe (bzgl. $t_0 = 2n^3$), die $(\frac{7}{2} + \varepsilon)$-stark wachsend sind. Wir unterscheiden zwei Fälle:

Fall 1: Es gibt eine Knotenmenge $S_{i_*}$ mit mindestens $kn$ partiellen Spalten:

In diesem Fall trägt jede der $kn$ partiellen Spalten zu mindestens einer vertikalen Randkante in $\delta_{G_{n\times\ell}}(S_{i_*})$ bei (vgl. Abbildung 7.13), m. a. W. ist $|\delta_{G_{n\times\ell}}(S_{i_*})| \geq kn$, also ist

$$\left|\bigcup_{i=1}^{k} \delta_{G_{n\times\ell}}(S_i)\right| \geq |\delta_{G_{n\times\ell}}(S_{i_*})| \geq k \cdot n,$$

was genau die gewünschte erweiterte isoperimetrische Eigenschaft darstellt.

Fall 2: Alle $S_i$ haben strikt weniger als $kn$ partielle Spalten:

Wir merken an dieser Stelle an, dass wir wegen Hilfslemma 7.3.8 für jede Menge $S_i$

also strikt weniger als $kn \overset{(k<n)}{<} n^2$ partielle Spalten haben.

Unser Ziel wird es sein, zu zeigen, dass wir in jeder der $n$ Zeile des Graphen mindestens $k$ horizontale Randkanten erhalten, woraus sofort die gewünschte Abschätzung $\left|\bigcup_{i=1}^{k} \delta_{G_{n\times\ell}}(S_i)\right| \geq k \cdot n$ folgt. Dies werden wir durch eine geschickte Anwendung von Lemma 7.3.7 erreichen. Hierzu benötigen wir die folgende Notation: Ist $r \in \{1, \ldots, n\}$ eine beliebige Zeilennummer des Gittergraphen, so definieren wir für alle Knotenmengen $S_i$ die Menge $S_i\restriction_r$ als die Menge aller Indizes derjenigen Spalten, die einen Knoten der Menge $S_i$ in Zeile $r$ enthalten. Für ein Beispiel siehe Abbildung 7.13.

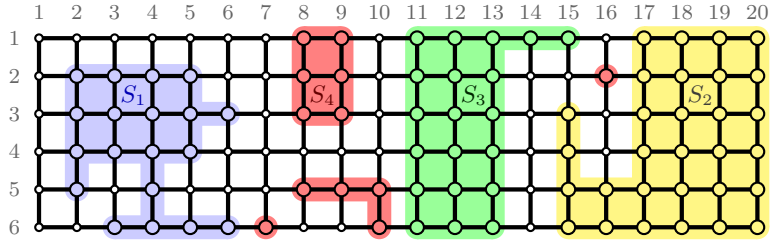



Abbildung 7.13.: Illustration der Reiheneinschränkung. Man beachte, dass die Bedingung $8n^3 \leq \ell \leq 2^n$ und das $(\frac{7}{2}+\varepsilon)$-starke Wachstum der Mengen aus zeichnerischen Gründen nicht gegeben sein kann. Beispielsweise gilt $S_1\restriction_3 = \{2,3,4,5,6\}$, während $S_1\restriction_5 = \{2,4\}$ ist. Ebenso sei beispielhaft erwähnt, dass alle Spalten von $S_1$ partiell sind und zu einer vertikalen Randkante beitragen, während die Spalten $11, 12, 13$ bzgl. $S_3$ voll sind und die Spalten $14, 15$ bzgl. $S_3$ partiell sind. Die Spalte 1 ist bspw. leer bzgl. jeder der Mengen $S_1, \ldots, S_4$.

Wir weisen darauf hin, dass $S_1$ so konstruiert ist, dass $\delta_{G_{6\times 20}}(S_1) = k \cdot n = 24$ ist. Im Sinne von Intuition 7.3.10 ist $S_1$ also nicht mehr im Wesentlichen ein „Block“. Hierdurch wäre die gewünschte Ungleichung bereits bewiesen.

Man beachte, dass jede Menge $S_i$ höchstens $|S_i|/n$ volle Spalten haben kann. Dies werden wir in (7.17) und (7.18) ausnutzen. Ebenso werden wir in (7.17) ausnutzen, dass $\left|S_i\restriction_r\right| \geq |\{\text{volle Spalten bzgl. } S_i\}|$ gilt: Jede volle Spalte taucht nämlich in $S_i\restriction_r$ auf, $S_i\restriction_r$ kann aber potentiell weitere Spaltenindizes beinhalten (siehe z. B. $S_3\restriction_1$).

Siehe auch Abbildung 7.14 für weitere beweisrelevante Konstruktionen.

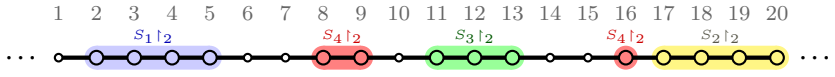



Abbildung 7.14.: Die Menge $\bigcup_{i=1}^{4} S_i\restriction_2$ aus Abbildung 7.13 im Graphen $G_{\mathbb{Z}}$. Man beachte, dass wir aus Darstellungsgründen auf die Eigenschaft des $(\frac{7}{2}+\varepsilon)$-starken Wachstums der Mengen keine Rücksicht genommen haben.

Um Lemma 7.3.7 anwenden zu können, wollen wir zeigen, dass das Verhältnis

$$\left|S_{i+1}\restriction_r\right| \Big/ \left|S_i\restriction_r\right| \geq 2 \quad \text{für alle } 1 \leq i \leq k-1 \text{ und alle } 1 \leq r \leq n$$

ist. Hierzu seien im Folgenden ein $i$ und ein $r$ wie oben fixiert. Wir beobachten: Da $S_i$ wie eingangs bemerkt höchstens $n^2$ partielle Spalten haben kann, ergibt sich

$$\left|S_i\restriction_r\right| \geq |\{\text{volle Spalten bzgl. } S_i\}| \geq \frac{|S_i|}{n} - n^2 \tag{7.17}$$

(für eine Erläuterung sowie ein Beispiel hierzu verweisen wir auf Abbildung 7.13). Weiterhin erhalten wir die Abschätzung

$$\left|S_i\restriction_r\right| \leq \frac{|S_i|}{n} + n^2, \tag{7.18}$$

denn es kann höchstens $\frac{|S_i|}{n}$ bzgl. $S_i$ volle Spalten geben und höchstens $n^2$ partielle Spalten bzgl. $S_i$ (wie eingangs bemerkt), also höchstens $\frac{|S_i|}{n} + n^2$ Spalten, die bzgl. $S_i$ voll *oder* partiell (also nicht-leer) sind – man beachte, dass nur in diesen beiden Fällen ein Spaltenindex in der Menge $S_i\restriction_r$ auftauchen kann. Unter Benutzung der Gleichungen (7.17) und (7.18) erhalten wir

$$\begin{aligned}
\frac{|S_{i+1}\restriction_r|}{|S_i\restriction_r|} &\geq \frac{\frac{|S_{i+1}|}{n} - n^2}{\frac{|S_i|}{n} + n^2} \overset{(\star)}{\geq} \frac{\frac{r|S_i|}{n} - n^2}{\frac{|S_i|}{n} + n^2} = r\left(\frac{\frac{|S_i|}{n} - \frac{n^2}{r}}{\frac{|S_i|}{n} + n^2}\right) \\
&\overset{(\dagger)}{=} r\left(1 - \frac{n^2 + \frac{n^2}{r}}{\frac{|S_i|}{n} + n^2}\right) = r\left(1 - \frac{\frac{rn^2+n^2}{r}}{\frac{|S_i|+n^3}{n}}\right) = r\left(1 - \frac{rn^2+n^2}{r} \cdot \frac{n}{|S_i|+n^3}\right) \\
&= r\left(1 - \frac{(r+1)n^3}{r(|S_i|+n^3)}\right) = r\left(1 - \frac{r+1}{\frac{r}{n^3}(|S_i|+n^3)}\right) = r\left(1 - \frac{r+1}{\frac{r|S_i|}{n^3} + r}\right) \\
&\overset{(\divideontimes)}{\geq} r\left(1 - \frac{r+1}{\frac{rt_0}{n^3} + r}\right) \overset{(\$)}{=} r\left(1 - \frac{r+1}{\frac{r \cdot 2n^3}{n^3} + r}\right) = r\left(1 - \frac{r+1}{3r}\right).
\end{aligned} \tag{7.19}$$

Bevor wir fortfahren, kommentieren wir die markierten Umformungsschritte kurz:

ad ($\star$): Wir haben hier benutzt, dass die Folge der Knotenmengen $r$-stark wachsend ist, d. h. $|S_{i+1}| \geq r|S_i|$ gilt.

ad ($\dagger$): Man beachte $1 = \frac{\frac{|S_i|}{n}+n^2}{\frac{|S_i|}{n}+n^2}$.

ad ($\divideontimes$): Wir benutzen, dass $S_i$ mittlere Größe hat, also dass $t_0 \leq |S_i|$ gilt.

ad ($\$$): Es ist $t_0 = 2n^3$.

Wegen

$$r\left(1 - \frac{r+1}{3r}\right) = r - \frac{r+1}{3} \overset{!}{>} 2 \iff r > \frac{7}{2} \tag{7.20}$$

ist der Quotient (7.19) nach unserer (sich aus dieser Abschätzung erklärenden) Wahl von $r = \frac{7}{2} + \varepsilon > \frac{7}{2}$ wie gewünscht größer als 2. Lemma 7.3.7 angewandt auf die Folge $S_1\restriction_r, \ldots, S_k\restriction_r$ impliziert also

$$\left|\bigcup_{i=1}^{k} \delta_{G_{\mathbb{Z}}}(S_i\restriction_r)\right| \geq k + c, \tag{7.21}$$

wobei $c$ die Anzahl der Zusammenhangskomponenten in $G_{\mathbb{Z}}[\bigcup_{i=1}^{k} S_i\restriction_r]$ ist.

Für alle Kanten aus $\bigcup_{i=1}^{k} \delta_{G_{\mathbb{Z}}}(S_i\restriction_r)$ außer $\{0,1\}$ und $\{\ell, \ell+1\}$ existiert eine korrespondierende horizontale Kante aus $\bigcup_{i=1}^{k} \delta_{G_{n\times\ell}}(S_i)$ in der Zeile $r$. Erneut unterscheiden wir zwei Fälle:

Fall 2a: $\bigcup_{i=1}^{k} \delta_{G_{\mathbb{Z}}}(S_i\restriction_r)$ enthält nur eine oder keine der Kanten $\{0,1\}$ oder $\{\ell, \ell+1\}$:
Per definitionem ist $c \geq 1$, folglich ergibt sich aus der Abschätzung (7.21) und obiger Vorbemerkung zur Fallunterscheidung die Existenz von mindestens $k$ horizontalen Randkanten in $\bigcup_{i=1}^{k} \delta_{G_{n\times\ell}}(S_i)$ in der Zeile $r$, wie gewünscht.

Fall 2b: $\bigcup_{i=1}^{k} \delta_{G_{\mathbb{Z}}}(S_i\restriction_r)$ enthält beide der Kanten $\{0,1\}$ und $\{\ell, \ell+1\}$:
Dann folgt unmittelbar $c \geq 2$; denn angenommen (um einen Widerspruch zu erhalten) es ist $c = 1$: Dann liegen die Knoten 1 und $\ell$ im Graphen $G_{\mathbb{Z}}$ in derselben Zusammenhangskomponente von $\bigcup_{i=1}^{k} S_i\restriction_r$, jedoch erhalten wir mit Hilfslemma 7.3.6 (ii) die Abschätzung

$$\begin{aligned}\left|\bigcup_{i=1}^{k} S_i\restriction_r\right| \leq \sum_{i=1}^{k} |S_i\restriction_r| < 2|S_k\restriction_r| &\overset{(7.18)}{\leq} 2\left(\frac{|S_k|}{n} + n^2\right) \\ &\leq 2\left(\frac{n\ell}{4\cdot n} + n^2\right) = \frac{\ell}{2} + 2n^2 < \ell, \end{aligned} \tag{7.22}$$

da $\ell \geq 8n^3 > 4n^2$ für alle $n \in \mathbb{N}$ ist. Ein Widerspruch! Also erhalten wir auch in diesem Fall aus Abschätzung (7.21) wie oben die Existenz von mindestens $k$ horizontalen Randkanten in $\bigcup_{i=1}^{k} \delta_{G_{n\times\ell}}(S_i)$.

Dies beendet unsere zwei Fallunterscheidungen und den Beweis. □

**Bemerkung 7.3.11** (Optimalität der Ungleichung)**.** Der Parameter $W$ in der obigen erweiterten isoperimetrischen Ungleichung 7.3.9 ist (bis auf additive Konstanten) so gewählt, dass die Ungleichung scharf ist: Lemma 7.1.7 (i) impliziert, dass wir $W$ eng an der Schnittweite des Gittergraphen gewählt haben. Es ist dadurch möglich Knotenmengen von mittlerer Größe zu konstruieren, die nur $\mathcal{O}(W)$ Randkanten haben.

### 7.3.2 Ein Komplexitätsmaß für Klauseln

Wie [Bec2017, S. 85] erinnern wir an die Arbeiten [CEI1996, IPS1999] und insbesondere Ben-Sasson und Wigdersons Artikel [BW2001]. Diese Artikel haben eine Vielzahl von Techniken im Beweissystem der Resolution etabliert, um eine untere Schranke an die Länge (bzw. Größe) einer Resolutionswiderlegung mit Hilfe einer unteren Schranke an die Weite einer Resolutionswiderlegung zu zeigen (diese letztgenannten Schranken sind nämlich im Allgemeinen deutlich leichter zu zeigen).

Orientiert an den Beweisen für untere Schranken für die Weite von Resolutionswiderlegungen spezieller Formeln in [BW2001, § 5], wollen wir Klauseln eine Komplexität zuordnen. Dazu betrachten wir zunächst das in [BW2001, Definition 5.1] studierte Komplexitätsmaß

$$\mu_{\mathcal{A}}(C) := \min_{\substack{S \subset \mathcal{A}, \\ S \models C}} |S|, \tag{7.23}$$

das einer Klausel $C$ die minimale Kardinalität einer Teilmenge $S$ einer gegebenen (unerfüllbaren) Axiomenmenge $\mathcal{A}$ zuordnet, sodass die semantische Implikation $S \models C$ gilt. Später, in Definition 7.3.23, werden wir das Maß für unsere Zwecke anpassen.

Diese Funktion erlaubt uns einerseits, Klauseln eines Resolutionsbeweises eine Komplexität zuzuordnen. Diese Komplexität ist so gewählt, dass grob abgeschätzt werden soll, wie „wertvoll“ eine Klausel im Beweis ist. Auf der anderen Seite lässt sich damit der Fortschritt des Beweises zu einem beliebigen Zeitpunkt messen.

Wir merken an, dass das Maß aus (7.23) folgende Eigenschaften aufweist (deren Beweise man sofort aus einer Modifikation des Beweises von Proposition 7.3.24 erhält): Es ist $\mu_{\mathcal{A}}(a) = 1$ für alle $a \in \mathcal{A}$ und das Maß ist *subadditiv*, d. h. gilt $\{C_1, C_2\} \vdash^{1}_{\mathfrak{R}} C$, dann ist $\mu_{\mathcal{A}}(C) \leq \mu_{\mathcal{A}}(C_1) + \mu_{\mathcal{A}}(C_2)$.

Insbesondere erhalten wir durch die Forderung der Subadditivität bzgl. Resolutionsschritten leicht die Existenz von Klauseln mit „Zwischenkomplexität“ in jeder Resolutionswiderlegung. Intuitiv entspricht dies einer Art Zwischenwertsatz.

**Satz 7.3.12.** *Ist $\pi$ eine Resolutionswiderlegung von $\mathcal{A}$, so gibt es eine Klausel $C$ im Beweis $\pi$ mit „Zwischenkomplexität“ $\frac{1}{3}\mu_{\mathcal{A}}(\square) \leq \mu_{\mathcal{A}}(C) \leq \frac{2}{3}\mu_{\mathcal{A}}(\square)$.*

*Beweis nach [Bec2017].* Angenommen, um einen Widerspruch zu erhalten, solch eine Klausel $C$ gibt es nicht in $\pi$. Dann muss die erste Klausel des Beweises, welche die Komplexität $\frac{2}{3}\mu_{\mathcal{A}}(\square)$ übersteigt, aus zwei Klauseln hergeleitet werden mit Komplexität echt kleiner als $\frac{1}{3}\mu_{\mathcal{A}}(\square)$, was der Subadditivität widerspricht. □

Das Argument konnte in [PI2000, BBI2016] sogar noch weiter verschärft werden: Es

gibt auch eine Klauseln $C'$ im Beweis mit $\frac{\mu_{\mathcal{A}}(C')}{\mu_{\mathcal{A}}(\square)} \in [\frac{1}{6}, \frac{1}{3})$, eine Klausel $C''$ mit $\frac{\mu_{\mathcal{A}}(C'')}{\mu_{\mathcal{A}}(\square)} \in [\frac{1}{12}, \frac{1}{6})$, et cetera. In [BW2001] wurde anhand einiger Beispiele gezeigt: Ist $\mathcal{A}$ passend gewählt und ist das betrachtete Beweissystem nicht zu mächtig, dann ist eine solche „Zwischenkomplexitäts-Klausel“ eine „komplizierte“ Boolesche Formel – besitzt also im Falle der Resolution viele Variablen. Hieraus erhält man die oben angesprochenen unteren Schranken an die Weite. So nützlich diese Technik in [BW2001] auch war, benötigen wir für Tradeoff-Argumente stärkere Techniken. Wir passen diese ursprüngliche Idee daher im nachfolgenden Abschnitt weiter an unsere Bedürfnisse an.

### 7.3.3 Stark beschränkte Komplexitätsmaße und Beweisepochen

**Definition 7.3.13** (Stark beschränkte Komplexitätsmaße; Korrigiert aus [Bec2017])**.**

(a) Eine Funktion $\mu^*$, welche Formeln eines Beweises $\pi$ eines Beweissystems $\mathfrak{S}$ auf die nicht-negativen ganzen Zahlen $\mathbb{N}_0$ abbildet, heißt *stark beschränkt*, falls

$$\varphi_1, \varphi_2 \vdash^1_{\mathfrak{S}} \varphi_3$$

(d. h. die Formel $\varphi_3$ kann in $\mathfrak{S}$ in einem Schritt aus $\varphi_1, \varphi_2$ gefolgert werden) impliziert, dass

$$\mu^*(\varphi_3) \leq 1 + \max\left\{\mu^*(\varphi_1), \mu^*(\varphi_2)\right\}$$

gilt. Die Zahl $\mu^*(\varphi)$ nennen wir auch $\mu^*$*-Komplexität* der Formel $\varphi$, weshalb wir eine Funktion $\mu^*$ wie oben auch als *stark beschränktes Komplexitätsmaß* bezeichnen.

(b) Weiter nennen wir ein stark beschränktes Komplexitätsmaß $\mu^*$ *stark beschränkt für die Menge* $\mathcal{A}$, falls gilt:

$$\mu^*(a) = 0 \quad \text{für alle } a \in \mathcal{A}.$$

Die Eigenschaft, dass sich die Komplexität einer Resolvente um höchstens 1 zur maximalen Komplexität der beiden Elternklauseln erhöht, wird uns einen wertvollen Dienst (ähnlich zu Satz 7.3.12) in Proposition 7.3.20 erweisen – dies ist der Grund für das Studium stark beschränkter Komplexitätsmaße.

**Bemerkung 7.3.14.** Man zeigt leicht, dass aus einem subadditiven Komplexitätsmaß $\mu_{\mathcal{A}}$ wie in (7.23) vermöge der Festlegung $\mu^*_{\mathcal{A}} := \lfloor \log_2 \mu_{\mathcal{A}} \rfloor$ ein stark beschränktes Komplexitätsmaß für die Menge $\mathcal{A}$ gewonnen werden kann.

Diese formalere Betrachtungsweise greift die Ideen des ursprünglichen Papers zum Tradeoff-Resultat auf: Obige Festlegung zu $\mu^*_{\mathcal{A}}$ korrespondiert mit den in [BBI2016, Definition 17]

erklärten Komplexitätsleveln. Dort wurde ein solches Level via

$$\mathcal{L}_i := \left\{ C \in \pi \,:\, 2^i t_0 < |\mathrm{crit}(C)| \leq 2^{i+1} t_0 \right\} \quad \text{für } 0 \leq i \leq \log_2\left(\frac{n\ell}{4t_0}\right)$$

definiert, wobei $\pi$ die Resolutionswiderlegung der Tseitin-Formel über $G_{n\times\ell}$ ist und $|\mathrm{crit}(\cdot)|$ zum Komplexitätsmaß aus Definition 7.3.23 umdefiniert werden kann. Wir verweisen insbesondere auf Gleichung (7.27), um den Kern unserer Idee zu unterstreichen.

Ist es uns möglich, ein stark beschränktes Komplexitätsmaß für unsere Zwecke zu definieren (dies werden wir in Definition 7.3.23 tun), erhalten wir hieraus einen *Fortschrittsbegriff* in verschiedenen *Epochen*, also Zeitabschnitten, des Beweises. Wir konkretisieren dies in der folgenden Definition, für die es aus Notationsgründen praktischer ist, die Formeln in einem Beweis mit $0, 1, 2, \ldots$ statt $1, 2, 3, \ldots$ zu nummerieren. Es ist außerdem nützlich, davon auszugehen, dass in jedem Schritt des Beweises entweder ein Axiom-Download oder eine Resolvierung stattfindet (ggfs. kombiniert mit einer Löschung). Wir erinnern an die Festlegung von $\mathrm{Le}(\pi)$ in Definition 4.1.7 (i).

**Definition & Konvention 7.3.15** (Epochen und Subepochen; Adaptiert aus [BBI2016])**.** Es sei $\pi = (\varphi_0, \ldots, \varphi_{t-1})$ ein Beweis in einem sequentiellen Beweissystem (z. B. Resolution; vgl. Abschnitt 3.5.2). Für die nachfolgende Definition sei $h \in \mathbb{N}$ die *Rekursionstiefe der Epochenunterteilung* und $m \in \mathbb{N}$ die *Subepochenanzahl pro Epoche*, wobei wir die Konvention treffen, dass diese Konstanten so gewählt sind, dass $M \cdot m^h = |\pi|$ für ein $M \in \mathbb{N}$ gilt.

a) i) Eine *Epoche des Levels* $1 \leq i \leq h$ ist ein Intervall von $M \cdot m^i$ aufeinanderfolgenden Formeln im Beweis $\pi$, sodass die erste Formel des Intervalls Index $j \cdot M \cdot m^i$ mit einem $j \in \mathbb{N}_0$ hat. Epochen des Levels 1 nennen wir auch *Blattepochen*.

a) ii) Ist $E$ eine Epoche des Levels $2 \leq i \leq h$, so sind die *Subepochen* von $E$ diejenigen $m$ Epochen des Levels $i-1$, die in der Epoche $E$ enthalten sind.

b) i) Ist $E$ eine Epoche des Levels $2 \leq i \leq h$, so ist die *charakteristische Menge von* $E$ die Menge aller Formeln, die am Ende irgendeiner Subepoche von $E$ im Speicher waren. Elemente der charakteristischen Menge von $E$ nennen wir auch *charakteristische Formeln* von $E$.

b) ii) Ist $E$ eine Blattepoche, so bezeichnen wir alle Formeln in $E$ als *charakteristisch*.

Um die Schreibweise kurz zu halten, sagen wir auch $\pi$ ist rekursiv mit Parametern $(h, m)$ in Epochen unterteilt.

Wir verweisen für eine Illustration dieser Definition auf Abbildung 7.15.

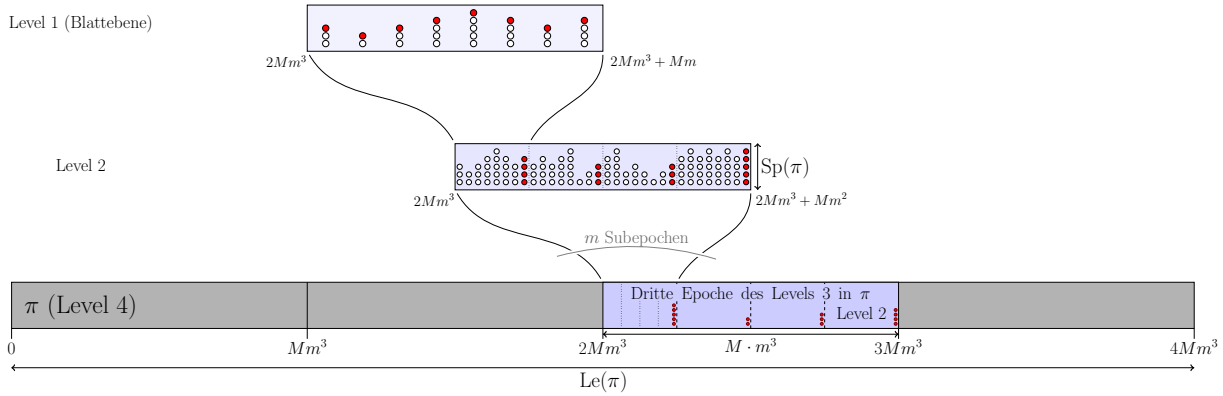



Abbildung 7.15.: Beispielhaft betrachten wir einen Beweis $\pi$ (in grau), der mit den Parametern $M = 2$, $m = 4$ und $h = 4$ rekursiv in Epochen zerlegt wurde. Die Punktstapel repräsentieren die zum jeweiligen Zeitpunkt gespeicherten Klausel, d. h. die Abbildung stellt ein *Zeit-Platz-Tableau* dar. Wir betrachten beispielhaft die dritte Epoche des Levels 3 im Beweis (im dunkelsten Blauton markiert). Diese besitzt $m$ Subepochen. Jede dieser Subepochen des Levels 2 besitzt ihrerseits wiederum $m$ Blattepochen. Alle während dieser Blattepoche gedownloadeten oder resolvierten Klauseln sind charakteristisch (die rote obersten Punkte der Stapel in der Blattepoche symbolisieren diese „frisch“ hinzugekommenen Klauseln). Die charakteristischen Klauseln der Epochen des Levels 2 finden sich jeweils zum Ende ihrer Blattepochen (Klauseln, die nicht charakteristisch sind, sind als weiße Punkte gezeichnet). Analog finden sich die charakteristischen Klauseln einer Epoche des Levels 3 jeweils an den Endpunkten ihrer $m$ Subepochen (wir haben hier aus Darstellungsgründen nur charakteristische Klauseln eingezeichnet).

Die einfachen Feststellungen der folgenden Notiz werden sich als von größter Wichtigkeit im Beweis des Haupttheorems 7.0.1 (iii) in Theorem 7.3.27 herausstellen. Insbesondere haben wir Definition & Konvention 7.3.15 derart (im Gegensatz zu [BBI2016, BNT2013, Bec2017]) modifiziert, dass wir einen Skalierungsfaktor $M$ mitdefinieren, den wir im Beweis von Theorem 7.3.27 geschickt wählen – dies stellt eine Kernidee des Beweises dar (auch in [BNT2013, Bec2017], wo das technische Detail des Skalierungsfaktors in der Definition nicht verwendet wird, im Beweis aber implizit benutzt wird).

**Notiz 7.3.16.** Es gelten folgende zwei Fakten über die Größe von Epochen und charakteristischen Mengen:

(a) Durch die obige Konvention haben Epochen des Levels $h$ Länge $T := \text{Le}(\pi) = M \cdot m^h$ (sind also der komplette Beweis), Epochen des Levels $h-1$ Länge $\frac{T}{m} = M \cdot m^{h-1}$, ..., Epochen des Levels 1 Länge $\frac{T}{m^{h-1}} = m \cdot M$.

(b) Blattepochen besitzen also genau $m \cdot M$ charakteristische Formeln; Epochen $E$ des Levels $2 \leq i \leq h$ besitzen maximal $m \cdot \text{Sp}(\pi)$ charakteristische Formeln, da an jedem der $m$ Subepochen-Endpunkte von $E$ höchstens Sp($\pi$) Formeln im Speicher sein können.

**Intuition 7.3.17.** Die charakteristische Menge einer Epoche kann als Ausschnitt der Formeln verstanden werden, die während der Epoche im Speicher sind. Anschaulich kann man sich die Menge als mehrere übereinandergelegte Folien mit „Schnappschüssen“ der sich im Speicher befindlichen Formeln vorstellen.

Je niedriger das Epochenlevel ist, desto genauer bildet die charakteristische Menge die Menge *aller* sich während der Epoche im Speicher befindlichen Formeln bzw. ihre Komplexitäten ab; die charakteristischen Mengen des Epochenlevels 1 sind präzise Abbildungen für die neu hinzugekommenen Klauseln.

**Hilfslemma 7.3.18** (Die maximale Anzahl an Epochen)**.** *Ist ein Beweis rekursiv mit Parametern $(h, m)$ in Epochen unterteilt, so gibt es höchstens $m^h$ Epochen.*

*Beweis.* Für $1 \leq i \leq h$ bezeichne $A(i)$ die Anzahl der Epochen des Levels $i$. Offensichtlich ist $E(h) = 1$, wie in Notiz 7.3.16 (a) erörtert, da die Epoche des Levels $h$ als ganzer Beweis festgelegt wurde. Da $m$ die Subepochenanzahl pro Epoche ist, erhalten wir mit der geometrischen Summe für die Gesamtanzahl an Epochen

$$\begin{aligned} &A(h) + A(h-1) + A(h-2) \cdots + A(2) + A(1) \\ &\qquad = 1 + m^1 + m^2 + \cdots + m^{h-2} + m^{h-1} = \frac{m^h - 1}{m - 1} \leq m^h - 1 \leq m^h. \end{aligned}$$

Lässt sich die geometrische Summe wegen $m = 1$ oben nicht anwenden, ist die Abschätzung des Hilfslemmas trivial, da es nur eine Epoche gibt. □

**Konvention 7.3.19.** Ohne Beschränkung der Allgemeinheit können wir im Folgenden davon ausgehen, dass die erste Klauselkonfiguration eines Resolutionsbeweises, abweichend von Definition 4.1.4 (iia), bereits die zu widerlegende Formel ist. Sollte dies nicht der Fall sein, könnten wir die Axiome der Formel zunächst in den Speicher „einlesen“. Da die folgenden Resultate allerdings superlinearen Platz betrachten und wir von Landau-Symbolen Gebrauch machen, ändern sich die Ergebnisse hierdurch nicht, die Beweise können aber mit weniger technischem Aufwand durchgeführt werden.

Inspiriert durch Satz 7.3.12 erhalten wir durch Einteilung des Beweises in Epochen nachfolgende Proposition.

**Proposition 7.3.20** (Eine Epoche mit vielen verschiedenen Komplexitäten)**.** *Es sei $F$ eine unerfüllbare* CNF*-Formel, sodass es ein stark beschränktes Komplexitätsmaß $\mu^*$ für $F$ im Beweissystem der Resolution gibt und $\mu^*(\square) =: L$ für ein passendes $L \in \mathbb{N}$ gilt. Weiter*

*sei die Widerlegung $\pi$ von $F$ rekursiv mit Parametern $(h, m)$ in Epochen unterteilt. Dann gilt für jede natürliche Zahl $k \leq \min\left\{m \operatorname{Sp}(\pi), \frac{T}{m^{h-1}}\right\}$* mindestens *eine der folgenden zwei Aussagen:*

*(i) Es existiert eine Blattepoche mit $L \cdot k^{-h+1}$ Formeln von paarweise verschiedener $\mu^*$-Komplexität (insbesondere hat die charakteristische Menge dieser Blattepoche $L \cdot k^{-h+1}$ Formeln von paarweise verschiedener $\mu^*$-Komplexität).*

*(ii) Es gibt eine Epoche des Levels $2 \leq i \leq h$, sodass die charakteristische Menge dieser Epoche mindestens $k$ Formeln von paarweise verschiedener $\mu^*$-Komplexität enthält.*

**Bemerkung 7.3.21.** Obige Proposition kann leicht auf sequentielle Beweissysteme verallgemeinert werden. Der nachfolgende Beweis kann dazu übernommen werden. Siehe hierfür [Bec2017, Lemma 4.55]. Unser Beweis beruht auf dem korrigierten Beweis [Bec2017, Lemma 4.55], kombiniert mit einer starken Modifikation von [BBI2016, Lemma 19].

*Beweis von Proposition 7.3.20.* Es sei $k \leq \min\left\{m \operatorname{Sp}(\pi), \frac{T}{m^{h-1}}\right\}$ eine beliebige natürliche Zahl. Durch diese Wahl ist dank Notiz 7.3.16 sichergestellt, dass $k$ nicht größer als die Kardinalität der größten charakteristischen Menge ist.

Gilt Aussage (ii), so sind wir bereits fertig, da nichts mehr zu zeigen ist. Daher können wir ohne Beschränkung der Allgemeinheit annehmen, dass Aussage (ii) nicht gilt und uns darauf beschränken, aus der Nichtgültigkeit von Aussage (ii) Aussage (i) zu folgern. Angenommen also es gibt keine Epoche des Levels $2 \leq i \leq h$, deren charakteristische Menge $k$ Formeln von paarweise verschiedener $\mu^*$-Komplexität enthält.

Induktiv definieren wir mit Hilfe dieser Annahme eine Folge von *Fortschritts-Lücken-Epochen* $(E_j)_{j=0}^{h-1}$ und Folgen natürlicher Zahlen $(i_{\text{low},j})_{j=0}^{h-1}, (i_{\text{high},j})_{j=0}^{h-1} \subset \{0, 1, 2, \ldots, L\}$, sodass für alle $j = 0, \ldots, h-1$ gilt:

- Epoche $E_j$ ist aus Level $h - j$, und
- alle sich im Speicher befindlichen Klauseln zu Beginn der Epoche $E_j$ haben $\mu^*$-Komplexität höchstens $i_{\text{low},j}$, und
- zumindest eine sich im Speicher befindliche Klausel am Ende der Epoche $E_j$ hat $\mu^*$-Komplexität mindestens $i_{\text{high},j}$, und
- es gilt $i_{\text{high},j} \geq i_{\text{low},j} + \dfrac{L}{k^j}$.

„$j = 0$“: Die Epoche $E_0$ soll aus Level $h$ sein und ist damit nach Notiz 7.3.16 (a) als der gesamte Beweis $\pi$ zu wählen. Zu Beginn des Beweises sind gemäß Konvention 7.3.19 lediglich Axiom-Klauseln von $F$ im Speicher. Diese haben $\mu^*$-Komplexität 0 nach Definition 7.3.13 (b), da $\mu^*$ ein stark beschränktes Komplexitätsmaß für $F$ ist. Am Ende des Beweises haben wir zumindest die leere Klausel im Speicher. Diese hat

nach Voraussetzung $\mu^*$-Komplexität $L$. Wir können also $i_{\text{low},0} := 0$ sowie $i_{\text{high},0} := L$ wählen. Wegen $L \geq 0 + \frac{L}{k^0} = L$ gilt damit auch die vierte Bedingung.

„$j \curvearrowright j+1$“: Angenommen wir haben $E_j$, $i_{\text{low},j}$ und $i_{\text{high},j}$ mit den vier Eigenschaften von oben gegeben. Wir betrachten die Partition der Epoche $E_j$ in ihre $m$ Subepochen des Levels $h-j-1$. Gemäß der Annahme, dass Aussage (ii) nicht gilt, hat die Epoche $E_j$ höchstens $k-1$ verschiedene $\mu^*$-Komplexitäten in ihrer charakteristischen Menge. Folglich gibt es, wie wir in Abbildung 7.16 beispielhaft illustriert und näher erläutert haben, ein echtes Teilintervall $[i_{\text{low},j+1}, i_{\text{high},j+1}] \subsetneq [i_{\text{low},j}, i_{\text{high},j}]$ mit Länge mindestens $\frac{i_{\text{high},j} - i_{\text{low},j}}{k} \geq \frac{L}{k^{j+1}}$, sodass keine Klausel aus der charakteristischen Menge von $E_j$ eine $\mu^*$-Komplexität in $[i_{\text{low},j+1}, i_{\text{high},j+1}]$ hat. Wir betrachten die erste Subepoche von $E_j$, welche mit einer Klausel von $\mu^*$-Komplexität mindestens $i_{\text{high},j+1}$ im Speicher endet (da sich am Ende der Epoche $E_j$ mindestens eine Klausel von $\mu^*$-Komplexität $i_{\text{high},j} \geq i_{\text{high},j+1}$ im Speicher befindet, muss eine solche Subepoche, wie gerade beschrieben, existieren). Da aber keine Klausel mit $\mu^*$-Komplexität mindestens $i_{\text{high},j+1}$ zu Beginn dieser Subepoche im Speicher war (da wir ansonsten die Vorgänger-Subepochen gemäß obiger Anweisung hätten wählen müssen) und keine Klausel mit $\mu^*$-Komplexität zwischen $i_{\text{low},j+1}$ und $i_{\text{high},j+1}$ zu Beginn irgendeiner Subepoche im Speicher ist, können wir die oben spezifizierte Subepoche als $E_{j+1}$ wählen.

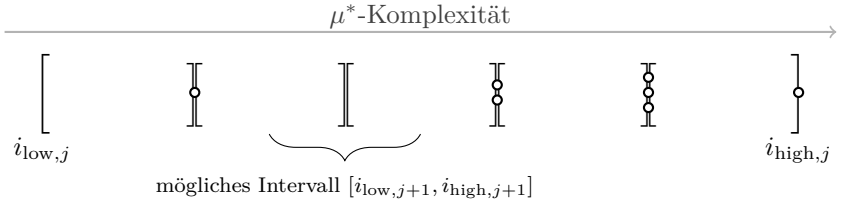



Abbildung 7.16.: Das Intervall $[i_{\text{low},j}, i_{\text{high},j}]$ wird äquidistant in $k$ gleich große Subintervalle partitioniert. Da es nur $k-1$ verschiedene $\mu^*$-Komplexitäten in $E_j$ gibt und mindestens eine Klausel $\mu^*$-Komplexität $i_{\text{high},j}$ besitzt, ergibt sich selbst bei einer „best-case“ Verteilung der übrigen $k-2$ Komplexitäten (beispielhaft angedeutet durch die Punkte) ein Intervall $[i_{\text{low},j+1}, i_{\text{high},j+1}]$ der Länge mindestens $\frac{i_{\text{high},j} - i_{\text{low},j}}{k} \geq \frac{L}{k^{j+1}}$, sodass keine Klausel eine Komplexität in diesem Intervall hat. Dies ist ein elementares Schubfach/Taubenschlag-Argument.

In der Abbildung ist beispielhaft $k = 5$, also $k - 1 = 4$.

Induktiv erhalten wir schließlich die Blattepoche $E_{h-1}$ und ein Intervall $[i_{\text{low},h-1}, i_{\text{high},h-1}]$ mit Länge mindestens $\frac{L}{k^{h-1}}$, sodass am Anfang der Epoche alle Klauseln im Speicher höchstens $\mu^*$-Komplexität $i_{\text{low},h-1}$ haben und sich am Ende der Epoche zumindest eine Klausel mit $\mu^*$-Komplexität $i_{\text{high},h-1}$ im Speicher befindet. Zwischenzeitlich müssen also

gemäß Satz 7.3.12 (angepasst auf stark beschränkte Komplexitätsmaße) Klauseln mit $\mu^*$-Komplexität zwischen $i_{\text{low},h-1}$ und $i_{\text{high},h-1}$ resolviert werden, also Klauseln mit mindestens $\frac{L}{k^{h-1}}$ paarweise verschiedenen $\mu^*$-Komplexitäten. Da alle Klauseln einer Blattepoche charakteristische Klauseln sind, folgt Aussage (ii), wie gewünscht. □

### 7.3.4 Die Verbindung zwischen Isoperimetrie und Komplexitätsmaßen

Um eine Verbindung zwischen Isoperimetrie und Komplexitätsmaßen herzustellen, benutzen wir die folgende Proposition, die durch den Artikel [BW2001] inspiriert wurde. Wir formulieren diese in einer allgemeineren Form, als für unsere Zwecke nötig, da der Beweis in der Sprache der linearen Algebra deutlich natürlicher geführt werden kann (man beachte jedoch, dass eine Umformulierung in die Sprache der Aussagenlogik möglich ist). Zudem merken wir an, dass durch diese allgemeinere Formulierung der Beweis des Tradeoff-Resultates in Theorem 7.3.27 durch eine geringfügige Modifikation (und eine geringfügige Anpassung des Komplexitätsmaßes in Definition 7.3.23) für das Beweissystem PCR übernommen werden (wir verweisen auf [Bec2017, § 4.7.4] und [BNT2013] für eine Ausführung dieser Beweismodifikation).

**Proposition 7.3.22.** *Es sei $G = (V, E)$ ein zusammenhängender Graph mit einer ungeraden Markierung $\chi$. Weiter sei $\varphi$ eine Disjunktion von linearen Gleichungen über dem Körper $\mathbb{F}_2$ in den Variablen $\{x_e : e \in E\}$ der Tseitin-Formel $\mathcal{T}(G, \chi)$. Außerdem sei $S$ eine minimale Teilmenge der Knoten $V$, sodass $\{\text{PARITY}_{v,\chi(v)}\}_{v\in S} \vDash \varphi$ gilt. Dann gilt die Inklusion*

$$\text{Var}(\varphi) \supset \{x_e : e \in \delta(S)\}.$$

Wir folgen im Beweis [Bec2017, Lemma 4.56].

*Beweis.* Aus den Voraussetzungen der Proposition folgt sofort, dass $S$ eine minimale Teilmenge der Knoten $V$ ist, sodass $\{\text{PARITY}_{v,\chi(v)}\}_{v\in S} \wedge \neg\varphi \vDash \square$ gilt. Da $\varphi$ eine Disjunktion von $\mathbb{F}_2$-Gleichungen ist, ist $\neg\varphi$ eine Konjunktion von $\mathbb{F}_2$-Gleichungen, weshalb $\mathscr{S} := \{\text{PARITY}_{v,\chi(v)}\}_{v\in S} \wedge \neg\varphi$ ein inkonsistentes CNF-Gleichungssystem darstellt.

Ohne Beschränkung der Allgemeinheit können wir annehmen, dass $\neg\varphi$ eine Lösung besitzt (bzw. erfüllbar ist); falls dies nicht der Fall ist, impliziert die Minimalität von $S$ wegen (der entsprechend angepassten Version von) Proposition A.0.1, dass $S = \varnothing$ gilt, wodurch die Aussage der Proposition nichtssagend ist.

Mittels elementarer linearer Algebra ist es also möglich den Widerspruch $1 \overset{\lightning}{=} 0$ aus dem System $\mathscr{S}$ durch eine Linearkombination $\mathscr{L}$ über dem Körper $\mathbb{F}_2$ herzuleiten. Angenommen, um einen Widerspruch zu erhalten, in eben dieser Linearkombination gibt es eine Glei-

chung $\mathrm{PARITY}_{v_*,\chi(v_*)}$ mit einem Koeffizienten 0. Dann ist es möglich, diese Gleichung aus der Linearkombination zu entfernen, sodass es immer noch möglich ist, aus dem Teilsystem $\mathscr{S} \setminus \mathrm{PARITY}_{v_*,\chi(v_*)}$ den Widerspruch $1 \overset{\lightning}{=} 0$ mit einer $\mathbb{F}_2$-Linearkombination herzuleiten. Ein Widerspruch zur Minimalität von $S$! Folglich hat jede $\mathrm{PARITY}_{v,\chi(v)}$-Gleichung in der Linearkombination $\mathscr{L}$ Koeffizient 1, was bedeutet, dass die $\mathbb{F}_2$-Summe der $\mathrm{PARITY}_{v,\chi(v)}$-Gleichungen inkonsistent mit $\neg\varphi$ ist.

Angenommen, erneut um einen Widerspruch zu erhalten, es gäbe eine Variable in dieser Summe, die keine Variable in $\neg\varphi$ ist. Dann wäre die Summe und $\neg\varphi$ erfüllbar (da wir angenommen haben, dass $\neg\varphi$ erfüllbar ist). Ein Widerspruch zu den Überlegungen unseres letzten Absatzes! Also muss die Inklusion

$$\mathrm{Var}\left(\sum_{v\in S} \mathrm{PARITY}_{v,\chi(v)}\right) \subset \mathrm{Var}(\neg\varphi) \tag{7.24}$$

gelten. Aufgrund der Struktur von Tseitin-Formeln taucht jede Variable $x_e$ in genau zwei PARITY-Gleichungen auf. Ist also $e \in \delta(S)$, dann taucht $x_e$ in genau einem der *Summanden* aus $\sum_{v\in S} \mathrm{PARITY}_{v,\chi(v)}$ auf, weshalb sich die Variable in der $\mathbb{F}_2$-Summe nicht „wegkürzt“, wir also die Inklusion

$$\{x_e \,:\, e \in \delta(S)\} \subset \mathrm{Var}\left(\sum_{v\in S} \mathrm{PARITY}_{v,\chi(v)}\right) \tag{7.25}$$

haben. Aus (7.24) und (7.25) folgt wegen $\mathrm{Var}(\neg\varphi) = \mathrm{Var}(\varphi)$ die Behauptung. □

Um die Beobachtung in Proposition 7.3.22 auf Komplexitätsmaße zu übertragen, und damit einen Zusammenhang zwischen Isoperimetrie in Graphen und Komplexität von Klauseln in einem Resolutionsbeweis herzustellen, konstruieren wir das im Folgenden definierte Komplexitätsmaß „geschickt“. Es wird sich herausstellen, dass das Standard-Komplexitätsmaß für Tseitin-Formeln wie in [BW2001] eingeführt, unseren Anforderungen genügt. Wir merken ebenfalls an, dass nachfolgendes Komplexitätsmaß mit der in [BBI2016] ursprünglich studierten Funktion $|\mathrm{crit}_{\mathcal{T}}(\cdot)|$ korrespondiert.

**Definition 7.3.23.** Es sei $G$ ein zusammenhängender Graph und $\chi$ eine ungerade Markierung des Graphen, sowie $\mathcal{T} := \mathcal{T}(G,\chi)$ die zugehörige Tseitin-Formel. Wir definieren das Komplexitätsmaß $\mu_{\mathcal{T}}$ für eine Klausel $C$ (über den Variablen der Tseitin-Formel) durch

$$\mu_{\mathcal{T}}(C) := \min_{\substack{\varnothing\neq S\subset V(G),\\ \bigwedge_{v\in S}\mathrm{PARITY}_{v,\chi(v)} \vDash C}} |S|.$$

Weiter bezeichnen wir mit $\text{Zeuge}_{\mu_\mathcal{T}}(C)$ eine[46] der Mengen $S$, die das obige Minimum realisieren.

Die obige Definition ist dabei dergestalt gewählt, dass die in [BW2001, S. 14f.] erörterten Eigenschaften gelten: Die Ursprungsklauseln haben Komplexität 1, die gewünschte leere Klausel hat „große“ Komplexität (also ist keine „kleine“ Teilmenge der Formel widersprüchlich, m. a. W. die Formel ist „hart“) und das Komplexitätsmaß ist subadditiv bzgl. Resolutionsschritten. Wir fassen diese Eigenschaften in folgender Proposition zusammen.

**Proposition 7.3.24.** *Es sei $G$ ein zusammenhängender Graph und $\chi$ eine ungerade Markierung des Graphen, sowie $\mathcal{T} := \mathcal{T}(G, \chi)$ die zugehörige Tseitin-Formel.*

*(i) Für jede Ursprungsklausel (d. h. jedes Axiom) $C \in \mathcal{T}$ gilt $\mu_\mathcal{T}(C) = 1$.*

*(ii) Es gilt $\mu_\mathcal{T}(\square) = |V(G)|$.*

*(iii) Das Komplexitätsmaß $\mu_\mathcal{T}$ ist* subadditiv*: Gilt $\{C_1, C_2\} \vdash^1_\mathfrak{R} C$, so ist $\mu_\mathcal{T}(C) \leq \mu_\mathcal{T}(C_1) + \mu_\mathcal{T}(C_2)$.*

*(iv) Jede zu den Randkanten von $\delta_G\Big(\text{Zeuge}_{\mu_\mathcal{T}}(C)\Big)$ korrespondierende Variable ist auch Variable in der Klausel $C$, in Zeichen*

$$\Big\{x_e \,:\, e \in \delta_G\Big(\text{Zeuge}_{\mu_\mathcal{T}}(C)\Big)\Big\} \subset \text{Var}(C) \quad \textit{für alle } C \in \mathcal{T}.$$

Wir beweisen nun Proposition 7.3.24 – dies entspricht für die Punkte (i) und (ii) dem Beweis des Hauptresultates aus [BW2001, § 6.1], dessen Ausführungen wir folgen werden.

*Beweis.* (i) Offensichtlich ist $\mathbb{N}_0$ der Bildbereich von $\mu_\mathcal{T}$. Ist $C \in \mathcal{T}$ eine Ursprungsklausel der Tseitin-Formel, so gilt nach Konstruktion (5.4) der Tseitin-Formel $C \in \text{PARITY}_{v,\chi(v)}$ für einen Knoten $v \in V(G)$. Für eben diesen Knoten $v$ gilt nun $\text{PARITY}_{v,\chi(v)} \models C$, d. h. $S = \{v\}$ ist ein Zeuge für den Fakt $\mu_\mathcal{T}(C) = 1$.

(ii) Tseitin-Formeln können nicht widerlegt werden, ohne die übersetzten Paritätsbedingungen (5.1) für alle Knoten zu benutzen. Wir zeigen diese Tatsache, indem wir nachweisen, dass für jede echte Teilmenge $V' \subsetneq V(G)$ die „verkleinerte“ Tseitin-Formel $\bigwedge_{v \in V'} \text{PARITY}_{v,\chi(v)}$ erfüllbar ist.

Dazu sei $v \in V(G) \setminus V'$. Wir betrachten die Formel $\mathcal{T}(G, \widetilde{\chi})$ mit der *geraden* Markierung

$$\widetilde{\chi}(u) := \begin{cases} 1 - \chi(u), & \text{falls } u = v \\ \chi(u), & \text{sonst.} \end{cases}$$

[46] Für unsere Zwecke ist es irrelevant, genau zu spezifizieren, welcher der möglicherweise mehreren Zeugen mit dieser Definition ausgewählt werden soll.

Theorem 5.0.6 liefert $\mathcal{T}(G, \widetilde{\chi}) \in \text{SAT}$. Wegen

$$\bigwedge_{v \in V'} \text{PARITY}_{v,\widetilde{\chi}(v)} \wedge \bigwedge_{v \in V(G) \setminus V'} \text{PARITY}_{v,\widetilde{\chi}(v)} = \mathcal{T}(G, \widetilde{\chi})$$

ist $\bigwedge_{v \in V'} \text{PARITY}_{v,\chi(v)} = \bigwedge_{v \in V'} \text{PARITY}_{v,\widetilde{\chi}(v)}$ eine Teilformel von $\mathcal{T}(G, \widetilde{\chi})$, also ebenfalls erfüllbar.

(iii) Wegen der Korrektheit des Resolutionskalküls 3.5.5 (a) folgt aus $\{C_1, C_2\} \vdash^1_{\mathfrak{R}} C$ direkt

$$\{C_1, C_2\} \vDash C. \tag{7.26}$$

Unmittelbar aus der Definition 7.3.23 von Zeugen ist klar, dass

$$\bigwedge_{v \in \text{Zeuge}_{\mu_\mathcal{T}}(C_1) \cup \text{Zeuge}_{\mu_\mathcal{T}}(C_2)} \text{PARITY}_{v,\chi(v)} \vDash C_i \quad \text{für } i = 1, 2$$

gilt. Damit gilt dank (7.26) auch

$$\bigwedge_{v \in \text{Zeuge}_{\mu_\mathcal{T}}(C_1) \cup \text{Zeuge}_{\mu_\mathcal{T}}(C_2)} \text{PARITY}_{v,\chi(v)} \vDash C.$$

Wegen der Definition 7.3.23 des Komplexitätsmaßes $\mu_\mathcal{T}$ erhalten wir schließlich

$$\begin{aligned} \mu_\mathcal{T}(C) &\leq \left|\text{Zeuge}_{\mu_\mathcal{T}}(C_1) \cup \text{Zeuge}_{\mu_\mathcal{T}}(C_2)\right| \\ &\leq \left|\text{Zeuge}_{\mu_\mathcal{T}}(C_1)\right| + \left|\text{Zeuge}_{\mu_\mathcal{T}}(C_2)\right| \\ &= \mu_\mathcal{T}(C_1) + \mu_\mathcal{T}(C_2). \end{aligned}$$

(iv) Da eine Klausel als Disjunktion von linearen Gleichungen über dem Körper $\mathbb{F}_2$ aufgefasst werden kann, folgt diese Aussage sofort aus Proposition 7.3.22. □

Durch Proposition 7.3.24 (iv) haben wir also einen Weg gefunden „Randeigenschaften" der Graphenstruktur auf die Weite von Klauseln zu übertragen. Wir werden diese Eigenschaft im folgenden Abschnitt nutzen, um zu zeigen, dass Isoperimetrie einen Zeit-Platz Tradeoff impliziert.

### 7.3.5 Isoperimetrie impliziert einen Zeit-Platz Tradeoff

**Proposition 7.3.25** (Verschiedene Komplexitäten implizieren kollektive Weite)**.** *Es sei $G$ ein zusammenhängender Graph, der die erweiterte isoperimetrische Eigenschaft mit Parametern $(W, t_0, r = 4)$ erfüllt. Außerdem sei $\chi$ eine ungerade Markierung dieses Graphen,*

*sowie* $\mathcal{T} := \mathcal{T}(G, \chi)$ *die zugehörige Tseitin-Formel. Weiter sei* $\mu^*_{\mathcal{T}}$ *das stark beschränkte Komplexitätsmaß für* $\mathcal{T}$*, das sich aus dem kanonischen Komplexitätsmaß* $\mu_{\mathcal{T}}$ *für Tseitin-Formeln aus Definition 7.3.23 via*

$$\mu^*_{\mathcal{T}}(C) = \begin{cases} 0, & \text{falls } \mu_{\mathcal{T}}(C) < t_0 \\ \left\lfloor \log_2\left(\frac{\mu_{\mathcal{T}}(C)}{t_0}\right)\right\rfloor, & \text{falls } t_0 \leq \mu_{\mathcal{T}}(C) \leq \frac{|V|}{4} \\ L := \left\lfloor \log_2\left(\frac{|V(G)|}{4t_0}\right)\right\rfloor, & \text{falls } \mu_{\mathcal{T}}(C) > \frac{|V|}{4} \end{cases} \tag{7.27}$$

*ergibt. Dann gilt für jede Folge von Klauseln* $C_1, \ldots, C_k$ *mit paarweise verschiedenen Bildern unter* $\mu^*_{\mathcal{T}}$ *in* $[0, L]$ *die folgende untere Schranke zur* kollektiven Weite *dieser Klauseln:*

$$\left|\bigcup_{i=1}^{k} \mathrm{Var}(C_i)\right| \geq \Omega(kW). \tag{7.28}$$

Um den Beweis von Proposition 7.3.25 übersichtlich zu halten, formulieren wir zunächst ein Hilfslemma, dessen Beweis wir auf den folgenden Seiten nachreichen.

Wir merken außerdem an, dass der Nachweis, dass $\mu^*_{\mathcal{T}}$ ein stark beschränktes Komplexitätsmaß ist, durch eine ähnliche Fallunterscheidung wie im folgenden Beweis von Hilfslemma 7.3.26 leicht nachgewiesen werden kann – wir verzichten an dieser Stelle auf einen formalen Beweis. Die starke Beschränktheit für $\mathcal{T}$ ergibt sich unmittelbar aus Proposition 7.3.24 (i). □

**Hilfslemma 7.3.26.** *Es seien* $\mu_{\mathcal{T}}$*,* $\mu^*_{\mathcal{T}}$*, sowie* $L$ *und weitere Bezeichnungen wie in Proposition 7.3.25. Ist weiter* $C_1, \ldots, C_k$ *eine beliebige Folge von Klauseln mit paarweise verschiedenen Bildern unter* $\mu^*_{\mathcal{T}}$ *in* $[0, L]$*, sodass*

$$\mu^*_{\mathcal{T}}(C_1) < \mu^*_{\mathcal{T}}(C_2) < \cdots < \mu^*_{\mathcal{T}}(C_k)$$

*gilt, dann gilt für alle* $i \in \{1, \ldots, k-3\}$ *die Implikation*

$$\mu^*_{\mathcal{T}}(C_i) + 3 \leq \mu^*_{\mathcal{T}}(C_{i+3}) \implies 4 \cdot \mu_{\mathcal{T}}(C_i) \leq \mu_{\mathcal{T}}(C_{i+3}). \tag{7.29}$$

Die Wahl in (7.27) war, neben den in Bemerkung 7.3.14 bereits erörterten Gründen, so getroffen, dass diese Implikation gilt. Wegen der Verwendung von Gaußklammern in (7.27) verzichten wir nicht auf einen formalen Beweis. Man beachte, dass Gaußklammern nötig sind, damit das Bild des Komplexitätsmaßes Teilmenge von $\mathbb{N}_0$ ist, was im Beweis von Proposition 7.3.20 ausgenutzt wurde.

*Beweis von Proposition 7.3.25.* Es sei $C_1, \ldots, C_k$ eine beliebige Folge von Klauseln mit

paarweise verschiedenen Bildern unter $\mu_{\mathcal{T}}^*$ in $[0, L]$. Dies erlaubt es uns, eine Permutation $\sigma\colon \{1, \dots, k\} \to \{1, \dots, k\}$ zu finden, die die Klauseln nach ihrem Wert unter $\mu_{\mathcal{T}}^*$ ordnet, d. h. es gilt

$$\mu_{\mathcal{T}}^*(C_{\sigma(1)}) < \mu_{\mathcal{T}}^*(C_{\sigma(2)}) < \cdots < \mu_{\mathcal{T}}^*(C_{\sigma(k)}).$$

Insbesondere gilt

$$\mu_{\mathcal{T}}^*(C_{\sigma(i+3)}) \geq \mu_{\mathcal{T}}^*(C_{\sigma(i)}) + 3 \quad \text{für alle } i = 1, \dots, k-3.$$

Hilfslemma 7.3.26 impliziert sofort

$$\mu_{\mathcal{T}}(C_{\sigma(i+3)}) \geq 4 \cdot \mu_{\mathcal{T}}(C_{\sigma(i)}) \quad \text{für alle } i = 1, \dots, k-3.$$

Wir erhalten

$$|\operatorname{Zeuge}_{\mu_{\mathcal{T}}}(C_{\sigma(i+3)})| \geq 4 \cdot |\operatorname{Zeuge}_{\mu_{\mathcal{T}}}(C_{\sigma(i)})| \quad \text{für alle } i = 1, \dots, k-3.$$

Mit anderen Worten ist $\left(\operatorname{Zeuge}_{\mu_{\mathcal{T}}}(C_{\sigma(3i+1)})\right)_{i=0}^{\lfloor \frac{k}{3} \rfloor - 1}$ eine 4-stark wachsende Folge von Knotenmengen im Graph $G$. Die nach Voraussetzung gültige erweiterte isoperimetrische Eigenschaft (7.14) liefert

$$\left| \bigcup_{i=0}^{\lfloor \frac{k}{3} \rfloor - 1} \delta_G\left(\operatorname{Zeuge}_{\mu_{\mathcal{T}}}(C_{\sigma(3i+1)})\right) \right| \geq \left\lfloor \frac{k}{3} \right\rfloor \cdot W.$$

Durch Proposition 7.3.24 (iv) erhalten wir

$$\left| \bigcup_{i=1}^{k} \operatorname{Var}(C_i) \right| \geq \left| \bigcup_{i=0}^{\lfloor \frac{k}{3} \rfloor - 1} \operatorname{Var}(C_{\sigma(3i+1)}) \right| \geq \left| \bigcup_{i=0}^{\lfloor \frac{k}{3} \rfloor - 1} \delta_G\left(\operatorname{Zeuge}_{\mu_{\mathcal{T}}}(C_{\sigma(3i+1)})\right) \right| \geq \left\lfloor \frac{k}{3} \right\rfloor \cdot W = \Omega(kW),$$

was den Beweis abschließt. □

*Beweis von Hilfslemma 7.3.26.* Es sei $i \in \{1, \dots, k-3\}$ fixiert. Da die Klauseln $C_i$ und $C_{i+3}$ nach Voraussetzung verschiedene $\mu_{\mathcal{T}}^*$-Komplexität haben, müssen wir lediglich die Fälle $\left(\mu_{\mathcal{T}}^*(C_i), \mu_{\mathcal{T}}^*(C_{i+3})\right)$„∈“$\big\{(0, \log), (0, L), (\log, \log), (\log, L)\big\}$, nicht aber die Fälle $(0, 0)$ und $(L, L)$, betrachten, wobei hier z. B. $\mu_{\mathcal{T}}^*(C) = \log$ bedeuten soll, dass $\mu_{\mathcal{T}}^*(C)$ gemäß der zweiten Spalte von (7.27) definiert sei.

Fall $(\log, \log)$: Gemäß Fallvoraussetzung gilt $\mu_{\mathcal{T}}^*(C_i) = \left\lfloor \log_2\left(\frac{\mu_{\mathcal{T}}(C_i)}{t_0}\right) \right\rfloor$, sowie $\mu_{\mathcal{T}}^*(C_{i+3}) =$

$\left\lfloor \log_2\left(\frac{\mu_{\mathcal{T}}(C_{i+3})}{t_0}\right)\right\rfloor$. Nach Voraussetzung (7.29) gilt

$$\left\lfloor \log_2\left(\frac{\mu_{\mathcal{T}}(C_i)}{t_0}\right)\right\rfloor + 3 = \mu^*_{\mathcal{T}}(C_i) + 3 \leq \mu^*_{\mathcal{T}}(C_{i+3}) = \left\lfloor \log_2\left(\frac{\mu_{\mathcal{T}}(C_{i+3})}{t_0}\right)\right\rfloor,$$

also ist

$$2^{\left\lfloor \log_2\left(\frac{\mu_{\mathcal{T}}(C_i)}{t_0}\right)\right\rfloor + 3} \leq 2^{\left\lfloor \log_2\left(\frac{\mu_{\mathcal{T}}(C_{i+3})}{t_0}\right)\right\rfloor}. \tag{7.30}$$

Wegen

$$4 \cdot \frac{\mu_{\mathcal{T}}(C_i)}{t_0} = 4 \cdot 2^{\log_2\left(\frac{\mu_{\mathcal{T}}(C_i)}{t_0}\right)} = 8 \cdot 2^{\log_2\left(\frac{\mu_{\mathcal{T}}(C_i)}{t_0}\right)-1} \leq 8 \cdot 2^{\left\lfloor \log_2\left(\frac{\mu_{\mathcal{T}}(C_i)}{t_0}\right)\right\rfloor} = 2^{\left\lfloor \log_2\left(\frac{\mu_{\mathcal{T}}(C_i)}{t_0}\right)\right\rfloor + 3}$$

und

$$2^{\left\lfloor \log_2\left(\frac{\mu_{\mathcal{T}}(C_{i+3})}{t_0}\right)\right\rfloor} \leq 2^{\log_2\left(\frac{\mu_{\mathcal{T}}(C_{i+3})}{t_0}\right)} = \frac{\mu_{\mathcal{T}}(C_{i+3})}{t_0}$$

folgt aus (7.30) sofort $4 \cdot \mu_{\mathcal{T}}(C_i) \leq \mu_{\mathcal{T}}(C_{i+3})$.

Fall $(\log, L)$: Gemäß Fallvoraussetzung gilt $\mu^*_{\mathcal{T}}(C_i) = \left\lfloor \log_2\left(\frac{\mu_{\mathcal{T}}(C_i)}{t_0}\right)\right\rfloor$, sowie $\mu^*_{\mathcal{T}}(C_{i+3}) = L$. Insbesondere ist $\frac{|V(G)|}{4} < \mu_{\mathcal{T}}(C_{i+3})$ nach (7.27). Nach Voraussetzung (7.29) gilt

$$\left\lfloor \log_2\left(\frac{\mu_{\mathcal{T}}(C_i)}{t_0}\right)\right\rfloor + 3 = \mu^*_{\mathcal{T}}(C_i) + 3 \leq \mu^*_{\mathcal{T}}(C_{i+3}) = L = \left\lfloor \log_2\left(\frac{|V(G)|}{4t_0}\right)\right\rfloor.$$

Wie im Fall $(\log, \log)$ folgert man hieraus in einer kurzen Rechnung

$$4 \cdot \frac{\mu_{\mathcal{T}}(C_i)}{t_0} \leq \frac{|V(G)|}{4t_0},$$

d. h. $16 \cdot \mu_{\mathcal{T}}(C_i) \leq |V(G)|$. Aus der Abschätzung

$$16 \cdot \mu_{\mathcal{T}}(C_i) \leq |V(G)| = 4 \cdot \frac{|V(G)|}{4} < 4 \cdot \mu_{\mathcal{T}}(C_{i+3})$$

folgt durch Kürzen sofort $4 \cdot \mu_{\mathcal{T}}(C_i) < \mu_{\mathcal{T}}(C_{i+3})$.

Fall $(0, \log)$: Gemäß Fallvoraussetzung gilt $\mu^*_{\mathcal{T}}(C_i) = 0$, sowie $\mu^*_{\mathcal{T}}(C_{i+3}) = \left\lfloor \log_2\left(\frac{\mu_{\mathcal{T}}(C_{i+3})}{t_0}\right)\right\rfloor$. Insbesondere ist $\mu_{\mathcal{T}}(C_i) < t_0$ nach (7.27). Nach Voraussetzung (7.29) gilt

$$\mu^*_{\mathcal{T}}(C_i) + 3 = 3 \leq \mu^*_{\mathcal{T}}(C_{i+3}) = \left\lfloor \log_2\left(\frac{\mu_{\mathcal{T}}(C_{i+3})}{t_0}\right)\right\rfloor.$$

Wie im Fall $(\log, \log)$ folgert man hieraus in einer kurzen Rechnung $8t_0 \leq \mu_{\mathcal{T}}(C_{i+3})$.

Dies liefert $4 \cdot \mu_{\mathcal{T}}(C_i) < 4t_0 < 8t_0 \leq \mu_{\mathcal{T}}(C_{i+3})$.

Fall $(0, L)$: Gemäß Fallvoraussetzung gilt $\mu^*_{\mathcal{T}}(C_i) = 0$, sowie $\mu^*_{\mathcal{T}}(C_{i+3}) = L$. Nach Voraussetzung (7.29) ist also

$$\mu^*_{\mathcal{T}}(C_i) + 3 = 3 \leq \mu^*_{\mathcal{T}}(C_{i+3}) = L = \left\lfloor \log_2 \left( \frac{|V(G)|}{4t_0} \right) \right\rfloor.$$

Eine einfache Rechnung liefert $8 \leq \frac{|V(G)|}{4t_0}$. Nach (7.27) haben wir außerdem $\mu_{\mathcal{T}}(C_i) < t_0$ und $\mu_{\mathcal{T}}(C_{i+3}) > \frac{|V(G)|}{4}$. Insgesamt also

$$4 \cdot \mu_{\mathcal{T}}(C_i) < 4t_0 < 8t_0 \leq \frac{|V(G)|}{4} < \mu_{\mathcal{T}}(C_{i+3}). \qquad \square$$

**Theorem 7.3.27** (Zeit-Platz Tradeoff durch Isoperimetrie)**.** *Ist $F$ eine unerfüllbare* CNF*-Formel mit einem stark beschränkten Komplexitätsmaß $\mu^*$ für $F$, welches die untere Schranke zur kollektiven Klauselweite* (7.28) *unter den in Proposition 7.3.25 genannten Voraussetzungen erfüllt, dann gilt der Zeit-Platz Tradeoff*

$$\mathrm{Le}(\pi) \geq \left( \frac{2^{\Omega(W)}}{\mathrm{Sp}(\pi)} \right)^{\Omega\left( \frac{\log_2 L}{\log_2 \log_2 L} \right)}$$

*mit $L := \mu^*(\square)$ für alle Resolutionswiderlegungen $\pi \colon F[\oplus_2] \vdash_{\mathfrak{R}} \square$ der XORification von $F$.*

*Beweis.* Wir fixieren einen beliebigen Beweis $\pi$ von der XORification $F[\oplus_2]$ mit Länge $T := \mathrm{Le}(\pi)$ und Platz $S := \mathrm{Sp}(\pi)$ und zerlegen ihn rekursiv mit Parametern $(h, m)$ in Epochen, d. h. jede Epoche wird rekursiv in $m$ Subepochen zerlegt bis zu einer Rekursionstiefe von $h$.

Wir spezifizieren nun die für diese Zerlegung und den weiteren Beweis benötigten Parameter: Dazu setzen wir zunächst $k := h$. Nun wählen wir $k \in \mathbb{N}$ so, dass

$$k \leq \min \left\{ mS, \frac{T}{m^{h-1}} \right\} \tag{7.31}$$

und

$$L \cdot k^{-h+1} > k \tag{7.32}$$

gilt, wobei wir $m$ so wählen, dass

$$m^h = \frac{T}{S} \tag{7.33}$$

gilt (dies korrespondiert mit der Wahl $M = S$ in Konvention 7.3.15, sodass gemäß Notiz 7.3.16 (a) & (b) die maximale Anzahl an charakteristischen Formeln für alle Epochen gleich ist und insbesondere die Menge $\left\{mS, \frac{T}{m^{h-1}}\right\}$, über die wir in (7.31) das Minimum gebildet haben, einelementig ist – vergleiche auch mit dem Zeit-Platz-Tableau in Abbildung 7.15).

Es sei weiter $\rho \sim \mathcal{Z}$ eine Zufallseinschränkung im erweiterten Sinne gemäß Definition 6.0.12. Wir halten fest, dass dank Notiz 6.0.14 die Gleichung $F[\oplus_2]\restriction_\rho = F$ gilt, also wie in Bemerkung 6.0.11 erörtert, die Einschränkung $\pi\restriction_\rho$ ein Beweis von $F$ ist. Wir halten einige Beobachtungen fest, aus denen der Tradeoff folgen wird:

(I) Da $\mu^*$ ein stark beschränktes Komplexitätsmaß für $F$ ist, lässt sich Proposition 7.3.20 anwenden, dank der wir

$$\operatorname*{Prob}_{\rho\sim\mathcal{Z}}\left[\begin{array}{c}\text{die charakteristische Menge einer Epoche}\\ \text{aus } \pi\restriction_\rho \text{ enthält mindestens } k \text{ Klauseln}\\ \text{von paarweise verschiedener } \mu^*\text{-Komplexität}\end{array}\right] = 1, \tag{7.34}$$

erhalten, denn es ist $L \cdot k^{-h+1} > k$, wie wir in (7.32) festgelegt hatten.

(IIa) Es sei nun $\mathscr{C}$ eine kleine Menge von Klauseln aus $\pi$. Wir werden nun jedoch zeigen, dass die Wahrscheinlichkeit, dass $\mathscr{C}\restriction_\rho$ Klauseln von $k$ paarweise verschiedener $\mu^*$-Komplexität enthält, gering ist: Fixiere dazu ein $k$-Tupel $(j_1, \dots, j_k)$, das $k$ Klauseln $C_{j_1}, \dots, C_{j_k}$ aus der Menge $\mathscr{C}$ spezifiziert. Angenommen die $k$ Klauseln $D_{j_1} := C_{j_1}\restriction_\rho, \dots, D_{j_k} := C_{j_k}\restriction_\rho$ aus $\mathscr{C}\restriction_\rho$ haben paarweise verschiedene $\mu^*$-Komplexität (d. h. mindestens $k-2$ dieser Klauseln haben paarweise verschiedene $\mu^*$-Komplexität aus $[1, L-1]$), dann muss nach Proposition 7.3.25 die kollektive Weite dieser $k$ Klauseln groß sein, präziser

$$\left|\operatorname{Var}\left(\bigvee_{i=1}^{k} D_{j_i}\right)\right| = \left|\bigcup_{i=1}^{k} \operatorname{Var}(D_{j_i})\right| \geq \Omega\Big((k-2)\cdot W\Big).$$

Jedoch liefert Satz 6.0.17 eine obere Schranke für die Wahrscheinlichkeit dieses Ereignisses: Setzen wir $\widehat{C} := \bigvee_{i=1}^{k} C_{j_i}$, dann ist

$$\begin{aligned}&\operatorname*{Prob}_{\rho\sim\mathcal{Z}}\left[\left|\operatorname{Var}\left(\bigvee_{i=1}^{k} D_{j_i}\right)\right| \geq \Omega\Big((k-2)\cdot W\Big)\right]\\ =&\operatorname*{Prob}_{\rho\sim\mathcal{Z}}\left[\left|\operatorname{Var}(\widehat{C}\restriction_\rho)\right| \geq \Omega\Big((k-2)\cdot W\Big)\right]\\ \leq&\left(\frac{3}{4}\right)^{\Omega((k-2)\cdot W)},\end{aligned}$$

was exponentiell klein in $k$ ist. Wir erhalten zusammen

$$\begin{aligned}\operatorname*{Prob}_{\rho\sim\mathcal{Z}}\left[A_{(D_{j_1},\ldots,D_{j_k})}\right] &:= \operatorname*{Prob}_{\rho\sim\mathcal{Z}}\left[\begin{array}{c}\text{die Klauseln } D_{j_1},\ldots,D_{j_k} \text{ aus } \mathscr{C}\restriction_\rho \text{ haben}\\ \text{paarweise verschiedene } \mu^*\text{-Komplexität}\end{array}\right]\\ &\leq \left(\frac{3}{4}\right)^{\Omega((k-2)\cdot W)}.\end{aligned}$$

Da wir das $k$-Tupel $(j_1,\ldots,j_k)$ zu Beginn fixiert haben, erhalten wir durch die $\sigma$-Subadditivität des Wahrscheinlichkeitsmaßes

$$\begin{aligned}&\operatorname*{Prob}_{\rho\sim\mathcal{Z}}\left[\begin{array}{c}\text{irgendein } k\text{-Tupel von Klauseln in } \mathscr{C}\restriction_\rho \text{ hat}\\ \text{paarweise verschiedene } \mu^*\text{-Komplexität}\end{array}\right]\\ &\leq \sum_{(C_{j_1},\ldots,C_{j_k})\in\binom{\mathscr{C}}{k}} \operatorname*{Prob}_{\rho\sim\mathcal{Z}}\left[A_{(C_{j_1}\restriction_\rho,\ldots,C_{j_k}\restriction_\rho)}\right]\\ &\leq \left|\binom{\mathscr{C}}{k}\right| \cdot \operatorname*{Prob}_{\rho\sim\mathcal{Z}}\left[A_{(C_{j_1}\restriction_\rho,\ldots,C_{j_k}\restriction_\rho)}\right]\\ &\leq |\mathscr{C}|^k \cdot \left(\frac{3}{4}\right)^{\Omega((k-2)\cdot W)},\end{aligned}$$

wobei wir ausgenutzt haben, dass $\left|\binom{\mathscr{C}}{k}\right| = \binom{|\mathscr{C}|}{k} \leq |\mathscr{C}|^k$ gilt.

(IIb) Fixieren wir eine beliebige charakteristische Menge $\operatorname{char}(E)$ einer Epoche $E$, so ergibt sich unmittelbar aus (IIa) die Abschätzung

$$\operatorname*{Prob}_{\rho\sim\mathcal{Z}}\left[\begin{array}{c}\operatorname{char}(E)\restriction_\rho \text{ enthält } k \text{ Klauseln mit paar-}\\ \text{weise verschiedener } \mu^*\text{-Komplexität}\end{array}\right] \leq (mS)^k \cdot \left(\frac{3}{4}\right)^{\Omega((k-2)\cdot W)},$$

da wegen (7.33) (bzw. der der Gleichung nachfolgenden Argumentation) die Abschätzung $|\operatorname{char}(E)| \leq mS$ für alle Epochen $E$ gilt. Da es gemäß Hilfslemma 7.3.18 höchstens $m^h$ Epochen geben kann, erhalten wir mit der $\sigma$-Subadditivität des Wahrscheinlichkeitsmaßes

$$\begin{aligned}&\operatorname*{Prob}_{\rho\sim\mathcal{Z}}\left[\begin{array}{c}\text{irgendein } \operatorname{char}(E)\restriction_\rho \text{ enthält } k \text{ Klauseln mit}\\ \text{paarweise verschiedener } \mu^*\text{-Komplexität}\end{array}\right]\\ &\leq (m^2S)^k \cdot \left(\frac{3}{4}\right)^{\Omega((k-2)\cdot W)} \leq \left(m^2S \cdot \left(\frac{3}{4}\right)^{\Omega(W)}\right)^k,\end{aligned} \tag{7.35}$$

wobei wir die Festlegung $h = k$ ausgenutzt haben.

Damit aus den Hauptergebnissen (7.34) aus (I) und (7.35) aus (IIb) jedoch kein Widerspruch entsteht, muss gelten

$$\left(m^2 S \cdot \left(\frac{3}{4}\right)^{\Omega(W)}\right)^k \geq 1.$$

Dies ist wegen $2^{-\Omega(W)} = \left(\frac{3}{4}\right)^{\Omega(W)}$ jedoch genau dann der Fall, wenn

$$m^2 \geq \frac{2^{\Omega(W)}}{S}$$

gilt. Aufgrund der Wahl (7.33) ergibt sich

$$\left(\frac{T}{S}\right)^{2/k} \geq \frac{2^{\Omega(W)}}{S},$$

also

$$T^{2/k} \geq \frac{2^{\Omega(W)}}{S} \cdot S^{2/k}$$

bzw.

$$T \geq \left(\frac{2^{\Omega(W)}}{S} \cdot S^{\frac{2}{k}}\right)^{\frac{k}{2}} = \frac{2^{\Omega(W)\cdot\frac{k}{2}}}{S^{\frac{k}{2}-1}} \geq \frac{2^{\Omega(W)\cdot\frac{k}{2}}}{S^{\frac{k}{2}}} = \left(\frac{2^{\Omega(W)}}{S}\right)^{k/2} = \left(\frac{2^{\Omega(W)}}{S}\right)^{\Omega(k)}.$$

Abschließend beobachten wir, dass wegen (7.32) die Ungleichung $k^k < L$ gilt. Man überprüft leicht, dass $k$ so groß wie $\frac{\log_2 L}{\log_2 \log_2 L}$ gewählt werden kann, ohne die Ungleichung zu verletzen. Wir erhalten also

$$T \geq \left(\frac{2^{\Omega(W)}}{S}\right)^{\Omega\left(\frac{\log_2 L}{\log_2 \log_2 L}\right)},$$

wie gewünscht. □

Als „Korollar“ erhalten wir unmittelbar unser Haupttheorem. Wir merken an, dass offensichtlich $n$ groß genug sein muss, damit der Exponent $\Omega(\log_2 \log_2 n / \log_2 \log_2 \log_2 n)$ einerseits definiert und andererseits positiv für alle größeren Werte von $n$ ist – dies ist unsere einzige Bedingung an $n$. Mithilfe eines Computer-Algebra-Programms prüft man leicht, dass $n \geq 5$ hinreichend ist und nicht weiter verbessert werden kann.

**Theorem 7.3.28** (Tradeoff (iii) in Haupttheorem 7.0.1)**.** *Für jede Resolutionswiderlegung $\pi_{n,\ell}$ von $F_{n,\ell} := \mathcal{T}(G_{n\times\ell}, \chi)[\oplus_2]$, wobei $G_{n\times\ell}$ der $(n \times \ell)$-Gittergraph mit $n \in \mathbb{N}_{\geq 5}$ und $8n^3 \leq \ell \leq 2^n$ sowie $\chi$ eine ungerade Markierung des Graphen ist, gilt der Zeit-Platz*

*Tradeoff*

$$\boxed{\mathrm{Le}(\pi_{n,\ell}) \geq \left(\frac{2^{\Omega(n)}}{\mathrm{Sp}(\pi_{n,\ell})}\right)^{\Omega\left(\frac{\log_2 \log_2 n}{\log_2 \log_2 \log_2 n}\right)}.}$$

*Beweis.* In Abschnitt 7.3.1 hatten wir in Proposition 7.3.9 eine erweiterte isoperimetrische Ungleichung mit Parametern ($W = n, t_0 = 2n^3, r = \frac{7}{2} + \varepsilon$) für Gittergraphen $G_{n\times\ell}$ mit $8n^3 \leq \ell \leq 2^n$ hergeleitet. Vermöge Proposition 7.3.25 (setze hierzu $\varepsilon = \frac{1}{2}$) gelingt es uns, ein stark beschränktes Komplexitätsmaß $\mu^*_{F_{n,\ell}}$ für $F_{n,\ell}$ zu definieren, sodass (7.28) unter den in Proposition 7.3.25 genannten Voraussetzungen gilt. Hiermit sind alle Voraussetzungen von Theorem 7.3.27 erfüllt, was uns den gewünschten Zeit-Platz Tradeoff liefert:

$$\mathrm{Le}(\pi_{n,\ell}) \geq \left(\frac{2^{\Omega(W)}}{\mathrm{Sp}(\pi_{n,\ell})}\right)^{\Omega\left(\frac{\log_2 L}{\log_2 \log_2 L}\right)} \geq \left(\frac{2^{\Omega(W)}}{\mathrm{Sp}(\pi_{n,\ell})}\right)^{\Omega\left(\frac{\log_2 \log_2 n}{\log_2 \log_2 \log_2 n}\right)}.$$

Hierbei haben wir in der letzten Abschätzung Gleichung (7.27) sowie Proposition 7.3.24 (ii) ausgenutzt, um

$$\begin{aligned} L = \left\lfloor \log_2\left(\frac{|V(G)|}{4t_0}\right)\right\rfloor &= \left\lfloor \log_2\left(\frac{n\ell}{4\cdot 2n^3}\right)\right\rfloor \\ &\geq \left\lfloor \log_2\left(\frac{n\cdot 8n^3}{4\cdot 2n^3}\right)\right\rfloor = \lfloor \log_2 n\rfloor \geq \log_2 n - 1 = \Omega(\log_2 n) \end{aligned} \tag{7.36}$$

zu erhalten. □

Bevor wir die Arbeit abschließen, führen wir retrospektiv Begründungen für die Wahl der Konstanten und Parameter dieses Kapitels auf:

**Kommentar zur Parameterwahl 7.3.29.** Unsere Parameterwahl während der Beweise war geprägt von einem Abwägen an Genauigkeit und Schärfe der Resultate bzw. Abschätzungen auf der einen Seite und einer möglichst eleganten Beweisführung auf der anderen Seite (wodurch einige Konstantenwahlen, der einfacheren Darstellung geschuldet, geringfügig schlechter gewählt wurden, als dies theoretisch möglich gewesen wäre). Hierdurch konnten wir vorhandene Resultate der Literatur verschärfen, verbessern bzw. korrigieren. Insbesondere wurde dabei Wert auf folgende kritische Stellen im Beweis gelegt, die ein natürliches Optimierungsproblem an die Konstantenwahl induzieren:

(a) Der Parameter $t_0$ der erweiterten isoperimetrischen Ungleichung muss wegen der Definition 7.3.3 (i) von Knotenmengen mittlerer Größe bzgl. $t_0$ so gewählt sein, dass $t_0 \leq \frac{|V|}{4}$ gilt. Durch diese Forderung konnte ein fundamentaler Fehler in den Arbei-

ten [BNT2013] sowie [Bec2017] behoben werden.

(b) Gleichzeitig wurde die Obergrenze $\frac{|V|}{4}$ der Kardinalität von Knotenmengen mittlerer Größe derart gewählt, dass Abschätzung (7.22) möglich ist. Dies korrigiert einen Fehler der Dissertation [Bec2017], in der es mit der dortigen Wahl der Obergrenze $\frac{|V|}{2}$ (im Kontrast zur dort behaupteten Aussage) nicht möglich ist, Abschätzung (7.22) zu treffen. Andere Wahlen wie z. B. $\frac{|V|}{2+\varepsilon}$ mit $\varepsilon > 0$ für die Obergrenze wären ebenfalls denkbar gewesen.

(c) Um eine elegante Abschätzung in (7.19 (\$)) zu gewährleisten, haben wir $t_0 = 2n^3$ gewählt. Eine Wahl $t_0 > n^3$ wäre an dieser Stelle ebenfalls zulässig gewesen; die Wahl $t_0 = n^3$ würde allerdings die Bedingung $r > 5$ in (7.20) liefern, was die Analyse in Proposition 7.3.25 deutlich verkompliziert. Jedoch benötigen wir, um die rechte Seite der Abschätzung (7.17) strikt positiv zu halten, sodass wir in der Abschätzungskette (7.19) nicht durch Null teilen müssen, dass $|S_i| > n^3$ ist, was die Wahl $t_0 = n^3$ endgültig ausschließt.

(d) Aus den obigen Punkten (a) und (c) ergibt sich die Forderung $|V| \geq 8n^3$ (was die zusätzliche Forderung $\ell > 4n^2$ für Abschätzung (7.22) redundant macht).

(e) Für die Abschätzung (7.36) erweist es sich schließlich in der Tat als nützlich, $\ell = 8n^3$ zu wählen. Dies verbessert die Resultate in [BBI2016, BNT2013] um einen polynomiellen Faktor. Zudem konnten wir die unpräzise Formulierung „für alle $n \in \mathbb{N}$ groß genug“ aus [BBI2016] eliminieren und das Ergebnis für alle $n \in \mathbb{N}_{\geq 5}$ zeigen, was das Resultat zusätzlich verbessert. Die Bedingung $n \in \mathbb{N}_{\geq 5}$ kann zudem nicht weiter verbessert werden, wie wir auf Seite 142 erörtert haben. Die erweiterte isoperimetrische Ungleichung wurde sogar für alle $n \in \mathbb{N}$ gezeigt, was die Resultate aus [BNT2013, Bec2017] verbessert.

Dies schließt den Beweis von Haupttheorem 7.0.1 ab. ▣

## 7.4 Offene Fragen

Wie in [BBI2016] ergeben sich aus dem Resultat in Haupttheorem 7.0.1 mehrere – bis heute offene – Fragen:

- Ist es möglich, den Exponenten $\Omega(\log_2 \log_2 n / \log_2 \log_2 \log_2 n)$ zu $\Omega(\log_2 n)$ zu verbessern?
- Etwas allgemeiner: Gibt es für jedes $k \in \mathbb{N}$ und jede Formel der Größe $n$, die einen Resolutionsbeweis der Länge $n^k$ besitzt, auch einen Resolutionsbeweis in Platz $\mathcal{O}(n)$

mit Länge $n^{\mathcal{O}(\log n)}$? Oder mit Länge $2^{n^\varepsilon}$ für ein $\varepsilon > 0$?

- Ebenso ist es möglich, das Tradeoff-Resultat als Separation zwischen *dynamischer Programmierung* und *Divide and Conquer* zu interpretieren (wir haben dies in Intuition 7.1.12 & 7.2.2 angedeutet). Ist es möglich diese Separation auch in anderen Resultaten wiederzufinden? Sind ähnliche Resultate mit anderen Paradigmen möglich?

Wir schließen mit der Bemerkung ab, dass die (damals) offene Frage aus [BBI2016], ob das Epochenargument, das wir in unserem Beweis benutzt haben, ebenfalls dazu verwendet werden kann, Tradeoffs in einem stärkeren Beweissystem als Resolution zu zeigen, positiv beantwortet werden kann. Dies wurde in [BNT2013, Theorem 4], [Bec2017] für das Beweissystem `PCR` durch geringfügige Modifikation der in diesem Kapitel getätigten Argumentation und ergänzender Technik erreicht.

# Appendix und Verzeichnisse

# ANHANG A

## Logik

Der erste Teil des Anhangs stellt einen einfachen Satz aus der Logik vor, den wir aufgrund der Häufigkeit der Benutzung in dieser Arbeit aufführen.

**Proposition A.0.1.** *Es sei $F$ eine aussagenlogische Formel. Dann ist $F$ eine Tautologie genau dann, wenn $\neg F$ unerfüllbar ist. Genauer:*

$$F \in \text{TAUT} \iff \neg F \notin \text{SAT} \iff \neg F \in \text{UNSAT}.$$

*Entsprechend gilt*

$$\neg F \in \text{TAUT} \iff F \notin \text{SAT} \iff F \in \text{UNSAT}.$$

Wir folgen für den Beweis grob der Darstellung in [Sch2013, S. 6].

*Beweis.* Die Formel $F$ ist genau dann eine Tautologie, wenn $F\alpha = \mathtt{1}$ für alle zu $F$ passenden Belegungen $\alpha$ gilt, was wiederum genau dann gilt, wenn $(\neg F)\alpha = \mathtt{0}$ für alle zu $\neg F$ passenden Belegungen $\alpha$ (was die gleichen, wie die zu $F$ passenden Belegungen sind) gilt, was genau dann der Fall ist, wenn $\neg F$ unerfüllbar ist. □

# ANHANG B

## Komplexitätstheoretische Resultate

Dieser Anhang sei der Darstellung einiger komplexitätstheoretischer Begriffe und Resultate gewidmet.

### B.1 Charakterisierungen der Komplexitätsklasse NP

Wir beginnen mit der „Standard“-Definition der Komplexitätsklasse NP, so wie sie in den meisten Standardbüchern zur Komplexitätstheorie eingeführt wird (wir folgen [AB2009, Abschnitt 2.1]). Insbesondere stimmt dies mit der ursprünglichen Definition der Klasse über nicht-deterministische Turingmaschinen überein – was den Namen NP erklärt, der für *non-deterministic polynomial-time* steht.

**Definition B.1.1** (Sprachakzeptanz-Definition der Komplexitätsklasse NP)**.** Eine Sprache $L \subset \Sigma^*$ ist in NP, falls es eine nicht-deterministische Turingmaschine $M$ und ein Polynom $p$ gibt, sodass für alle Wörter $x \in \Sigma^*$ gilt:

(i) Bei Eingabe des Wortes $x$ hält $M$ nach höchstens $p(|x|)$ Schritten, d. h. jeder *Berechnungspfad* von $M$ auf $x$ hat maximale Länge $p(|x|)$.

(ii) Es gilt $x \in L$ genau dann, wenn es mindestens einen akzeptierenden Berechnungspfad von $M$ auf $x$ gibt.

Eine weitere Definition der Komplexitätsklasse NP, die insbesondere im Zusammenhang mit SAT-Solvern und Beweissystemen die „anschaulichere“ ist, führen wir im Folgenden an.

**Definition B.1.2** (Suchproblem-Definition der Komplexitätsklasse NP)**.** Eine Sprache $L \subset \Sigma^*$ ist in NP, falls es eine deterministische Polynomial-Zeit Turingmaschine $M$ und ein Polynom $p$ gibt, sodass für alle $x \in \Sigma^*$ die Äquivalenz

$$x \in L \iff \text{es gibt ein } y \in \Sigma^* \text{ mit } |y| \leq p(|x|) \text{ und } M(x, y) = 1$$

gilt.

Dabei heißt $y$ *Zertifikat* von $x$. Ist $x \in L$, so nennen wir $y$ einen *Beweis* oder *Zeugen* für $x$ (bzgl. der Sprache $L$ und der Turingmaschine $M$).

Man beachte den engen Zusammenhang zu polynomiell-beschränkten Beweissystemen, den wir in Satz 3.3.8 dargestellt haben. Diesen Satz formulieren wir im Sinne der obigen Definitionen hier erneut als Notiz.

**Notiz B.1.3** (Äquivalenz der Definitionen der Komplexitätsklasse NP)**.** Die zwei Definitionen von NP als die Klasse der durch eine nicht-deterministische Turingmaschine in Polynomial-Zeit lösbaren Probleme (B.1.1) und als die Klasse der durch eine deterministische Turingmaschine in Polynomial-Zeit verifizierbaren Probleme (B.1.2) sind äquivalent. Um es mit RECKHOWs Worten aus [Rec1975, S. 46] auszudrücken:

> This ... confirms the intuitive feeling that 'guess and verify' ... accurately characterizes the computing power of nondeterministic polynomial-time acceptors.

*Beweis.* Beweise dieser bekannten Äquivalenz finden sich z. B. in [Pap1994, Proposition 9.1], [Sip2006, Theorem 7.20] oder [AB2009, Abschnitt 2.1]. Wir verweisen zudem auf den Beweis des Satzes 3.3.8, der lediglich geringfügig umgeschrieben werden muss, um einen Beweis für Notiz B.1.3 zu erhalten. □

## B.2 Reduktionen und Vollständigkeit

Die Sprache SAT ist mindestens so schwer, wie jede andere Sprache in NP: Besitzt SAT einen Polynomialzeit-Algorithmus, so besitzen alle Probleme in NP einen solchen Algorithmus. Diese Eigenschaft wird NP-Schwere genannt. Wir wollen sie im Folgenden definieren.

Dazu seien im Folgenden $\Sigma$ und $\Gamma$ zwei Alphabete.

**Definition B.2.1.** (i) Eine *Stringfunktion* ist eine Funktion von $\Sigma^*$ nach $\Gamma^*$.

(ii) Eine Stringfunktion $f\colon \Sigma^* \to \Gamma^*$ ist *berechenbar in Zeit* $T(n)$, falls es eine deterministische Mehrband-Turingmaschine $M$ gibt, die bei Eingabe jedes Strings $x \in \Sigma^*$ nach höchstens $T(|x|)$ Schritten hält und $f(x)$ auf dem Arbeitsband stehen hat.

(iii) Eine Stringfunktion $f\colon \Sigma^* \to \Gamma^*$ ist *berechenbar in polynomieller Zeit*, falls es ein Polynom $p\colon \mathbb{N}_0 \to \mathbb{R}_0^+$ gibt, sodass $f$ berechenbar in Zeit $p(n)$ ist.

**Definition B.2.2.** Es seien $L_1 \subset \Sigma^*$ und $L_2 \subset \Gamma^*$ zwei Sprachen. Wir nennen $L_1$ *polynomiell many-one-reduzierbar* auf $L_2$, in Zeichen $L_1 \preccurlyeq_m^P L_2$, falls es eine in polynomieller Zeit berechenbare Stringfunktion $f\colon \Sigma^* \to \Gamma^*$ gibt, eine *Reduktion*, sodass für alle $x \in \Sigma^*$ die

Äquivalenz

$$x \in L_1 \iff f(x) \in L_2$$

gilt. Siehe Abbildung B.1.

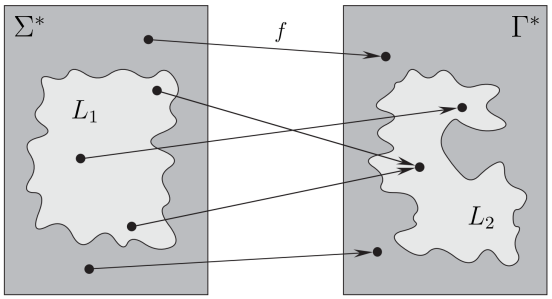



Abbildung B.1.: Eine Illustration einer Reduktion $f\colon \Sigma^* \to \Gamma^*$ mit der Eigenschaft $x \in L_1 \iff f(x) \in L_2$ für alle $x \in \Sigma^*$. Adaptiert aus [CLRS2009].

Offensichtlich ist $\preccurlyeq_m^P$ *transitiv*, d. h. gilt $A \preccurlyeq_m^P B$ und $B \preccurlyeq_m^P C$, dann auch $A \preccurlyeq_m^P C$.

**Lemma B.2.3.** *Es seien $L_1 \subset \Sigma^*$ und $L_2 \subset \Gamma^*$ zwei Sprachen mit $L_1 \preccurlyeq_m^P L_2$. Gilt $L_2 \in \mathsf{P}$ (bzw. $L_2 \in \mathsf{NP}$), so ist auch $L_1 \in \mathsf{P}$ (bzw. $L_1 \in \mathsf{NP}$).*

*Beweis.* Siehe z. B. [CLRS2009, Lemma 34.3]. □

Die folgende Definition ist für unsere Bedürfnisse angepasst und ist bewusst nicht so allgemein wie möglich gehalten, um keine zusätzlichen Reduktionsbegriffe einzuführen.

**Definition B.2.4.** Es sei $\mathsf{C} \in \{\mathsf{NP}, \mathsf{co\text{-}NP}\}$ eine Komplexitätsklasse.

(i) Eine Sprache $A \subset \Sigma^*$ heißt $\mathsf{C}$-*schwer* (engl. $\mathsf{C}$-*hard*) (bzgl. $\preccurlyeq_m^P$), falls für alle $L \in \mathsf{C}$ die Reduktion $L \preccurlyeq_m^P A$ gilt.

(ii) Eine Sprache $A \subset \Sigma^*$ heißt $\mathsf{C}$-*vollständig* (engl. $\mathsf{C}$-*complete*), falls $A$ Mitglied in $\mathsf{C}$ ist und $\mathsf{C}$-schwer ist.

Trivialerweise ist $\mathsf{P} \subset \mathsf{NP}$. Ob jedoch $\mathsf{P} = \mathsf{NP}$ oder $\mathsf{P} \subsetneq \mathsf{NP}$ gilt, ist eines der ungelösten Millennium-Probleme. Diesbezüglich ist das folgende Theorem nützlich: Gibt es einen Polynomial-Zeit Algorithmus für *ein* $\mathsf{NP}$-schweres Problem, so gibt es Polynomial-Zeit Algorithmen für *alle* Probleme in $\mathsf{NP}$, es würde also $\mathsf{P} = \mathsf{NP}$ gelten.

**Theorem B.2.5.**

*(i) Ist eine Sprache $L \in \mathsf{P}$ und $\mathsf{NP}$-vollständig, dann ist $\mathsf{P} = \mathsf{NP}$.*

*(ii) Für jede $\mathsf{NP}$-vollständige Sprache $L$ gilt $L \in \mathsf{P}$ genau dann, wenn $\mathsf{P} = \mathsf{NP}$ ist.*

*Beweis.* Es genügt (ii) zu zeigen, da dies eine stärkere Aussage als (i) ist. Wir orientieren uns an [ST2013, Observation auf S. 156].

Gilt $\mathsf{P} = \mathsf{NP}$, dann ist wegen $L \in \mathsf{NP}$ offensichtlich auch $L \in \mathsf{P}$.

Ist umgekehrt $L \in \mathsf{P}$ und $L'$ eine beliebige Sprache in $\mathsf{NP}$, dann folgt aus der $\mathsf{NP}$-Vollständigkeit von $L$ direkt $L' \preccurlyeq_m^P L$. Hieraus lässt sich $L' \in \mathsf{P}$ mit Lemma B.2.3 folgern. Also ist $\mathsf{P} = \mathsf{NP}$. □

**Theorem B.2.6.** *(i) Ist $A$ eine $\mathsf{NP}$-vollständige Sprache, dann ist $A \in \mathsf{co\text{-}NP}$ genau dann, wenn $\mathsf{NP} = \mathsf{co\text{-}NP}$ gilt.*

*(ii) Ist $A$ eine $\mathsf{co\text{-}NP}$-vollständige Sprache, dann ist $A \in \mathsf{NP}$ genau dann, wenn $\mathsf{NP} = \mathsf{co\text{-}NP}$ gilt.*

*Beweis.* Wir zeigen lediglich (i); die zweite Aussage (ii) zeigt man analog.

„⟶“: Es sei $A$ eine $\mathsf{NP}$-vollständige Sprache, die zudem in $\mathsf{co\text{-}NP}$ liegt. Wir zeigen die Gleichheit $\mathsf{NP} = \mathsf{co\text{-}NP}$, indem wir beide Inklusionen nachweisen.

„⊂“: Es sei $L \in \mathsf{NP}$. Da $A$ eine $\mathsf{NP}$-schwere Sprache ist, gilt nach Definition $L \preccurlyeq_m^P A$. Folglich gilt dann auch $\overline{L} \preccurlyeq_m^P \overline{A}$. Wegen $A \in \mathsf{co\text{-}NP}$ ist $\overline{A} \in \mathsf{NP}$, also ist $\overline{L} \in \mathsf{NP}$ nach Lemma B.2.3, d. h. $L \in \mathsf{co\text{-}NP}$.

„⊃“: Ist $L \in \mathsf{co\text{-}NP}$, dann ist $\overline{L} \in \mathsf{NP}$. Gemäß der Inklusion $\mathsf{NP} \subset \mathsf{co\text{-}NP}$, die wir im vorherigen Schritt etabliert haben, gilt also auch $\overline{L} \in \mathsf{co\text{-}NP}$, d. h. $L \in \mathsf{NP}$.

„⟵“: Diese Richtung der Äquivalenz ist trivial. □

**Theorem B.2.7.** *Ist $L$ eine $\mathsf{NP}$-vollständige Sprache, so ist $\overline{L}$ eine $\mathsf{co\text{-}NP}$-vollständige Sprache.*

*Beweis.* Es sei $L$ eine $\mathsf{NP}$-vollständige Sprache. Gemäß Definition ist dann $\overline{L} \in \mathsf{co\text{-}NP}$. Um zu zeigen, dass $\overline{L}$ eine $\mathsf{co\text{-}NP}$-schwere Sprache ist, wählen wir eine beliebige Sprache $A \in \mathsf{co\text{-}NP}$ und wollen zeigen, dass $A \preccurlyeq_m^P \overline{L}$ gilt. Dazu bemerken wir zunächst, dass $\overline{A} \in \mathsf{NP}$ ist und dass es wegen der $\mathsf{NP}$-Vollständigkeit von $L$ eine in Polynomialzeit berechenbare Stringfunktion $f$ gibt, sodass

$$x \in \overline{A} \iff f(x) \in L \quad \text{für alle } x$$

gilt. Dies ist aber äquivalent zu

$$x \notin \overline{A} \iff f(x) \notin L \quad \text{für alle } x,$$

was wiederum äquivalent ist zu

$$x \in A \iff f(x) \in \overline{L} \quad \text{für alle } x.$$

Also ist $f$ auch eine Reduktion von $A$ nach $\overline{L}$, d. h. es gilt $A \preccurlyeq_m^P \overline{L}$. Also ist $\overline{L}$ eine co-NP-vollständige Sprache. □

**Korollar B.2.8.** *Die Sprache* TAUT *ist* co-NP*-vollständig.*

*Beweis.* (I) *Mitgliedschaft in* co-NP*:* Wir erinnern daran, dass eine Sprache $L \subset \Sigma^*$ in co-NP liegt, falls $\overline{L} = \left\{x \in \Sigma^* \,:\, x \notin L\right\} \in \mathsf{NP}$ ist. Da aber $\overline{\text{TAUT}}$ gerade die Menge der Booleschen Formeln ist, die keine Tautologie sind, ist klar, dass $\overline{\text{TAUT}} \in \mathsf{NP}$ gilt: Eine nicht-deterministische Turingmaschine muss lediglich eine Belegung finden, die eine gegebene Formel *falsifiziert*, was durch den trivialen „rate & verifiziere"-Algorithmus in Polynomialzeit möglich ist.

(II) *Vollständigkeit für* co-NP*:* Um schließlich zu zeigen, dass TAUT vollständig für co-NP ist, genügt es wegen Theorem B.2.7 zu zeigen, dass $\overline{\text{TAUT}}$ eine NP-vollständige Sprache ist.

Es ist bekannt, dass SAT eine NP-vollständige Sprache ist und es gilt $\text{SAT} \preccurlyeq_m^P \overline{\text{TAUT}}$, denn für jede Formel $F$ gilt

$$F \in \text{SAT} \iff \neg F \in \overline{\text{TAUT}}$$

analog zu Proposition A.0.1. Wie eben in (I) gezeigt, ist $\overline{\text{TAUT}}$ zudem in NP, also folgt, dass $\overline{\text{TAUT}}$ eine NP-vollständige Sprache ist, wie gewünscht. □

Ähnlich zeigt man die co-NP-Vollständigkeit von UNSAT.

## B.3 Mögliche Beziehungen zwischen P, NP und co-NP

Viele fundamentale Fragen neben dem $\mathsf{P} \stackrel{?}{=} \mathsf{NP}$-Problem zu Beziehungen verschiedener Komplexitätsklassen sind – trotz großer Forschungsanstrengungen – ungelöst. So ist z. B. unklar, ob $\mathsf{NP} \stackrel{?}{=} \text{co-NP}$ ist. Da P jedoch komplementabgeschlossen ist, wissen wir, dass $\mathsf{P} \subset \mathsf{NP} \cap \text{co-NP}$ ist (weshalb der Schnitt $\mathsf{NP} \cap \text{co-NP}$ nicht-leer ist). Erneut ist unklar, ob Gleichheit gilt, oder ob es eine Sprache in $\mathsf{NP} \cap \text{co-NP} \setminus \mathsf{P}$ gibt. In Abbildung B.2 haben wir die vier Beziehungsmöglichkeiten dargestellt. Dies zeigt die bedauerlicherweise unvollständige Kenntnis der genauen Beziehung zwischen P und NP.

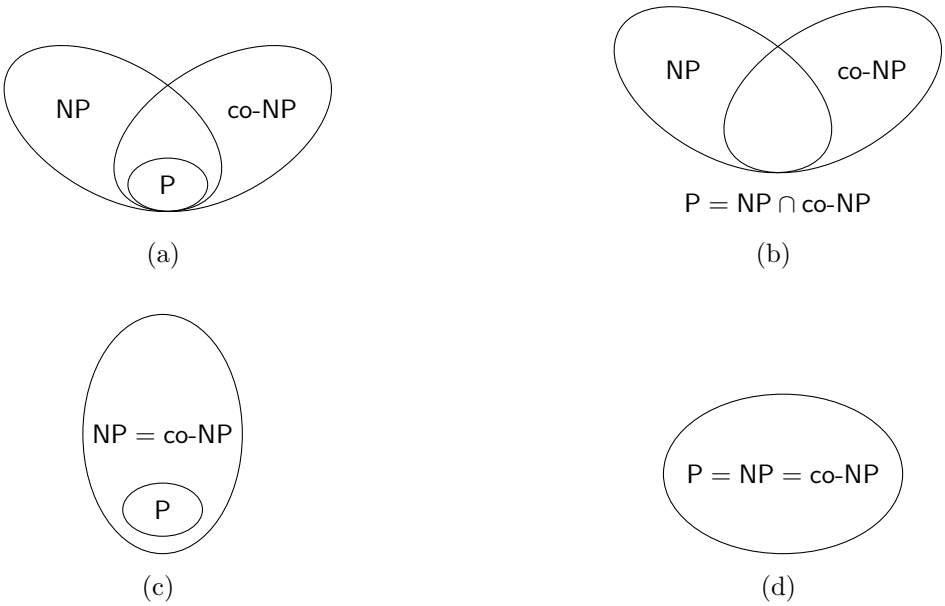



Abbildung B.2.: Die vier möglichen Beziehungen zwischen den Komplexitätsklassen P, NP und co-NP jeweils als Euler-Diagramm dargestellt; adaptiert aus [CLRS2009]. (a) $\mathsf{NP} \neq \mathsf{co\text{-}NP}$ und $\mathsf{P} \neq \mathsf{NP} \cap \mathsf{co\text{-}NP}$. Die meisten Forscher sehen diesen Fall als am wahrscheinlichsten an. (b) Der Fall $\mathsf{P} = \mathsf{NP} \cap \mathsf{co\text{-}NP}$ mit nicht-komplementabgeschlossenem NP. (c) Ist NP komplementabgeschlossen gilt $\mathsf{NP} = \mathsf{co\text{-}NP}$. Man beachte, dass dies nicht $\mathsf{P} = \mathsf{NP}$ notwendig macht. (d) $\mathsf{P} = \mathsf{NP} = \mathsf{co\text{-}NP}$ gilt als unwahrscheinlichster Fall. Er führt u. a. zu einem vollständigen *Kollaps der Polynomialzeithierarchie* PH auf P (siehe z. B. [AB2009, Theorem 5.6] hierzu).

Man beachte, dass in den Fällen (a) und (b) wegen $\mathsf{NP} \neq \mathsf{co\text{-}NP}$ notwendigerweise $P \neq \mathsf{NP}$ gilt.

# ANHANG C

## Tseitin-Transformationen

Zweck dieses Anhangs ist es, die im Haupttext oft angesprochene Tseitin-Transformation zu erklären.

Wir wiederholen zu Beginn dieses Kapitels eine Definition aus den Präliminarien: Zwei Formeln $F$ und $G$ heißen *(semantisch) äquivalent*, wenn für alle Belegungen $\alpha$ mit der Eigenschaft $\mathrm{Var}(\alpha) = \mathrm{Var}(F) \cup \mathrm{Var}(G)$ gilt, dass $F\alpha = G\alpha$ ist.

Andererseits heißen zwei Formeln $F$ und $G$ *sat-äquivalent* (auch: *erfüllbarkeitsäquivalent*), falls $F$ erfüllbar ist, genau dann, wenn $G$ erfüllbar ist.

**Beispiel C.0.1.** Ist $F \notin \mathrm{TAUT} \cup \mathrm{UNSAT}$ eine Formel, die weder tautologisch noch unerfüllbar ist, so sind $F$ und $\neg F$ nicht äquivalent, aber sat-äquivalent.

**Theorem C.0.2.** *Für jede Formel $F$ gibt es eine äquivalente* CNF*-Formel $G$. Im Allgemeinen benötigt die Transformation von $F$ zu $G$ exponentiell viel Berechnungszeit (in der Länge $|F|$ der Eingabeformel). Insbesondere gibt es* keinen *Polynomialzeit-Algorithmus, der beliebige aussagenlogische Formeln in äquivalente Formeln in* CNF *umformt.*

*Beweis.* Siehe z. B. [ST2013, 1.2]. Für die Konstruktion einer expliziten Folge von Formeln $F_n$ mit $2n$ Literalen, für die jede äquivalente Formel in CNF mindestens $2^n$ Klauseln besitzt, führen wir das Beispiel aus [KL1994, Lemma 2.3.4] an: Für $n \in \mathbb{N}$ sei

$$F_n := \bigvee_{1 \leq i \leq n} (x_{i,1} \wedge x_{i,2})$$

mit paarweise verschiedenen Variablen $x_{1,1}, x_{1,2}, x_{2,1}, x_{2,2}, \ldots, x_{n,1}, x_{n,2} \in \mathcal{V}$. Offensichtlich hat die Formel $F_n$ Größe $\mathrm{size}(F_n) = 2n$ bzw. Länge $|F_n| = \mathcal{O}(n)$. Eine minimale äquivalente Formel in CNF ist jedoch

$$G_n = \bigwedge_{j_1, \ldots, j_n \in \{1,2\}} (x_{1,j_1} \vee \cdots \vee x_{n,j_n})$$

mit $2^n$ Klauseln bzw. Länge $|G_n| = n2^n = \Omega(2^n)$. □

Dennoch ist es möglich eine Formel in polynomieller Zeit in eine *sat-äquivalente* Formel in CNF (sogar 3-CNF) zu transferieren, welche eine Größe linear zur ursprünglichen Größe der Formel hat.

**Definition & Theorem C.0.3** (Tseitin-Transformation)**.** *Für jede Formel $F$ gibt es eine sat-äquivalente Formel $G$ in* 3-CNF*, sodass $|G| = \mathcal{O}(|F|)$ gilt (d. h. $|G|$ ist höchstens linear größer als $|F|$). Die Konstante der $\mathcal{O}$-Notation ist unabhängig von der Formel $F$. Die Transformation von $F$ nach $G$ ist in Polynomialzeit berechenbar. Wir schreiben $G = \mathrm{Tseitin}(F)$.*

*Beweis.* Für einen Beweis dieses bekannten Resultats siehe z. B. [ST2013, 1.3] oder TSEITINs Originalarbeit [Tse1968]. Ebenfalls empfehlenswert für einen „Beweis durch Beispiel" ist NORDSTRÖMs Dissertation [Nor2008, S. 36f.] (vgl. auch unser unten stehendes Beispiel C.0.5), welche die Idee, eine neue Variable für jede Subformel (siehe Definition C.0.4) zu definieren, sehr deutlich macht. Für die Längenabschätzung der entstehenden Formel siehe z. B. [KL1994, Algorithmus 2.4 & 2.5, Satz 2.3.8]. Hieraus ergibt sich auch die Unabhängigkeit der Konstante in der $\mathcal{O}$-Notation von $F$: Für jede $\wedge$- bzw. $\vee$-Operation in der Formel benötigt man höchstens 3 neue Klauseln in der Codierung in 3-CNF. □

**Definition C.0.4.** Ist $F$ eine aussagenlogische Formel, so ist $\mathrm{Sub}(F)$, die *Menge der Subformeln* (bzw. *Teilformeln*) *von* $F$, rekursiv wie folgt definiert:

1. $\mathrm{Sub}(\phi) = \{\phi\}$ für alle $\phi \in \mathcal{V} \cup \{0, 1\}$,
2. $\mathrm{Sub}(\neg F) = \{\neg F\} \cup \mathrm{Sub}(F)$,
3. $\mathrm{Sub}(F \circ G) = \{F \circ G\} \cup \mathrm{Sub}(F) \cup \mathrm{Sub}(G)$ für $\circ \in \{\wedge, \vee, \to\}$.

Wie NORDSTRÖM ist der Autor dieser Arbeit der Ansicht, dass man sich die Idee der Transformation am Besten durch ein Beispiel veranschaulicht. Es ist nicht schwer aus diesem einen formalen Beweis zu notieren.

**Beispiel C.0.5** („Beweis" der Tseitin-Transformation C.0.3 durch ein Beispiel)**.** Da wir lediglich eine sat-äquivalente Formeln finden wollen, dürfen wir neue Variablen einführen: Dies tun wir für jede Subformel. Wir verbinden diese Subformeln mit den zu ihnen korrespondierenden Variablen durch ein Bikonditional und konjugieren anschließend alle Bedingungen (inklusive der Bedingung für die gesamte Formel) miteinander. Anschließend können wir jede Bedingung separat in CNF umformen. Ist beispielsweise

$$F := \Big((x \vee y) \wedge z\Big) \to (\neg w),$$

so sind die Subformeln von $F$ (der Länge nach geordnet):

$$
\begin{aligned}
&x, y, z, w,\\
&\neg w,\\
&(x \vee y),\\
&(x \vee y) \wedge z,\\
&\Big((x \vee y) \wedge z\Big) \to (\neg w).
\end{aligned}
$$

Wir führen die Hilfsvariablen $h_1, \ldots, h_4$ ein und verknüpfen die Subformeln $\neg w$, $(x \vee y)$, $(x \vee y) \wedge z$ sowie $\Big((x \vee y) \wedge z\Big) \to (\neg w)$ mit diesen Hilfsvariablen durch das Bikonditional (wobei wir entsprechende Teilformeln bereits durch die jeweiligen Hilfsvariablen substituieren):

$$
\begin{aligned}
&h_1 \leftrightarrow \neg w,\\
&h_2 \leftrightarrow x \vee y,\\
&h_3 \leftrightarrow h_2 \wedge z,\\
&h_4 \leftrightarrow h_3 \to h_1.
\end{aligned}
$$

Durch Konjunktion dieser Bikonditionale zusammen mit der Bedingung $h_4$ (die für die ursprüngliche Formel $F$ steht) als eigene Klausel ergibt sich:

$$\text{Tseitin}_{\text{vorläufig}}(F) := h_4 \wedge (h_4 \leftrightarrow h_3 \to h_1) \wedge (h_3 \leftrightarrow h_2 \wedge z) \wedge (h_2 \leftrightarrow x \vee y) \wedge (h_1 \leftrightarrow \neg w).$$

Die Formel $\text{Tseitin}(F)$ ergibt sich hieraus durch Umformung der einzelnen Konjunkte. Beispielhaft betrachten wir

$$
\begin{aligned}
h_2 \leftrightarrow (x \vee y) &\equiv h_2 \to (x \vee y) \wedge \Big((x \vee y) \to h_2\Big)\\
&\equiv (\neg h_2 \vee x \vee y) \wedge \Big(\neg(x \vee y) \vee h_2\Big)\\
&\equiv (\neg h_2 \vee x \vee y) \wedge \Big((\neg x \wedge \neg y) \vee h_2\Big)\\
&\equiv (\neg h_2 \vee x \vee y) \wedge (\neg x \vee h_2) \wedge (\neg y \vee h_2).
\end{aligned}
$$

Man beachte, dass es im letzten Schritt durch Anwendung des Distributivgesetzes (im Gegensatz zu Theorem C.0.2) nur noch zu einem konstanten Anstieg der Konjunktionen kommen kann.

Trivialerweise gilt $\text{Tseitin}(F) \vDash F$. Man zeigt leicht, dass auch die Umkehrung gilt, die beiden Formeln also sat-äquivalent sind.

Da SAT in NP liegt, ist auch $k$-CNF-SAT $\in$ NP für $k \geq 3$ (man beachte, dass 2-SAT $\in$ P liegt, da Resolventen von 2-Klauseln ebenfalls höchstens 2 Literale enthalten, es also höchstens $\mathcal{O}(n^2)$ Resolventen für eine Formeln in $n$ Variablen geben kann, wovon alle potenziell Möglichen in polynomieller Zeit generiert werden können und auf die leere Klausel überprüft werden können. Vergleiche [ST2013, § 3.1] sowie den `Davis-Putnam`-Algorithmus 1). Die Tseitin-Transformation aus Definition & Theorem C.0.3 kann als eine Reduktionsabbildung vom NP-vollständigen SAT-Problem zu 3-SAT aufgefasst werden. Wir erhalten daher sofort das folgende Korollar.

**Korollar C.0.6.** 3-SAT *und* $k$-CNF-SAT *für* $k \geq 3$ *sind* NP*-vollständig.*

# ANHANG D

## Relationen

Wir geben im Folgenden einige Definitionen relevanter Relationen für den Themenkomplex der $p$-Simulation. Für weitergehende Ausführungen und Beispiele (auf die wir fast gänzlich verzichten wollen) verweisen wir auf das Buch [SRI2013], an welchem wir uns für diesen Anhang orientiert haben.

### D.1 Relationsarten: Äquivalenzen, Quasi- & Partialordnungen

**Definition D.1.1.** Eine (*zweistellige* oder *binäre*) *Relation* $R$ zwischen zwei Mengen $M$ und $N$ ist eine Teilmenge des kartesischen Produktes $M \times N = \{(m, n) : m \in M, n \in N\}$, in Zeichen $R \subset M \times N$. Ist $M = N$, so sprechen wir von einer (*homogenen*) *Relation auf* $M$.

Im Rest des Anhangs D sei $M$ immer eine Menge.

**Bemerkung D.1.2.** Ist $R$ eine binäre Relation, so bevorzugt man häufig die *Infix-Notation* $aRb$ statt der Schreibweise $(a, b) \in R$.

Äquivalenzrelationen spielen für die folgenden Diskussionen eine wichtige Rolle. Der Vollständigkeit halber führen wir die Definition hierzu auf.

**Definition D.1.3.** (i) Eine Relation $\sim$ auf $M$ heißt *Äquivalenzrelation*, falls sie *reflexiv*, *symmetrisch* und *transitiv* ist, d. h. für alle $a, b, c \in M$ gilt:
(a) Reflexivität: $a \sim a$,
(b) Symmetrie: $a \sim b$ impliziert $b \sim a$,
(c) Transitivität: $a \sim b$ zusammen mit $b \sim c$ impliziert $a \sim c$.
Gilt für zwei Elemente $a \sim b$, so sagen wir, dass $a$ zu $b$ *äquivalent* ist.
(ii) Jede Äquivalenzrelation $\sim$ auf $M$ legt *Äquivalenzklassen* auf $M$ fest: Für $a \in M$ definieren wir die zugehörige $\sim$-*Äquivalenzklasse* $[a]_\sim$ als die Menge aller Elemente

aus $M$, die zu $a$ äquivalent sind, in Zeichen

$$[a]_\sim := \{b \in M \,:\, b \sim a\}.$$

(iii) Die Menge aller $\sim$-Äquivalenzklassen

$$M/\!\sim := \{[a]_\sim \,:\, a \in M\}$$

bezeichnen wir als *Quotientenmenge* von $M$ bzgl. $\sim$.

**Bemerkung D.1.4** (Fortsetzung von Bemerkung D.1.2)**.** Häufig verwendet man statt $R$ bei einer nicht-symmetrischen Relation auch das Symbol $\leq$ (oder vergleichbare Symbole) und spricht wie bei reellen Zahlen von *kleineren* und *größeren* Elementen (vgl. Definition & Lemma D.3.1 im Folgenden).

**Definition D.1.5.** Eine binäre Relation $\lesssim$ auf $M$ heißt *Quasiordnung* (auch: *Präordnung*, engl. *preordner*), falls sie *reflexiv* und *transitiv* ist, d. h. für alle $a, b, c \in M$ gilt:
(a) Reflexivität: $a \lesssim a$,
(b) Transitivität: $a \lesssim b$ zusammen mit $b \lesssim c$ impliziert $a \lesssim c$.
Sind $a, b \in M$, so schreiben wir $a \not\lesssim b$, falls $a \lesssim b$ *nicht* gilt. Das Tupel $(M, \lesssim)$ nennt man *quasigeordnete Menge* (engl. *preordered set*, auch *proset*).

**Beispiel & Satz D.1.6.** Das Konzept der $p$-Simulation definiert eine Quasiordnung $\preccurlyeq_p$ auf der Menge $\mathfrak{M}$ der aussagenlogischen Beweissysteme.

*Beweis.* Um die Reflexivität von $\preccurlyeq_p$ nachzuweisen, wählt man $f = \mathrm{id}_{\Sigma^*}$ als *Simulationsfunktion* in Definition 3.3.12 (i).

Ist $V_3 \preccurlyeq_p V_2$ via $g$ und $V_2 \preccurlyeq_p V_1$ via f, so ist $V_3 \preccurlyeq_p V_1$ via $h := f \circ g$, was offensichtlich eine in Polynomial-Zeit berechenbare Funktion ist, da es $f$ und $g$ waren. □

Wir führen zudem den Begriff der partiellen Ordnung ein, der ursprünglich in [DM1941] eingeführt wurde. Eine partielle Ordnung ist eine Quasiordnung mit der zusätzlichen Eigenschaft der *Antisymmetrie*.

**Definition D.1.7.** Eine binäre Relation $\leq$ auf $M$ heißt *Partialordnung* (auch: *partielle Ordnung*, *Halbordnung* oder *Teilordnung*, engl. *partial order*), falls sie *reflexiv*, *antisymmetrisch* und *transitiv* ist, d. h. für alle $a, b, c \in M$ gilt:

(a) Reflexivität: $a \leq a$,
(b) Antisymmetrie: $a \leq b$ zusammen mit $b \leq a$ impliziert $a = b$,
(c) Transitivität: $a \leq b$ zusammen mit $b \leq c$ impliziert $a \leq c$.

Das Tupel $(M, R)$ nennt man *partiell geordnete Menge* (engl. *partially ordered set*, auch *poset*).

## D.2 Maximale Elemente und das größte Element

**Definition D.2.1.** Es sei $(M, \lesssim)$ eine quasigeordnete Menge, $N \subset M$ eine Teilmenge von $M$ sowie $x \in N$.

(i) Das Element $x$ heißt *maximales Element* von $N$, falls für alle $y \in N$, für die $x \lesssim y$ gilt, sofort $y \lesssim x$ gilt.
(ii) Das Element $x$ heißt *größtes Element* von $N$, falls $y \lesssim x$ für alle $y \in N$ gilt.

**Notiz D.2.2.** Die Begriffe maximales Element und größtes Element sind i. A. nicht äquivalent[47].

Es kann außerdem mehrere maximale Elemente in quasigeordneten Mengen geben. Ist $x$ das größte Element von $M$, dann ist $x$ auch das einzige maximale Element von $M$. Die Umkehrung gilt jedoch nicht: Selbst wenn $M$ nur ein maximales Element besitzt, ist dieses nicht automatisch größtes Element von $M$.

Man beachte zudem, dass maximale Elemente nicht zu existieren brauchen. Man betrachte etwa $M := \{q \in \mathbb{Q} : 1 \leq q^2 \leq 2\} \subset \mathbb{Q}$ und beachte $\sqrt{2} \notin \mathbb{Q}$.

**Beispiel D.2.3.** Wir betrachten die Mengen $M := \{0, 1\}^3$ sowie $M' := M \setminus \{(1, 1, 1)\}$ jeweils zusammen mit der „komponentenweise-kleiner-gleich“-Relation

$$(x_1, x_2, x_3) \leq^3 (y_1, y_2, y_3) \ :\Longleftrightarrow\ x_i \leq y_i \text{ für alle } i = 1, 2, 3,$$

welche eine Partialordnung auf $M$ und $M'$ definiert. Zur Veranschaulichung haben wir das Hasse-Diagramm in Abbildung D.1 angegeben. In der partiell geordneten Menge $(M, \leq^3)$ ist $(1, 1, 1)$ das größte Element und einzige maximale Element. In der partiell geordneten Menge $(M', \leq^3)$ sind die drei Elemente $(0, 1, 1), (1, 0, 1), (1, 1, 0)$ maximale Elemente. Es gibt jedoch kein größtes Element.

[47] Es sei denn, die Quasiordnung ist eine *Totalordnung* $\leq$. Dies ist eine Partialordnung mit der zusätzlichen Eigenschaft der *Totalität*, d. h. für alle $a, b \in M$ gilt $(a \leq b) \vee (b \leq a)$.

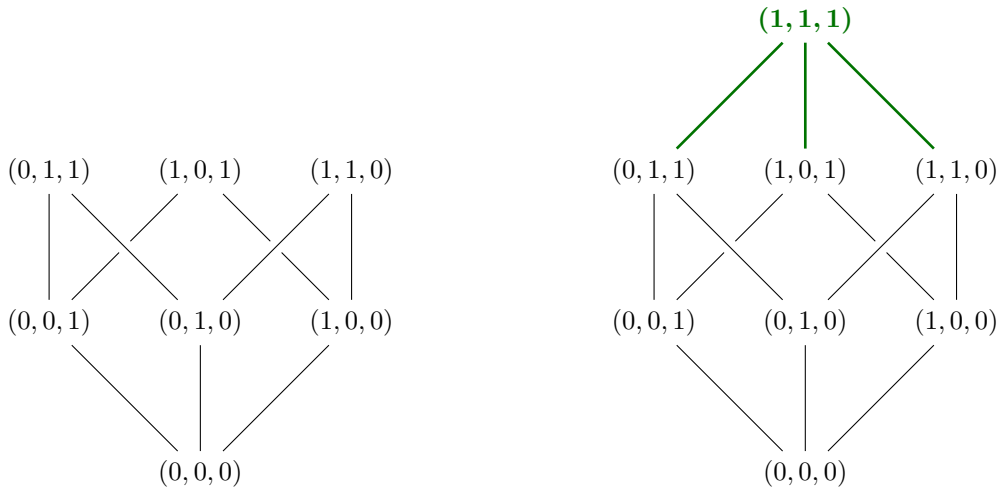



Abbildung D.1.: Das Hasse-Diagramm für $(M', \leq^3)$ ist in schwarzer Farbe auf der linken Seite abgedruckt. Das Hasse-Diagramm für $(M, \leq^3)$ ergibt sich rechts durch Hinzunahme des fetten und olivgrünen Teils. Ein *Hasse-Diagramm* ist eine graphische Darstellungsmöglichkeit endlicher partiell geordneter Mengen als gerichteter Graph. Die Elemente aus $M$ bilden dabei die Knoten des Graphen. Zwei Elemente $a, b \in M$ werden durch eine Kante verbunden, falls $a < b$ gilt und es kein Element $c \in M$ gibt mit $a < c < b$. Dabei haben wir

$$a < b \ :\Longleftrightarrow \ a \leq b \text{ und } a \neq b$$

für zwei Elemente $a, b \in M$ gesetzt. Die Richtung der Kante von $a$ nach $b$ deuten wir nicht durch Pfeile an, sondern dadurch, dass $b$ höher als $a$ im Diagramm steht. Insbesondere lässt man Schleifen zur Kennzeichnung der Reflexivität weg.

## D.3 Partialordnungen aus Quasiordnungen: Kerne

Die zusätzliche Eigenschaft der Antisymmetrie bei Partialordnungen ist in der Praxis oft zu restriktiv: So ist z. B. die Teilbarkeitsrelation $|$ eine Quasiordnung auf der Menge der ganzen Zahlen $\mathbb{Z}$, jedoch ist sie keine Partialordnung, da bspw. $1|-1$, und auch $-1|1$ gilt, jedoch 1 von $-1$ verschieden ist.

Einem ähnlichen „Dilemma“ sind wir bei Beweissystemen, die sich gegenseitig $p$-simulieren, begegnet. Diese sind lediglich $p$-äquivalent, nicht aber notwendigerweise gleich. Die folgenden Betrachtungen dienen dazu, dieses Problem zu lösen.

Wir beginnen dazu mit dem folgenden Lemma, das sich unmittelbar aus obigen Definitionen ergibt und das wir nutzen wollen, um weitere Sprechweisen einzuführen.

**Definition & Lemma D.3.1.** *Ist $\lesssim$ eine Quasiordnung auf $M$ und sind $a, b \in M$, so können höchstens vier Fälle auftreten:*

1. $(a \lesssim b) \wedge (b \not\lesssim a)$*, in Worten* $a$ *ist* echt kleiner *als* $b$*.*
2. $(a \not\lesssim b) \wedge (b \lesssim a)$*, in Worten* $a$ *ist* echt größer *als* $b$*.*

*3.* $(a \lesssim b) \wedge (b \lesssim a)$ *, in Worten* $a$ *ist* äquivalent *zu* $b$*.*
*4.* $(a \not\lesssim b) \wedge (b \not\lesssim a)$*, in Worten* $a$ *und* $b$ *sind* nicht vergleichbar*.*

Motiviert durch den Fall der Äquivalenz in Punkt 3 der obigen Auflistung definieren wir zu jeder Quasiordnung eine Äquivalenzrelation.

**Definition & Satz D.3.2.** *Ist* $\lesssim$ *eine Quasiordnung auf* $M$*, so definieren wir die kanonische Äquivalenzrelation* $\sim$ *auf* $M$ *durch*

$$a \sim b \ :\Longleftrightarrow\ (a \lesssim b) \wedge (b \lesssim a), \tag{D.1}$$

*die wir* Kern *der Quasiordnung* $\lesssim$ *nennen.*

*Beweis.* Die Relation $\sim$ ist symmetrisch gemäß Definition und reflexiv, da die Quasiordnung $\lesssim$ reflexiv ist. Die Transitivität von $\sim$ erhält man durch das zweimalige Anwenden der Transitivität von $\lesssim$. □

Anschaulich bedeutet dies: Zwei Elemente sind bezüglich $\sim$ äquivalent, wenn sie via $\lesssim$ gegenseitig in beide Richtungen vergleichbar sind.

**Beispiel D.3.3.** Das Tupel $(\mathbb{C}, \lesssim)$ mit

$$z \lesssim w \ :\Longleftrightarrow\ |z| \leq |w| \quad \text{für } z, w \in \mathbb{C} \tag{D.2}$$

ist eine quasigeordnete Menge. Dabei bezeichne $\mathbb{C}$, wie in den Präliminarien erörtert, die Menge der komplexen Zahlen, $|\cdot|$ den Absolutbetrag auf $\mathbb{C}$ und $\leq$ die Kleiner-Gleich-Relation auf $\mathbb{R}$.

Allerdings ist $\lesssim$ keine Partialordnung, da die Antisymmetrie verletzt wäre: Z. B. ist $1 \lesssim \mathrm{i}$ und $\mathrm{i} \lesssim 1$, jedoch ist $1 \neq \mathrm{i}$.

Der Kern $\sim$ der Quasiordnung $\lesssim$ ist definiert durch

$$z \sim w \ \overset{(D.1)}{:\Longleftrightarrow}\ (z \lesssim w) \wedge (w \lesssim z) \ \overset{(D.2)}{\Longleftrightarrow}\ (|z| \leq |w|) \wedge (|w| \leq |z|) \ \Longleftrightarrow\ |z| = |w|,$$

m. a. W. sind zwei komplexe Zahlen genau dann äquivalent bzgl. $\sim$, wenn ihr Absolutbetrag gleich ist. Ist $z_0 \in \mathbb{C}$ eine beliebige komplexe Zahl, so ist ihre $\sim$-Äquivalenzklasse

$$[z_0]_\sim = \{z \in \mathbb{C} \,:\, |z| = |z_0|\} \tag{D.3}$$

ein Kreis in der komplexen Ebene mit Radius $|z_0|$ um den Nullpunkt $0 \in \mathbb{C}$.

Vermöge des Kerns lässt sich zu jeder Quasiordnung eine kanonische Partialordnung festlegen.

**Definition & Satz D.3.4.** *Ist $\lesssim$ eine Quasiordnung auf $M$ und ${}^{M}/_{\sim} = \{[a]_\sim : a \in M\}$ die Quotientenmenge bezüglich des Kerns der Quasiordnung $\lesssim$, so definieren wir die* kanonische Partialordnung zu $\lesssim$ *durch die wohldefinierte Festlegung*

$$[a]_\sim \leq [b]_\sim \ :\Longleftrightarrow\ a \lesssim b. \tag{D.4}$$

**Beispiel D.3.5** (Fortsetzung von Beispiel D.3.3)**.** Auf ${}^{\mathbb{C}}/_{\sim}$ ist die kanonische Partialordnung zu $\lesssim$ gegeben durch

$$[z]_\sim \leq [w]_\sim \ \overset{\text{(D.4)}}{:\Longleftrightarrow}\ z \lesssim w \ \overset{\text{(D.2)}}{\Longleftrightarrow}\ |z| \leq |w|,$$

was wegen (D.3) repräsentantenunabhängig ist. Die Äquivalenzklasse einer Zahl $z$ ist also genau dann kleiner als die zu einer Zahl $w$, falls $z$ auf einem Kreis um 0 mit kleinerem Radius liegt, als der Radius des Kreises um 0, auf dem $w$ liegt.

**Hauptbeispiel D.3.6.** Wir haben auf Seite 35 bzw. in Beispiel & Satz D.1.6 gesehen, dass das Konzept der $p$-Simulation eine Quasiordnung $\preccurlyeq_p$ auf der Menge $\mathfrak{M}$ der aussagenlogischen Beweissysteme definiert.

Das Konzept der $p$-Äquivalenz stellt eine Äquivalenzrelation $\equiv_p$ dar, die wegen Definition 3.3.13 (i) und (D.1) der Kern von $\preccurlyeq_p$ ist.

Folglich erhalten wir auf ${}^{\mathfrak{M}}/_{\equiv_p} = \left\{[V]_{\equiv_p} : V \in \mathfrak{M}\right\}$ die kanonische Partialordnung $\leq_p$ zu $\preccurlyeq_p$ durch *Faktorisierung*, d. h. für zwei aussagenlogische Beweissysteme $V_1, V_2 \in \mathfrak{M}$ gilt

$$[V_1]_{\equiv_p} \leq_p [V_2]_{\equiv_p} \ :\Longleftrightarrow\ V_1 \preccurlyeq_p V_2.$$

**Notiz D.3.7** (Übernommen von [SRI2013], Seite 118)**.** Wie oftmals in der Mathematik beim Arbeiten mit Äquivalenzrelationen fällt in der Praxis die Trennung zwischen partiellen Ordnungen und Quasiordnungen oftmals unscharf aus: Da eine Quasiordnung $\lesssim$ durch Identifizieren bezüglich $\sim$ äquivalenter Objekte zu einer partiellen Ordnung wird, ist der formal korrekte Begriff oftmals lediglich eine Frage des Abstraktionsniveaus der angenommenen Perspektive.

# Literaturverzeichnis

Wir zitieren bei Artikeln, die in Konferenzbänden und einem Journal erschienen sind oder in mehr als einem Journal erschienen sind, jeweils die (aktuellere) Journal-Variante und verweisen auf die vorausgehende Veröffentlichung mittels folgender Abkürzungen:

| | |
|---|---|
| BEATCS | Bulletin of the EATCS |
| CCC | Computational Complexity Conference |
| ECCC | Electronic Colloquium on Computational Complexity |
| FOCS | Annual IEEE Symposium on Foundations of Computer Science |
| ICCAD | International Conference on Control, Automation and Diagnosis |
| STACS | Symposium on Theoretical Aspects of Computer Science |
| STOC | Annual ACM Symposium on the Theory of Computing |

Weiter finden folgende Abkürzungen Verwendung:

| | |
|---|---|
| AAAI | Association for the Advancement of Artificial Intelligence |
| ACM | Association for Computing Machinery |
| AFIPS | American Federation of Information Processing Societies |
| ASI | Advanced Science Institutes |
| EACSL | European Association for Computer Science Logic |
| EATCS | European Association for Theoretical Computer Science |
| IAS | Institute for Advanced Study |
| IBM | International Business Machines Corporation |
| IEEE | Institute of Electrical and Electronics Engineers |
| KTH | Kungliga Tekniska Högskolan (Königliche Technische Hochschule) |
| MIT | Massachusetts Institute of Technology |
| NoNA | Nordic Network on Algorithms |
| SIAM | Society for Industrial and Applied Mathematics |
| SIGLOG | Special Interest Group on Logic and Computation |
| SIGACT | Special Interest Group on Algorithms and Computation Theory |
| SoSe | Sommersemester |
| WiSe | Wintersemester |

Alle weiteren Abkürzungen sind durch den jeweiligen Eintrag selbsterklärend.


[Aar2016] Scott Aaronson. $\mathsf{P} \stackrel{?}{=} \mathsf{NP}$. In *Open Problems in Mathematics*. Hrsg. von John Forbes Nash, Jr. und Michael Th. Rassias. S. 1–122. Cham, Schweiz: Springer International Publishing, Juli 2016.
(Zitiert auf Seite 28).

[AB2009] Sanjeev Arora und Boaz Barak. *Computational Complexity: A Modern Approach*. Cambridge, Vereinigtes Königreich et al.: Cambridge University Press, Juni 2009.
(Zitiert auf den Seiten 1, 7, 16, 17, 151, 152, 156).

[ABLM2008] Carlos Ansótegui, María Luisa Bonet, Jordi Levy und Felip Manyà. Measuring the Hardness of SAT Instances. In *Proceedings of the 23rd AAAI Conference on Artificial Intelligence (AAAI '08). Chicago, Illinois, USA, July 13–17, 2008*. Hrsg. von Dieter Fox und Carla P. Gomes. S. 222–228. Menslo Park, Kalifornien: AAAI Press, Juli 2008.
(Zitiert auf Seite 71).

[ABRW2002] Michael Alekhnovich, Eli Ben-Sasson, Alexander A. Razborov und Avi Wigderson. Space Complexity in Propositional Calculus. *SIAM Journal on Computing*, 31(4), S. 1184–1211, Feb. 2002. Vorausgehende Version erschienen in STOC '00.
(Zitiert auf den Seiten 2, 3, 58, 59, 65).

[ACL+2014] Eric Allender, Shiteng Chen, Tiancheng Lou, Periklis A. Papakonstantinou und Bangsheng Tang. Width-Parameterized SAT: Time-Space Tradeoffs. *Theory Of Computing*, 10(12), S. 297–339, Okt. 2014.
(Zitiert auf den Seiten 109, 111).

[AD2008] Albert Atserias und Víctor Dalmau. A Combinatorial Characterization of Resolution Width. *Journal of Computer and System Sciences*, 74(3), S. 323–334, Mai 2008. Vorausgehende Version erschienen in CCC '03.
(Zitiert auf Seite 2).

[AJPU2007] Michael Alekhnovich, Jan Johannsen, Toniann Pitassi und Alasdair Urquhart. An Exponential Separation between Regular and General Resolution. *Theory of Computing*, 3(1), S. 81–102, Mai 2007. Vorausgehende Version erschienen in STOC '02.
(Zitiert auf Seite 56).

[Ajt1994] Miklós Ajtai. The Complexity of the Pigeonhole Principle. *Combinatorica*, 14(4), S. 417–433, Dez. 1994. Vorausgehende Version erschienen in FOCS '88.
(Zitiert auf Seite 46).

[ALN2016] Albert Atserias, Massimo Lauria und Jakob Nordström. Narrow Proofs May Be Maximally Long. *ACM Transactions on Computational Logic*, 17(3), 19:1–19:30, Juli 2016. Vorausgehende Version erschienen in CCC '14.
(Zitiert auf Seite 66).

[AR2008] Michael Alekhnovich und Alexander A. Razborov. Resolution Is Not Automatizable Unless W[P] is Tractable. *SIAM Journal on Computing*, 38(4), S. 1347–1363, Aug. 2008. Vorausgehende Version erschienen in FOCS '01.
(Zitiert auf Seite 38).

[AR2011] Michael Alekhnovich und Alexander A. Razborov. Satisfiability, Branch-Width and Tseitin Tautologies. *Computational Complexity*, 20(4), S. 649–678, Dez. 2011. Vorausgehende Version erschienen in FOCS '02.
(Zitiert auf den Seiten 108, 109).

[AS2009] Gilles Audemard und Laurent Simon. Predicting Learnt Clauses Quality in Modern SAT Solvers. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI '09). Pasadena, California, USA, July 11–17, 2009*. Hrsg. von Craig Boutilier. S. 399–404. Menslo Park, Kalifornien: AAAI Press, Juli 2009.
(Zitiert auf Seite 28).

[BBI2016] Paul Beame, Chris Beck und Russell Impagliazzo. Time-Space Trade-offs in Resolution: Superpolynomial Lower Bounds for Superlinear Space. *SIAM Journal on Computing*, 45(4), S. 1612–1645, Aug. 2016. Vorausgehende Versionen erschienen in STOC '12 und ECCC '11. Siehe auch die Dissertation [Bec2017].
(Zitiert auf den Seiten vi, 3–5, 71, 72, 76, 81, 84, 93, 94, 97, 98, 101, 108, 109, 111, 125–128, 130, 133, 144, 145).

[BDG+2004] Maria Luisa Bonet, Carlos Domingo, Ricard Gavaldà, Alexis Maciel und Toniann Pitassi. Non-Automatizability of Bounded-Depth Frege Proofs. *Computational Complexity*, 13(1-2), S. 47–68, Dez. 2004.
(Zitiert auf Seite 38).

[Bea2004] Paul Beame. Proof Complexity. In *Computational Complexity Theory*. Hrsg. von Steven Rudich und Avi Wigderson. Bd. 10 der Reihe *IAS/Park City Mathematics Series*, S. 199–246. Providence/Princeton: American Mathematical Society & Institute for Advanced Study, Aug. 2004.
(Zitiert auf den Seiten 23, 27, 37, 38, 46).

[Bec2017] Chris Beck. *Time and Space in Proof Complexity*. Dissertation. Princeton University, 2017.
(Zitiert auf den Seiten 2, 4, 5, 47, 81, 87, 90, 93, 96, 101, 108, 109, 111, 115, 121, 125, 126, 128, 130, 132, 144, 145, 168).

[Bec2018] Chris Beck. *Persönliche Korrespondenz sowie Vortragsnotizen.* März 2018.
(Zitiert auf den Seiten vi, 2, 101, 111, 121).

[Ben2007] Eli Ben-Sasson. *Persönliche Korrespondenz zwischen Eli Ben-Sasson und Jakob Nordström.* Siehe hierzu das Literaturverzeichnis in [Nor2013]. 2007.
(Zitiert auf Seite 3).

[Ben2009a] Eli Ben-Sasson. *Course 236604: Propositional Proof Complexity.* Technion Israel Institute of Technology, Haifa, Israel. 2009. URL: `http://www.cs.technion.ac.il/~idddo/notes_2008_Fall.pdf` (besucht am 01. 07. 2018).
(Zitiert auf den Seiten 39, 51, 53, 65, 66, 68, 69).

[Ben2009b] Eli Ben-Sasson. Size-Space Tradeoffs for Resolution. *SIAM Journal on Computing*, 38(6), S. 2511–2525, Mai 2009. Vorausgehende Version erschienen in STOC '02.
(Zitiert auf den Seiten 71, 84, 85, 87, 90).

[BG2003] Eli Ben-Sasson und Nicola Galesi. Space Complexity of Random Formulae in Resolution. *Random Structures & Algorithms*, 23(1), S. 92–109, Mai 2003. Vorausgehende Version erschienen in CCC '01.
(Zitiert auf den Seiten 3, 65).

[BH2008] Samuel R. Buss und Jan Hoffmann. The NP-Hardness of Finding a Directed Acyclic Graph for Regular Resolution. *Theoretical Computer Science*, 396(1-3), S. 271–276, Mai 2008.
(Zitiert auf Seite 55).

[BHJ2008] Samuel R. Buss, Jan Hoffmann und Jan Johannsen. Resolution Trees with Lemmas: Resolution Refinements that Characterize DLL Algorithms with Clause Learning. *Logical Methods in Computer Science*, 4(4), Dez. 2008.
(Zitiert auf Seite 53).

[BHMW2009] Armin Biere, Marijn J. H. Heule, Hans van Maaren und Toby Walsh, Hrsg. *Handbook of Satisfiability.* Bd. 185 der Reihe *Frontiers in Artificial Intelligence and Applications.* Amsterdam: IOS Press, Feb. 2009.
(Zitiert auf den Seiten 2, 25, 29, 65, 69).

[Bie2010] Armin Biere. *Lingeling, Plingeling, PicoSAT and PrecoSAT at* SAT *Race 2010.* Technical Report 10/1, FMV Reports Series, Institute for Formal Models und Verification, Johannes Kepler University, Linz, Österreich, Aug. 2010.
(Zitiert auf Seite 28).

[BIW2004] Eli Ben-Sasson, Russell Impagliazzo und Avi Wigderson. Near Optimal Separation of Tree-Like and General Resolution. *Combinatorica*, 24(4), S. 585–603, Sep. 2004.
(Zitiert auf Seite 56).

[BK2014] Olaf Beyersdorff und Oliver Kullmann. Unified Characterisations of Resolution Hardness Measures. In *Proceedings of the 17th International Conference on Theory and Applications of Satisfiability Testing (SAT '14). Held as Part of the Vienna Summer of Logic, VSL 2014. Vienna, Austria, July 14–17, 2014.* Hrsg. von Carsten Sinz und Uwe Egly. Bd. 8561 der Reihe *Lecture Notes in Computer Science*, S. 170–187. Cham, Schweiz et al.: Springer, Aug. 2014.
(Zitiert auf Seite 71).

[BKS2004] Paul Beame, Henry A. Kautz und Ashish Sabharwal. Towards Understanding and Harnessing the Potential of Clause Learning. *Journal of Artificial Intelligence Research*, 22, S. 319–351, Dez. 2004.
(Zitiert auf den Seiten 3, 69).

[Bla1937] Archie Blake. *Canonical Expressions in Boolean Algebra.* Dissertation. University of Chicago, 1937.
(Zitiert auf Seite 47).

[Blå2005] Viktor Blåsjö. The Isoperimetric Problem. *The American Mathematical Monthly*, 112(6), S. 526–566, 2005.
(Zitiert auf Seite 113).

[BLW1976] Norman L. Biggs, E. Keith Lloyd und Robin J. Wilson. *Graph Theory, 1736–1936*. Oxford et al.: Oxford University Press, 1976.
(Zitiert auf Seite 173).

[BM1988] László Babai und Shlomo Moran. Arthur-Merlin Games: A Randomized Proof System, and a Hierarchy of Complexity Classes. *Journal of Computer and System Sciences*, 36(2), S. 254–276, Apr. 1988. Aufbauend auf László Babais Artikel *Trading Group Theory for Randomness* in STOC '85.
(Zitiert auf Seite 58).

[BN2008] Eli Ben-Sasson und Jakob Nordström. Short Proofs May Be Spacious: An Optimal Separation of Space and Length in Resolution. In *Proceedings of the 49th Annual IEEE Symposium on Foundations of Computer Science (FOCS '08). Philadelphia, Pennsylvania, USA, October 25–28, 2008*. S. 709–718. Los Alamitos, Kalifornien / Washington / Tōkyō: IEEE Computer Society Press, 2008.
(Zitiert auf den Seiten 3, 71).

[BN2011] Eli Ben-Sasson und Jakob Nordström. Understanding Space in Proof Complexity: Separations and Trade-offs via Substitutions (Extended Abstract). In *Proceedings of the 2nd Symposium on Innovations in Computer Science (ICS '10). Tshinghua University, Beijing, China, January 7–9, 2011*. Hrsg. von Bernard Chazelle. S. 401–416. Beijing: Tshinghua University Press, 2011.
(Zitiert auf den Seiten 3, 58, 71, 72, 84).

[BNT2013] Chris Beck, Jakob Nordström und Bangsheng Tang. Some Trade-off Results for Polynomial Calculus: Extended Abstract. In *Proceedings of the 45th Annual ACM Symposium on Theory of Computing (STOC '13). Palo Alto, California, USA, June 1–4, 2013*. Hrsg. von Dan Boneh, Tim Roughgarden und Joan Feigenbaum. S. 813–822. New York: ACM, Mai 2013.
(Zitiert auf den Seiten vi, 3–5, 47, 71, 81, 84, 87, 90, 93, 111, 128, 132, 144, 145).

[Bol1988] Béla Bollobás. The Isoperimetric Number of Random Regular Graphs. *European Journal of Combinatorics*, 9(3), S. 241–244, Mai 1988.
(Zitiert auf Seite 64).

[Bon2018] Ilario Bonacina. *Space in Weak Propositional Proof Systems*. 1. Aufl. Cham, Schweiz: Springer International Publishing, Jan. 2018.
(Zitiert auf den Seiten 1, 2, 33, 46, 47, 51, 53, 56–60, 62, 63, 66, 69, 84, 97).

[BP1996] Paul Beame und Toniann Pitassi. Simplified and Improved Resolution Lower Bounds. In *Proceedings of the 37th Annual IEEE Symposium on Foundations of Computer Science (FOCS '96). Burlington, Vermont, USA, 14–16 October, 1996*. S. 274–282. Los Alamitos, Kalifornien / Washington / Tōkyō: IEEE Computer Society Press, Okt. 1996.
(Zitiert auf Seite 62).

[BP2001] Paul Beame und Toniann Pitassi. Propositional Proof Complexity: Past, Present, and Future. In *Current Trends in Theoretical Computer Science: Entering the 21st Century*. Hrsg. von Gheorghe Păun, Grzegorz Rozenberg und Arto Salomaa. S. 42–70. Singapur et al.: World Scientific Publishing, 2001. Vorausgehende Version erschienen in BEATCS '96.
(Zitiert auf den Seiten 23, 25–27, 33, 34, 38, 46, 47, 61).

[BP2007] Joshua Buresh-Oppenheim und Toniann Pitassi. The complexity of resolution refinements. *Journal of Symbolic Logic*, 72(4), S. 1336–1352, Dez. 2007.
(Zitiert auf Seite 56).

[Bru2014] Henning Bruhn-Fujimoto. *Optimierung I*. Vorlesungsmanuskript SoSe 2014. Institut für Optimierung und Operations Research, Universität Ulm, 2014.
(Zitiert auf den Seiten 7, 29).

[Bru2015] Henning Bruhn-Fujimoto. *Combinatorial Optimisation II*. Vorlesungsmanuskript SoSe 2015. Institut für Optimierung und Operations Research, Universität Ulm, 2015.
(Zitiert auf den Seiten 29, 47).

[BS1997] Roberto J. Bayardo, Jr. und Robert Schrag. Using CSP Look-Back Techniques to Solve Real-World SAT Instances. In *Proceedings of the 14th AAAI Conference on Articial Intelligence (AAAI '97). Providence, Rhode Island, USA, July 27–31, 1997*. Hrsg. von Ben Kuipers und Bonnie Webber. S. 203–208. Menslo Park, Kalifornien: AAAI Press, Juli 1997.
(Zitiert auf den Seiten 28, 69).

[Bus1987] Samuel R. Buss. Polynomial Size Proofs of the Propositional Pigeonhole Principle. *Journal of Symbolic Logic*, 52(4), S. 916–927, Dez. 1987.
(Zitiert auf den Seiten 45, 52).

[Bus1995] Samuel R. Buss. Some Remarks on Lengths of Propositional Proofs. *Archive for Mathematical Logic*, 34(6), S. 377–394, Dez. 1995.
(Zitiert auf Seite 46).

[Bus1998] Samuel R. Buss. An Introduction to Proof Theory. In *Handbook of Proof Theory*. Hrsg. von Samuel R. Buss. Bd. 137 der Reihe *Studies in Logic and the Foundations of Mathematics*, S. 1–78. Amsterdam et al.: Elsevier Science, Juli 1998. Siehe auch [Bus1999].
(Zitiert auf den Seiten 23, 41, 46).

[Bus1999] Samuel R. Buss. Propositional Proof Complexity: An Introduction. In *Computational Logic*. Hrsg. von Ulrich Berger und Helmut Schwichtenberg. Bd. 165 der Reihe *NATO ASI Series, Series F: Computer and System Sciences*, S. 127–178. Berlin et al.: Springer, Apr. 1999.
(Zitiert auf den Seiten 30, 36, 42, 45–47, 171).

[Bus2006] Samuel R. Buss. Polynomial-size Frege and Resolution Proofs of st-Connectivity and Hex Tautologies. *Theoretical Computer Science*, 357(1-3), S. 35–52, Juli 2006.
(Zitiert auf Seite 100).

[Bus2012] Samuel R. Buss. Towards $\mathsf{NP} - \mathsf{P}$ via Proof Complexity and Search. *Annals of Pure and Applied Logic*, 163(7), S. 906–917, Juli 2012.
(Zitiert auf Seite 34).

[BW2000] Gertrud Bauer und Markus Wenzel. Computer-Assisted Mathematics at Work: The Hahn-Banach Theorem in Isabelle/Isar. In *Proceedings of the International Workshop on Types for Proofs and Programs (TYPES '99). Lökeberg, Sweden, June 12–16, 1999. Selected Papers*. Hrsg. von Thierry Coquand, Peter Dybjer, Bengt Nordström und Jan Smith. Bd. 1956 der Reihe *Lecture Notes in Computer Science*, S. 61–76. Berlin et al.: Springer, Okt. 2000.
(Zitiert auf Seite 24).

[BW2001] Eli Ben-Sasson und Avi Wigderson. Short Proofs are Narrow – Resolution made Simple. *Journal of the ACM*, 48(2), S. 149–169, März 2001. Vorausgehende Versionen erschienen in STOC '99 und CCC' 99.
(Zitiert auf den Seiten 26, 38, 66, 76, 79, 81, 125, 126, 132–134).

[CEI1996] Matthew Clegg, Jeffery Edmonds und Russell Impagliazzo. Using the Groebner Basis Algorithm to Find Proofs of Unsatisfiability. In *Proceedings of the 28th Annual ACM Symposium on Theory of Computing (STOC '96). Philadelphia, Pennsylvania, USA, May 22–24, 1996*. S. 174–183. New York: ACM, Mai 1996.
(Zitiert auf den Seiten 38, 125).

[Chv1973] Vašek Chvátal. Edmonds Polytopes and a Hierarchy of Combinatorial Problems. *Discrete Mathematics*, 4(4), S. 305–337, Apr. 1973. Wiederveröffentlicht in der 35th Special Anniversary Issue der Discrete Mathematics, 306(10-11), S. 886–904, Mai 2006.
(Zitiert auf Seite 47).

[CJW2006] James A. Carlson, Arthur Jaffe und Andrew Wiles. *The Millennium Prize Problems*. Providence, Rhode Island: American Mathematical Society, Juni 2006.
(Zitiert auf den Seiten 1, 28, 172).

[CK2002] Peter Clote und Evangelos Kranakis. *Boolean Functions and Computation Models*. Teil der Reihe *Texts in Theoretical Computer Science. An EATCS Series*. Hrsg. von Wilfried Brauer, Grezegorz Rozenberg und Arto Salomaa. Berlin et al.: Springer, Sep. 2002.
(Zitiert auf den Seiten 52, 65).

[CL2014] Chin-Liang Chang und Richard Char-Tung Lee. *Symbolic Logic and Mechanical Theorem Proving*. Teil der Reihe *Computer Science and Applied Mathematics*. Erstveröffentlicht 1973. Cambridge, Massachusetts: Academic Press, Juni 2014.
(Zitiert auf Seite 52).

[CLRS2009] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest und Clifford Stein. *Introduction to Algorithms*. 3. Aufl. Cambridge, Massachusetts: MIT Press, Juli 2009.
(Zitiert auf den Seiten 153, 156).

[CM1997] Stephen A. Cook und David G. Mitchell. Finding Hard Instances of the Satisfiability Problem: A Survey. In *Proceedings of a DIMACS Workshop on the Satisfiability Problem: Theory and Applications. Piscataway, New Jersey, USA, March 11–13, 1996*. Hrsg. von Ding-Zhu Du, Jun Gu und Panos M. Pardalos. Bd. 35 der Reihe *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*, S. 1–18. Providence, Rhode Island: American Mathematical Society, Okt. 1997.
(Zitiert auf den Seiten 65, 66, 68).

[Coo1971] Stephen A. Cook. The Complexity of Theorem-Proving Procedures. In *Proceedings of the 3rd Annual ACM Symposium on Theory of Computing (STOC '71). Shaker Heights, Ohio, USA, May 3–5, 1971*. Hrsg. von Michael A. Harrison, Ranan B. Banerji und Jeffrey D. Ullman. S. 151–158. New York: ACM, 1971.
(Zitiert auf den Seiten 17, 28).

[Coo2006] Stephen A. Cook. The P versus NP Problem. In *The Millennium Prize Problems*. Hrsg. von James A. Carlson, Arthur Jaffe und Andrew Wiles. S. 87–104. Providence, Rhode Island: American Mathematical Society, Juni 2006. Siehe [CJW2006].
(Zitiert auf den Seiten 1, 28).

[CPRS2001] Cristian S. Calude, Gheorghe Păun, Grzegorz Rozenberg und Arto Salomaa. *Multiset Processing: Mathematical, Computer Science, and Molecular Computing Points of View*. 1. Aufl. Bd. 2235 der Reihe *Lecture Notes in Computer Science*. Hrsg. von Gerhard Goos, Juris Hartmanis und Jan van Leeuwen. Berlin et al.: Springer, 2001.
(Zitiert auf Seite 19).

[CR1974a] Stephen A. Cook und Robert A. Reckhow. On the Lengths of Proofs in the Propositional Calculus (Preliminary Version). In *Proceedings of the 6th Annual ACM Symposium on Theory of Computing (STOC '74). Seattle, Washington, USA, April 30 – May 2, 1974*. S. 135–148. New York: ACM, 1974. Korrigierte Version in [CR1974b].
(Zitiert auf den Seiten 1, 27, 29–32, 35, 45, 46).

[CR1974b] Stephen Cook und Robert Reckhow. Corrections for "On the Lengths of Proofs in the Propositional Calculus (Preliminary Version)". *ACM SIGACT News*, 6(3), S. 15–22, Juli 1974.
(Zitiert auf Seite 172).

[CR1979] Stephen A. Cook und Robert A. Reckhow. The Relative Efficiency of Propositional Proof Systems. *Journal of Symbolic Logic*, 44(1), S. 36–50, März 1979.
(Zitiert auf den Seiten 1, 45).

[CS1988] Vašek Chvátal und Endre Szemerédi. Many Hard Examples for Resolution. *Journal of the ACM (JACM)*, 35(4), S. 759–768, Okt. 1988.
(Zitiert auf den Seiten 2, 65).

[Dal2013] Dirk van Dalen. *Logic and Structure*. 5. Aufl. Teil der Reihe *Universitext*. Hrsg. von Sheldon Axler et al. London et al.: Springer, 2013.
(Zitiert auf den Seiten 7, 14).

[DLL1962] Martin Davis, George Logemann und Donald W. Loveland. A Machine Program for Theorem-Proving. *Communications of the ACM*, 5(7), S. 394–397, Juli 1962.
(Zitiert auf den Seiten 47, 67).

[DM1941] Ben Dushnik und E. W. Miller. Partially Ordered Sets. *American Journal of Mathematics*, 63(3), S. 600–610, Juli 1941.
(Zitiert auf Seite 162).

[DP1960] Martin Davis und Hilary Putnam. A Computing Procedure for Quantification Theory. *Journal of the ACM (JACM)*, 7(3), S. 201–215, Juli 1960.
(Zitiert auf den Seiten 47, 67).

[EJL+2016] Jan Elffers, Jan Johannsen, Massimo Lauria, Thomas Magnard, Jakob Nordström und Marc Vinyals. Trade-offs Between Time and Memory in a Tighter Model of `CDCL` SAT Solvers. In *Proceedings of the 19th International Conference on Theory and Applications of Satisfiability Testing (SAT '16). Bordeaux, France, July 5–8, 2016*. Hrsg. von Nadie Creignou und Daniel Le Berre. Bd. 9710 der Reihe *Lecture Notes in Computer Science*, S. 160–176. Cham, Schweiz: Springer International Publishing, Juli 2016.
(Zitiert auf Seite 69).

[ES2004] Niklas Eén und Niklas Sörensson. An Extensible SAT-Solver. In *Proceedings of the 6th International Conference on Theory and Applications of Satisfiability Testing (SAT '03). Santa Margherita Ligure, Italy, May 5–8, 2003. Selected Revised Papers*. Hrsg. von Enrico Giunchiglia und Armando Tacchella. Bd. 2919 der Reihe *Lecture Notes in Computer Science*, S. 502–518. Berlin et al.: Springer, 2004.
(Zitiert auf Seite 28).

[ET2001] Juan Luis Esteban und Jacobo Torán. Space Bounds for Resolution. *Information and Computation*, 171(1), S. 84–97, Nov. 2001. Vorausgehende Version erschienen in STACS '99. Siehe auch [Tor1999] für vorausgegangene Resultate.
(Zitiert auf den Seiten 2, 58, 59, 65, 81, 96, 97, 105).

[Eul1741] Leonhard Euler. Solutio problematis ad geometriam situs pertinentis. *Commentarii academiae scientiarum Petropolitanae*, 8, S. 128–140, 1741. Deutsche Übersetzung in [Spe1925], Englische Übersetzung und Republikation in [BLW1976] (The solution of a problem relating to the geometry of position).
(Zitiert auf den Seiten 24, 75).

[Fer2000] Christian G. Fermüller. Implicational Completeness of Signed Resolution. In *Automated Deduction in Classical and Non-Classical Logics: Selected Papers*. Hrsg. von Ricardo Caferra und Gernot Salzer. Bd. 1761 der Reihe *Lecture Notes in Artificial Intelligence (Subseries of Lecture Notes in Computer Science)*, S. 167–174. Berlin et al.: Springer, Apr. 2000.
(Zitiert auf Seite 52).

[FP1983] John Franco und Marvin C. Paull. Probabilistic Analysis of the Davis Putnam Procedure for Solving the Satisfiability Problem. *Discrete Applied Mathematics*, 5(1), S. 77–87, Jan. 1983.
(Zitiert auf Seite 65).

[Fre1879] Gottlob Frege. *Begriffsschrift: Eine der Arithmetischen Nachgebildete Formelsprache des Reinen Denkens*. Halle an der Saale: Verlag von L. Nebert, 1879.
(Zitiert auf den Seiten 25, 41, 44).

[Gal2009] Nicola Galesi. *Introduction to Proof Complexity*. NoNA Summer School on Complexity Theory, St. Petersburg, Russland. Aug. 2009. URL: `https://logic.pdmi.ras.ru/ssct09/slides/SSCT09_Nicola_Galesi_Lecture_1.pdf`; sowie URL: `https://logic.pdmi.ras.ru/ssct09/slides/SSCT09_Nicola_Galesi_Lecture_2.pdf`. (Besucht am 01. 07. 2018).
(Zitiert auf den Seiten 27, 33, 35, 37, 88).

[Gen1935] Gerhard Gentzen. Untersuchungen über das Logische Schließen. *Mathematische Zeitschrift*, 39(1), S. 176–210, 405–431, 1935.
(Zitiert auf Seite 46).

[GMR1989] Shafi Goldwasser, Silvio Micali und Charles Rackoff. The Knowledge Complexity of Interactive Proof Systems. *SIAM Journal on Computation*, 18(1), S. 186–208, 1989. Vorausgehende Version erschienen in STOC '85.
(Zitiert auf Seite 58).

[GN2013] Michael R. Genesereth und Nils J. Nilsson. *Logische Grundlagen der Künstlichen Intelligenz*. Teil der Reihe *Künstliche Intelligenz*. Hrsg. von Wolfgang Bibel und Walter von Hahn. Übers. von Michael Tarnowski. Braunschweig/Wiesbaden: Vieweg, 2013.
(Zitiert auf Seite 47).

[Göd1956] Kurt Gödel. *Brief an John von Neumann des 20. März 1956*. Manuscript Division of the Library of Congress, Washington, D. C. 1956. Für einen Originalabdruck des Briefes siehe z. B. [Sip1992].
(Zitiert auf Seite 28).

[Gom1958] Ralph E. Gomory. *An Algorithm for Integer Solutions to Linear Programs*. Technical Report 1, Princeton IBM Mathematics Research Project, 1958.
(Zitiert auf Seite 47).

[GTT2018] Nicola Galesi, Navid Talebanfard und Jacobo Torán. Cops-Robber Games and the Resolution of Tseitin Formulas. *Accepted Paper der 21st International Conference on Theory and Applications of Satisfiability Testing (SAT '18)*, Apr. 2018. Die SAT '18 Konferenz wird vom 9.–12. Juli 2018 in Oxford, Vereinigtes Königreich stattfinden. Preprint erhältlich unter URL: `http://users.math.cas.cz/talebanfard/tseitin-final.pdf`. (Besucht am 01. 07. 2018).
(Zitiert auf den Seiten 79, 81, 96).

[Hak1985] Armin Haken. The Intractability of Resolution. *Theoretical Computer Science*, 39(2-3), S. 297–308, Aug. 1985.
(Zitiert auf den Seiten 2, 3, 61, 62).

[Har1993] Juris Hartmanis. Gödel, von Neumann and the $\mathsf{P} \stackrel{?}{=} \mathsf{NP}$ problem. In *Current Trends in Theoretical Computer Science: Essays and Tutorials*. Hrsg. von Herbert Edelsbrunner, Hartmut Ehring und Yuri Gurevich. Bd. 40 der Reihe *World Scientific Series in Computer Science*, S. 445–450. Singapur et al.: World Scientific, Aug. 1993.
(Zitiert auf Seite 28).

[Hås2017] Johan Håstad. On Small-Depth Frege Proofs for Tseitin for Grids. In *Proceedings of the 58th Annual IEEE Symposium on Foundations of Computer Science (FOCS '17). Berkeley, California, USA, October 15–17, 2017*. S. 97–108. Los Alamitos, Kalifornien / Washington / Tōkyō: IEEE Computer Society Press, 2017.
(Zitiert auf Seite 79).

[HL2011] Martin Hofmann und Martin Lange. *Automatentheorie und Logik*. Teil der Reihe *eXamen.press*. Heidelberg et al.: Springer, März 2011.
(Zitiert auf Seite 7).

[HLW2006] Shlomo Hoory, Nathan Linial und Avi Wigderson. Expander Graphs and Their Applications. *Bulletin of the American Mathematical Society*, 43(4), S. 439–561, Okt. 2006.
(Zitiert auf Seite 113).

[HMU2011] John E. Hopcroft, Rajeev Motwani und Jeffrey D. Ullman. *Einführung in Automatentheorie, Formale Sprachen und Berechenbarkeit*. 3., aktualisierte Aufl. Teil der Reihe *Pearson Studium – IT*. München et al.: Pearson Education, März 2011.
(Zitiert auf Seite 15).

[Hof2007] Jan Hoffmann. *Resolution Proofs and `DLL` Algorithms with Clause Learning*. Diplomarbeit, Ludwig-Maximilians-Universität München, 2007.
(Zitiert auf den Seiten 7, 33, 39, 41, 69).

[IPS1999] Russell Impagliazzo, Pavel Pudlák und Jirí Sgall. Lower Bounds for the Polynomial Calculus and the Gröbner Basis Algorithm. *Computational Complexity*, 8(2), S. 127–144, Nov. 1999.
(Zitiert auf Seite 125).

[Jär2008] Matti Järvisalo. *Structure-Based Satisfiability Checking: Analyzing and Harnessing the Potential*. Dissertation. Helsinki University of Technology resp. Aalto University, Espoo, Finnland, 2008.
(Zitiert auf Seite 69).

[JMNŽ2012] Matti Järvisalo, Arie Matsliah, Jakob Nordström und Stanislav Živný. Relating Proof Complexity Measures and Practical Hardness of SAT. In *Proceedings of the 18th International Conference on Principles and Practice of Constraint Programming (CP '12). Québec City, Québec, Canada, October 8–12, 2012*. Hrsg. von Michela Milano. Bd. 7514 der Reihe *Lecture Notes in Computer Science*, S. 316–331. Heidelberg et al.: Springer, 2012.
(Zitiert auf den Seiten 3, 69, 71).

[Juk2012] Stasys Jukna. *Boolean Function Complexity: Advances and Frontiers*. Bd. 27 der Reihe *Algorithms and Combinatorics*. Hrsg. von William J. Cook, Ronald Graham, Bernhard Korte, László Lovász, Avi Wigderson und Günter M. Ziegler. Heidelberg et al.: Springer, Jan. 2012.
(Zitiert auf den Seiten 47, 56, 62, 63, 79, 81).

[KL1994] Hans Kleine Büning und Theodor Lettmann. *Aussagenlogik: Deduktion und Algorithmen*. Teil der Reihe *Leitfäden und Monographien der Informatik*. Hrsg. von Hans-Jürgen Appelrath, Volker Claus, Günther Hotz und Klaus Waldschmidt. Stuttgart: B. G. Teubner Verlag, Jan. 1994.
(Zitiert auf den Seiten 47, 48, 50–52, 58, 157, 158).

[KMT2003] Johannes Köbler, Jochen Messner und Jacobo Torán. *Optimal Proof Systems Imply Complete Sets for Promise Classes*. Resultate dieses Papers sind in STACS '98 und CCC '98 erschienen. Juli 2003.
(Zitiert auf Seite 36).

[Koz1977] Dexter Kozen. Lower Bounds for Natural Proof Systems. In *Proceedings of the 18th Annual IEEE Symposium on Foundations of Computer Science (FOCS '77). Providence, Rhode Island, USA, October 31 – November 1, 1977*. S. 254–266. Long Beach, Kalifornien: IEEE Computer Society, 1977.
(Zitiert auf Seite 58).

[KP1989] Jan Krajíček und Pavel Pudlák. Propositional Proof Systems, the Consistency of First Order Theories and the Complexity of Computations. *Journal of Symbolic Logic*, 54(3), S. 1063–1079, Sep. 1989.
(Zitiert auf Seite 36).

[KPW1995] Jan Krajíček, Pavel Pudlák und Alan Woods. An Exponential Lower Bound to the Size of Bounded Depth Frege Proofs of the Pigeonhole Principle. *Random Structures & Algorithms*, 7(1), S. 15–40, Aug. 1995.
(Zitiert auf Seite 46).

[Kra2005] Jan Krajíček. Proof Complexity. In *European Congress of Mathematics. Stockholm, June 27 – July 2, 2004*. Hrsg. von Ari Laptev. S. 221–231. Zürich: European Mathematical Society Publishing House, 2005.
(Zitiert auf Seite 25).

[Kra2018] Jan Krajíček. *Proof Complexity*. Lužnice, Tschechien / Cambridge, Vereinigtes Königreich: Cambridge University Press. Mai 2018. Finaler Vorabentwurf des Buches. Erhältlich unter URL: `https://www.karlin.mff.cuni.cz/~krajicek/prf2.pdf`. (Besucht am 01. 07. 2018).
(Zitiert auf den Seiten 23, 25, 27).

[KS1993] Ephraim Korach und Nir Solel. Tree-Width, Path-Width, and Cutwidth. *Discrete Applied Mathematics*, 43(1), S. 97–101, Mai 1993.
(Zitiert auf Seite 98).

[KS1994] Scott Kirkpatrick und Bart Selman. Critical Behavior in the Satisfiability of Random Boolean Expressions. *Science*, 264(5163), S. 1297–1301, Mai 1994.
(Zitiert auf Seite 65).

[Lau2017a] Massimo Lauria. *Introduction to Proof Complexity – Lecture 3: Lower Bounds Based on Resolution Width*. Vorlesungsmanuskript Fall 2015. Department of Mathematical und Computing Science, Tōkyō Institute of Technology, 2017. URL: `http://www.massimolauria.net/courses/2015.ProofComplexity/lecture3.pdf` (besucht am 01. 07. 2018).
(Zitiert auf Seite 77).

[Lau2017b] Massimo Lauria. *Introduction to Proof Complexity – Lecture 8: Space Complexity in Proof Complexity*. Vorlesungsmanuskript Fall 2015. Department of Mathematical und Computing Science, Tōkyō Institute of Technology, 2017. URL: `http://www.massimolauria.net/courses/2015.ProofComplexity/lecture8.pdf` (besucht am 01. 07. 2018).
(Zitiert auf Seite 71).

[Lee1967] Richard Char-Tung Lee. *A Completeness Theorem and a Computer Program for Finding Theorems Derivable from Given Axioms*. Dissertation. University of California, Berkeley, 1967.
(Zitiert auf Seite 52).

[Lev1973] Leonid Anatolevich Levin. Universal Sequential Search Problems (Übersetzung von Russisch ins Englische). *Problemy Peredachi Informatsii (Problems of Information Transmission)*, 9(3), S. 115–116, 1973.
(Zitiert auf Seite 17).

[Lip2010] Richard J. Lipton. *The* $\mathsf{P} = \mathsf{NP}$ *Question and Gödel's Lost Letter*. New York et al.: Springer, Sep. 2010.
(Zitiert auf Seite 28).

[Lov1969] Donald W. Loveland. A Simplified Format for the Model Elimination Theorem-Proving Procedure. *Journal of the ACM (JACM)*, 16(3), S. 349–363, Juli 1969.
(Zitiert auf Seite 46).

[Lov1970] Donald W. Loveland. A Linear Format for Resolution. In *Proceedings of the Symposium on Automatic Demonstration. Versailles, France, December 1968*. Hrsg. von Michel Laudet, Daniel Lacombe, L. Nolin und Marcel-Paul Schützenberger. Bd. 125 der Reihe *Lecture Notes in Mathematics*, S. 147–162. Berlin et al.: Springer, 1970.
(Zitiert auf Seite 47).

[Lov1978] Donald W. Loveland. *Automated Theorem Proving: A Logical Basis*. Bd. 6 der Reihe *Fundamental Studies in Computer Science*. Amsterdam / New York / Oxford: North-Holland Publishing Company / Elsevier, 1978.
(Zitiert auf Seite 46).

[Löw1908] Leopold Löwenheim. Über das Auflösungsproblem im Logischen Klassenkalkül. *Sitzungsberichte der Berliner Mathematischen Gesellschaft*, 7, S. 90–94, 1908.
(Zitiert auf Seite 47).

[Löw1910] Leopold Löwenheim. Über die Auflösung von Gleichungen im Logischen Gebietekalkül. *Mathematische Annalen*, 68(2), S. 169–207, 1910.
(Zitiert auf Seite 47).

[Löw1913] Leopold Löwenheim. Über Transformationen im Gebietekalkül. *Mathematische Annalen*, 73, S. 245–272, 1913.
(Zitiert auf Seite 47).

[MMZ+2001] Matthew W. Moskewicz, Conor F. Madigan, Ying Zhao, Lintao Zhang und Sharad Malik. `Chaff`: Engineering an Efficient SAT Solver. In *Proceedings of the 38th Design Automation Conference (DAC '01). Las Vegas, Nevada, USA, June 18–22, 2001*. S. 530–535. New York: ACM, 2001.
(Zitiert auf den Seiten 3, 28, 69).

[MPDP2010] Rafael Martí, Juan J. Pantrigo, Abraham Duarte und Eduardo G. Pardo. Cutwidth Minimization Problem. In *Optsicom Project*. University of Valencia, 2010. URL: `http://www.optsicom.es/cutwidth/` (besucht am 01. 07. 2018).
(Zitiert auf Seite 98).

[MS1999] João P. Marques-Silva und Karem A. Sakallah. `GRASP`: A Search Algorithm for Propositional Satisfiability. *IEEE Transactions on Computers*, 48(5), S. 506–521, Mai 1999. Vorausgehende Version erschienen in ICCAD '96.
(Zitiert auf den Seiten 3, 28, 69).

[Nas1955] John Nash. *Brief an die United States National Security Agency (NSA)*. FOIA (Freedom of Information Act) Reports and Releases: Declassified Documents of the National Security Agency. 1955. URL: `https://www.nsa.gov/news-features/declassified-documents/nash-letters/assets/files/nash_letters1.pdf` (besucht am 01. 07. 2018).
(Zitiert auf Seite 28).

[NH2013] Jakob Nordström und Johan Håstad. Towards an Optimal Separation of Space and Length in Resolution. *Theory of Computing*, 9(14), S. 471–557, Mai 2013. Vorausgehende Version erschienen in STOC '08.
(Zitiert auf den Seiten 3, 71).

[Nor2001] Jakob Nordström. *Stålmarck's Method versus Resolution: A Comparative Theoretical Study*. Masterarbeit. KTH Royal Institute of Technology, Stockholm, 2001.
(Zitiert auf den Seiten 7, 14, 51, 53).

[Nor2008] Jakob Nordström. *Short Proofs May Be Spacious: Understanding Space in Resolution*. Dissertation. KTH Royal Institute of Technology, Stockholm, 2008.
(Zitiert auf den Seiten 7, 27, 36–38, 41, 45, 46, 50, 51, 53, 55, 59, 60, 71, 88, 158).

[Nor2009] Jakob Nordström. Narrow Proofs May Be Spacious: Separating Space and Width in Resolution. *SIAM Journal on Computing*, 39(1), S. 59–121, Mai 2009. Vorausgehende Version erschienen in STOC '06.
(Zitiert auf Seite 3).

[Nor2012] Jakob Nordström. *Time-Space Trade-offs in Proof Complexity*. 17th Estonian Winter School in Computer Science, Palmse, Estonia. 2012. Siehe [Nor2013, Nor2016a]. Übergeordnete URL: `http://www.csc.kth.se/~jakobn/teaching/ewscs12/`. (Besucht am 01. 07. 2018).
(Zitiert auf Seite 25).

[Nor2013] Jakob Nordström. Pebble Games, Proof Complexity, and Time-Space Trade-offs. *Logical Methods in Computer Science*, 9(3), Article 15, Sep. 2013.
(Zitiert auf den Seiten 2, 3, 47, 56, 58, 60, 63, 69, 71, 84, 94, 169, 177).

[Nor2015] Jakob Nordström. On the Interplay Between Proof Complexity and SAT Solving. *ACM SIGLOG News*, 2(3), S. 19–44, Juli 2015. Siehe auch [Nor2016b].
(Zitiert auf den Seiten 2, 29, 61, 63–66, 68, 69, 178).

[Nor2016a] Jakob Nordström. *DD2442 Seminars on Theoretical Computer Science Autumn 2016: Proof Complexity. Lecture 1 & Lecture 3*. KTH Royal Institute of Technology, Stockholm. 2016. Siehe auch [Nor2013]. Übergeordnete URL: `http://www.csc.kth.se/utbildning/kth/kurser/DD2442/semteo16/`. (Besucht am 01. 07. 2018).
(Zitiert auf den Seiten 27, 29, 30, 51, 57, 64, 177).

[Nor2016b] Jakob Nordström. *On the Interplay Between Proof Complexity and* SAT *Solving*. Survey Talk, National Research University Higher School of Economics, Moskau. Apr. 2016. Für das zugehörige Survey Paper siehe [Nor2015]. Die Folien des Vortrags sind erhältlich unter URL: `http://www.csc.kth.se/~jakobn/research/TalkMoscow1604Interplay.pdf`. Eine Aufzeichnung des Vortrags findet sich unter URL: `https://www.youtube.com/watch?v=MKTWkVvameY`. (Besucht am 01. 07. 2018).
(Zitiert auf den Seiten 29, 109, 177).

[Pap1994] Christos H. Papadimitriou. *Computational Complexity*. Reading, Massachusetts: Addison Wesley Longman, 1994.
(Zitiert auf den Seiten 7, 15, 31, 152).

[PBI1993] Toniann Pitassi, Paul Beame und Russell Impagliazzo. Exponential Lower Bounds for the Pigeonhole Principle. *Computational Complexity*, 3(2), S. 97–140, Juni 1993. Vorausgehende Version erschienen in STOC '92.
(Zitiert auf Seite 46).

[PD2010] Knot Pipatsrisawat und Adnan Darwiche. On Modern Clause-Learning Satisfiability Solvers. *Journal of Automated Reasoning*, 44(3), S. 277–301, März 2010.
(Zitiert auf Seite 3).

[PD2011] Knot Pipatsrisawat und Adnan Darwiche. On the Power of Clause-Learning SAT Solvers as Resolution Engines. *Artificial Intelligence*, 175(2), S. 512–525, Feb. 2011.
(Zitiert auf den Seiten 3, 69).

[PI2000] Pavel Pudlák und Russell Impagliazzo. A Lower Bound for DLL Algorithms for $k$-SAT (Preliminary Version). In *Proceedings of the 11th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA '00). San Francisco, California, USA, January 9–11, 2000*. Hrsg. von David B. Shmoys. S. 128–136. New York / Philadelphia: ACM & SIAM, 2000.
(Zitiert auf Seite 125).

[Pit2002] Toniann Pitassi. *CS 2429 – Propositional Proof Complexity – Lecture #2*. University of Toronto. Sep. 2002. URL: `https://www.cs.toronto.edu/~toni/Courses/Proofcomplexity/Lectures/Lecture2/lecture2.pdf` (besucht am 01. 07. 2018).
(Zitiert auf den Seiten 37, 56, 68).

[Pit2011] Toniann Pitassi. Propositional Proof Complexity: A Survey on the State of the Art, Including Some Recent Results. In *Proceedings of the 26th Annual IEEE Symposium on Logic in Computer Science (LICS '11). Toronto, Ontario, Canada, June 21–24, 2011*. S. 119. Los Alamitos, Kalifornien / Washington / Tōkyō: IEEE Computer Society Press, Juni 2011.
(Zitiert auf Seite 61).

[Pra1960] Dag Prawitz. An Improved Proof Procedure. *Theoria*, 26(2), S. 102–139, Aug. 1960.
(Zitiert auf Seite 47).

[Pud2008] Pavel Pudlák. Twelve Problems in Proof Complexity. In *Proceedings of the 3rd International Computer Science Symposium in Russia (CSR '08): Computer Science – Theory and Applications. Moscow, Russia, June 7–12, 2008*. Hrsg. von Edward A. Hirsch, Alexander A. Razborov, Alexei L. Semenov und Anatol Slissenko. Bd. 5010 der Reihe *Lecture Notes in Computer Science*, S. 13–27. Berlin et al.: Springer, 2008.
(Zitiert auf den Seiten 1, 46, 61).

[Qui1955] Willard Van Orman Quine. A Way to Simplify Truth Functions. *The American Mathematical Monthly*, 62(9), S. 627–631, Nov. 1955.
(Zitiert auf Seite 47).

[Rac2007] Nicolas Rachinsky. *The Complexity of Resolution Refinements and Satisfiability Algorithms*. Diplomarbeit. Ludwig-Maximilians-Universität München, 2007.
(Zitiert auf den Seiten 11, 33, 35, 53, 56).

[Raz2001] Alexander A. Razborov. Proof Complexity of Pigeonhole Principles. In *Proceedings of the 5th International Conference on Developments in Language Theory (DLT '01). Vienna, Austria, July 16–21, 2001. Revised Papers.* Hrsg. von Werner Kuich, Grzegorz Rozenberg und Arto Salomaa. Bd. 2295 der Reihe *Lecture Notes in Computer Science*, S. 100–116. Berlin et al.: Springer, 2001.
(Zitiert auf Seite 61).

[Raz2003] Alexander A. Razborov. Resolution Lower Bounds for the Weak Functional Pigeonhole Principle. *Theoretical Computer Science*, 303(1), S. 233–243, Juni 2003.
(Zitiert auf Seite 63).

[Raz2004a] Ran Raz. Resolution Lower Bounds for the Weak Pigeonhole Principle. *Journal of the ACM (JACM)*, 51(2), S. 115–138, März 2004. Vorausgehende Version erschienen in STOC '02.
(Zitiert auf Seite 63).

[Raz2004b] Alexander A. Razborov. Resolution Lower Bounds for Perfect Matching Principles. *Journal of Computer and System Sciences*, 69(1), S. 3–27, Aug. 2004. Vorausgehende Version erschienen in CCC '02.
(Zitiert auf Seite 63).

[Rec1975] Robert A. Reckhow. *On the Lengths of Proofs in the Propositional Calculus.* Dissertation. University of Toronto, 1975. URL: `https://www.cs.toronto.edu/~sacook/homepage/reckhow_thesis.pdf` (besucht am 01. 07. 2018).
(Zitiert auf den Seiten 1, 29–31, 39, 41, 42, 47, 152).

[Rob1965] John Alan Robinson. A Machine-Oriented Logic Based on the Resolution Principle. *Journal of the ACM (JACM)*, 12(1), S. 23–41, Jan. 1965.
(Zitiert auf Seite 47).

[Sav1970] Walter J. Savitch. Relationships Between Nondeterministic and Deterministic Tape Complexities. *Journal of Computer and System Sciences*, 4(2), S. 177–192, Apr. 1970.
(Zitiert auf Seite 111).

[Sch1997] Uwe Schöning. Resolutions Proofs, Exponential Bounds and Kolmogorov Complexity. In *Proceedings ot the 22nd International Symposium on Mathematical Foundations of Computer Science (MFCS '97). Bratislava, Slovakia, August 25–29, 1997.* Hrsg. von Igor Prívara und Peter Ružička. Bd. 1295 der Reihe *Lecture Notes in Computer Science*, S. 110–116. Berlin et al.: Springer, 1997.
(Zitiert auf den Seiten 64, 76, 81).

[Sch2011] Friedmar Schulz. *Analysis 1.* 2. Auflage. München: Oldenburg Verlag, 2011.
(Zitiert auf Seite 120).

[Sch2013] Uwe Schöning. *Logik.* Vorlesungsmanuskript SoSe 2014. Institut für Theoretische Informatik, Universität Ulm, 2013.
(Zitiert auf den Seiten 7, 11, 13, 48, 51, 149).

[Sch2016] Georg Schnitger. *Theoretische Informatik 2 – Teil IV Algorithmische Fragestellungen der Logik: Kapitel 7 Beweissysteme für die Aussagenlogik.* Vorlesungsmanuskript SoSe 2016. Siehe [Sch2018a] für die aktualisierte Version. Professur für Theoretische Informatik, Institut für Informatik, Goethe-Universität Frankfurt am Main, 2016.
(Zitiert auf den Seiten 41, 43).

[Sch2018a] Georg Schnitger. *Theoretische Informatik 2.* Vorlesungsmanuskript SoSe 2018. Professur für Theoretische Informatik, Institut für Informatik, Goethe-Universität Frankfurt am Main, 2018. URL: `http://www.thi.cs.uni-frankfurt.de/lehre/gl2/sose18/gl2_sose18_skript.pdf` (besucht am 01. 07. 2018).
(Zitiert auf den Seiten 41, 179).

[Sch2018b] Nicole Schweikardt. *Logik in der Informatik*. Vorlesungsmanuskript WiSe 2017/18. Lehrstuhl Logik in der Informatik, Institut für Informatik, Humboldt-Universität zu Berlin, Feb. 2018. URL: `https://www2.informatik.hu-berlin.de/logik/lehre/WS17-18/Logik/downloads/LogInf-Skript.pdf` (besucht am 01. 07. 2018).
(Zitiert auf den Seiten 7, 9, 11, 14, 40, 67, 68).

[Seg2007] Nathan Segerlind. The Complexity of Propositional Proofs. *Bulletin of Symbolic Logic*, 13(4), S. 417–481, Dez. 2007.
(Zitiert auf den Seiten 23, 61).

[Sip1992] Michael Sipser. The History and Status of the P Versus NP Question. In *Proceedings of the 24th Annual ACM Symposium on Theory of Computing (STOC '92). Victoria, British Columbia, Canada, May 4–6, 1992*. Hrsg. von Sambasiva Rao Kosaraju, Michael Ralph Fellows, Avi Wigderson und John A. Ellis. S. 603–618. New York: ACM, 1992.
(Zitiert auf den Seiten 28, 174).

[Sip2006] Michael Sipser. *Introduction to the Theory of Computation*. 2. Auflage. Boston: Thomson Course Technology, 2006.
(Zitiert auf Seite 152).

[Spe1925] Andreas Speiser. *Klassische Stücke der Mathematik*. Zürich: Orell Füssli, 1925.
(Zitiert auf Seite 173).

[SRI2013] Bernhard Steffen, Oliver Rüthing und Malte Isberner. *Grundlagen der Höheren Informatik: Induktives Vorgehen*. Teil der Reihe *eXamen.press*. Berlin et al.: Springer Vieweg, Okt. 2013.
(Zitiert auf den Seiten 161, 166).

[ST2013] Uwe Schöning und Jacobo Torán. *The Satisfiability Problem: Algorithms and Analyses*. Bd. 3 der Reihe *Mathematics for Applications (Mathematik für Anwendungen)*. Berlin: Lehmanns Media, Juli 2013.
(Zitiert auf den Seiten vi, 3, 7, 17, 29, 47, 48, 55, 61, 62, 64–69, 88, 154, 157, 158, 160).

[Sze2005] Stefan Szeider. The Complexity of Resolution with Generalized Symmetry Rules. *Theory of Computing Systems*, 38(2), S. 171–188, Feb. 2005. Vorausgehende Version erschienen in STACS '03.
(Zitiert auf Seite 53).

[Tap2009] Richard A. Tapia. *The Remarkable Life of the Isoperimetric Problem: The World's Most Influential Mathematics Problem*. 15th Conference for African American Researchers in the Mathematical Sciences, Rice University, Houston, Texas. Juni 2009. URL: `https://rusmp.rice.edu/sites/g/files/bxs1546/f/The%20remarkable%20life%20of%20the%20isoperimetric%20problem.pdf` (besucht am 01. 07. 2018).
(Zitiert auf Seite 113).

[Tor1999] Jacobo Torán. Lower Bounds for Space in Resolution. In *Proceedings of the 13th International Computer Science Logic Workshop (CSL '96). Annual Conference of the EACSL. Madrid, Spain, September 20–25, 1999*. Hrsg. von Jörg Flum und Mario Rodríguez-Artalejo. Bd. 1683 der Reihe *Lecture Notes in Computer Science*, S. 362–373. Berlin et al.: Springer, 1999.
(Zitiert auf den Seiten 76, 79, 173).

[Tor2017] Jacobo Torán. *Komplexitätstheorie*. Vorlesungsmanuskript SoSe 2017. Institut für Theoretische Informatik, Universität Ulm, 2017.
(Zitiert auf den Seiten 7, 16, 70, 71).

[Tse1968] Grigori S. Tseitin. On the Complexity of Derivation in Propositional Calculus. *Studies in Constrained Mathematics and Mathematical Logic*, 1968. Siehe auch [Tse1983].
(Zitiert auf den Seiten 55, 76, 81, 158).

[Tse1983] Grigori S. Tseitin. On the Complexity of Derivation in Propositional Calculus. In *Automation of Reasoning 2: Classical Papers on Computational Logic 1967–1970*. Hrsg. von Jörg Siekmann und Graham Wrightson. Teil der Reihe *Symbolic Computation*, S. 466–483. Berlin et al.: Springer, 1983.
(Zitiert auf den Seiten 52, 61, 180).

[Urq1987] Alasdair Urquhart. Hard Examples for Resolution. *Journal of the ACM (JACM)*, 34(1), S. 209–219, Jan. 1987.
(Zitiert auf den Seiten 2, 52, 64, 79, 81).

[Urq1995] Alasdair Urquhart. The Complexity of Propositional Proofs. *Bulletin of Symbolic Logic*, 1(4), S. 425–467, Dez. 1995.
(Zitiert auf den Seiten 23, 46, 56).

[Urq2011] Alasdair Urquhart. A Near-Optimal Separation of Regular and General Resolution. *SIAM Journal on Computing*, 40(1), S. 107–121, Feb. 2011.
(Zitiert auf Seite 56).

[Vol2013] Lutz Volkmann. *Fundamente der Graphentheorie*. Teil der Reihe *Springer Lehrbuch Mathematik*. Wien et al.: Springer, Okt. 2013.
(Zitiert auf den Seiten 7, 75).

[WCR1964] Lawrence Wos, Daniel Carson und George Robinson. The Unit Preference Strategy in Theorem Proving. In *Proceedings of the 3rd Fall Joint Computer Conference (FJCC '64). San Francisco, California, USA, October 27–29, 1964*. S. 615–621. AFIPS, 1964.
(Zitiert auf Seite 47).

[Weg1987] Ingo Wegener. *The Complexity of Boolean Functions*. Teil der Reihe *Wiley-Teubner Series in Computer Science*. Hoboken, New Jersey: John Wiley & Sons, Sep. 1987.
(Zitiert auf Seite 29).

[Wig2011] Avi Wigderson. The Gödel Phenomenon in Mathematics: A Modern View. In *Kurt Gödel and the Foundations of Mathematics: Horizons of Truth*. Hrsg. von Matthias Baaz, Christos H. Papadimitriou, Hilary W. Putnam, Dana S. Scott und Charles L. Harper, Jr. S. 475–508. Cambridge, Vereinigtes Königreich et al.: Cambridge University Press, 2011.
(Zitiert auf Seite 28).

[ZH1994] Hantao Zhang und Jieh Hsiang. Solving Open Quasigroup Problems by Propositional Reasoning. In *Proceedings of the International Computer Symposium (ICC '94). Hsinchu, Taiwan, December 12–15, 1994*. Hsinchu, Taiwan: Department of Computer Science und Information Engineering, National Chiao Tung University, 1994.
(Zitiert auf Seite 29).

[ZMMM2001] Lintao Zhang, Conor F. Madigan, Matthew H. Moskewicz und Sharad Malik. Efficient Conflict Driven Learning in a Boolean Satisfiability Solver. In *Proceedings of the 2001 International Conference on Computer-Aided Design (ICCAD '01). San Jose, California, USA, November 4–8, 2001*. Hrsg. von Rolf Ernst. S. 279–285. Los Alamitos, Kalifornien / Washington / Tōkyō: IEEE Computer Society Press, 2001.
(Zitiert auf Seite 3).

# Index und Symbolverzeichnis

## Symbole

## G



## H



## I



## K

## S



## T

## W



## X



## Z