Parametrisierte Komplexität in der polynomiellen Hierarchie[1]

Ronald de Haan[2]

Abstract: In dieser Arbeit erweitern wir die Theorie der parametrisierten Komplexität, um Probleme, die von höheren Ebenen der polynomiellen Hierarchie stammen, adäquat analysieren zu können. Wir erweitern die bekannten Konzepte und Methoden in grundlegender Weise, um auch die bemerkenswerte Effektivität von existierenden SAT-Solvern theoretisch zu berücksichtigen. Wir demonstrieren, dass unser neues Instrumentarium es ermöglicht, die exakte Komplexität einer Vielzahl von fundamentaler Berechnungsproblemen zu bestimmen, und in Folge deren theoretische Schwere einzuordnen und zu vergleichen. Die betrachteten Probleme stammen aus vielen Bereichen der Informatik, z.B. der Künstlichen Intelligenz, Wissensräpresentation, Verifikation und Optimierung.

1 Einführung

Viele wichtige Berechnungsprobleme haben sich als schwer und vermutlich nicht in Polynomialzeit lösbar herausgestellt. Die Forschung der letzten zwei Jahrzehnten hat unter Anderem die folgenden zwei Ansätze hervorgebracht, die mit solchen Problemen umgehen können: (a) festparameter-handhabbare (*fixed-parameter tractable*, FPT) Algorithmen aus dem Gebiet der parametrisierten Komplexität, und (b) Lösung durch Übersetzung in eine äquivalente Eingabe des auslagenlogischen Erfüllbarkeitsproblems (SAT) und die nachfolgende Lösung mittels eines SAT-Solvers. In der vorliegenden Arbeit untersuchen wir die Kombination dieser beiden Ansätze, nämlich die Methode der *festparameter-handhabbaren Übersetzung nach SAT*, die in gewisser Weise das Beste aus den beiden Welten (a und b) vereint, und das Potenzial besitzt, die erstaunliche Mächtigkeit heutiger SAT-Solver auch auf Probleme, die schwerer als NP sind, anzuwenden. Um theoretisch zu erforschen, in wieweit dies tatsächlich möglich ist, sind die bekannten theoretischen Konzepte und Methoden nur unzureichend.

In dieser Arbeit entwickeln wir daher entsprechende neue Konzepte und Methoden, die im Kern aus einer Reihe neuer parameterisierter Komplexitätsklassen bestehen, die hauptsächlich zwischen der ersten und zweiten Stufe der polynomiellen Hierarchie (PH) angesiedelt sind. Diese neuen Klassen ermöglichen es uns, die exakte Komplexität von vielen Berechnungsproblemen zu bestimmen, und theoretische Evidenz zu liefern, ob die Probleme eine festparameter-handhabbare Übersetzung nach SAT erlauben oder nicht.

Die betrachteten Probleme entstammen einem weiten Feld von Anwendungsgebieten, wie zum Beispiel der Künstliche Intelligenz, der Wissensräpresentation, der Verifikation und der Optimierung. Wir fassen die Ergebnisse unserer Untersuchungen von konkreten Problemen in Form eines *Kompendiums mit über 75 Einträgen* zusammen.

[1] Englischer Titel der Arbeit: "Parameterized Complexity in the Polynomial Hierarchy"

[2] Institute for Logic, Language and Computation, University of Amsterdam, R.deHaan@uva.nl

2 Hintergrund: Nicht-Handhabbarkeit meistern

Nicht-Handhabbarkeit ist ein wichtiges Konzept und Forschungsthema der theoretischen Informatik mit Relevanz für fast alle Bereiche menschlicher Aktivität. Beispielsweise gibt es zahlreiche Such- und Optimierungsprobleme, die erhebliche Herausforderungen an die InformatikerInnen stellen, die an Methoden zur praktischen Lösung dieser Probleme arbeiten. Um solche Probleme zu lösen, ist es oft wichtig festzustellen, in welchem Kontext diese Probleme handhabbar (*tractable*) sind, und in welchem Kontext dies nicht der Fall ist (Nicht-Handhabbarkeit, *intractability*).

Wegen der mathematischen Schwierigkeit (oder sogar Unmöglichkeit) in wichtigen Fällen die Nicht-Handhabbarkeit von Problemen absolut (also ohne zusätzlicher Annahmen) zu zeigen, ist das Standardverfahren im Feld der Komplexitätstheorie, die relative Nicht-Handhabbarkeit. Das heisst, mittels geeigneter Algorithmen, die ein Problem in ein anderes Problem übersetzen (sogenannte Reduktionen), kann festgestellt werden, dass das eine Problem mindestens so schwer zu lösen ist wie das andere. Damit ist es auch möglich, die schwersten Probleme innerhalb einer Komplexitätsklasse zu identifizieren. Solche Probleme werden dann als *vollständig* für die Komplexitätsklasse bezeichnet. Ein prominentes Beispiel ist die Komplexitätsklasse NP, für die schon seit den 1970er Jahren hunderte von vollständigen Problemen bekannt sind. Eines der (offenen) Milleniumsprobleme ist die Frage, ob NP-vollständigen Probleme in deterministischer Polynomialzeit gelöst werden können. Viele Forscher vermuten, dass das nicht der Fall ist.

Wenn für ein Problem kein Lösungsverfahren bekannt ist, das für alle möglichen Problemeingaben eine Lösung in Polynomialzeit finden kann, also z.B. für ein als NP-vollständig bekanntes Problem, muss das Problem ja in der Praxis trotzdem gelöst werden. Die InformatikerInnen müssen daher weiter nach möglichst effizienten Methoden suchen, die das Problem zumindest für viele in die Praxis wichtigen Fällen effizient lösen können. Tatsachlich hat die Informatikforschung der letzten Jahrzehnte mehrere Ansätze entwickelt, die mit nicht-handhabbaren Problemen umgehen können. Zwei dieser Ansätze dienen als Grundlage für die vorliegende Arbeit, und deshalb möchten wir diese beiden Ansätze und die ihnen zu Grunde liegenden Ideen hier kurz vorstellen und diskutieren: (a) *festparameter-handhabbare Algorithmen* aus dem Gebiet der parametrisierten Komplexität [DF13, FG06], und (b) *Lösung durch Übersetzung in eine äquivalente Instanz des auslagenlogischen Erfüllbarkeitsproblems (SAT)* und die nachfolgende Lösung mithilfe eines SAT-Solvers [Bi09].

Beide dieser Ansätze wurden erfolgreich verwendet, um vor allem NP-vollständige Probleme, unter Berücksichtigung von zusätzlichen Eigenschaften von Problemeingaben, effizient zu lösen. Hierbei zielt der Ansatz (a) hauptsächlich auf theoretische Laufzeitgarantien, und Ansatz (b) hauptsächlich auf die praktische Lösbarkeit, sogar im industriellen Massstab, ab. In beiden Fällen ist eine möglichst gute Skalierbarkeit in Hinblick auf die Eingabegröße das Ziel.

(a) Festparameter-Handhabbarkeit Der erste der beiden Ansätze erweitert den Standardbegriff der Handhabbarkeit (Lösbarkeit in Polynomialzeit) auf Laufzeiten, die zwar exponentiell sein können, wo allerdings der exponentielle Faktor auf einen bestimmten

Aspekt der Eingabe (repräsentiert durch dem Parameter) limitiert ist. Dieser Parameter ist typischerweise eine ganze Zahl, die eine bestimmte Eigenschaft der Problemeingabe repräsentiert und in realistischen Eingaben als klein angenommen werden kann. Dieser erweiterte Bergriff der Handhabbarkeit wird als Festparameter-Handhabbarkeit (*fixed-parameter tractability, fpt*) bezeichnet. Genauer gesagt, betrachtet man hierbei Laufzeiten, die von einer Funktion folgender Form beschränkt werden können: $f(k) \cdot n^c$, wobei n die Eingabegröße und k den Wert des Parameters bezeichnet; c ist eine Konstante (die weder von n noch von k abhängt), und f ist eine (möglicherweise exponentielle) Funktion. Hier ist es wesentlich zu beachten, dass für jeden konstanten Wert des Parameters k eine polynomielle Laufzeit vorliegt, wobei die Ordnung c des Polynoms *nicht* vom Parameter abhängt (im Gegensatz zu Laufzeiten der Form n^k). Parametrierte Komplexität ist das Forschungsgebiet, das sich mit festparameter-handhabbaren Algorithmen (kurz *fpt-Algorithmen*) und deren Grenzen beschäftigt. Dieses Forschungsgebiet hat sich in den letzten zwei Jahrzehnten sehr schön entfaltet (siehe, zum Beispiel [Bo12, Cy15, DF13]).

Die Laufzeit von Algorithmen und die Schwere von Problemen in diesem zweidimensionalen Raum zu betrachten, der von Eingangröße und Parameterwert aufgespannt wird, hat sich als sehr sinnvoll und weitreichend herausgestellt. Viele Probleme, die ohne Parametrisierung NP-vollständig sind, können mit geeigneten Parametern als festparameter-handhabbar identifiziert werden, und lassen Algorithmen zu, die Eigenschaften von Problemeingaben explizit ausnützen und das Problem einer effizienten Lösung zugänglich machen. Es wurde auch eine parametrisierte Komplexitätstheorie entwickelt, die auf sogenannten *fpt-Reduktionen* zwischen parameterisierten Problemen aufbaut. Hierbei wird ein Problem in ein anderes mittels eines fpt-Algorithmus übersetzt, wobei der neue Parameterwert durch eine Funktion des alten beschränkt bleibt.

Im Prinzip lässt sich dieser Ansatz auch auf Probleme anwenden, die ohne Parametrisierung schwerer als NP sind, also auf höheren Ebenen der polynomiellen Hierarchie angesiedelt sind. Jedoch um solche Probleme festparameter-handhabbar zu machen, ist man bei der Wahl von geeigneten Parametern dazu gezwungen, sehr stark einschränkende Parameter zu wählen, bei denen nicht zu erwarten ist, dass sie in realistischen Problemeingaben klein sind. Daher hatten bisher fpt-Algorithmen wenig Erfolg bei Problemen, die von höheren Ebenen der polynomiellen Hierarchie stammen.

(b) Übersetzung nach SAT Der zweite der beiden Ansätze basiert auf der Übersetzung der Problemeingabe in eine entscheidungsäquivalente Eingabe für ein Zielproblem, für das es leistungsstarke heuristische Methoden („Solver") gibt. Neben der ganzzahligen linearen Optimierung (*integer linear programming*, ILP) ist das aussagenlogische Erfüllbarkeitsproblem (*satisfiability problem*, SAT) wohl das prominenteste Beispiel für solch ein Zielproblem, für das sehr leistungsstarke Solver verfügbar sind.

SAT
Eingabe: Eine aussagenlogische Formel φ in konjunktiver Normalform (KNF).
Frage: Ist φ erfüllbar?

Das Zusammenspiel von verschieden algorithmischen Methoden (wie z.B. dem Klausellernen) und extrem effizienten Datenstrukturen (wie z.B. den beobachteten Literalen) hat die verfügbaren SAT-Solver extrem (und durchaus überraschend) leistungsfähig gemacht. Heute werden grosse SAT-Eingaben mit hunderttausenden von Variablen routinemässig im industriellen Einsatz gelöst. Die Verifikation von Hardware und Software ist eines der bedeutendsten Einsatzgebiete der SAT-Solver, die aber auch in vielen anderen Bereichen sehr erfolgreich verwendet werden. Diese besondere Leistungsfähigkeit der heutigen SAT-Solver stellt die Informatikforschung vor die interessante Frage nach einer theoretischen Erklärung [Va14]. Ein Aspekt der vorliegenden Arbeit ist es, diese bisher unerklärte Leisungsfähigkeit der SAT-Solver, gewissermassen als *black box* im theoretischen Modell zu integrieren.

Da das Erfüllbarkeitsproblem NP-vollständig ist [Co71], kann im Prinzip jedes NP-vollständige Problem in Polynomialzeit nach SAT übersetzt werden, und das gibt tatsächlich für viele NP-vollständigen Probleme auch einen praktikablen Lösungsansatz. Für Probleme, die (vermutlich) schwerer als NP sind, also z.B. Probleme deren Komplexität auf höheren Stufen der polynomiellen Hierarchie angesiedelt ist, ist eine Übersetzung nach SAT in Polynomialzeit (vermutlich) nicht möglich. Deshalb war bisher der Ansatz der Übersetzung nach SAT primär auf Probleme in NP beschränkt.

3 Nicht-Handhabbarkeit durch die Kombination der beiden Ansätze überwinden

In der vorliegenden Arbeit betrachten wir die vor kurzem aufgekommene Idee, die beiden obigen Ansätze zu kombinieren [FS13, PRS13]. Probleme werden mit Hilfe eines fpt-Algorithmus (statt eines Polynomialzeitalgorithmus) nach SAT übersetzt (oder „reduziert"). Dadurch kann die Fähigkeit von fpt-Algorithmen, bestimmte Eigenschaften von Problemeingaben explizit in Form von Parametern zu berücksichtigen und algorithmisch auszunützen, mit der bemerkenswerten Leistungsstärke von SAT-Solvern kombiniert werden. Solche festparameter-handhabbaren Reduktionen nach SAT (kurz *fpt-SAT-Reduktionen*) stellen ein großes Potential dar, um den Einsatzbereich von SAT-Solvern maßgeblich zu erweitern. Andererseits brauchen die Parameter für eine fpt-SAT-Reduktion bedeutend weniger einschränkend zu sein, als für einen fpt-Algorithmus, der das Problem gänzlich löst.

In anderen Worten, der Ansatz der fpt-SAT-Reduktionen vereint die beiden einzelnen Ansätze (a) und (b) zu einem neuen Ansatz, der mächtiger ist als die beiden einzelnen Ansätze für sich genommen.

Eine Lücke in der bekannten parametrisierten Komplexitätstheorie Viele der algorithmischen Methoden, die in der parameterisierten Algorithmik entwickelt worden sind, lassen sich auch auf die Entwicklung von fpt-SAT-Reduktionen anwenden. Jedoch sind die bekannten komplexitätstheoretischen Konzepte und Methoden nur sehr eingeschränkt dazu geeignet, die Grenzen von fpt-SAT-Reduktionen aufzuzeigen. Viele der Probleme, für die eine fpt-SAT-Reduktion in Frage kommt, befinden sich in einem unerforschten Bereich der Komplexitätslandschaft (siehe Abbildung 1).

Um besser zu verstehen, was mit fpt-SAT-Reduktionen machbar ist, und was nicht, ist eine strukturierte Komplexitätsanalyse notwendig, die auch die Einführung neuer Konzepte beinhaltet.

Beispiel: Minimierung von DNF-Formeln Um die Unzulänglichkeiten bekannter theoretischer Konzepte zu veranschaulichen, betrachten wir zunächst als Beispiel das Problem der Minimierung aussagenlogischer Formeln. Die Aufgabe dieses Problems ist es, durch das Entfernen von Literalen eine aussagenlogische Formel φ_1 in disjunktiver Normalform (DNF) in eine möglichst kleine, logisch äquivalente DNF Formel φ_2 umzuwandeln. Dies ist ein grundlegendes Problem, das für viele praktischen Anwendungen relevant ist. Da dieses Problem Σ_2^p-vollständig ist (d.h. schwerer als NP) [Um00], bietet der Ansatz der Polynomialzeitübersetzungen nach SAT keine praktikable Lösung für dieses Problem. Der Ansatz der fpt-SAT-Reduktionen hat jedoch das Potential, ein Lösungsverfahren zu ermöglichen.

Wenn man als Parameter die Anzahl der von φ_1 zu entfernen Literalen nimmt, scheint eine fpt-SAT-Reduktion tatsächlich in Reichweite zu sein. Für jede konstante Anzahl der zu entfernen Literalen kann das Problem ja in Polynomialzeit in eine SAT-Eingabe übersetzt werden. Wir zeigen in dieser Arbeit, dass dieses parametrisiertes Problem sich in der Lücke zwischen bekannten Komplexitätsklassen (siehe Abbildung 1) befindet. Deswegen reichen die bisher bekannten parametrisierten Komplexitätsklassen nicht aus, um zulänglich untersuchen zu können, ob eine fpt-SAT-Reduktion für dieses Problem möglich ist oder nicht. In der vorliegenden Arbeit erzielen wir theoretische Resultate, die starke Hinweise dafür liefern, dass für das besprochene Minimierungsproblem und für viele weitere Beispielsprobleme keine fpt-SAT-Reduktionen möglich sind.

4 Ziel: Entwicklung und Anwendung neuer Konzepte und Methoden

Das Ziel der vorliegenden Arbeit ist es, eine strukturierte und grundlegende komplexitätstheoretische Analyse zu ermöglichen, die Potential und Grenzen der Methode der fpt-SAT-Reduktion aufzeigt. Hierbei ist es uns besonders wichtig, uns bei der Entwicklung der Methoden durch die Analyse von konkreten Problemen aus verscheiden Bereichen der Informatik leiten zu lassen. Damit soll sichergestellt werden, dass die entwickelten Konzepte auch tatsächlich anwendbar sind, und nicht rein abstrakte und inhaltsleere Kon-

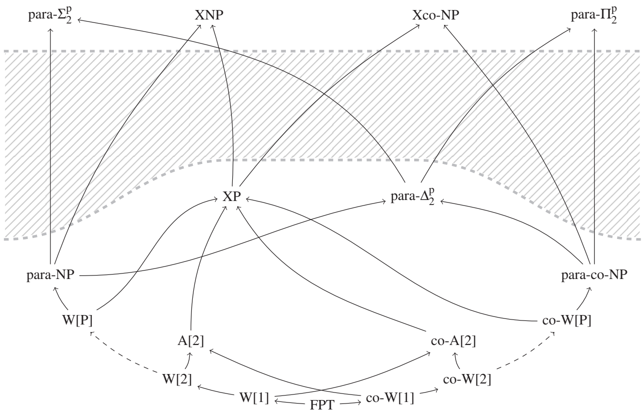

Abb. 1: Die Lücke (gestrichelter grauer Bereich) zwischen den bisher bekannten parametrisierten Komplexitätsklassen: in diesem Bereich wurden vor dieser Arbeit keine Komplexitätsklassen untersucht, obwohl sich dort viele fundamentale parametrisierten Probleme befinden.

strukte bleiben. Die vorgestellte Arbeit ist von den folgenden vier für uns wesentlichen Fragestellungen geprägt:

1. *Was ist das Potential von festparameter-handhabbaren Übersetzungen auf SAT, und macht es einen Unterschied, ob in eine oder mehrere SAT-Eingaben übersetzt wird?* Bisher wurden nur festparameter-handhabbare Übersetzungen in eine einzelne SAT-Eingabe betrachtet, was die parametrisierte Komplexitätsklasse para-NP ergibt. Selbst in diesem Fall ist diese Klasse bisher hauptsächlich nur für negative Resultate herangezogen worden.
2. *Können wir theoretische Methoden entwickeln, um die Möglichkeit von fpt-SAT-Reduktionen in bestimmten Fällen auszuschließen?* Das vorhandene theoretische Instrumentarium ist unzureichend. Es gibt zwar einige bekannten parametrisierten Komplexitätsklassen (z.B. die Klasse para-Σ_2^p), die man benutzen kann, um die Unmöglichkeit von fpt-SAT-Reduktionen aufzuzeigen. Die bekannten Klassen sind jedoch nur in sehr drastischen Fällen anwendbar, wo die Unmöglichkeit von fpt-SAT-Reduktionen mehr oder weniger offensichtlich ist.
3. *Können wir eine feingliedrige parametrisierten Komplexitätstheorie entwickeln, die dazu geeignet ist, die Komplexität verschiedener parametrisierter Probleme auszuwerten, die ohne Parametrisierung zwischen der ersten und zweiten Stufe der polynomiellen Hierarchie angesiedelt sind (d.h. zwischen* para-NP *und* para-Σ_2^p*)?* Solch eine Theorie ist erforderlich um passende untere und obere Schranken für die Komplexität vieler interessanten parametrisierten Problemen zu finden, die in zahlreichen Bereichen der Informatik auftreten.
4. *Für welche nicht-handhabbaren Probleme, die im Bereich der Künstliche Intelligenz und in anderen Bereichen der Informatik auftreten, bieten fpt-SAT-Reduktionen Potential für einen praktikablen Lösunganſatz?* Viele relevante Probleme befinden sich auf der zweiten Ebene der polynomiellen Hierarchie, und können möglicherweise mit dem Ansatz der fpt-SAT-Reduktion effizient gelöst werden. Es fehlt eine strukturierte Untersuchung des Potentials dieses Ansatzes für solche Probleme.

5 Beiträge dieser Arbeit

Als nächstes beschreiben wir die Beiträge, die wir in dieser Arbeit leisten, um die vier vorgenannten Forschungsfragen zu beantworten.

(1) Wir untersuchen die verschiedenen Möglichkeiten, um parametrisierte Probleme anhand festparameter-handhabbarer Übersetzungen in eine oder mehrere SAT-Instanzen zu lösen, und in wieweit eine Übersetzung in mehrere SAT-Instanzen zu einem mächtigeren Verfahren führt.

- Wir liefern die erste strukturierte parametrisierte Komplexitätsuntersuchung, die auf das Enthaltensein in den Klassen para-NP und para-co-NP als primäre positive Ergebnisse abzielt.
- Wir führen verschiedene parametrisierte Komplexitätsklassen ein, die Problemen enthalten, die mittels einer festparameter-handhabbaren Übersetzung in mehrere SAT-Eingaben gelöst werden können. Am wichtigsten hat sich die Klasse $\text{FPT}^{\text{NP}}[\text{few}]$ erwiesen, die Pro-

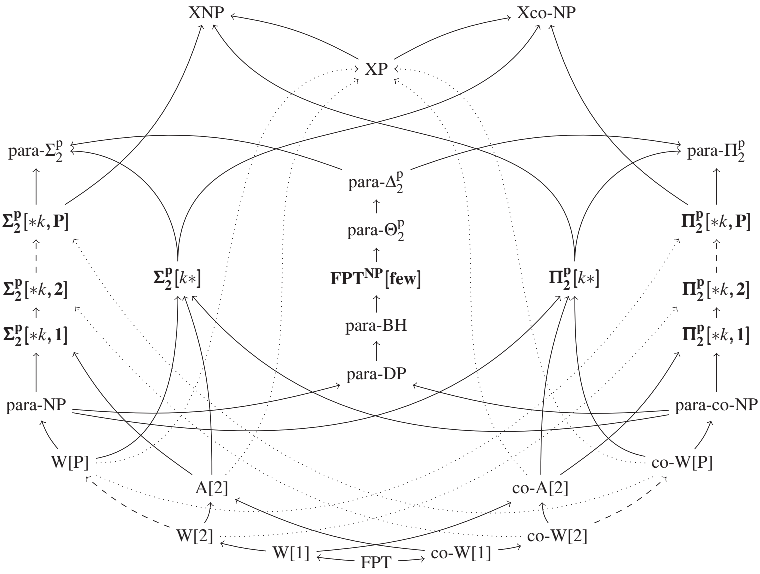

Abb. 2: Die wichtigste parametrisierte Komplexitätsklassen, die wir in dieser Arbeit entwickeln (fett hervorgehoben), und ihre Lage in der breiteren Landschaft parametrisierter Komplexitätsklassen. Die Klassen $\Sigma_2^p[*k,1]$–$\Sigma_2^p[*k,P]$, $\Sigma_2^p[k*]$, $\Pi_2^p[k*]$, und $\Pi_2^p[*k,1]$–$\Pi_2^p[*k,P]$ befinden sich in der Lücke, auf die wir in Abbildung 1 hingewiesen hatten.

bleme beinhaltet, die auf eine vom Parameterwert abhängige Anzahl von SAT-Eingaben übersetzt werden können (siehe Abbildung 2).

- Wir entwickeln theoretische Methoden, um untere Schranken für die erforderliche Anzahl der SAT-Eingaben bei festparameter-handhabbaren Übersetzungen nach SAT zu bestimmen.

(2) Um eine Untersuchung der Grenzen des Potentials von fpt-SAT-Reduktionen zu ermöglichen, zeigen wir, wie mittels Methoden der parametrisierten Komplexität in bestimmten Fällen Hinweise erhalten werden können, dass fpt-SAT-Reduktionen nicht möglich sind.

- Wir zeigen, dass A[2]-Schwere (A[2] ist eine bekannte parametrisierte Komplexitätsklasse) darauf hindeutet, dass für ein parametrisiertes Problem keine fpt-SAT-Reduktion möglich ist (weder eine Übersetzung in eine einzelne, noch in mehrere SAT-Eingaben).

(3) Wir entwickeln neue parametrisierte Komplexitätsklassen, mit denen man feine unterschiedlichen Grade der Komplexität parametrisierter Varianten von Problemen, die von höheren Ebenen der polynomiellen Hierarchie stammen, differenzieren kann.

- Wir liefern starke theoretische Argumente dafür, dass die bekannten parametrisierte Komplexitätsklassen unzureichend sind, um den genauen Grad der Komplexität vieler parametrisierten Problemen festzustellen. Dies gilt insbesondere für Probleme, deren nicht-parametrisierte Formen sich auf höheren Ebenen der polynomiellen Hierarchie befinden. Das heißt, wir zeigen, dass sich viele fundamentale parametrisierten Probleme in der Lücke befinden, auf die wir in Abbildung 1 hingewiesen haben.

- Wir führen neue parametrisierte Komplexitätsklassen ein, die die parametrisierte Komplexitätslandschaft zwischen der ersten und zweiten Ebene der polynomiellen Hierarchie repräsentieren. Diese Klassen sind in Abbildung 2 abgebildet.

  Diese parametrisierten Komplexitätsklassen basieren auf gewichteten Varianten des Erfüllbarkeitsproblems für quantifizierte boolesche Formeln mit einer einzelnen Quantorenalternierung. Z.B., die Klasse $\Sigma_2^{\mathrm{p}}[k*]$ basiert auf folgendem Problem:

$\Sigma_2^{\mathrm{p}}[k*]$-WSAT *Input:* Eine quantifizierte boolesche Formel $\varphi = \exists X.\forall Y.\psi$, und eine positive ganze Zahl k. *Parameter:* k. *Question:* Gibt es eine Wahrheitsbelegung α für X mit Hamming-Gewicht k (d.h., α erfüllt genau k Variablen) so dass für alle Wahrheitsbelegungen β auf Y es gilt, dass ψ erfüllt wird durch $\alpha \cup \beta$?

- Wir erhellen die Frage, welche Rolle der Nichtdeterminismus in diesen parametrisierten Komplexitätsklassen spielt, indem wir alternative Charakterisierungen dieser Klassen aufzeigen.

  Wir charakterisieren die Klassen $\Sigma_2^{\mathrm{p}}[k*]$, $\Sigma_2^{\mathrm{p}}[*k,t]$, $\Pi_2^{\mathrm{p}}[k*]$ und $\Pi_2^{\mathrm{p}}[*k,t]$ mittels alternierender Turingmaschinen (mit passenden Schranken für die Laufzeit sowie Einschränkungen von Art und Ausmaß des verfügbare Nichtdeterminismus); dies führt zu einem Pendant des Satzes von *Cook-Levin*.

- Wir zeigen, dass die neuen parametrisierten Komplexitätsklassen eine Vollständigkeitstheorie erzeugen, die es ermöglicht, die genaue Komplexität fundamentaler parametrisierter Probleme zu ermitteln. Dies war mit den bisher bekannten parametrisierten Komplexitätsklassen nicht möglich.

- Außerdem stellen wir Verbindungen zwischen den neu entwickelten Klassen und anderen Bereichen der Komplexitätstheorie her, wie z.B. der *nicht-uniformen Komplexität* und der *subexponentiellen Komplexität*.

  Wir erzielen beispielsweise ein parametrisiertes Pendant des Satzes von *Karp-Lipton*, das eine Verbindung zwischen den neu entwickelten parametrisierten Klassen und den nicht-uniformen parametrisierten Komplexitätsklassen herstellt.

- Wir verallgemeinern die Klassen $\Sigma_2^{\mathrm{p}}[k*]$, $\Sigma_2^{\mathrm{p}}[*k,t]$, $\Pi_2^{\mathrm{p}}[k*]$ und $\Pi_2^{\mathrm{p}}[*k,t]$, indem wir ähnliche parametrisierte Komplexitätsklassen auf höheren Ebenen der polynomiellen Hierarchie, nämlich die Klassen $\Sigma_i^{\mathrm{p}}[w,t]$ und $\Pi_i^{\mathrm{p}}[w,t]$, für beliebige Zeichenketten $w \in \{*,k\}^i$, einführen.

(4) Wir initiieren eine strukturierte Untersuchung der parametrisierten Komplexität von Problemen aus verscheidenen Bereichen der Informatik (wie z.B. KI), bei der wir uns besonders dafür interessieren, ob eine fpt-SAT-Reduktion möglich ist.

- Wir identifizieren verschiedene bekannte praktische Methoden aus der Literatur (z.B. das *bounded model checking* für bestimmte Zeitlogiken, die in der formalen Verifikation von Bedutung sind), die als fpt-SAT-Reduktionen betrachtet werden können. Dies zeigt auf, dass fpt-SAT-Reduktionen auch tatsächlich zu praktikablen Lösungen führen können.

Problem	Parameter	Klasse
Disjunktives Answer-Set Programming *(Ist ein disjunctives Answer-Set Programm konsistent?)*	# der ‘kontingenten’ Regeln # der disjunctiven Regeln # der nicht-dual-Horn Regeln	$\Sigma_2^p[k*]$ $\Sigma_2^p[*k, \mathrm{P}]$ $\Sigma_2^p[*k, \mathrm{P}]$
DNF Minimierung *(Kann man eine DNF Formel φ durch das Entfernen von Literalen in eine kleinere, äquivalente DNF Formel φ' umwandeln?)*	# der zu entfernen Literalen	$\Sigma_2^p[k*]$
Abductives Schließen *(Kann eine Menge von Konsequenzen durch eine Teilmenge von Hypothesen erklärt werden, unter Berücksichtigung einer logischen Theorie (einer KNF Formel)?)*	# der nicht-Horn Klauseln # der Klauseln der Größe > 2	$\Sigma_2^p[*k, \mathrm{P}]$ $\Sigma_2^p[*k, 1]$
Modellprüfung für Prädikatenlogik ($\exists\forall$) *(Erfüllt ein Modell einen prädikatlogischen Satz mit einem $\exists\forall$-Quantorenpräfix?)*	# der existentiellen Variablen	$\Sigma_2^p[k*]$
Implikantminimierung *(Kann man ein Implikant I einer DNF Formel φ durch das Entfernen von Literalen in einen kleineren Implikanten I' von φ umwandeln?)*	# der zu entfernen Literalen Größe der Implikant I'	$\Sigma_2^p[k*]$ $\Sigma_2^p[k*]$
‘Robuste’ Bedingungserfüllung *(Kann jede Teillösung für eine Menge von Bedingungen auf eine vollständige Lösung erweitert werden?)*	Größe der Teillösungen	$\Sigma_2^p[k*]$
Reparatur logischer Inkonsistenzen *(Kann man eine Teilmenge von einer inkonsistenten Menge aussagelogischer Formeln entfernen, damit die Menge logisch konsistent wird, und dabei bestimmte Formeln erhalten bleiben?)*	# der zu entfernen Formeln	$\mathrm{FPT}^{\mathrm{NP}}[\mathrm{few}]$

Tab. 1: Kleine Auswahl von Problemen aus unserem Kompendium, die für die neu entwickelten parametrisierten Komplexitätsklassen vollständig sind.

- Wir setzen das entwickelte theoretische Instrumentarium dazu ein, um zu untersuchen, für welche der vielen natürlichen parametrisierten Varianten von Problemen aus diversen Bereichen der Informatik, die wir betrachten, fpt-SAT-Reduktionen möglich sind.

  Wir fassen diese Ergebnisse in Form eines ausführlichen Kompendiums (im Garey-Johnson Stil [GJ79]) mit über 75 Einträgen zusammen. Siehe Tabelle 1 für einen kleinen Ausschnitt der Vollständigkeitsergebnisse aus diesem Kompendium.

6 Abschließende Bemerkungen

Die vorliegende Arbeit erweitert die Theorie der parametrisierten Komplexität in geeigneter Weise, um vor allem Probleme, die schwerer als NP sind, zu analysieren. Dabei haben wir auch eine Reihe neuer parameterisierter Komplexitätsklassen eingeführt, die einen bisher unerforschten Bereich in der Komplexitätslandschaft abdecken. Die lange Liste der konkreten Probleme, die wir als vollständig für einzelne neu eingeführte Komplexitätsklassen zeigen konnten, ist ein guter Hinweis dafür, dass die neuen Klassen sinnvoll und nützlich sind. Weiters haben wir die Robustheit der neuen Komplexitätsklassen durch verschiedene alternative Charakterisierungen untermauert. Wir sehen unsere Resultate und Konzepte als den Anfang einer rigorosen komplexitätstheoretischen Untersuchung vieler weiterer

Probleme, wofür wir passende Werkzeuge zur Verfügung stellen. Weiters sehen wir unsere Resultate auch als den ersten Schritt zu einer weiter ausdifferenzierten Theorie, die interessante und herausfordernde Fragestellungen für die komplexitätstheoretische Forschung stellt.

Literaturverzeichnis

[Bi09] Biere, Armin; Heule, Marijn; van Maaren, Hans; Walsh, Toby, Hrsg. Handbook of Satisfiability, Jgg. 185 in Frontiers in Artificial Intelligence and Applications. IOS Press, 2009.

[Bo12] Bodlaender, Hans L.; Downey, Rod; Fomin, Fedor V.; Marx, Dániel, Hrsg. The Multivariate Algorithmic Revolution and Beyond, Jgg. 7370 in Lecture Notes in Computer Science. Springer Verlag, 2012.

[Co71] Cook, Stephen A.: The Complexity of Theorem-Proving Procedures. In: Proc. 3rd Annual Symp. on Theory of Computing. S. 151–158, 1971.

[Cy15] Cygan, Marek; Fomin, Fedor V.; Kowalik, Lukasz; Lokshtanov, Daniel; Marx, Dániel; Pilipczuk, Marcin; Pilipczuk, Michal; Saurabh, Saket: Parameterized Algorithms. Springer, 2015.

[DF13] Downey, Rodney G.; Fellows, Michael R.: Fundamentals of Parameterized Complexity. Texts in Computer Science. Springer Verlag, 2013.

[FG06] Flum, Jörg; Grohe, Martin: Parameterized Complexity Theory, Jgg. XIV in Texts in Theoretical Computer Science. Springer Verlag, Berlin, 2006.

[FS13] Fichte, Johannes Klaus; Szeider, Stefan: Backdoors to Normality for Disjunctive Logic Programs. In: Proceedings of the 27th AAAI Conference on Artificial Intelligence (AAAI 2013). AAAI Press, S. 320–327, 2013.

[GJ79] Garey, Michael R.; Johnson, David R.: Computers and Intractability. W. H. Freeman and Company, New York, San Francisco, 1979.

[PRS13] Pfandler, Andreas; Rümmele, Stefan; Szeider, Stefan: Backdoors to Abduction. In (Rossi, Francesca, Hrsg.): Proceedings of the 23rd International Joint Conference on Artificial Intelligence (IJCAI 2013). AAAI Press/IJCAI, 2013.

[Um00] Umans, Christopher: Approximability and Completeness in the Polynomial Hierarchy. Dissertation, University of California, Berkeley, 2000.

[Va14] Vardi, Moshe Y.: Boolean satisfiability: theory and engineering. Communications of the ACM, 57(3):5, März 2014.

Ronald de Haan studierte von 2007 bis 2010 im Bachelor-Studiengang Sprachwissenschaft und im Bachelor-Studiengang Künstliche Intelligenz an der Universität Utrecht, sowie von 2010 bis 2012 im Master-Studiengang Computational Logic an der Technischen Universität Dresden, an der Freien Universität Bozen, und an der Technischen Universität Wien. In 2016 schloss er sein Doktorat ab an der Technischen Universität Wien. Zur Zeit forscht er als Postdoc am Institute for Logic, Language and Computation der Universität von Amsterdam.