Platz- und Schaltkreiskomplexität von MSO-beschreibbaren Problemen auf baumartig zerlegbaren Strukturen*

Michael Elberfeld

International Computer Science Institute
1947 Center Street, Suite 600
Berkeley, California 94704, USA

elberfeld@icsi.berkeley.edu

Abstract: Dieser Beitrag ist eine deutschsprachige Kurzfassung der Dissertation von Michael Elberfeld [Elb12]. Die Dissertation entwickelt und bearbeitet Fragestellungen aus den Bereichen der Theoretischen Informatik und Mathematischen Logik. Sie untersucht die Platz-, Schaltkreis- und Beschreibungskomplexität von Problemen, die sich durch Formeln in monadischer Logik zweiter Stufe beschreiben lassen und deren Eingaben eine beschränkte Baumweite oder -tiefe besitzen. Die gewonnenen Resultate werden angewendet, um die Komplexität konkreter Entscheidungs-, Zähl- und Optimierungsprobleme aus verschiedenen Anwendungsgebieten zu klassifizieren.

1 Einleitung

Informatik und Mathematische Logik sind verknüpfte Forschungsfelder. Während die Informatik sich mit der *Lösung* von *Berechnungsproblemen* wie dem Finden eines kürzesten Weges in einer Landkarte beschäftigt, untersucht die Mathematische Logik *Eigenschaften* von Strukturen wie Graphen und deren *Beschreibbarkeit* durch logische Formeln. Obwohl diese Fragestellungen auf den ersten Blick recht unterschiedlich erscheinen mögen, gibt es doch Gemeinsamkeiten: Zum einen lassen sich Berechnungsprobleme oft eindeutig und kompakt durch logische Formeln beschreiben. Zum anderen kann man die Frage, ob eine logische Struktur eine Formel erfüllt, selbst als Berechnungsproblem auffassen und nach Algorithmen hierzu suchen. Verbindet man beide Ideen, kann man individuelle Berechnungsprobleme lösen ohne neue spezialisierte Programme schreiben zu müssen. Es reicht aus eine Beschreibung des Problems in Form einer logischen Formel zu entwickeln. Zusammen mit dem Löser, der die Formel auswertet, ergibt sich ein Programm für das Problem. Viele wissenschaftliche Arbeiten haben bisher die *Zeitkomplexität* dieser Art von Logik-basierten Berechnungsproblemen untersucht; also die Frage, wie schnell man

*Englischer Titel der Dissertation: „Space and Circuit Complexity of Monadic Second-Order Definable Problems on Tree-Decomposable Structures."

ein Problem lösen kann. In der Dissertation, von der diese Kurzfassung handelt, liegt der Fokus auf der *Platz-*, und *Schaltkreiskomplexität* von durch logischen Formeln beschreibbaren Problemen; also die Frage, mit wie viel Arbeitsspeicher oder mit welcher Art von integrierten Schaltkreisen ein Problem gelöst werden kann. Ebenfalls wird die eng verwandte *Beschreibungskomplexität* von Problemen eine Rolle spielen; die Frage, in wie weit man logische Problembeschreibungen vereinfachen kann. Hieraus können sich indirekt effizientere Algorithmen ergeben. Seien sie nun schnell, verbrauchen wenig Platz oder haben kleine Schaltkreise.

Allgemeine algorithmische Resultate ergeben sich, wenn man ausdrucksstarke Logiken betrachtet, deren Formeln viele Probleme beschreiben können. Durch die NP-schwere vieler Logik-basierter Probleme ist die Suche nach effizienten Algorithmen aber oft eine waghalsige Gradwanderung. Eines der raffiniertesten Resultate in diesem Gebiet, das eine perfekte Balance zwischen großer Ausdrucksstärke und algorithmischer Effizienz erreicht, ist ein Satz von Courcelle. Dieser sagt, dass sich jedes durch eine Formel in monadischer Logik zweiter Stufe beschreibbare Problem in linearer Zeit lösen lässt, wenn man Eingabestrukturen betrachtet, die eine beschränkte Baumweite haben [Cou90]. Um aufbauend auf dem Satz von Courcelle das Ziel und die Beträge der vorgestellten Dissertation zu diskutieren, werden nun zunächst die Begriffe der monadischen Logik zweiter Stufe und der Baumweite besprochen.

Die *monadische Logik zweiter Stufe* (MSO-*Logik*) verallgemeinert die Prädikatenlogik erster Stufe dahingehend, dass man nicht nur einzelne Elemente einer logischen Struktur durch Quantoren adressieren kann, sondern dies auch für Teilmengen von Elementen und Relationen möglich ist (diese Variante der MSO-Logik nennt man oft auch GSO- oder MSO_2-Logik). Zum Beispiel beschreibt die MSO-Formel $\varphi :=$

$$\exists R\, \exists G\, \exists B\, \forall v \Big(R(v) \vee G(v) \vee B(v) \wedge \\ \forall w \left(E(v,w) \rightarrow \neg(R(v) \wedge R(w)) \wedge \neg(G(v) \wedge G(w)) \wedge \neg(B(v) \wedge B(w)) \right) \Big)$$

genau die Graphen (Kanten werden durch das Relationssymbol E adressiert), deren Knoten sich so in drei Mengen (adressiert durch die Symbole R, G und B) aufteilen lassen, dass benachbarte Knoten in verschiedenen Mengen liegen. Durch Auswertung dieser Formel wird entschieden, ob ein gegebener Graph 3-färbbar ist; also das formale *Entscheidungsproblem* 3-FÄRBBARKEIT gelöst. Zum Beispiel erfüllt der Graph  die obige Formel, der Graph  aber nicht. Neben Berechnungsproblemen, bei denen es hauptsächlich auf eine Entscheidung zwischen „ja“ und „nein“ ankommt, geht es bei *Zähl-* und *Optimierungsproblemen* jeweils um die Frage, wie viele Lösungen einer bestimmten Art es gibt oder wie gut man etwas lösen kann. Diese Probleme kann man häufig auch durch logische Formeln beschreiben. Die MSO-Formel $\psi(X) := \forall v \left(X(v) \vee \exists w \left(X(w) \wedge E(v,w) \right) \right)$ beschreibt zum Beispiel genau die (durch X adressierten) Mengen von Knoten eines Graphen, die alle Knoten abdecken (jeder Knoten des Graphen ist Teil der Menge oder benachbart zu einem Knoten der Menge). Die zugehörigen Zähl- und Optimierungsprobleme fragen für einen gegebenen Graphen nach der Anzahl solcher Lösungen beziehungsweise der Größe einer kleinsten Lösung. Beispielsweise kann der Graph  durch 17 verschiedene Knotenmengen abgedeckt werden, wobei die kleinste Menge aus 3 Knoten besteht. Um

die Gemeinsamkeiten der drei Problemarten (Entscheidungs-, Zähl- und Optimierungsprobleme) bei deren Lösung auszunutzen wird in der Arbeit das Konzept des *Lösungshistogramms* eingeführt: Für das gerade betrachtete Paar von Formel und Graph ist dies das Feld $0 \mid 0 \mid 3 \mid 8 \mid 5 \mid 1$ mit 6 Einträgen, die von links nach rechts durch 0 bis 5 indiziert werden. Entsprechend des Indexes steht an jeder Position die Anzahl der verschiedenen abdeckenden Mengen – *Lösungen* – dieser Größe. Ausgehen von einem Lösungshistogramm für eine MSO-Formel und einer logischen Struktur kann man entscheiden, ob es eine Lösung gibt, die Anzahl der Lösungen durch Aufaddieren bestimmen und die Größe der kleinsten oder größten Lösung ablesen.

Ein großer Vorteil der MSO-Logik ist ihre große Ausdrucksstärke und die einfache Art, mit der man Berechnungsprobleme beschreiben kann. Ein großer Nachteil ist, dass MSO-*beschreibbare Probleme* im algorithmischen Sinn schwer lösbar sein können: Es gibt NP-schwere MSO-beschreibbare Entscheidungsprobleme (äquivalent zum Entscheidungsproblem der Aussagenlogik) und #P-schwere MSO-beschreibbare Zählprobleme (äquivalent zum Zählproblem der Aussagenlogik). Wenn man Problembeschreibungen selbst als Eingabe verarbeiten möchte, dann ist dies sogar schwer für PSPACE (die Klasse aller auf polynomiellem Platz lösbaren Probleme). Um der algorithmischen Schwere zu entgehen, aber trotzdem die große Ausdrucksstärke der MSO-Logik zu haben, werden beim Satz von Courcelle nur Eingabestrukturen betrachtet, die eine beschränkte Baumweite haben.

Die *Baumweite* ist ein Maß für die Ähnlichkeit eines Graphen (allgemein einer logischen Struktur) zu Bäumen. Um die Baumweite zu messen, wird ein Graph als eine *Baumzerlegung* kodiert. Dies ist ein Baum, dessen Knoten so mit Teilgraphen markiert sind, dass Abdeckungs- und Zusammenhangsbedingungen erfüllt sind: Die *Abdeckungsbedingung* sagt, dass jede Kante des Graphen in einem Teilgraphen vorkommen muss. Die *Zusammenhangsbedingung* sagt, dass die Teilgraphen, in denen ein Knoten des Graphen vorkommt, nicht beliebig verstreut sein dürfen, sondern im Baum zusammenhängend sind. Mithilfe von Baumzerlegungen kann man die Methode der dynamischen Programmierung anwenden, um zum Beispiel 3-FÄRBBARKEIT zu lösen: Ausgehend von den Blättern einer Baumzerlegungen berechnet man für jeden Teilgraphen eine Menge von Färbungen, die sich zu Färbungen für den schon betrachteten darunterliegenden Graph erweitern lassen. Wenn die Knoten der zur Verfügung stehenden Baumzerlegung nur durch Teilgraphen konstanter Größe markiert sind, ergibt sich ein effizienter Lösungsansatz. Formal spricht man von einer Menge von Graphen mit *beschränkter Baumweite*, falls es eine natürliche Zahl w gibt, so dass man für jeden Graph aus der Menge eine Baumzerlegung finden kann, die nur Teilgraphen mit maximal w vielen Knoten enthält. Der Satz von Courcelle verallgemeinert den obigen Lösungsansatz für 3-FÄRBBARKEIT so, dass er auf jedes MSO-beschreibbare Probleme anwendbar ist. Im zugehörigen Beweis wird zuerst eine Baumzerlegung konstruiert und dann die MSO-Formel entlang der Zerlegung ausgewertet.

Wenn man dem Satz von Courcelle und seinem Beweis zum ersten Mal begegnet, dann scheint er nur eingeschränkt einsetzbar zu sein: Klassen von Graphen mit beschränkter Baumweite enthalten zum Beispiel nicht alle planaren Graphen oder alle vollständigen Graphen. Des weiteren führt der Beweis von Courcelle zwar auf Algorithmen mit linearer Laufzeit, in deren „O-Notationen“ verstecken sich aber große Konstanten und sie sind somit nicht direkt praktisch anwendbar. Nichtsdestotrotz spielt Courcelles Satz eine entschei-

dende Rolle bei der Entwicklung neuer effizienter Algorithmen. Bevor man viel Arbeit in einen wirklich praktikablen Algorithmus für ein Problem investiert, kann man Courcelles Satz verwenden, um erst einmal zu wissen, ob effiziente Algorithmen möglich sind. Des weiteren hat sich herausgestellt, dass es viele Berechnungsprobleme gibt, deren Eingaben zwar keine beschränkte Baumweite haben oder die nicht MSO-beschreibbar sind, bei denen Courcelles Satz aber als Teil einer größeren Lösungsstrategie verwendet werden kann.

In der Dissertation wird untersucht, in wie weit man die Vorteile von MSO-beschreibbaren Problemen auf Eingaben mit beschränkter Baumweite für die Untersuchung der Platz- und Schaltkreiskomplexität verwenden kann. Durch Beschränkung von Speicherplatz oder der Größe und Tiefe von Schaltkreisen werden die verwendeten algorithmischen Ansätze oft unverhältnismäßig komplex. Die Dissertation untersucht, in wie weit man MSO-basierte Problembeschreibungen und Baumzerlegungen verwenden kann, um algorithmische Ansätze in der Platz- und Schaltkreiskomplexität gleichzeitig zu verallgemeinern und einfacher zugänglich zu machen. Hierbei spielt auch die Beschreibungskomplexität von Problemen eine Rolle, denn durch einfachere Problembeschreibungen erhält man in der Regel auch effizientere Algorithmen. Konkret schlagen sich die Resultate nieder als eine Reihe mathematischer *Sätze* mit Beispielen für deren *Anwendungen* sowie der Erweiterung bestehender und Entwicklung neuer *Beweismethoden*.

Die Beiträge der Arbeit werden in den folgenden Abschnitten 2 bis 5 nach Art der verwendeten Berechnungsmodelle sortiert dargestellt. Im Abschnitt 6 werden die Beiträge nebeneinandergestellt und darauf aufbauend zukünftige Forschungsrichtungen diskutiert.

2 Beschränkte Baumtiefe und Logik erster Stufe

Der erste Teil der Dissertation beschäftigt sich mit der Beschreibungskomplexität von MSO-beschreibbaren Problemen und untersucht die folgende Frage: „Wann ist es möglich, die algorithmisch einfacher zu handhabende Prädikatenlogik erster Stufe (FO-Logik) anstatt der monadischen Logik zweiter Stufe zur Beschreibung von Problemen zu verwenden?“ Das heißt, unter welchen Umständen kann man MSO-Formeln in äquivalente FO-Formeln umwandeln? Es ist bekannt, dass dies nicht möglich ist, wenn man beliebige Graphen oder auch nur Pfad-Graphen betrachtet. Man kann zum Beispiel mit einer MSO-Formel beschreiben, dass ein Pfad eine gerade Anzahl von Knoten enthält. Dies ist mit einer FO-Formel, die nur einzelne Elemente mit ihren Quantoren binden kann, nicht möglich. Der allgemeine Grund hierfür ist, dass FO-Formeln nur Eigenschaften ausdrücken können, die sich auf lokale Umgebungen von Knoten in einem Graph beziehen; wie zum Beispiel die Eigenschaft, dass ein Graph einen Pfad einer konstanten Länge als Teilgraph enthält oder jeder Knoten einen Nachbar besitzt. Dies wirft die Frage auf, was bei Graphen passiert, in denen es keine langen Pfade gibt. Interessanterweise sind dies genau die Graphen, für die es Baumzerlegungen gibt, die gleichzeitig eine beschränkte Weite haben und deren zugrundeliegender Baum eine beschränkte Tiefe besitzt; jede Klasse solcher Graphen hat eine *beschränkte Baumtiefe*. Auf diesen Graphen haben MSO- und FO-Logik dieselbe Ausdrucksstärke:

Satz 2.1. *Sei $\mathcal{C}$ eine Klasse von Graphen mit beschränkter Baumtiefe. Für jede* MSO*-Formel φ gibt es eine* FO*-Formel ψ, so dass jeder Graph $G \in \mathcal{C}$ die Formel φ genau dann erfüllt, wenn dies auch für die Formel ψ gilt.*

Anwendung fand dieser Satz in einer Arbeit [EGT12], die auch untersucht, wann sich die Ausdrucksstärken von MSO- und FO-Logik unterscheiden. Unter bestimmten Abschlussbedingungen an die Klasse $\mathcal{C}$ ist dies der Fall, wenn $\mathcal{C}$ keine beschränkte Baumweite hat.

Für den Beweis des obigen Satzes könnte man versuchen dem typischen Beweisschema von Courcelles Satz zu folgen. Das heißt, erst eine Baumzerlegung beschränkter Weite und Tiefe mit FO-Formeln zu definieren und dann zu verwenden, um die MSO-Formel entlang der Baumzerlegung auszuwerten. Leider schlägt dieser Ansatz fehl. Zwar kann man in FO-Logik verifizieren, ob eine gegebene Baumzerlegung zum Graphen passt, man kann aber nicht immer eindeutig eine solche Zerlegung definieren; die FO-Formel kann nicht unbedingt zwischen verschiedenen Zerlegungen unterscheiden und einfach die (lexikographisch) kleinste herauspicken. In der Arbeit wird daher folgender neuer Ansatz entwickelt: Zuerst wird eine Formel ρ verwendet, die der rekursiven Definition von Graphen mit beschränkter Baumtiefe entspricht [NO12]. Diese wird dann zu einer Formel ψ erweitert, die die ursprüngliche MSO-Formel auswertet, in dem sie induktiv Mengen von MSO-Formeln (MSO-*Typen*) beschreibt, die von Teilstrukturen der gesamten Struktur erfüllt sind. Am Ende stellt sie fest, ob die gesuchte MSO-Formel φ im MSO-Typ des gesamten Graphen enthalten ist. Den Induktionsschritt (wie man die MSO-Typen von Teilgraphen zusammenfügt um den MSO-Typ des gesamten Graphen zu bekommen) kann man auf verschiedene Arten beweisen: In [EGT12] wurde hierzu ein nicht-konstruktiver auf Ehrenfeucht–Fraïssé-Spielen basierender Ansatz verwendet. In der Dissertation wurde ein konstruktiver Ansatz aufbauend auf einem neu entwickelten Automaten-Modell, dem *Multimengen-Baumautomaten*, verwendet. Multimengen-Baumautomaten finden auch im folgenden Abschnitt Verwendung.

3 Beschränkte Baumtiefe und Schaltkreise konstanter Tiefe

Durch die geringe Beschreibungskomplexität von MSO-beschreibbaren Entscheidungsproblemen auf Graphen mit beschränkter Baumtiefe ergibt sich aus bekannten Resultaten der Komplexitätstheorie, dass diese auch durch Boolesche Schaltkreise konstanter Tiefe gelöst werden können (einen Überblick hierzu findet sich im Buch von Immerman [Imm99]). Die betrachteten Probleme liegen daher in der Komplexitätsklasse AC^0, wobei man den Aufbau der verwendeten Schaltkreise in DLOGTIME verifizieren kann – sie sind DLOGTIME-*uniform*. Dies gilt für jede in der Dissertation betrachtete Familie von Schaltkreisen. (Uniforme Schaltkreise werden im Buch von Vollmer [Vol99] behandelt.)

Korollar 3.1. *Sei $\mathcal{C}$ eine Klasse von Graphen mit beschränkter Baumtiefe. Für jede* MSO*-Formel φ gibt es* DLOGTIME*-uniforme* AC^0*-Schaltkreise, die für Graphen aus $\mathcal{C}$ das durch φ beschriebene Entscheidungsproblem lösen.*

Analog zum Korollar, mit dem man Entscheidungsprobleme durch Booleschen Schaltkreise lösen kann, wird in der Arbeit ein Satz entwickelt, mit dem man Zählprobleme durch arithmetische Schaltkreise lösen kann. Die zugehörigen Berechnungsprobleme liegen daher in der Komplexitätsklasse GapAC^0 (einen Überblick über GapAC^0 und verwandte Klassen gibt Allender [All04]). Der folgende Satz löst hierbei nicht nur Zählprobleme, sondern berechnet Lösungshistogramme:

Satz 3.2. *Sei $\mathcal{C}$ eine Klasse von Graphen mit beschränkter Baumtiefe. Für jede* MSO*-Formel $\varphi(X)$ gibt es* DLOGTIME*-uniforme* GapAC^0*-Schaltkreise, die für Graphen aus $\mathcal{C}$ das durch $\varphi(X)$ beschriebene Histogramm (als Zahl kodiert) berechnen.*

Korollar und Satz finden Anwendung bei der Bestimmung oberer komplexitätstheoretischer Schranken für Berechnungsprobleme, die (1) MSO-beschreibbar sind und deren Eingaben man auf Graphen mit beschränkter Baumtiefe einschränkt oder (2) sich (vollständig) in MSO-beschreibbare Probleme über Graphen mit einer konstanten Baumtiefe umwandeln lassen. Ein prominentes Beispiel des ersten Typs ist die Berechnung der Anzahl von Paarungen eines Graphen. Dieses Problem ist im allgemeinen schwer für $\#\mathrm{P}$, aber man kann es bei beschränkter Baumtiefe in $\#\mathrm{AC}^0$ lösen. Ein Beispiel des zweiten Typs ist SUBSET-SUM mit unär kodierten Gewichten. Dieses Problem ist in pseudopolynomieller Zeit lösbar und mit einer Graph-basierten Darstellung des zugehörigen dynamischen Programmierungsansatzes kann man zeigen, dass es in NL (nichtdeterministischem logarithmischem Platz) liegt. Da es sich auf die Berechnung von MSO-beschreibbaren Lösungshistogrammen über Graphen mit beschränkter Baumweite reduzieren lässt, kann man durch Anwendung von Resultaten aus der Dissertation zeigen, dass es in TC^0 liegt.

Zum Beweis von Satz 3.2 wird die initiale Idee aus dem vorherigen Abschnitt zur Konstruktion von Baumzerlegungen wieder aufgegriffen: Es wird zuerst eine Baumzerlegung beschränkter Weite und Tiefe konstruiert. Interessanterweise geschieht dies unter Zuhilfenahme der im vorherigen Abschnitt entwickelten Beschreibung von Strukturen mit beschränkter Baumtiefe durch FO-Formeln. Danach wird die MSO-Formel in einen äquivalenten Multimengen-Baumautomaten umgewandelt. Zuletzt wird gezeigt, wie man den Automaten durch arithmetische Schaltkreise simulieren kann. Hierzu werden bestehende Ansätze aus dem Bereich der Linearzeitalgorithmen aufgegriffen und stark erweitert, um auch mit Bäumen konstanter Tiefe und unbeschränktem Grad arbeiten zu können.

Nachdem uns die Frage nach dem Unterschied in der Ausdrucksstärke von MSO- und FO-Logik auf das Konzept der beschränkten Baumtiefe und Schaltkreise konstanter Tiefe geführt hat, kommen wir in den nächsten beiden Abschnitten wieder auf das am Anfang erwähnte Konzept der beschränkten Baumweite zurück.

4 Baumzerlegungen in Klammerdarstellung und Schaltkreise logarithmischer Tiefe

Im Beweis von Courcelles Satz wird zuerst eine Baumzerlegung für die in der Eingabe kodierte Struktur konstruiert und dann ein zur MSO-Formel äquivalenter Baumautomat ent-

lang der Zerlegung simuliert. In diesem Abschnitt werden zuerst die Ergebnisse der Arbeit besprochen, die sich auf den zweiten Schritt beziehen. Hierbei zeigt sich, dass schon die Auswertung der MSO-Formel entlang einer gegebenen (und möglichst zugänglich kodierten) Zerlegung eine breite Anwendung bei der Untersuchung von Schaltkreiskomplexitätsklassen besitzt. Die entwickelten Methoden helfen Resultate aus der Komplexitätstheorie zu verallgemeinern und gleichzeitig einfacher zu beweisen.

Es ist bekannt, dass die Komplexität von Berechnungsproblemen auf Bäumen stark von der Kodierung des Eingabebaums abhängt. Zum Beispiel ist es vollständig für L (deterministischer logarithmischer Platz), die transitive Hülle eines Baumes zu berechnen. Es wird aber einfacher (vollständig für TC^0), wenn man einen Baum wie auf der rechten Seite gezeigt als *Klammerausdruck* [[ ] [[ ] [ ]]] kodiert. Damit bei der Untersuchung der Komplexität von Problemen wie der Auswertung von Booleschen oder arithmetischen Ausdrücken die Kodierung der Eingabe nicht die Komplexität des eigentlichen Problems überdeckt, verwendet man meist den algorithmisch einfacheren Klammerausdruck. Mit dieser Kodierung sind die obigen Probleme in logarithmischer Tiefe lösbar, wobei die verwendeten Gatter im Fall von Booleschen Ausdrücken ebenfalls Boolesch und im Fall von arithmetischen Ausdrücken ebenfalls arithmetisch sind [BCGR92]. In beiden Fällen haben die Gatter nur beschränkt viele Eingaben; dies entspricht jeweils den Komplexitätsklassen NC^1 und $\#\mathrm{NC}^1$. Die folgenden Sätze verallgemeinern die obigen Resultate:

Satz 4.1. *Sei $\mathcal{C}$ eine Klasse von Graphen mit beschränkter Baumweite. Für jede* MSO*-Formel φ gibt es* DLOGTIME*-uniforme* NC^1*-Schaltkreise, die für Graphen aus $\mathcal{C}$, die zusammen mit einer Baumzerlegung in Klammerdarstellung gegeben sind, das durch φ beschriebene Entscheidungsproblem lösen.*

Satz 4.2. *Sei $\mathcal{C}$ eine Klasse von Graphen mit beschränkter Baumweite. Für jede* MSO*-Formel $\varphi(X)$ gibt es* DLOGTIME*-uniforme* $\#\mathrm{NC}^1$ *Schaltkreise, die für Graphen aus $\mathcal{C}$, die zusammen mit einer Baumzerlegung in Klammerdarstellung gegeben sind, das durch $\varphi(X)$ beschriebene Histogramm (als Zahl kodiert) berechnen.*

Neben der Anwendung der obigen Sätze auf das Lösen von Booleschen oder arithmetischen Ausdrücken, ist ein weiteres Beispiel das Lösen von durch Automaten definierter Probleme, deren Eingaben XML-Dateien sind. Die Anwendung der obigen Sätze ist möglich, da XML-Dateien baumartige Strukturen durch (angereicherte) Klammerdarstellungen kodieren.

Zum Beweis des Satzes zu Entscheidungsproblemen kann zum Beispiel die MSO-Formel in einen äquivalenten Baumautomaten umwandeln und dann bestehende Ergebnisse zur Simulation von Baumautomaten verwenden. Diese Ergebnisse basieren auf der Beweistechnik von Buss et al. [BCGR92] mit der Boolesche Ausdrücke beliebiger Tiefe durch Schaltkreise logarithmischer Tiefe ausgewertet werden können. Kern dieser Technik ist es, rekursiv Teile aus einem bestehenden Ausdruck auszukoppeln, diese auszuwerten und die Ergebnisse so zu kombinieren, dass sich ein Ergebnis für den gesamten Ausdruck ergibt. Wenn man die ausgekoppelten Teile groß genug wählt, ergibt sich eine logarithmisch tiefe Rekursion, die sich mit DLOGTIME-uniformen NC^1-Schaltkreisen implementieren lässt. Um auch den Satz zur Konstruktion von Histogrammen zu beweisen, wird in der Disserta-

tion eine alternative Beweistechnik entwickelt: Anstatt einen Problem-basierten Ansatz zu verwenden, der *gleichzeitig* balanciert und auswertet, werden diese Schritte *getrennt*. Zuerst wird die gegebene Baumzerlegung balanciert. Dies geschieht durch Anwendung von Methoden zur Baumkontraktion. Danach wird ein zur MSO-Formel äquivalenter Baumautomat auf der balancierten (und damit logarithmisch tiefen) Baumzerlegung simuliert.

Was passiert, wenn man sich nicht in der komfortablen Situation befindet, dass schon eine Baumzerlegung in der Eingabe enthalten ist, wird im nächsten Abschnitt besprochen.

5 Beschränkte Baumweite und logarithmischer Platz

In diesem Abschnitt betrachten wir MSO-beschreibbare Probleme auf Graphen mit beschränkter Baumweite. Da die folgenden zwei Sätze L-vollständige Probleme wie das Erreichbarkeitsproblem in Bäumen abdecken, beantworten sie die Frage nach der Komplexität dieser MSO-beschreibbaren Probleme:

Satz 5.1. *Sei $\mathcal{C}$ eine Klasse von Graphen mit beschränkter Baumweite. Für jede* MSO*-Formel φ ist das Entscheidungsproblem für Graphen aus $\mathcal{C}$ in* L *berechenbar.*

Satz 5.2. *Sei $\mathcal{C}$ eine Klasse von Graphen mit beschränkter Baumweite. Für jede* MSO*-Formel $\varphi(X)$ ist das Histogramm für Graphen aus $\mathcal{C}$ in* L *berechenbar.*

Die obigen Sätze werden in der Dissertation auf zwei Arten *angewendet*. Zum einen kann man mit ihnen zeigen, dass MSO-beschreibbare Probleme auf Graphen mit beschränkter Baumweite L-berechenbar sind. Dies umfasst zum Beispiel das Erreichbarkeitsproblem oder die Frage, ob ein gegebener Graph eine Paarung seiner Knoten besitzt. Verwendet man die Sätze als Teil einer größeren Lösungsstrategie, kann man auch Probleme auf Graphen (potentiell) unbeschränkter Baumweite lösen. Dies ist zum Beispiel die Frage, ob ein gegebener Graph einen Kreis gerader Länge enthält: Bei hoher Baumweite ist ein solcher Kreis immer vorhanden. Falls die Baumweite beschränkt ist, verwendet man die Sätze um herauszufinden, ob es einen (MSO-beschreibbaren) Pfad gerader Länge gibt.

Um die Sätze zu beweisen, folgt die Dissertation dem gängigen Beweisschema, bei dem man zuerst eine Zerlegung berechnet und dann, ausgehend von dieser Zerlegung, das betrachtete MSO-beschreibbare Problem löst. Die Resultate aus dem vorherigen Abschnitt zur Lösung des zweiten Schrittes kann man in diesem Abschnitt wiederverwenden, da jede $\#\mathrm{NC}^1$-berechenbare Funktion auch L-berechenbar ist (einen Überblick und Details hierzu gibt Allender [All04]). Der noch verbleibende Schritt zum Beweis der obigen Sätze ist durch den folgenden Satz gegeben:

Satz 5.3. *Sei $\mathcal{C}$ eine Klasse von Graphen mit beschränkter Baumweite. Man kann in* L *für jeden Graphen aus $\mathcal{C}$ eine Baumzerlegung mit beschränkter Weite (in Klammerdarstellung) berechnen.*

Um Baumzerlegungen zu konstruieren wird in der Arbeit das Konzept der Deskriptorzerlegung entwickelt. Mithilfe des Satzes von Reingold [Rei08] werden zuerst Deskriptorzer-

legungen konstruiert. Danach werden Teilmengenbedingungen der Deskriptorzerlegung genutzt, um eine Baumzerlegung aus der Deskriptorzerlegung zu gewinnen.

6 Fazit

Das Ziel der Dissertation war es, die Verwendung von MSO-Logik für die Untersuchung der Platz- und Schaltkreiskomplexität von Entscheidungs-, Optimierungs-, und Zählproblemen zu ermöglichen. Dies wurde durch eine Reihe mathematischer Sätze erreicht, die jeweils eine Aussage über die Komplexität MSO-beschreibbarer Probleme treffen, deren Eingaben baumartige Graphen sind. Abbildung 1 zeigt dies zusammenfassend. Die Ergebnisse verallgemeinern bestehende Resultate, wie zum Beispiel das Auswerten von arithmetischen Ausdrücken in $\#\mathrm{NC}^1$, und ermöglichen es auch neue Aussagen über die Komplexität von Problemen zu treffen, wie zum Beispiel das Finden von gerichteten Pfaden in Graphen mit beschränkter Baumweite in L.

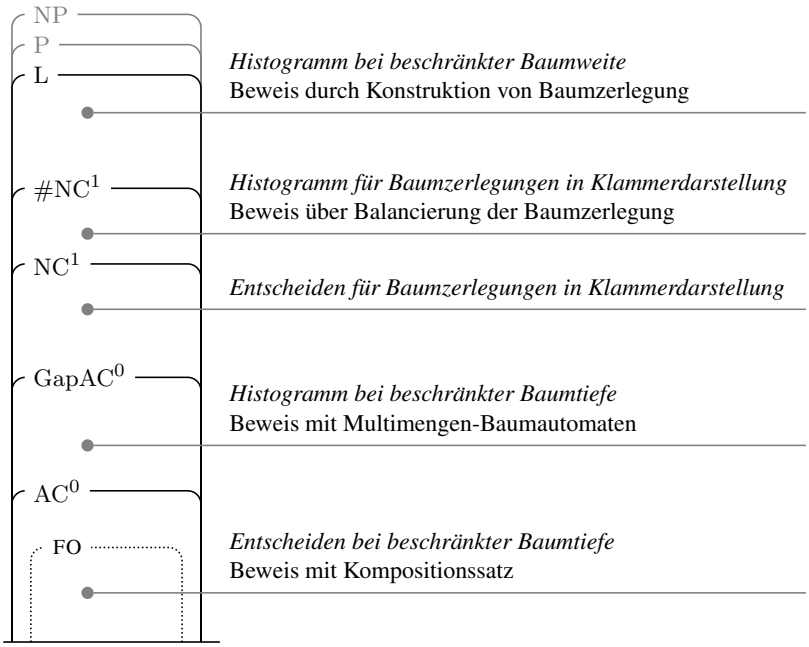

Abbildung 1: Die in der Dissertation entwickelten Klassifizierungen von MSO-beschreibbaren Problemen. Die Probleme unterscheiden sich in der Art ihrer Lösung – Entscheidungs- und Histogrammproblem – und in der Art der betrachteten Eingabe – Graphen mit beschränkter Baumtiefe, Graphen mit gegebener Baumzerlegung in Klammerdarstellung und Graphen mit beschränkter Baumweite.

Es bleiben eine Reihe offener Fragen zurück: Aufbauend auf jedem Satz kann man Fragen, ob der Satz immer noch gilt, wenn man anstatt MSO-Logik eine Logik mit größerer

Ausdrucksstärke verwenden. Auch wäre es interessant zu wissen, ob ein Satz noch gilt, wenn man größere Mengen von Graphen als Eingabe betrachtet. Allgemein stellt sich die Frage, ob es ähnliche Sätze (basierend auf einer Logik, graphentheoretischen Konzepten und vielen Anwendungen) für andere Komplexitätsklassen gibt.

Literatur

[All04] Eric Allender. Arithmetic Circuits and Counting Complexity Classes. In Jan Krajíček, Hrsg., *Complexity of Computations and Proofs*, Jgg. 13 of *Quaderni di Matematica*, Seiten 33–72. Seconda Universita di Napoli, 2004.

[BCGR92] S. Buss, S. Cook, A. Gupta und V. Ramachandran. An optimal parallel algorithm for formula evaluation. *SIAM Journal on Computing*, 21(4):755–780, 1992.

[Cou90] Bruno Courcelle. Graph rewriting: An algebraic and logic approach. In Jan van Leeuwen, Hrsg., *Handbook of Theoretical Computer Science, Volume B: Formal Models and Semantics*, Seiten 193–242. Elsevier and MIT Press, 1990.

[EGT12] Michael Elberfeld, Martin Grohe und Till Tantau. Where First-Order and Monadic Second-Order Logic Coincide. In *Proceedings of the 27th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS 2012)*, Seiten 265–274. IEEE Computer Society, 2012.

[Elb12] Michael Elberfeld. *Space and Circuit Complexity of Monadic Second-Order Definable Problems on Tree-Decomposable Structures*. Universität zu Lübeck, 2012. Dissertation.

[Imm99] Neil Immerman. *Descriptive complexity*. Springer, 1999.

[NO12] Jaroslav Nešetřil und Patrice Ossona de Mendez. *Sparsity: Graphs, Structures, and Algorithms*. Springer, 2012.

[Rei08] Omer Reingold. Undirected connectivity in log-space. *Journal of the ACM*, 55(4):1–24, 2008.

[Vol99] Heribert Vollmer. *Introduction to Circuit Complexity: A Uniform Approach*. Springer, 1999.

Michael Elberfeld absolvierte von 2002 bis 2007 ein Studium der Informatik an der Universität zu Lübeck, das er im September 2007 als Diplom-Informatiker abschloss. Nach dem Studium war er vom Oktober 2007 bis September 2012 als wissenschaftlicher Mitarbeiter am Institut für Theoretische Informatik der Universität zu Lübeck tätig. Seine Forschung konzentrierte sich Anfangs auf das Design und die Anwendung von Algorithmen in der Bioinformatik. Im Sommer 2009 kam die Forschung im Bereich der Komplexitäts- und Modelltheorie hinzu, die in die hier besprochene Dissertation mündete. Seit September 2012 wird er durch ein Postdoktoranden-Stipendium der DAAD gefördert und arbeitet an platzeffizienten Algorithmen zur Lösung von Pfadproblemen am ICSI in Berkeley und NII in Tokio.