Obere und untere Schranken für eingeschränkte Parity-Branchingprogramme

Dissertation
zur Erlangung des Doktorgrades
der Mathematisch-Naturwissenschaftlichen Fakultäten
der Georg-August-Universität zu Göttingen

vorgelegt von
Henrik Brosenne
aus
Göttingen

Göttingen 2006

D 7

Referent:	Prof. Dr. Stephan Waack
Korreferent:	Prof. Dr. Carsten Damm
Tag der mündlichen Prüfung:	18.04.2006

Danksagung

An dieser Stelle möchte ich mich bei allen bedanken, die mich während meines Promotionsstudiums unterstützt haben, insbesondere bei denen, die nachfolgend nicht aufgeführt sind.
Mein größter Dank gilt meinem Doktorvater *Prof. Dr. Stephan Waack*, der mich während meiner Promotionszeit umfassend betreut hat. Schon während meines Diplomstudiums hat er mich in das Gebiet der Branchingprogramme eingearbeitet. Der Großteil der Ergebnisse dieser Arbeit ist erst durch die von ihm vorgeschlagenen Denkansätze zustandegekommen.
Maßgeblichen Anteil am Gelingen dieser Arbeit hat *Prof. Dr. Robert M. Switzer*. Zum einen, durch meine Beschäftigung als sein Assistent, zum anderen, beruht ein großer Teil meiner Kenntnisse der praktischen Informatik auf seiner Ausbildung und der nachfolgenden sehr ergiebigen Zusammenarbeit.
Mein Dank gilt auch *Prof. Dr. Carsten Damm*, der immer bereit war sein umfassendes Wissen der Theoretischen Informatik mit mir zu teilen. Er nahm sich oft die Zeit, um in ausführlichen Gesprächen viele Anregungen zu geben.
Auch meinem Mitstreiter *Dr. Matthias Homeister* möchte ich danken. Seine Sichtweise der Probleme war für mich immer sehr inspirierend.
Weiterhin bin ich der Deutschen Forschungsgemeinschaft zu Dank verpflichtet. Das Forschungsprojekt „Datenstrukturen für Boolesche Funktionen: Komplexität und algebraische Struktur ihrer Darstellung und Algorithmen zu ihrer Handhabung“ ermöglichte den Beginn meiner Promotion. Das Ende meines Promotion wurde durch eine Überbrückungsfinanzierung des Instituts für Numerische und Angewandte Mathematik der Universität Göttingen unterstützt, dafür möchte ich mich beim Institutsvorstand bedanken.
Ein besondere Dank gilt meiner Familie, insbesondere meinen Eltern, für ihre in jeder Beziehung vorbildliche Unterstützung.
Außerordentlichen Dank verdient meine Frau *Mareike*. Sie hat mich während der Promotionszeit ertragen müssen und mich trotzdem immer ermutigt.

Inhaltsverzeichnis

1 Einleitung

1.1 Obere und untere Schranken

Zu den wesentlichen Forschungszielen der Komplexitätstheorie als Teilgebiet der Theoretischen Informatik gehört die Klassifizierung von Problemen entsprechend ihrer Komplexität, das heißt, dem zu ihrer Lösung notwendigen Aufwand. Die Komplexität eines Problems wird immer bezüglich eines Berechnungsmodells angegeben.
Eine bedeutende Klasse von Problemen sind die Entscheidungsprobleme. Entscheidungsprobleme haben zu einer gegebenen Eingabe als Lösung nur zwei mögliche Antworten, *akzeptieren* (bzw. ja oder 1) oder *verwerfen* (bzw. nein oder 0). Jedes derartige Problem lässt sich auch als das Wortproblem einer formalen Sprache auffassen. Eine Sprache wird akzeptiert, wenn genau die Worte, die zu der Sprache gehören, akzeptiert werden.
Die Turingmaschine ist ein Berechnungsmodell, das gut geeignet ist zum Akzeptieren von Sprachen. Die grundlegenden Komplexitätsklassen fassen die Sprachen zusammen, die von einer Turingmaschine, mit in der Eingabenlänge beschränkten Ressourcen, akzeptiert werden können. Die betrachteten Ressourcen sind in der Regel Rechenzeit und Speicherplatzbedarf.
Es ist im Allgemeinen nicht möglich und auch nicht sinnvoll, den genauen Verbrauch einer Ressource anzugeben. Vielmehr wird der Ressourcenverbrauch im schlechtesten Fall betrachtet und für diesen wird eine asymptotische Schranke, also eine Schranke für immer größer werdende, potentiell unendlich große Eingabenlängen, angegeben. Dazu werden üblicherweise die Landau-Symbole $\mathcal{O}$ und Ω verwendet. Seien $f, g : \mathbb{N} \to \mathbb{N}$ zwei Abbildungen über den natürlichen Zahlen. Die Abbildung g ist eine asymptotische obere (untere) Schranke für f, man schreibt $f = \mathcal{O}(g)$ ($f = \Omega(g)$), wenn $g(n)$ für fast alle $n \in \mathbb{N}$ um einen konstanten positiven Faktor größer (kleiner) ist als $f(n)$.

$$f = \mathcal{O}(g) \iff \exists\, c > 0, n_0 \in \mathbb{N} \text{ sodass } \forall\, n > n_0 \text{ gilt } f(n) \leq c \cdot g(n)$$
$$f = \Omega(g) \iff \exists\, c > 0, n_0 \in \mathbb{N} \text{ sodass } \forall\, n > n_0 \text{ gilt } f(n) \geq c \cdot g(n)$$

Für verschiedene asymptotische Schranken werden konkrete Bezeichnungen verwendet, die allerdings nicht in allen Aufsätzen dieselbe Bedeutung haben. In Abbildung 1.1 wird aufgelistet, welche Bezeichnungen in dieser Arbeit benutzt werden und welche Bedeutung sie, abhängig von der Eingabenlänge n, haben.

$\mathcal{O}(1)$	*konstant*
$\mathcal{O}(n)$	*linear*
$n^{\mathcal{O}(1)}$	*polynomiell*
$\mathcal{O}(\log_2 n)$	*logarithmisch*
$2^{(\log_2 n)^{\mathcal{O}(1)}}$	*quasipolynomiell*
$2^{n^{1/\mathcal{O}(1)}}$	*superpolynomiell*
$2^{\mathcal{O}(n)}$	*exponentiell*

Abbildung 1.1: Konkrete Bezeichnungen für asymptotische Schranken.

Die grundlegenden Komplexitätsklassen sind P und L. Die Klasse P umfasst alle von einer Turingmaschine in polynomieller Zeit akzeptierbaren Sprachen. L ist die Klasse der Probleme, die von einer Turingmaschine bei logarithmischem Speicherplatzbedarf akzeptiert werden können, wobei der für Eingabe und Ausgabe benötigte Speicherplatz nicht mitgerechnet wird.

Ein wichtiges Konzept der theoretischen Informatik ist der Nichtdeterminismus. Eine (deterministische) Turingmaschine legt für eine Konfiguration genau eine nachfolgende Konfiguration fest. Eine Eingabe wird akzeptiert, wenn der eindeutige Berechnungsweg erfolgreich ist. Bei einer nichtdeterministischen Turingmaschine kann es für eine Konfiguration eine Menge von möglichen nachfolgenden Konfigurationen geben, aus der eine ausgewählt wird. Es gibt verschiedene Akzeptierungsmodi für nichtdeterministische Turingmaschinen. Beim existenziellen Akzeptierungsmodus wird eine Eingabe akzeptiert, wenn mindestens ein erfolgreicher Berechnungsweg möglich ist. Das Gegenteil des existenziellen ist der universelle Akzeptierungsmodus, der eine Eingabe verwirft, wenn mindestens ein nicht erfolgreicher Berechnungsweg möglich ist. Das heißt, der universelle Akzeptierungsmodus akzeptiert eine Eingabe genau dann, wenn alle möglichen Berechnungswege erfolgreich sind. Hingegen wird beim Parity-Akzeptierungsmodus ($\oplus$-Akzeptierungsmodus) eine Eingabe akzeptiert, wenn die Anzahl aller möglichen erfolgreichen Berechnungswege ungerade ist.

Die Komplexitätsklassen NP, co-NP, $\oplus$P und NL, co-NL, $\oplus$L sind die nichtdeterministischen Gegenstücke zu P und L. Diese Komplexitätsklassen fassen die Sprachen zusammen, die von nichtdeterministischen Turingmaschinen mit polynomieller Rechenzeit und existenziellen (NP), universellem (co-NP) oder $\oplus$-Akzeptierungsmodus ($\oplus$P) bzw. logarithmischem Speicherplatzbedarf und entsprechendem Akzeptierungsmodus akzeptiert werden.

Die klassische Turingmaschine ist ein *uniformes* Berechnungsmodell, das Eingaben beliebiger Länge bearbeiten kann. Viele wichtige Berechnungsmodelle sind *nichtuniform*, das heißt, die Eingabenlänge ist fest vorgegeben. Insbesondere hardwarenahe Berechnungsmodelle, zum Beispiel Schaltkreise und

Branchingprogramme, berechnen Boolesche Funktionen $f : \{0,1\}^n \to \{0,1\}$. Die Menge der Booleschen Funktionen in n Variablen wird mit Bool_n bezeichnet. Die formale Definition von Branchingprogrammen erfolgt in Abschnitt 1.2.

Zu den grundlegenden (uniformen) Komplexitätsklassen gibt es nichtuniforme Entsprechungen. Eine nichtuniforme Turingmaschine erhält neben der Eingabe noch einen Hinweis (advice), der nur von der Länge der Eingabe abhängt. Die Komplexitätsklassen P/poly (NP/poly, co-NP/poly, $\oplus$P/poly) enthalten alle Booleschen Funktionen, die von einer nichtuniformen (nichtdeterministischen) Turingmaschine (und entsprechendem Akzeptierungsmodus) mit einem Hinweis polynomieller Länge in polynomieller Zeit berechnet werden können. L/poly (NL/poly, co-NL/poly, $\oplus$L/poly) sind, bei logarithmischem Speicherplatzbedarf, entsprechend definiert.

Mittlerweile gibt es eine kaum überschaubare Vielfalt von Komplexitätsklassen [AK05]. Jedes Berechnungsmodell definiert mit einem festgelegten Ressourcenverbrauch eine eigene Komplexitätsklasse.

Obere und untere Schranken für den Ressourcenverbrauch bezüglich eines Berechnungsmodells, ordnen eine Boolesche Funktion in die entsprechende Komplexitätsklasse ein. Als Beispiel wird die Klasse P-BP, der Funktionen mit polynomiell großen Branchingprogrammen, betrachtet. Sei $f = (f_n)_{n>0}$ mit $f_n : \{0,1\}^n \to \{0,1\}$ eine Familie von Booleschen Funktionen. Es gilt $f \in$ P-BP, wenn es für jedes n ein deterministisches Branchingprogramm polynomiell beschränkter Größe gibt, das die Funktion f_n darstellt. Gibt es hingegen für alle n eine nicht polynomielle untere Schranke für die Größe jedes deterministischen Branchingprogramms, das f_n darstellt, gilt $f \notin$ P-BP.

Nichtdeterministische Branchingprogramme haben, genau wie nichtdeterministische Turingmaschinen, für eine Eingabe mehrere akzeptierende Berechnungswege, die mit verschiedenen Akzeptierungsmodi bewertet werden können. Die Komplexitätsklassen P-$\vee$BP, P-$\wedge$BP und P-$\oplus$BP umfassen alle Funktionen, die von polynomiell großen nichtdeterministischen Branchingprogrammen mit existenziellem, universellem und Parity-Akzeptierungsmodus dargestellt werden können.

Die Komplexitätsklassen nichtdeterministischer Branchingprogramme polynomieller Größe entsprechen denen nichtuniformer Turingmaschinen mit logarithmischem Speicherplatzbedarf. Meinel [Mei90] konnte zeigen,

$$\begin{aligned} \text{P-}\vee\text{BP} &= \text{NL/poly} \ , \\ \text{P-}\wedge\text{BP} &= \text{co-NL/poly} \ , \\ \text{P-}\oplus\text{BP} &= \oplus\text{L/poly} \ . \end{aligned}$$

Einige interessante obere Schranken für nichtdeterministische Branchingprogramme wurden schon bewiesen.

Szelepcsényi [Sze87] und Immerman [Imm88] zeigten NL/poly = co-NL/poly. Daraus folgt, P-∨BP = P-∧BP.
Wigderson [Wig94] beobachtete NL/poly ⊆ ⊕L/poly. Dieses Ergebnis wurde von Beimel und Gál [BG98] erweitert. Mit Hilfe von arithmetischen Branchingprogrammen wird ein eleganter Beweis für P-∨BP ⊆ P-⊕BP angegeben.
Allerdings ist der Beweis von unteren Schranken für die Größe von Branchingprogrammen für explizit definierte Funktionen und damit der Beweis von scharfen Relationen zwischen den Komplexitätsklassen ein grundlegendes, offenes Problem der Komplexitätstheorie. Deshalb werden eingeschränkte Branchingprogramme untersucht, für die teilweise schon starke Techniken zum Beweis von unteren Schranken entwickelt wurden.
Ein Branchingprogramm mit einmaligen Tests (BP1) liest auf jedem Pfad jede Variable höchstens einmal. Große untere Schranken für deterministische BP1s sind bekannt [Zák84, Weg88]. Weiterhin werden superpolynomielle untere Schranken für nichtdeterministische BP1s mit existenziellem oder universellem Akzeptierungsmodus von Jukna [Juk89] und Krause, Meinel und Waack [KMW91] bewiesen. Aber die verwendeten Methoden lassen sich nicht auf nichtdeterministische BP1s mit Parity-Akzeptierungsmodus übertragen. Der Beweis von unteren Schranken für dieses Modell ist ein interessantes, offenes Problem. Um sich der Lösung dieses Problems weiter zu nähern, werden Branchingprogramme mit stärkeren Einschränkungen betrachtet.
Ein Branchingprogramm ist *oblivious*, wenn auf jedem Pfad die Variablen, bis auf Auslassungen, in derselben Reihenfolge gelesen werden. In der Variablenordnung, die die Reihenfolge bestimmt, dürfen Variablen mehrfach vorkommen. Die Länge des oblivious Branchingprogramms ist die Länge der Variablenordnung. Krause [Kra92] beweist eine superpolynomielle untere Schranke für die Größe nichtdeterministischer oblivious Branchingprogramme linearer Länge mit existenziellem oder universellem Akzeptierungsmodus. Auch für dieses Modell ist mit Parity-Akzeptierungsmodus keine Technik zum Beweis von unteren Schranken bekannt.
Eine Verbindung der beiden Modelle sind oblivious Branchingprogramme mit einmaligen Tests, kurz OBDDs (ordered binary decision diagrams). In OBDDs werden auf allen Berechnungspfaden die Variablen immer in derselben Reihenfolge durchlaufen und auf einem beliebigen Berechnungspfad wird jede Variable höchstens einmal getestet. Exponentielle untere Schranken für deterministische OBDDs werden von Bryant [Bry86] gezeigt. Bryant [Bry91] gibt ebenfalls exponentielle untere Schranken für deterministische OBDDs an, die das mittlere Bit der Multiplikation über den natürlichen Zahlen berechnen. Die dabei benutzte Technik kann verwendet werden, um untere Schranken für die von Gergov und Meinel [GM96] eingeführten nichtdeterministischen OBDDs mit Parity-Akzeptierungsmodus (⊕OBDDs) zu beweisen.
Die von Waack [Waa01] vorgenommene algebraische Charakterisierung von ⊕OBDDs lieferte neue Einsichten und ermöglicht einen anderen Ansatz zum Beweis von

unteren Schranken für nichtdeterministische Branchingprogramme mit Parity-Akzeptierungsmodus. Dabei wird eine Besonderheit des Parity-Akzeptierungsmodus ausgenutzt. $(\{0,1\}, \oplus, \wedge)$ die Booleschen Konstanten mit den Booleschen Operatoren Exklusiv-Oder und Und als Addition und Multiplikation bilden einen Körper. Dieser Körper ist isomorph zu $\mathbb{F}_2$, dem Primkörper mit 2 Elementen. Somit kann $\mathbb{B}_n := (\mathrm{Bool}_n, \oplus, \wedge)$, die Menge der Booleschen Funktionen in n Variablen mit den Booleschen Operatoren Exklusiv-Oder und Und als Addition und Multiplikation mit Skalaren, als $\mathbb{F}_2$-Vektorraum aufgefasst werden. Weiterhin erlaubt die Variablenordnung eine Zerlegung der Knotenmenge des $\oplus$OBDDs in disjunkte Level. Für jeden Level wird ein Untervektorraum von $\mathbb{B}_n$ definiert. Dieser $\mathbb{F}_2$-Vektorraum wird von allen Unterfunktionen der darzustellenden Funktion auf den noch zu lesenden Variablen aufgespannt. Die Anzahl der Knoten eines Levels für eine gegebene Funktion ist mindestens die Dimension über $\mathbb{F}_2$ des Quotientenraums des Vektorraums dieses Levels und des Vektorraums des nächst kleineren Levels.
Für den existenziellen und den universellen Akzeptierungsmodus funktioniert dieser Ansatz nicht, denn $\mathbf{B}_0 := (\{0,1\}, \vee, \wedge)$, die Booleschen Konstanten mit den Operatoren Oder und Und als Addition und Multiplikation, ist nur ein kommutativer Halbring.
Üblicherweise wird nicht so strikt zwischen der Menge der Booleschen Funktionen in n Variablen Bool_n, der Booleschen Algebra $(\mathrm{Bool}_n, \neg, \vee, \wedge)$, dem Booleschen Halbring $\mathbf{B}_n \cong (\mathrm{Bool}_n, \vee, \wedge)$ und dem Booleschen Körper $\mathbb{F}_2 \cong (\mathrm{Bool}_n, \oplus, \wedge)$ unterschieden. In dieser Arbeit ist die Unterscheidung, aufgrund der Betrachtung verschiedener Akzeptierungsmodi, aber nötig.

Die Ideen für den Beweis von unteren Schranken für $\oplus$OBDDs können auf weniger stark eingeschränkte $\oplus$BP1s übertragen werden, wenn man $\oplus$OBDDs so verallgemeinert, dass die erforderlichen Eigenschaften erhalten bleiben. Eine solche Verallgemeinerung sind graphgesteuerte $\oplus$BP1s. Graphgesteuerte $\oplus$BP1s legen über eine Graphordnung für jede Eingabe eine Variablenordnung fest. Fordert man zudem noch Wohlstrukturiertheit, ist die Einteilung der Knotenmenge in disjunkte Level und eine Festlegung, welche Level benachbart sind, möglich. In [Bro00] wird eine algebraische Charakterisierung von wohlstrukturierten graphgesteuerten $\oplus$BP1s vorgenommen.
Das erste Hauptergebnis für eingeschränkte $\oplus$BP1s ist eine Technik zum Beweis von unteren Schranken für wohlstrukturierte graphgesteuerte $\oplus$BP1, die auf der algebraischen Charakterisierung beruht (Korollar 2.3). Dabei wird für eine gegebene Boolesche Funktion eine untere Schranke für die Größe eines wohlstrukturierten graphgesteuerten $\oplus$BP1s, das die Funktion darstellt, nur durch Invarianten der Funktion und der Graphsteuerung beschrieben.
Lässt man die Forderung nach Wohlstrukturiertheit für ein graphgesteuertes $\oplus$BP1 fallen, ist keine Einteilung in Level mehr möglich. Ein Kriterium für untere Schranken

von graphgesteuerten ⊕BP1s (Korollar 2.7) ist das zweite Hauptergebnis für eingeschränkte ⊕BP1s. Dieses Kriterium beruht auf der Betrachtung der Mengen von Unterfunktionen der dargestellten Funktion, die von verschiedenen Variablenmengen abhängen. Wiederum wird der Beweis mit der linearen Algebra und $\mathbb{F}_2$-Vektorräumen geführt.
Das dritte Hauptergebnis für eingeschränkte ⊕BP1s ist der Beweis von P-graphgesteuertes-⊕BP1 $\subsetneq$ P-⊕BP1 (Satz 2.12). Dazu wird eine Funktion angegeben, die mit ⊕BP1s polynomieller Größe darstellbar ist, aber nur mit graphgesteuerten ⊕BP1s superpolynomieller Größe dargestellt werden kann.
Ein Nebenergebnis für eingeschränkte ⊕BP1s ist die Beobachtung, dass die Eigenschaft, graphgesteuert zu sein, für ein nichtdeterministisches BP1 sehr natürlich ist (Satz 2.16).

Eine Möglichkeit obere Schranken etwas abzuschwächen, ist die Approximation einer gegebenen Funktion. Eine Funktion wird durch eine andere Funktion approximiert, wenn die beiden Funktionen, bis auf einen kleinen Anteil aller Eingaben, übereinstimmen.
Um verschiedene Modelle für die Darstellung von Booleschen Funktionen mit ⊕OBDDs zu approximieren, werden arithmetische OBDDs verwendet. Mit arithmetischen OBDDs kann über dem zugehörigen Halbring oder Körper gerechnet werden. Das erste Nebenergebnis zur Approximation mit ⊕OBDDs ist, dass sich arithmetische OBDDs über Körpern der Charakteristik 2 durch ⊕OBDDs simulieren lassen (Satz 3.2). Dazu wird ein Ergebnis von Beimel und Gál [BG98] an arithmetische OBDDs angepasst und durch die Ausnutzung des Frobenius-Automorphismus von endlichen Körpern verbessert (Lemma 3.1). Bei der Simulation geht der Logarithmus der Ordnung des Körpers als Exponent in die Größe des ⊕OBDDs mit ein. Das zweite Nebenergebnis zur Approximation mit ⊕OBDDs lautet wie folgt. Bei vorgegebenem Fehler kann jedes nichtdeterministische OBDD mit existenziellem oder universellem Akzeptierungsmodus durch ein arithmetisches OBDD über einem endlichen Körper approximiert werden (Satz 3.5), wenn der Logarithmus der Ordnung des Körpers größer als der Kehrwert des Fehlers ist.
Die Kombination dieser Ergebnisse, wobei der Körper und der umgekehrte Fehler dieselbe Größenordnung haben sollen, ergibt folgendes Hauptergebnis zur Approximation mit ⊕OBDDs. Nichtdeterministische OBDDs polynomieller Größe mit existenziellem oder universellem Akzeptierungsmodus können von ⊕OBDDs quasipolynomieller Größe mit umgekehrt quasipolynomiellem Fehler approximiert werden (Korollar 3.6).
Ein drittes Nebenergebnis zur Approximation mit ⊕OBDDs ist, dass die vorgestellten Methoden zur Simulation von arithmetischen OBDDs und Approximation von nichtdeterministischen OBDDs optimal sind.
Das zweite Hauptergebnis zur Approximation mit ⊕OBDDs, beschreibt die

Approximation von disjunktiven Formen, in deren Literale quasipolynomiell große $\oplus$OBDDs eingesetzt sind, durch $\oplus$OBDDs (Satz 3.10). Dabei wird eine Idee von Smolensky [Smo87] zur Approximation der Disjunktion verwendet.

Ein k-OBDD ist ein oblivious Branchingprogramm der Länge kn und die Variablenordnung, die die Reihenfolge bestimmt, in der die Variablen durchlaufen werden, ist eine k-fache Hintereinanderstellung derselben Anordnung der Variablenmenge. Also eine Erweiterung von OBDDs, in der die Variablen k-mal in derselben Reihenfolge durchlaufen werden können. Das erste Hauptergebnis für k-OBDDs ist, unabhängig vom Akzeptierungsmodus bringt der konstant oft wiederholte Test bezüglich einer Variablenordnung keinen Vorteil (Korollar 4.2). Das zweite Hauptergebnis für k-OBDDs ist die Erweiterung dieses Ergebnisses auf randomisierte k-OBDDs.

1.2 Branchingprogramme

1.2.1 Nichtdeterministische Branchingprogramme

Ein *Branchingprogramm* (*BP*) auf der *Variablenmenge* $\{X_1, \ldots, X_n\}$ hat als Grundgerüst einen gerichteten kreisfreien Graphen (V, E) mit Knotenmenge V und Kantenmenge E, in dem Mehrfachkanten erlaubt sind. Der Graph hat genau zwei graphtheoretische *Senken*, die jeweils mit 0 und 1 markiert sind und mit 0-Senke (t_0) und 1-Senke (t_1) bezeichnet werden. Alle anderen Knoten werden *Verzweigungsknoten* genannt. Jeder Verzweigungsknoten ist mit einer Variable aus der Variablenmenge $\{X_1, \ldots, X_n\}$ und jede Kante ist mit einem Wert aus $\{0, 1\}$ markiert. Ein mit $X_i \in \{X_1, \ldots, X_n\}$ markierter Knoten wird X_i-Knoten und eine mit $\delta \in \{0, 1\}$ markierte Kante wird δ-Kante genannt.
Ein BP ist *deterministisch*, wenn jeder Verzweigungsknoten genau eine ausgehende 0- und 1-Kante hat.
Sei $\mathcal{B}$ ein BP auf der Variablenmenge $\{X_1, \ldots, X_n\}$ und (V, E) der zugrunde liegende Graph. Jede Eingabe $\alpha = (\alpha_1, \ldots, \alpha_n) \in \{0, 1\}^n$ *aktiviert* alle ausgehenden α_i-Kanten jedes X_i-Knotens. Ein Pfad in (V, E) ist ein von α *aktivierter Pfad*, wenn alle Kanten des Pfades von α aktiviert werden. Für ein Knotenpaar v, w ist ein *Berechnungspfad* in $\mathcal{B}$ von v nach w unter $\alpha \in \{0, 1\}^n$ ein von α aktivierter Pfad von v nach w in (V, E). Ein Berechnungspfad in $\mathcal{B}$ wird *vollständig* genannt, wenn er in einer Senke endet. Für jedes Knotenpaar v, w und jede Eingabe $\alpha \in \{0, 1\}^n$ ist $\mathrm{comp}_{v,w}(\alpha)$ definiert als die Anzahl der Berechnungspfade von v nach w unter α.
Jedem Knoten v wird eine Boolesche Funktion Res_v, die so genannte Resultatsfunktion, zugeordnet, indem die Anzahl der vollständigen Berechnungspfade, die in einer Senke enden mit einem *Akzeptierungsmodus* interpretiert wird.

Der existenzielle oder $\vee$-Akzeptierungsmodus akzeptiert die Eingabe $\alpha \in \{0,1\}^n$, wenn mindestens ein vollständiger Berechnungspfad unter der Eingabe α in der 1-Senke endet.

$$\mathrm{Res}_v^{(\vee)}(\alpha) := 1 \Longleftrightarrow \mathrm{comp}_{v,t_1}(\alpha) > 0$$

Der universelle oder $\wedge$-Akzeptierungsmodus akzeptiert, wenn alle vollständigen Berechnungspfade in der 0-Senke enden.

$$\mathrm{Res}_v^{(\wedge)}(\alpha) := 1 \Longleftrightarrow \mathrm{comp}_{v,t_0}(\alpha) = 0$$

Und der Parity- oder $\oplus$-Akzeptierungsmodus akzeptiert, wenn die Anzahl der vollständigen Berechnungspfade, die in der 1-Senke enden, ungerade ist.

$$\mathrm{Res}_v^{(\oplus)}(\alpha) := 1 \Longleftrightarrow \mathrm{comp}_{v,t_1}(\alpha) \equiv 1 \pmod 2$$

Ein nichtdeterministisches BP mit festgelegtem Akzeptierungsmodus wird diesem entsprechend, jeweils existentielles oder $\vee$BP, universelles oder $\wedge$BP und Parity- oder $\oplus$BP genannt.
Es fällt auf, dass alle drei Akzeptierungsmodi für nichtdeterministische BPs jeweils nur die vollständigen Berechnungspfade in *eine* Senke berücksichtigen. Das heißt, auf die andere Senke und die zu ihr führenden Pfade kann verzichtet werden.
Der einem BP zugrunde liegende Graph erlaubt mehrere Kanten zwischen zwei Knoten, allerdings sind mehrere *gleichmarkierte* Kanten zwischen zwei Knoten nicht sinnvoll. Die Menge der gleichmarkierten Kanten zwischen zwei Knoten kann, konform zum jeweiligen Akzeptierungsmodus, reduziert werden, ohne dass sich die Resultatsfunktion des BPs ändert. Beim existenziellen und universellen Akzeptierungsmodus entsprechen mehrere gleichmarkierte Kanten einer solchen Kante. Beim Parity-Akzeptierungsmodus werden mehrere gleichmarkierte Kanten auf die Anzahl der gleichmarkierten Kanten modulo zwei (eine oder keine) reduziert.
In dieser Arbeit haben nichtdeterministische BPs mit existenziellem, universellem oder Parity-Akzeptierungsmodus nur eine Senke t und keine gleichmarkierten Kanten zwischen zwei Knoten.
Die Größe eines nichtdeterministischen BPs $\mathcal{B}$ ist die Anzahl der Knoten des zugrunde liegenden Graphen (V, E) und wird mit $|\mathcal{B}| := |V|$ bezeichnet. Aus der obigen Bemerkung über Mehrfachkanten folgt, dass die Anzahl der Kanten quadratisch von der Anzahl der Knoten abhängt, $|E| \leq 2|V|^2$.

Für den randomisierten Nichtdeterminismus muss das Modell etwas erweitert werden. In einem randomisierten BP hat jeder Branchingknoten mindestens eine ausgehende 0- und 1-Kante. Für jede Eingabe $\alpha = (\alpha_1, \ldots, \alpha_n) \in \{0,1\}^n$ und jeden Verzweigungsknoten v gibt es ein gleichverteiltes Zufallselement $E(\alpha, v)$, das Werte

über der Menge, der von v ausgehenden, unter der Eingabe α aktivierten Kanten annimmt. Das heißt, ist v mit X_i markiert, wird zufällig und gleichverteilt aus allen, von v ausgehenden und mit α_i markierten Kanten, eine Kante $E(\alpha, v)$ ausgewählt. Für alle $\alpha \in \{0,1\}^n$ und alle Knoten v sind die Zufallskanten $E(\alpha, v)$ unabhängig. Für jede Eingabe $\alpha \in \{0,1\}^n$ und jedes Knotenpaar v, w, wird eine $\{0,1\}$-wertige Zufallsvariable $\mathcal{B}(\alpha, v, w)$ definiert, die genau dann den Wert 1 annimmt, wenn der zufällige Berechnungspfad, der in v startet unter α zu w führt.
Zu diesem Modell passt der Majoritäts-Akzeptierungsmodus, der wie folgt definiert ist.

$$\mathrm{Res}_v^{(\mathrm{maj})}(\alpha) := 1 \quad \Longleftrightarrow \quad \Pr[\mathcal{B}(\alpha, v, t_1) = 1] \geq \Pr[\mathcal{B}(\alpha, u, t_0) = 1]$$

Diese Definition ist äquivalent zu $\Pr[\mathcal{B}(a, v, t_1) = 1] \geq 1/2$.
Als Maß für die Größe eines randomisierten BPs reicht die Anzahl der Knoten des zugrunde liegenden Graphen nicht aus. Ein randomisiertes BP enthält noch zusätzliche Informationen, denn mehrere gleichmarkierte Kanten zwischen zwei Knoten sind sinnvoll. Weil die Anzahl der Kanten nicht von der Anzahl der Knoten abhängt, muss die Anzahl der Kanten des zugrunde liegenden Graphen in die Größe mit eingehen. Sei v ein beliebiger Knoten und Succ_v^δ für $\delta \in \{0,1\}$ die Menge der δ-Nachfolger von v, also die Knoten, zu denen es mindestens eine von v ausgehende δ-Kante gibt. Die zu v gehörige gleichverteilte Zufallsvariable über den von v ausgehenden Kanten, induziert eine, in der Regel nicht gleichverteilte, diskrete Wahrscheinlichkeitsverteilung auf den δ-Nachfolgern von v. Für jeden Knoten v wird die Entropie H_v^δ der Verteilung seiner δ-Nachfolger berücksichtigt.
Die Wahrscheinlichkeit p_w eines Knotens $w \in \mathrm{Succ}_v^\delta$ ist der Kehrwert der Anzahl von δ-Kanten, die von v nach w führen. Sei k_v^δ die Anzahl aller von v ausgehenden δ-Kanten. Es gilt $p_w \geq 1/k_v^\delta$ für alle $w \in \mathrm{Succ}_v^\delta$. Die Entropie ist wie üblich definiert und kann nach oben abgeschätzt werden.

$$H_v^\delta := -\sum_{w \in \mathrm{Succ}_v^\delta} p_w \log_2(p_w) \leq \sum_{w \in \mathrm{Succ}_v^\delta} p_w \log_2(k_v^\delta) = \log_2(k_v^\delta)$$

Für jedes $v \in V$ und $\delta \in \{0,1\}$ wird die Anzahl der δ-Nachfolger durch die Anzahl der Knoten abgeschätzt, $|\,\mathrm{Succ}_v^\delta\,| \leq |V|$ und die Anzahl der ausgehenden δ-Kanten kann durch die Anzahl aller Kanten abgeschätzt werden, $\log_2(k_v^\delta) \leq \log_2(|E|)$. Dann ergibt sich $|\mathcal{B}| := |V|^2 \log(|E|)$ als vernünftiges Maß für die Größe eines randomisierten BPs $\mathcal{B}$. Das entspricht auch der Intuition, für jeden Knoten die Menge der Nachfolger und für jeden Nachfolger die Wahrscheinlichkeit (binär kodiert) zu speichern, mit der er ausgewählt wird.
Eine saubere Definition des Größenmaßes ist wichtig, denn, wie man sieht, hat ein randomisiertes BP mit polynomieller Anzahl von Knoten und superpolynomieller

Anzahl von Kanten trotzdem polynomielle Größe.

Wie gesehen, stellen nichtdeterministische und randomisierte BPs mit entsprechendem Akzeptierungsmodus an jedem Knoten eine Boolesche Funktion dar. Einem Branchingprogramm wird genau eine Boolesche Funktion zugeordnet, indem ein Knoten des BPs, die Quelle s, ausgezeichnet wird. Dabei muss es sich nicht unbedingt um eine graphtheoretische Quelle handeln. Die Boolesche Funktion des BPs $\mathcal{B}$ ist, mit entsprechendem Akzeptierungsmodus, die Resultatsfunktion der Quelle, $\mathrm{Res}_{\mathcal{B}}(\alpha) := \mathrm{Res}_s(\alpha)$.

1.2.2 Branchingprogramme mit einmaligen Tests

Das Lesen der Eingabe zu reglementieren, ist eine Möglichkeit für die Einschränkung von Berechnungsmodellen. Berechnungsmodelle mit einmaligen Tests (read-once) dürfen auf einem Berechnungsweg jedes Bit der Eingabe maximal einmal lesen. Aufbauend auf einmaligen Tests sind noch weitere Einschränkungsmöglichkeiten sinnvoll.
In *Branchingprogrammen mit einmaligen Tests*, kurz *BP1s*, darf jede Variable in der Menge der Knotenmarkierungen auf jedem Pfad von der Quelle in die Senke maximal einmal vorkommen. Branchingprogramme mit einmaligen Tests wurden von Zák [Zák84] und Wegener [Weg88] eingeführt.
Eine schärfere Reglementierung sind *einmalige geordnete Tests*. Das heißt, auf einem Berechnungsweg wird jedes Bit einer Eingabe nur einmal und die Bits jeder Eingabe werden in derselben Reihenfolge gelesen.
Sei $\mathcal{B}$ ein Branchingprogramm mit einmaligen geordneten Tests auf der Variablenmenge $\{X_1, \ldots, X_n\}$. Dann wird die Variablenordnung $(X_{\pi(i)}, \ldots, X_{\pi(n)})$ durch eine feste Permutation π der Indizes der Variablen festgelegt. In $\mathcal{B}$ werden auf allen Pfaden von der Quelle in die Senke, bis auf Auslassungen, die Variablen, in der durch die Variablenordnung festgelegten Reihenfolge, durchlaufen.
Branchingprogramme mit einmaligen geordneten Tests werden entgegen aller anderen Bezeichungskonventionen in dieser Arbeit mit *OBDD* (*ordered binary decision diagram*) abgekürzt. Die theoretischen Informatiker sprechen überwiegend von Branchingprogrammen, während die praktischen Informatiker die Abkürzung BDD (binary decision diagram) vorziehen. Allerdings hat sich die Abkürzung OBDD allgemein durchgesetzt. Bryant [Bry86] betrachtete deterministische OBDDs und legte damit den Grundstein für die erfolgreiche Verwendung von Branchingprogrammen als Datenstruktur für Boolesche Funktionen in der praktischen Informatik.
Zwischen Berechnungsmodellen mit einmaligen Tests, die Bits einer Eingabe können auf einem Berechnungsweg in einer beliebigen Reihenfolge gelesen werden, und denen mit einmaligen geordneten Tests, die Bits jeder Eingabe müssen auf einem Berechnungsweg in derselben Reihenfolge gelesen werden, liegt noch eine

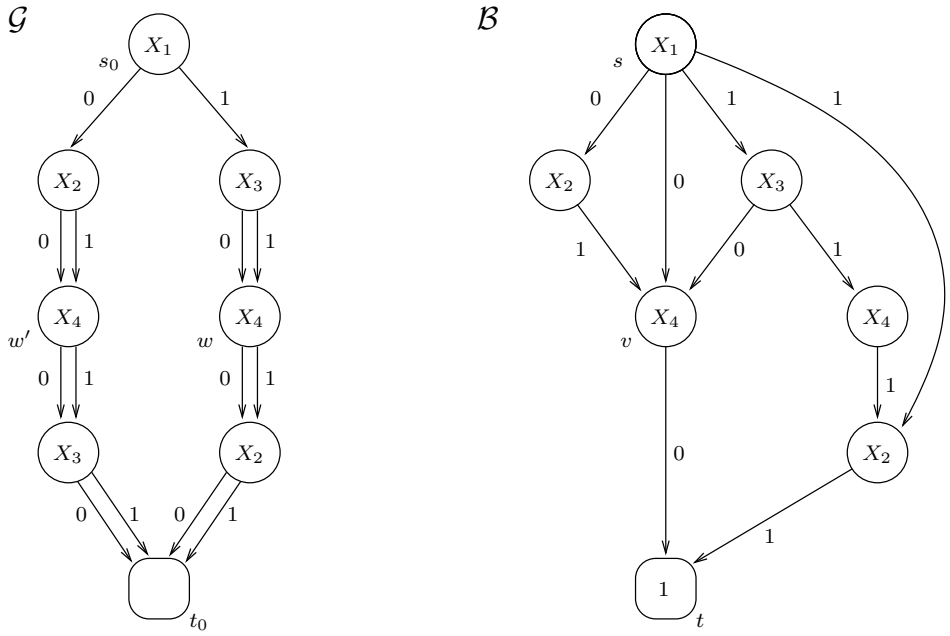

Abbildung 1.2: Graphordnung $\mathcal{G}$ und graphgesteuertes BP1 $\mathcal{B}$.

weitere Art der Einschränkung. Für jede Eingabe wird eine Lesereihenfolge der Eingabebits vorgeschrieben, aber unterschiedliche Eingaben können unterschiedlichen Lesereihenfolgen zugeordnet werden.

Durch eine *Graphordnung* kann verschiedenen Eingaben unterschiedliche Variablenordnungen zugeordnet werden. Eine Graphordnung $\mathcal{G}$ der Variablenmenge $\{X_1, \ldots, X_n\}$ ist einem deterministischen BP1 sehr ähnlich. Es gibt einen zugrunde liegenden kreisfreien Graphen (V_0, E_0) mit einer graphtheoretischen Quelle und einer Senke. Mit Ausnahme der Senke ist jeder Knoten mit einer Variable aus $\{X_1, \ldots, X_n\}$ markiert und hat genau zwei ausgehende Kanten, die jeweils mit 0 und 1 markiert sind. Darüber hinaus kommt jede Variable unter den Knotenmarkierungen aller Pfade von der Quelle in die Senke genau einmal vor.

Aus der Definition folgt, dass es für jede Eingabe $\alpha \in \{0,1\}^n$ in der Graphordnung einen eindeutigen vollständigen Berechnungspfad gibt, auf dem jede Variable genau einmal getestet wird. Die Folge der Variablen auf diesem Berechnungspfad ist die von der Graphordnung $\mathcal{G}$ unter der Eingabe α definierten Variablenordnung.

Für ein *graphgesteuertes BP1* $\mathcal{B}$ mit Graphordnung $\mathcal{G}$ gilt, für jede Eingabe $\alpha \in \{0,1\}^n$ ist die Folge, der auf einem beliebigen Berechnungspfad in $\mathcal{B}$ unter α getesteten Variablen, eine Teilfolge der von $\mathcal{G}$ unter α definierten Variablenordnung.

Eine spezielle Einschränkung von graphgesteuerten BP1s ist die *Wohlstrukturiertheit*. Sei $\mathcal{B} = (V, E)$ ein wohlstrukturiertes graphgesteuertes BP1 mit Graphordnung $\mathcal{G} =$

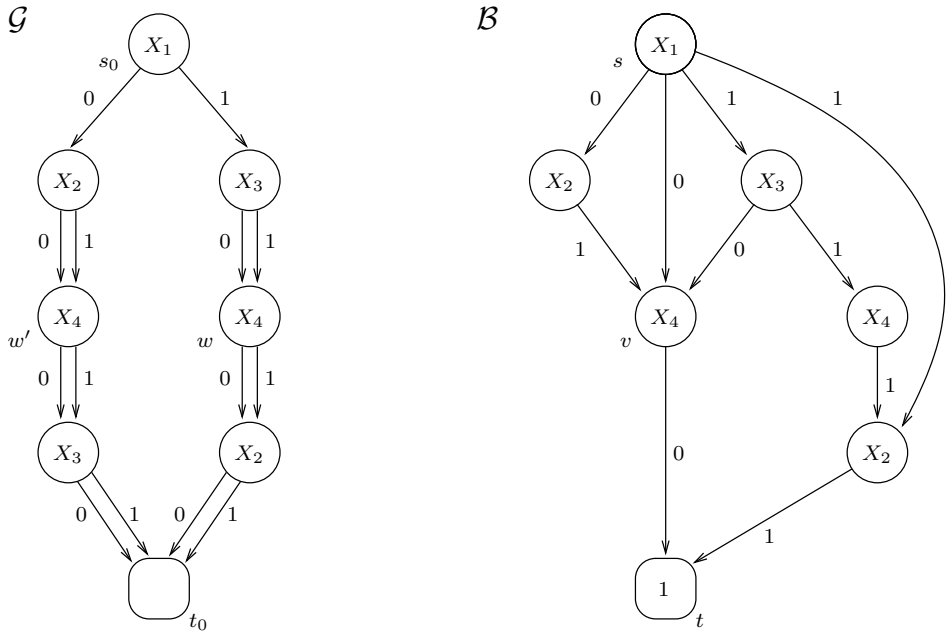

Abbildung 1.3: Wohlstrukturiertes graphgesteuertes BP1 $\mathcal{B}$ mit Graphordnung $\mathcal{G}$.

(V_0, E_0), dann gibt es eine Funktion level : $V \to V_0$, mit folgenden Eigenschaften.

Die Senke t von $\mathcal{B}$ wird auf die Senke t_0 von $\mathcal{G}$ abgebildet, level$(t) = t_0$. Die Knoten v und level(v) sind mit derselben Variable markiert. Sei $\alpha \in \{0,1\}^n$ eine beliebige Eingabe, dann gilt für jeden Knoten v der zu einem Berechnungspfad in $\mathcal{B}$ unter α gehört, level(v) gehört zu dem eindeutigen vollständigen Berechnungspfad in $\mathcal{G}$ unter α.

Das graphgesteuerte BP1 $\mathcal{B}$ mit Graphordnung $\mathcal{G}$ in Abbildung 1.2 ist nicht wohlstrukturiert. Für den Knoten v in $\mathcal{B}$ kann kein Knoten level(v) in $\mathcal{G}$ angegeben werden. Es gibt Eingaben für die v und w und Eingaben für die v und w' durchlaufen werden. Damit $\mathcal{B}$ wohlstrukturiert wird, muss eine Kopie v' von v erzeugt werden, sodass v nur für die Eingaben, die $\mathcal{G}$ über w laufen und v' nur für die Eingabe die $\mathcal{G}$ über w' laufen, durchlaufen wird (siehe Abbildung 1.3). Hier deutet sich schon an, dass ein wohlstrukturiertes graphgesteuertes BP1 aus einem graphgesteuerten BP1 durch hinzufügen von Knoten erzeugt werden kann. Offensichtlich muss dabei die Größe der Graphordnung berücksichtigt werden (siehe Lemma 2.18).

Graphordnungen wurden für deterministische BP1s von Gergov und Meinel [GM93] sowie Sieling und Wegener [SW95] unabhängig voneinander eingeführt.

1.2.3 Branchingprogramme mit mehrfachen Tests

Als Erweiterung von Berechnungsmodellen mit einmaligen Tests liegt es nahe, mehrfache Tests zuzulassen. Auch bei diesem Modell gibt es verschiedene Ausprägungen, wie die Anzahl der Tests und die Lesereihenfolge reglementiert werden können.
Genau wie bei einmaligen Tests ist die einfachste Möglichkeit mehrfache Tests zu definieren, festzulegen wie oft jedes Bit der Eingabe auf einem Berechnungsweg gelesen werden darf.
In *Branchingprogrammen mit k-fachen Tests*, (*BPks*), darf jede Variable in der Menge der Knotenmarkierungen auf jedem Pfad von der Quelle in die Senke maximal k-mal vorkommen.
Eine weitere Möglichkeit, Berechnungsmodelle mit mehrfachen Tests zu definieren, ist für die Bits jeder Eingabe eine globale Lesereihenfolge festzulegen, in der die Eingabebits auch mehrfach vorkommen dürfen. Die Anzahl der Tests pro Bit ist somit indirekt durch die Länge der Lesereihenfolge beschränkt. Die zu diesem Modell passenden Branchingprogramme werden *oblivious BP* genannt. Zu einem oblivious BP $\mathcal{B}$ gehört eine Variablenordnung $\sigma = (\sigma_1, \ldots, \sigma_n)$ über der Variablenmenge $\{X_1, \ldots, X_n\}$, in der Variablen auch mehrfach vorkommen dürfen. Es gilt, auf jedem Pfad in $\mathcal{B}$ ist die Folge der durchlaufenen Variablen eine Teilfolge von σ. Die *Länge* eines oblivious BPs ist die Länge $|\sigma|$ der zugehörigen Variablenordnung. Untere Schranken für oblivious BPs der Länge $\ell = kn$, wobei k eine Konstante oder eine unterlineare Funktion in der Anzahl der Variablen n ist, finden sich in [AM88], [Kra91], [KW91] und [BNS92].
Die Eingabebits auf einem Berechnungsweg mehrfach in derselben Reihenfolge zu lesen, ist eine Abwandlung des Prinzips der globalen Lesereihenfolge.
Sei $\mathcal{B}$ ein *Branchingprogramm mit k-fachen geordneten Tests* (k-OBDD). Genau wie bei einmaligen geordneten Tests legt eine feste Permutation π der Indizes der Variablenmenge $\{X_1, \ldots, X_n\}$ eine Variablenfolge $(X_{\pi(1)}, \ldots, X_{\pi(n)})$ fest. Sei σ die Variablenordnung, die durch Hintereinanderstellen von k Kopien der Variablenfolge entsteht, dann ist das k-OBDD $\mathcal{B}$ ein oblivious BP der Länge kn mit Variablenordnung σ.

1.3 Arithmetische Branchingprogramme

Arithmetische Branchingprogramme sind ein allgemeineres Modell, in dem sich nichtdeterministische und randomisierte Branchingprogramme zusammenfassen lassen. Arithmetische Branchingprogramme werden über Halbringen definiert. Alle Halbringe, die in dieser Arbeit verwendet werden, sind *kommutativ*.
Ein Ziel, das durch die Verwendung von arithmetischen Branchingprogrammen erreicht werden soll, ist, Eigenschaften von arithmetischen BPs zu beweisen und

diese auf alle Modelle zu übertragen, ohne dass eine Unterscheidung der einzelnen Akzeptierungsmodi nötig ist. Durch Anwendung von Techniken aus der linearen Algebra können für arithmetische Branchingprogramme insbesondere Ergebnisse zur Approximation von nichtdeterministischen Branchingprogrammen erzielt werden.
Ein arithmetisches Branchingprogramm ist auf einer Variablenmenge $\{X_1, \ldots, X_n\}$ über einem Halbring $\mathbf{R}$ definiert. Das Grundgerüst ist wieder ein gerichteter kreisfreier Graph (V, E) mit Knotenmenge V und Kantenmenge E, in dem Mehrfachkanten erlaubt sind. Der Graph hat eine graphtheoretische Senke, die mit $1 \in \mathbf{R}$ markiert ist und mit t bezeichnet wird. Jeder Verzweigungsknoten ist mit einer Variable aus $\{X_1, \ldots, X_n\}$ markiert. Die Kanten sind mit einem Wert $\delta \in \{0, 1\}$ markiert und haben zusätzlich ein Gewicht $\omega \in \mathbf{R}$.
Sei $\alpha \in \{0, 1\}$ eine beliebige Belegung der Variablenmenge $\{X_1, \ldots, X_n\}$. Für ein Knotenpaar v, w ist das *Gewicht eines Berechnungspfades* das Produkt in $\mathbf{R}$ der Gewichte der durchlaufenen Kanten und $\text{weight}_{v,w}(\alpha)$ ist die Summe in $\mathbf{R}$ der Gewichte aller Berechnungspfade von v nach w unter α.
Die Definition der Gewichtsfunktion entspricht einer rekursiven Beschreibung, die sehr hilfreich zum Verständnis des Rechnens mit arithmetischen BPs ist. Das Gewicht der Senke liegt fest, $\text{weight}_{t,t}(\alpha) = 1_{\mathbf{R}}$. Sei Out_v^δ die Menge der von v ausgehenden δ-Kanten. Für jede Kante $e \in \text{Out}_v^\delta$ ist ω_e das Gewicht der Kante und u_e der Endknoten der Kante. Somit ergibt sich,

$$\text{weight}_{v,t} = (1 - X_i) \sum_{e \in \text{Out}_v^0} \omega_e \, \text{weight}_{u_e,t} \; + X_i \sum_{e \in \text{Out}_v^1} \omega_e \, \text{weight}_{u_e,t} \quad .$$

Durch Bewertung der Gewichtsfunktion mit verschiedenen Akzeptierungsmodi wird jedem Knoten v eine Boolesche Funktion Res_v zugeordnet.
Gebräuchlich sind, für ein beliebiges, festes $\tau \in \mathbf{R}$, der Nicht-τ- oder $(\neq \tau)$-Akzeptierungsmodus und der Exakt-τ- oder $(= \tau)$-Akzeptierungsmodus.

$$\text{Res}_v^{(\neq\tau)}(\alpha) := 1 \iff \text{weight}_{v,t}(\alpha) \neq \tau \in \mathbf{R}$$
$$\text{Res}_v^{(=\tau)}(\alpha) := 1 \iff \text{weight}_{v,t}(\alpha) = \tau \in \mathbf{R}$$

Über geordneten Halbringen sind auch Akzeptierungsmodi, die ein beliebiges, festes $\tau \in \mathbf{R}$ als Grenzwert festlegen, der Minor-τ- oder $(\leq \tau)$- Akzeptierungsmodus und der Major-τ- oder $(\geq \tau)$-Akzeptierungsmodus, sinnvoll.

$$\text{Res}_v^{(\geq\tau)}(\alpha) := 1 \iff \text{weight}_{v,t}(\alpha) \geq \tau \in \mathbf{R}$$
$$\text{Res}_v^{(\leq\tau)}(\alpha) := 1 \iff \text{weight}_{v,t}(\alpha) \leq \tau \in \mathbf{R}$$

Ein arithmetisches BP über dem Halbring $\mathbf{R}$ wird mit $\mathbf{R}$-BP bezeichnet. Ein $\mathbf{R}$-BP mit gegebenem Akzeptierungsmodus, zum Beispiel $(\neq \tau)$, bezeichnet man als $\mathbf{R}$-$(\neq \tau)$-BP.

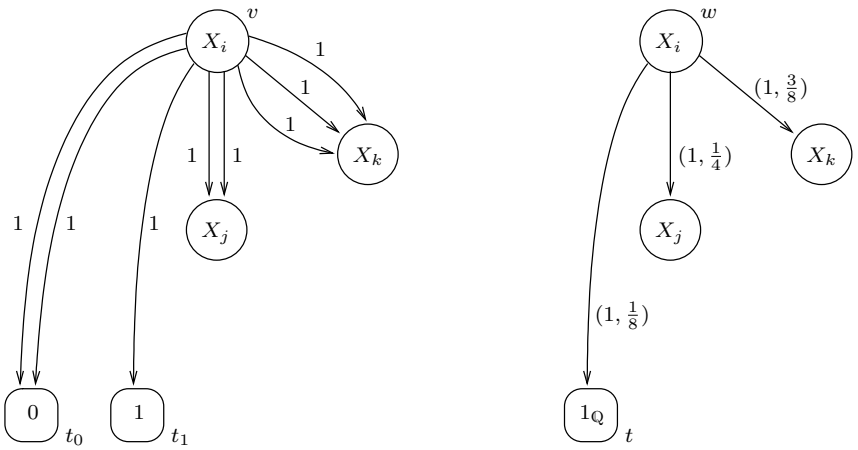

Abbildung 1.4: Umwandlung des Knotens v eines randomisierten BPs zum Knoten w eines $\mathbb{Q}$-BPs mit Wahrscheinlichkeits-Nebenbedingung und vice versa.

Genau wie bei nichtdeterministischen und randomisierten Branchingprogrammen werden in dieser Arbeit nur arithmetische BPs mit einer Quelle s betrachtet. Das Gewicht eines arithmetischen BPs $\mathcal{B}$ ist das Gewicht der Quelle über der Senke, $\text{weight}_{\mathcal{B}}(\alpha) := \text{weight}_{s,t}(\alpha)$. Die von $\mathcal{B}$ dargestellte Boolesche Funktion ist mit gewähltem Akzeptierungsmodus, zum Beispiel $(\neq \tau)$, die Resultatsfunktion der Quelle, $\text{Res}_{\mathcal{B}}^{(\neq\tau)}(\alpha) := \text{Res}_{s}^{(\neq\tau)}(\alpha)$.
In arithmetischen BPs können, wie bei nichtdeterministischen BPs, gleichmarkierte Kanten zwischen zwei Knoten durch eine Kante ersetzt werden. Die neue Kante erhält als Gewicht die Summe der Gewichte der wegfallenden Kanten.
Die Größe eines arithmetischen BPs ist die Anzahl der Knoten des zugrunde liegenden Graphen. Allerdings muss die binäre Kodierung der Kantenmarkierungen zusätzlich mitberücksichtigt werden.

Wie passen nun arithmetische BPs und die vorher eingeführten Modelle zusammen?
Man betrachtet ein arithmetisches BP über dem Booleschen Halbring $\mathbf{B}_0$ und nimmt an, dass alle Kanten und somit auch alle Berechnungspfade das Gewicht $1 \in \mathbf{B}_0$ haben. Dann ist die Bedingung, es gibt einen (keinen) vollständigen Berechnungspfad, gleichbedeutend mit, die Summe der Gewichte der Berechnungspfade ist ungleich (gleich) null. Daraus folgt, $\vee$BPs sind gleichzusetzen mit $\mathbf{B}_0$-$(\neq 0)$-BPs und $\wedge$BPs mit $\mathbf{B}_0$-$(= 0)$-BPs. Mit derselben Überlegung und dem Körper $\mathbb{F}_2$, dem Primkörper mit 2 Elementen, erhält man eine Übereinstimmung von $\oplus$BPs und $\mathbb{F}_2$-$(\neq 0)$-BPs.

Randomisierte BPs einzupassen ist etwas aufwendiger. Dazu werden arithmetische BPs über dem Körper $\mathbb{Q}$ der rationalen Zahlen verwendet, die zusätzlich folgende *Wahrscheinlichkeits-Nebenbedingungen* erfüllen. Alle Kantengewichte sind positiv und die Summe der Gewichte ausgehender, gleichmarkierter Kanten eines Knotens ist kleiner gleich 1.
Ein randomisiertes BP kann direkt in ein $\mathbb{Q}$-BP mit Wahrscheinlichkeits-Nebenbedingung umgewandelt werden. Die 1-Senke wird zur einzigen Senke. Die 0-Senke und alle zu ihr führenden Pfade können gelöscht werden. Die ausgehenden Kanten eines Knotens mit gleicher Markierung erhalten als Gewicht den Kehrwert ihrer Anzahl. Ausgehende, gleichmarkierte Kanten eines Knotens, die zum selben Knoten führen, werden in einer Kante zusammengelegt. Das Gewicht dieser Kanten ist die Summe der ursprünglichen Kantengewichte.
Bei der Umwandlung eines $\mathbb{Q}$-BPs mit Wahrscheinlichkeits-Nebenbedingungen in ein randomisiertes BP gibt es zwei Probleme. Zum einen haben die Kanten verschiedene Gewichte, das heißt, sie sind nicht gleichwahrscheinlich. Zu anderen summieren sich die Gewichte nicht unbedingt zu 1, das heißt, es fehlen Kanten in die 0-Senke.
Für die Umwandlung wird zuerst die vorhandene Senke zur 1-Senke und die fehlende 0-Senke wird hinzugefügt. Die mit $\delta \in \{0, 1\}$ markierten Kanten eines Knotens werden jeweils getrennt behandelt. Die fehlenden, in die 0-Senke führenden, δ-Kanten jedes Knotens werden ergänzt. Dazu wird die Summe S der Gewichte aller von v ausgehenden δ-Kanten gebildet. Anschließend erhält v eine neue δ-Kante mit Gewicht $1 - S$ in die 0-Senke.
Alle ausgehenden δ-Kanten eines Knotens v bekommen das gleiche Gewicht. Die Kantengewichte sind, als rationale Zahlen, Brüche natürlicher Zahlen. Der Hauptnenner h, der Gewichte aller von v ausgehenden δ-Kanten, wird bestimmt. Jedes Kantengewicht kann als Bruch mit Nenner h betrachtet werden. Mit jeder der, von v ausgehenden, δ-Kanten wird auf dieselbe Weise verfahren. Sei r/h das Gewicht einer solchen δ-Kante, dann wird diese Kante durch r δ-Kanten mit Gewicht $1/h$ ersetzt.
In einem so veränderten arithmetischen BP haben alle ausgehenden, gleichmarkierten Kanten eines Knotens dasselbe Gewicht und diese Gewichte summieren sich zu 1. Nun fallen alle Gewichte weg und stattdessen erhält jeder Knoten das zu randomisierten BPs gehörende Zufallselement. So erhält man ein randomisiertes BP, das mit Majoritäts-Akzeptierungsmodus dieselbe Funktion darstellt wie das ursprüngliche $\mathbb{Q}$-BP mit Wahrscheinlichkeits-Nebenbedingung und $(\geq 1/2)$-Akzeptierungsmodus.
Die Umwandlung von einem Modell in das andere ist für die 1-Nachfolger eines Knotens in Abbildung 1.4 illustriert.
Bei der Simulation von randomisierten BPs durch $\mathbb{Q}$-BPs geht die Anzahl der ausgehenden, gleichmarkierten Kanten eines Knotens als Nenner in die Kantengewichte mit ein. Umgekehrt bestimmt der Hauptnenner der Kantengewichte des $\mathbb{Q}$-BPs die Anzahl der Kanten im randomisierten BP. Die Zusatzinformation, die ein randomisiertes BP über die Entropie der Wahrscheinlichkeitsverteilung enthält

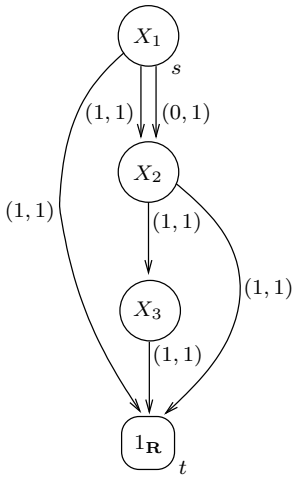

Abbildung 1.5: Arithmetisches BP mit Kantenmarkierungen $(\delta, \omega) \in \{0,1\} \times \mathbf{R}$.

$|V| \log(k)$, entspricht dem Platz, der für die Kodierung der Kantengewichte benötigt wird $|V| \log(h)$, da Hauptnenner h und Anzahl der Kanten k gleich sind.

Ein einfaches Beispiel für ein arithmetisches BP ist in Abbildung 1.5 angegeben. Sei $\mathcal{B}$ das abgebildete arithmetische BP, es gilt,

$$\text{weight}_{\mathcal{B}}(X_1, X_2, X_3) = X_1 + X_2 \cdot (1 + X_3) \ .$$

Sind die Kantengewichte aus $\mathbf{B}_0$ und wird der $(\neq 0)$- oder der $(=0)$-Akzeptierungsmodus angewendet ergibt sich,

$$\begin{aligned} \text{Res}_{\mathcal{B}}^{(\neq 0)}(X_1, X_2, X_3) &= X_1 \vee X_2 \ , \\ \text{Res}_{\mathcal{B}}^{(=0)}(X_1, X_2, X_3) &= \neg X_1 \wedge \neg X_2 \ . \end{aligned}$$

Hingegen gilt mit Kantengewichten aus $\mathbb{F}_2$ und $(\neq 0)$-Akzeptierungsmodus,

$$\text{Res}_{\mathcal{B}}^{(\neq 0)}(X_1, X_2, X_3) = X_1 \oplus (X_2 \wedge \neg X_3) \ .$$

Mit arithmetischen BPs kann man sehr gut rechnen. Sei $\mathbf{R}$ ein Halbring, $r \in \mathbf{R}$ und seien $\mathcal{B}', \mathcal{B}''$ zwei $\mathbf{R}$-BPs. Wenn man die jeweiligen Gewichtsfunktionen mit den $\mathbf{R}$-BPs $\mathcal{B}'$ und $\mathcal{B}''$ identifiziert, sind die Operationen über $\mathbf{R}$ einfach nachzubilden. Multiplikation mit einem Skalar $\mathcal{B} := r \cdot \mathcal{B}'$, wird durch Multiplikation der Kantengewichte aller von der Quelle ausgehender Kanten mit r erreicht. Addition $\mathcal{B} := \mathcal{B}' + \mathcal{B}''$, wird durch Kombination der Quellen nachgebildet. Sind beide Quellen mit derselben Variable markiert, werden die Quellen verschmolzen. Sind sie mit

unterschiedlichen Variablen markiert, wird die eine Quelle über eine 0- und 1-Kante, jeweils mit Gewicht 1, Nachfolger der anderen. Die Multiplikation $\mathcal{B} := \mathcal{B}' \cdot \mathcal{B}''$ wird durch Ersetzen der Senke von $\mathcal{B}'$ durch die Quelle von $\mathcal{B}''$ (oder umgekehrt) simuliert. Alle diese Operationen können ohne zusätzlichen Aufwand an Rechenzeit oder Speicherplatz durchgeführt werden.
Leider gilt das für arithmetische OBDDs, das in dieser Arbeit verwendete Berechnungsmodell, nicht ohne Einschränkung. Die wichtigste Operation, die Multiplikation zweier arithmetischer OBDDs, erfordert quadratischen Aufwand. Dazu wird die aus der Theorie der endlichen Automaten bekannte Produktkonstruktion verwendet. Der Produktgraph wird durch einen simultanen Durchlauf der einzelnen Graphen erzeugt. Die Kantengewichte passen sich in diese Methode problemlos ein. Das Gewicht einer Produktkante ist das Produkt der Gewichte aller beteiligten Kanten.

1.4 Gliederung

Kapitel 1 enthält eine kurze Einführung in die Theorie der oberen und unteren Schranken. Weiterhin werden alle betrachteten Varianten von Branchingprogrammen definiert.
Die nachfolgenden Kapitel sind alle locker verknüpft, durch die Verwandtschaft der jeweils behandelten Branchingprogramme zu nichtdeterministischen OBDDs.
Kapitel 2 behandelt wohlstrukturierte graphgesteuerte $\oplus$BP1s und graphgesteuerte $\oplus$BP1s. Diese Berechnungsmodelle erweitern $\oplus$OBDDs, indem die Forderung nach einer globalen Variablenordnung abgeschwächt wird. Für beide Modelle werden Techniken für den Beweis von unteren Schranken angegeben und diese auf explizit definierte Funktionen angewendet.
In Kapitel 3 werden verschiedene Modelle zur Darstellung von Booleschen Funktionen mit $\oplus$OBDDs approximiert. Als Hilfsmittel kommen dabei arithmetische OBDDs zum Einsatz. Arithmetische OBDDs erweitern nichtdeterministische OBDDs durch zusätzliche Kantengewichte aus einem Halbring. Somit wird jede Eingabe in diesen Halbring abgebildet und man kann mit arithmetischen OBDDs über dem Halbring rechnen. Erkenntnisse aus der linearen Algebra, insbesondere Abschätzungen zur Anzahl der Nullstellen eines eingeschränkten Polynoms, werden für überraschende Approximationsergebnisse benutzt.
Mit nichtdeterministischen k-OBDDs, einer Erweiterung von nichtdeterministischen OBDDs, die das k-fache Lesen der Eingabe, allerdings immer in derselben Reihenfolge, erlaubt, beschäftigt sich Kapitel 4. Wieder kommen arithmetische OBDDs zum Einsatz. Zum einen kann so der existenzielle, der universelle und den Parity-Akzeptierungsmodus gleichzeitig behandelt werden. Und zum anderen werden die Berechnungen, bei der Betrachtung von randomisierten k-OBDDs, durch die

Benutzung von arithmetischen OBDDs einfacher.
Der Anhang dient der Bequemlichkeit des Lesers. Anhang A erläutert, die in Kapitel 2 verwendeten Eigenschaften von Linearen Codes und ist ein Ausschnitt aus der Monographie von Jukna [Juk01]. Anhang B stellt einige Techniken zur Abschätzung der Anzahl der Nullstellen von eingeschränkten Polynomen vor. Diese Techniken sind von Zippel [Zip79], Schwarz [Sch80] und Blum, Chandra, Wegman [BCW80] benutzt worden, zur Abschätzung für die Wahrscheinlichkeit eine Nullstelle zu wählen, bei zufälliger und gleichverteilter Wahl einer Belegung der Variablen des Polynoms.

1.5 Veröffentlichungen

Ein Großteil der Ergebnisse der folgenden Veröffentlichungen findet sich in der vorliegenden Arbeit wieder.

[BHW01] Henrik Brosenne, Matthias Homeister und Stephan Waack. Graph Driven Free Parity-BDDs: Algorithms and Lower Bounds. In *Proceedings of the 26th Symposium on Mathematical Foundations of Computer Science (MFCS 2001), Lecture Notes in Computer Science*, 2136:212–223, Springer Verlag, 2001.

[BHW02] Henrik Brosenne, Matthias Homeister und Stephan Waack. Characterizing the Complexity of Boolean Functions represented by Well–Structured Graph–Driven Parity–FBDDs. *RAIRO – Theoretical Informatics and Applications*, 36(3):229–247, EDP Sciences, 2002.

[BHW03] Henrik Brosenne, Matthias Homeister und Stephan Waack. Lower Bounds for General Graph-Driven Read-Once Parity Branching Programs. In *Proceedings of the 28th Symposium on Mathematical Foundations of Computer Science (MFCS 2003), Lecture Notes in Computer Science*, 2747:290–299, Springer, 2003.

[BDHW05] Henrik Brosenne, Carsten Damm, Matthias Homeister und Stephan Waack. On Approximation by $\oplus$OBDDs. In *Proceedings of the 7th International Symposium on Representations and Methodology of Future Computing Technologies*, 2005.

[BHW06] Henrik Brosenne, Matthias Homeister und Stephan Waack. Nondeterministic ordered binary decision diagrams with repeated tests and various modes of acceptance. *Information Processing Letters*, 98:6–10, Elsevier, 2006.

2 Eingeschränkte ⊕BP1s

Graphgesteuerte Parity-Branchingprogramme mit einmaligen Tests (graphgesteuerte ⊕BP1s) reglementieren einerseits das Lesen der Eingabe stärker als Parity-Branchingprogramme mit einmaligen Tests (⊕BP1s) und sind andererseits weniger streng als Parity-Branchingprogramme mit einmaligen geordneten Tests (⊕OBDDs). Aber unterscheidet sich die Komplexitätsklasse von graphgesteuerten ⊕BP1s jeweils von den beiden, zu den anderen Modellen gehörenden, Komplexitätsklassen? Der Beantwortung dieser Frage wird sich in diesem Kapitel schrittweise genähert. Entscheidend dabei ist, dass es der Parity-Akzeptierungsmodus erlaubt, die Komplexität einer Booleschen Funktion mit Mitteln der linearen Algebra zu beschreiben.
In Abschnitt 2.1 werden wohlstrukturierte graphgesteuerte ⊕BP1s untersucht. Die Komplexität einer Booleschen Funktion, die von einem wohlstrukturierten graphgesteuerten ⊕BP1 dargestellt wird, kann nur abhängig von der Funktion und der Graphordnung, mit Hilfe der linearen Algebra beschrieben werden. Aus dieser Beschreibung wird ein Kriterium für untere Schranken abgeleitet und eine superpolynomielle untere Schranke für die charakteristische Funktion eines bestimmten linearen Codes bewiesen.
Die Forderung nach Wohlstrukturiertheit wird in Abschnitt 2.2 fallen gelassen. Erneut wird die lineare Algebra benutzt, um eine Beschreibung für die Komplexität einer Booleschen Funktion, die von einem graphgesteuerten ⊕BP1s dargestellt wird, anzugeben, deren Invarianten die Funktion und die Graphordnung sind. Daraus wird ein sehr einfaches Kriterium für untere Schranken abgeleitet. Dieses Kriterium wird auf die charakteristischen Funktionen von linearen Codes und Permutationsmatrizen angewendet. Weiterhin wird eine zu dem Kriterium für untere Schranken passende Funktion eingeführt, die aber mit einem ⊕BP1 polynomieller Größe dargestellt werden kann.
Abschnitt 2.3 ordnet die zu graphgesteuerten und wohlstrukturierten graphgesteuerten ⊕BP1s gehörenden Komplexitätsklassen mit den erzielten Ergebnissen, unter Einbeziehung bekannter oberer und unterer Schranken, in eine Hierarchie von ausgewählten Komplexitätsklassen ein. Damit wird auch die Frage positiv beantwortet, ob graphgesteuerte ⊕BP1s eine echte Erweiterung von ⊕OBDDs und eine echte Einschränkung von ⊕BP1s sind.
Obwohl in diesem Kapitel Techniken zum Beweis von unteren Schranken für graphgesteuerte und wohlstrukturierte graphgesteuerte ⊕BP1s angegeben werden, ist

keine Funktion bekannt, die die Komplexitätsklassen dieser beide Modelle trennt. In Abschnitt 2.4 wird untersucht wo, trotz der engen Verwandtschaft der beiden Modelle, eine Lösung dieses Problems ansetzen könnte. Weiterhin wird gezeigt, dass graphgesteuerte ⊕BP1s eine sehr natürliche Einschränkung von ⊕BP1s sind.
Die wesentlichen Ergebnisse dieses Kapitels sind bereits veröffentlicht. Wohlstrukturierten graphgesteuerten ⊕BP1s werden in [BHW01] und in überarbeiteter Form in [BHW02] behandelt. Die Überlegungen zu graphgesteuerten ⊕BP1s finden sich in [BHW03].

2.1 Wohlstrukturierte graphgesteuerte ⊕BP1s

2.1.1 Algebraische Charakterisierung

Die algebraische Charakterisierung der Komplexität von Funktionen, die mit wohlstrukturierten graphgesteuerten ⊕BP1s dargestellt werden, ist in [Bro00] ausführlich beschrieben und wird hier nur zitiert.
Die Wohlstrukturiertheit ist eine starke Eigenschaft, die eine Zerlegung der Knotenmenge in disjunkte Mengen ermöglicht und eine Ordnungsrelation auf diesen Mengen definiert.
Die Knotenmenge eines wohlstrukturierten graphgesteuerten ⊕BP1s $\mathcal{B} = (V, E)$ mit Graphordnung $\mathcal{G} = (V_0, E_0)$ wird durch die Levelfunktion $\text{level} : V \to V_0$ in disjunkte Mengen, Level genannt, zerlegt. Der Level eines Knotens w der Graphordnung $\mathcal{G}$ ist definiert als die Menge aller Knoten von $\mathcal{B}$, die nach w abgebildet werden, $\mathcal{N}_w(\mathcal{B}) := \{v \in V \mid \text{level}(v) = w\}$.
Sei $\mathcal{G}$ eine beliebige Graphordnung mit Quelle s_0 und Senke t_0, dann kann auf der Knotenmenge des zugrundeliegenden Graphen (V_0, E_0) folgende Halbordnung definiert werden. Seien $u, w \in V_0$ zwei beliebige Knoten. Es gilt $u \leq w$ genau dann, wenn alle Pfade von w in die Senke t_0 über u führen. Insbesondere gilt $w \leq w$ und $w \leq t_0$ für alle $w \in V_0$.
Die Halbordnung definiert für jeden Knoten $w \in V_0$ eine Menge von *Vorgängern* (bezüglich der Relation) $\{u \in V_0 \mid u \leq w,\ u \neq w\}$. Nach Definition führen *alle* Pfade von einem Knoten in die Senke über *jeden* Vorgänger, deshalb stehen alle Vorgänger eines Knoten paarweise zueinander in Relation. Das heißt, die Menge der Vorgänger eines Knotens ist total geordnet. Somit kann zu jedem Knoten ein eindeutiger *direkter Vorgänger*, der größte Knoten in der Menge aller Vorgänger, bestimmt werden. Sei $w \in V_0$ ein beliebiger Knoten der Graphordnung und $u \in V_0$ der direkte Vorgänger von w. Dann gilt, $v \leq u$ für alle $v \in V_0$ mit $v \leq w$ und $v \neq w$.
Jedem Knoten w der Graphordnung $\mathcal{G}$ wird folgender $\mathbb{F}_2$-Vektorraum zugeordnet.

$$\mathbb{B}_w(f) := \text{span}_{\mathbb{F}_2} \bigcup_{u \leq w} \{f|_{\alpha(\pi)} \mid \pi \text{ ist ein Pfad in } \mathcal{G} \text{ von } s_0 \text{ nach } u\}$$

Sei π ein Pfad in der Graphordnung von der Quelle s_0 zum Knoten w, dann ist $\alpha(\pi)$ die Teilbelegung der Variablen $\{X_1, \ldots, X_n\}$, die kanonisch durch den Pfad π bestimmt wird.
Damit ist die notwendige Notation eingeführt, um die algebraische Charakterisierung der Komplexität einer Funktion zu formulieren.

Satz 2.1. *Sei $\mathcal{B}$ ein minimales wohlstrukturiertes graphgesteuertes ⊕BP1 mit Graphordnung $\mathcal{G}$, das die Boolesche Funktion $f \in \mathrm{Bool}_n$ darstellt. Sei w ein beliebiger Verzweigungsknoten der Graphordnung $\mathcal{G}$ und u dessen direkter Vorgänger, bezüglich der Halbordnung auf der Knotenmenge von $\mathcal{G}$.*
Dann gilt,

$$|\mathcal{N}_w(\mathcal{B})| = \dim_{\mathbb{F}_2} \mathbb{B}_w(f) - \dim_{\mathbb{F}_2} \mathbb{B}_u(f) \ .$$

2.1.2 Ein Kriterium für untere Schranken

Satz 2.1 enthält implizit schon ein Kriterium für untere Schranken. Wie üblich wird ein Schnitt durch das Branchingprogramm gelegt. Bei wohlstrukturierten graphgesteuerten ⊕BP1s heißt das, alle Level, die zu den Knoten der Graphordnung einer bestimmten Tiefe gehören, werden betrachtet. Offensichtlich reicht es nicht die Summe der Dimensionen aller Räume $\mathbb{B}_w(f)$, der Knoten w gleicher Tiefe in der Graphordnung, nach unten abzuschätzen. Angenommen w_1, w_2 sind zwei Knoten der Graphordnung mit gleicher Tiefe und u ist der direkte Vorgänger beider Knoten. Dann ist der Vektorraum $\mathbb{B}_u(f)$ sowohl in $\mathbb{B}_{w_1}(f)$ als auch in $\mathbb{B}_{w_2}(f)$ enthalten. Aber wenn man für jeden Knoten w, aus der Menge der Knoten gleicher Tiefe, jeweils einen großen Teilraum von $\mathbb{B}_w(f)$ finden kann, der mit dem Raum $\mathbb{B}_u(f)$, des direkte Vorgängers u von w, einen belanglosen Durchschnitt hat, liefert Satz 2.1 eine untere Schranke.
Um eine entsprechende Klasse von Funktionen zu definieren, wird eine Idee von Jukna [Juk99] an diese Vorüberlegungen angepasst.
Eine Funktion f auf der Variablenmenge $\{X_1, \ldots, X_n\}$ hängt wesentlich von einer Variable X_i ab, wenn das Setzen von X_i auf unterschiedliche Konstanten zu unterschiedlichen Funktionen führt. Das heißt für Boolesche Funktionen, f hängt wesentlich von X_i ab genau dann, wenn $f|_{X_i=0} \neq f|_{X_i=1}$.

Definition 2.2. *Sei $f \in \mathrm{Bool}_n$ eine Boolesche Funktion auf der Variablenmenge $\{X_1, \ldots, X_n\}$. Die Funktion f wird* streng k-gemischt *genannt, mit $k < n$, wenn Folgendes gilt. Für jede k-elementige Teilmenge $V \subset \{X_1, \ldots, X_n\}$ hängen alle nicht trivialen Linearkombinationen der 2^k Unterfunktionen von f, die durch Belegung der Variablen aus V entstehen, wesentlich von jeder Variable aus der Teilmenge $\{X_1, \ldots, X_n\} \backslash V$ ab.*

Offensichtlich sind die 2^k Unterfunktionen einer streng k-gemischten Funktion, die durch Setzen der gleichen k Variablen auf alle möglichen Belegungen entstehen, linear

unabhängig. Denn jede nicht triviale Linearkombination dieser Funktionen hängt von jeder der verbleibenden Variablen wesentlich ab und ist somit nicht die Null-Funktion. Aus Satz 2.1 und der Definition von streng k-gemischten Funktionen kann ein Kriterium für untere Schranken gefolgert werden.

Korollar 2.3. *Die Größe jedes wohlstrukturierten graphgesteuerten ⊕BP1s, das eine streng k-gemischte Funktion darstellt, ist nach unten beschränkt durch 2^k.*

Beweis. Sei $\mathcal{G}$ eine beliebige Graphordnung auf den Booleschen Variablen $\{X_1, \ldots, X_n\}$. Sei w ein beliebiger Verzweigungsknoten von $\mathcal{G}$ der $k+1$ Knoten von der Quelle entfernt ist. Sei u der direkte Vorgänger von w bezüglich der Halbordnung auf der Knotenmenge von $\mathcal{G}$. Ohne Beschränkung der Allgemeinheit wird angenommen, dass w mit X_{k+1} markiert ist und das $\{X_1, \ldots, X_k\}$ die Variablenmenge ist, die auf jedem Pfad π in $\mathcal{G}$ von der Quelle s_0 nach w durchlaufen wird. Es sei $\alpha(\pi) \in \{0,1\}^k$, die zum Pfad π gehörende Belegung der Variablen $\{X_1, \ldots, X_k\}$.
Sei Π_w die Menge aller Pfade, die in $\mathcal{G}$ von s_0 zu w führen und $F_w := \{f|_{\alpha(\pi)} \mid \pi \in \Pi_w\}$ die Menge der zugehörigen Unterfunktionen. Es gilt, $|\Pi_w| = |F_w|$, denn f ist streng k-gemischt und somit sind alle $f|_{\alpha(\pi)}$ linear unabhängig.
Mit Satz 2.1 reicht es zu zeigen

$$\dim_{\mathbb{F}_2} \mathbb{B}_w(f) - \dim_{\mathbb{F}_2} \mathbb{B}_u(f) \geq |F_w| \ .$$

Zum Beweis wird der Quotientenraum $\mathbb{B}_w(f)/\mathbb{B}_u(f)$ betrachtet. Der Quotientenraum wird von den Äquivalenzklassen $g \oplus \mathbb{B}_u(f) = \{g \oplus h \mid h \in \mathbb{B}_u(f)\}$ der Funktionen $g \in \mathbb{B}_w(f)$ gebildet. Es gilt, $\dim_{\mathbb{F}_2} \mathbb{B}_w(f) - \dim_{\mathbb{F}_2} \mathbb{B}_u(f) = \dim_{\mathbb{F}_2} (\mathbb{B}_w(f)/\mathbb{B}_u(f))$.
Da f streng k-gemischt ist, hängt jede Linearkombination der Funktionen in F_w wesentlich von X_{k+1} ab. Das heißt, die Funktionen in F_w haben paarweise verschiedene Äquivalenzklassen. Weiterhin sind die Funktionen in F_w linear unabhängig. Daraus folgt, $\dim_{\mathbb{F}_2}(\mathbb{B}_w(f)/\mathbb{B}_u(f)) \geq |F_w|$. □

2.1.3 Untere Schranke für lineare Codes

Mit Korollar 2.3 kann eine exponentielle untere Schranke für die charakteristische Funktion bestimmter linearer Codes bewiesen werden. Einige Grundlagen über lineare Codes sind in Anhang A zusammengestellt.

Satz 2.4. *Sei $C \in \mathbb{F}_2^n$ ein linearer Code mit Minimal-Abstand d und $C^\perp$ dessen dualer Code mit Minimal-Abstand $d^\perp$.*
Dann ist die Größe jedes wohlstrukturierte graphgesteuerte ⊕BP1s, das die charakteristische Funktion f_C von C darstellt, nach unten beschränkt durch $2^{\min(d,d^\perp)-2}$.

Beweis. Sei $k := \min(d, d^\perp) - 2$. Mit Korollar 2.3 reicht es zu zeigen, dass f_C streng k-gemischt ist. Ohne Beschränkung der Allgemeinheit sei $\{X_1, \ldots, X_k\}$ eine beliebige k-elementige Teilmenge der Variablenmenge und X_{k+1} eine weitere Variable.
Sei $A = \{\alpha_1, \ldots, \alpha_{2^k}\}$ die Menge aller möglichen Belegungen der Variablen $X_1, \ldots, X_k$. Es ist zu zeigen, jede nicht triviale Linearkombination der Funktionen $f_C(\alpha_1, X_{k+1}, X_{k+2}, \ldots, X_n), \ldots, f_C(\alpha_{2^k}, X_{k+1}, X_{k+2}, \ldots, X_n)$ hängt wesentlich von X_{k+1} ab.
Der Minimal-Abstand $d^\perp$ des dualen Codes $C^\perp$ ist größer k, daraus folgt, C ist k-universell. Der Raum $\mathbb{F}_2^k$, repräsentiert durch $\alpha_1, \ldots, \alpha_{2^k}$, kann als Projektion von C auf die ersten k Koordinaten jedes Codewortes aufgefasst werden. Daraus folgt, für jedes α_i mit $i = 1, \ldots, 2^k$ gibt es (mindestens) eine Belegung β_i der Variablen $X_{k+1}, X_{k+2}, \ldots, X_n$, sodass $(\alpha_i, \beta_i) \in C$, also $f_C(\alpha_i, \beta_i) = 1$.
Für $i = 1, \ldots, 2^k$ sei β_i' die Belegung, die aus β_i durch Invertieren des Wertes der Variable X_{k+1} entsteht. Es gilt,

$$\begin{aligned} f_C(\alpha_i, \beta_j) &= 0 \quad \text{für } i, j = 1, \ldots, 2^k \text{ mit } i \neq j \\ f_C(\alpha_i, \beta_j') &= 0 \quad \text{für } i, j = 1, \ldots, 2^k \ , \end{aligned}$$

denn der Minimal-Abstand des Codes C ist mindestens $k+2$. Aus $(\alpha_i, \beta_j) \in C$ würde $w((\alpha_i, \beta_j) \oplus (\alpha_j, \beta_j)) \leq k$ folgen und aus $(\alpha_i, \beta_j') \in C$ würde $w((\alpha_i, \beta_j') \oplus (\alpha_j, \beta_j)) \leq k+1$ folgen (die Definition des Hamming-Gewichts w findet sich in Anhang A).
Angenommen es gibt eine nichtleere Teilmenge $I \subseteq \{1, \ldots, 2^k\}$, sodass die Linearkombination $g := \sum_{i \in I} f_C(\alpha_i, X_{k+1}, X_{k+2}, \ldots, X_n)$ nicht wesentlich von X_{k+1} abhängt. Sei $j \in I$ ein beliebiger Index. Es gilt $g(b_j) = 1$ und, weil g nicht von X_{k+1} abhängt, folgt $g(b_j') = 1$. Das heißt, es gibt einen Index $i \in I$ mit $f_C(\alpha_i, \beta_j') = 1$. Das ist ein Widerspruch. □

Um eine konkrete untere Schranke zu erhalten, werden Reed-Muller-Codes betrachtet. Der binäre Reed-Muller-Code $R(r, \ell)$ mit Ordnung r der Länge $n = 2^\ell$ ist die Menge der Graphen aller Polynome in ℓ Variablen über $\mathbb{F}_2$ mit Grad kleiner gleich r. Der Code $R(r, \ell)$ ist linear mit Minimal-Distanz $2^{\ell - r}$. Der duale Code von $R(r, \ell)$ ist $R(\ell - r - 1, \ell)$ (siehe [MS77]). Das nachstehende Korollar folgt unmittelbar aus Satz 2.4.

Korollar 2.5. *Sei $n = 2^\ell$ und $r = \lfloor n/\ell \rfloor$. Dann ist die Größe jedes wohlstrukturierten graphgesteuerten ⊕BP1s, das die charakteristische Funktion des Codes $R(r, \ell)$ darstellt, nach unten beschränkt durch $2^{\Omega(\sqrt{n})}$.*

Diese untere Schranke ist nicht ausreichend um ⊕BP1s gegen wohlstrukturierte graphgesteuerte ⊕BP1s abzugrenzen. Es ist nicht bekannt ob die charakteristische Funktion jedes linearen Codes mit ⊕BP1s darstellbar ist. Allerdings beobachtete Jukna [Juk95], dass die Negation der charakteristischen Funktion jedes linearen Codes von nichtdeterministischen OBDDs polynomieller Größe mit existenziellen

Akzeptierungsmodus dargestellt werde kann. Es gilt, P-∨OBDD $\subset$ P-∨BP. Wigderson [Wig94] zeigte P-∨BP = P-⊕BP. Somit folgt, weil die Negation für ⊕BP einfach ist, die charakteristische Funktion jedes linearen Codes ist mit ⊕BPs polynomieller Größe darstellbar. Das heißt,

$$\text{P-wohlstrukturiertes-graphgesteuertes-}\oplus\text{BP1} \subsetneq \text{P-}\oplus\text{BP} \ .$$

2.2 Graphgesteuerte ⊕BP1s

2.2.1 Ein Kriterium für untere Schranken

Die algebraische Charakterisierung der Komplexität von Funktionen, die mit wohlstrukturierten graphgesteuerten ⊕BP1s dargestellt werden, wird ermöglicht durch die Zerlegung der Knotenmenge in disjunkte Level und die auf diesen Levels vorhandene Ordnungsrelation. Bei graphgesteuerten ⊕BP1s gibt es keine Levelstruktur. Das Branchingprogramm muss als Ganzes betrachtet werden. Allerdings ist auch hier das Ziel die minimale Anzahl der Knoten eines graphgesteuerten ⊕BP1s mit Graphordnung $\mathcal{G}$ für die Boolesche Funktion f so zu charakterisieren, dass nur die Graphordnung $\mathcal{G}$ und die Funktion f Invarianten der Beschreibung sind.
Sei $\mathcal{B} = (V, E)$ ein graphgesteuertes ⊕BP1 auf der Variablenmenge $\{X_1, \ldots, X_n\}$ mit Graphordnung $\mathcal{G} = (V_0, E_0)$, das die Boolesche Funktion f darstellt. Es werden zwei Vektorräume über $\mathbb{F}_2$, dem Primkörper mit 2 Elementen, definiert. Der Raum $\mathbb{B}(\mathcal{B})$ wird von den Resultatsfunktionen aller Knoten von $\mathcal{B}$ aufgespannt.

$$\mathbb{B}(\mathcal{B}) := \text{span}_{\mathbb{F}_2}\{\text{Res}_v \mid v \in V\}$$

Jeder von der Quelle s_0 ausgehende Pfad π in $\mathcal{G}$ bestimmt kanonisch eine Teilbelegung $\alpha(\pi)$ der Variablenmenge $\{X_1, \ldots, X_n\}$. Der Raum $\mathbb{B}_{\mathcal{G}}(f)$ wird von allen Unterfunktionen $f|_{\alpha(\pi)}$ aufgespannt, wobei π jeder Pfad von der Quelle zu jedem Knoten der Graphordnung $\mathcal{G}$ ist.

$$\mathbb{B}_{\mathcal{G}}(f) := \text{span}_{\mathbb{F}_2} \bigcup_{w \in V_0} \{f|_{\alpha(\pi)} \mid \pi \text{ ist Pfad in } \mathcal{G} \text{ von } s_0 \text{ nach } w\} \ .$$

Satz 2.6. *Sei $\mathcal{B}$ ein graphgesteuertes ⊕BP1 auf der Variablenmenge $\{X_1, \ldots, X_n\}$ mit Graphordnung $\mathcal{G}$, dass die Boolesche Funktion $f \in \text{Bool}_n$ darstellt.*
Dann gilt,

$$|\mathcal{B}| \geq \dim_{\mathbb{F}_2} \mathbb{B}_{\mathcal{G}}(f) \ .$$

Beweis. Aus der Definition von $\mathbb{B}(\mathcal{B})$ folgt, $|\mathcal{B}| \geq \dim_{\mathbb{F}_2} \mathbb{B}(\mathcal{B})$.
Sei $f|_{\alpha(\pi)}$ ein beliebiges erzeugendes Element des Vektorraumes $\mathbb{B}_{\mathcal{G}}(f)$. Weil $\mathcal{G}$ eine Graphordnung für $\mathcal{B}$ ist, aktiviert die Teilbelegung $\alpha(\pi)$ in $\mathcal{B}$ Pfade von der Quelle zu Knoten $v_1, \ldots, v_\nu$. Das heißt, $f|_{\alpha(\pi)} = \sum_{j=1}^{\nu} \mathrm{Res}_{v_j}$. Daraus folgt, $\mathbb{B}_{\mathcal{G}}(f)$ ist Untervektorraum von $\mathbb{B}(\mathcal{B})$. □

Die Dimension eines Vektorraumes kann nach unten abgeschätzt werden durch die Kardinalität einer Menge von linear unabhängigen Elementen des Vektorraumes. Damit ergibt sich ein für die Anwendung gut geeignetes Korollar.

Korollar 2.7. *Seien $\pi_1, \ldots, \pi_\nu$ an der Quelle beginnende Pfade in $\mathcal{G}$. Seien $\alpha_1, \ldots, \alpha_\nu$ die zu diesen Pfaden gehörigen Teilbelegungen.*
Sind die Unterfunktionen $f|_{\alpha_1}, \ldots, f|_{\alpha_\nu}$ linear unabhängig, dann gilt,

$$|\mathcal{B}| \geq \nu \ .$$

Beweis. Der von $\{f|_{\alpha_1}, \ldots, f|_{\alpha_\nu}\}$ aufgespannte Raum ist ein Untervektorraum von $\mathbb{B}_{\mathcal{G}}(f)$ und hat Dimension ν. □

Aus der Definition 2.2 für streng k-gemischte Funktionen kann durch Anwendung von Korollar 2.7 eine schwache untere Schranke für solche Funktionen gefolgert werden.

Korollar 2.8. *Die Größe jedes graphgesteuerten ⊕BP1s mit Graphordnung $\mathcal{G}$, das eine streng k-gemischte Funktion darstellt, ist nach unten beschränkt durch $2^k/|\mathcal{G}|$.*

In den nachfolgenden Abschnitten werden stärkere untere Schranken gezeigt, die unabhängig von der Größe der Graphordnung sind.
Für die unteren Schranken in den folgenden Abschnitten wird eine spezielle Notation verwendet. Eine Eingabe $\alpha \in \{0,1\}^n$ für eine Funktion auf n Variablen $\{X_1, \ldots, X_n\}$ ist gleichzusetzen mit einer Belegung der Variablen mit Werten aus $\{0,1\}$. Die Belegung einer Variable X_k wird mit $\alpha(X_k)$ bezeichnet. Die Einschränkung einer Funktion durch die Belegung α einer Teilmenge der Variablenmenge $\{X_1, \ldots, X_n\}$ wird wie üblich mit $f|_\alpha$ bezeichnet. Allerdings wird die eingeschränkte Funktion $f|_\alpha$ trotzdem als Funktion auf $\{X_1, \ldots, X_n\}$ betrachtet.

2.2.2 Untere Schranke für lineare Codes

Satz 2.9. *Sei $C \in \mathbb{F}_2^n$ ein linearer Code mit Minimal-Abstand d und $C^\perp$ dessen dualer Code mit Minimal-Abstand $d^\perp$.*
Dann ist die Größe jedes graphgesteuerten ⊕BP1s, das die charakteristische Funktion f_C von C darstellt, nach unten beschränkt durch $2^{\min(d,d^\perp)-1}$.

Beweis. Sei $\mathcal{B}$ ein graphgesteuertes ⊕BP1 mit Graphordnung $\mathcal{G}$, das die Funktion f_C darstellt. Sei W_0 die Menge der Knoten der Graphordnung $\mathcal{G}$, die genau k Knoten von der Quelle entfernt sind, wobei $k := \min\{d, d^\perp\} - 1$ ist. Das heißt, auf jeden Pfad π in $\mathcal{G}$, der von der Quelle zu einem Knoten aus W_0 führt, werden genau k Variablen getestet. Für $m := 2^k$ seien $\alpha_1, \ldots, \alpha_m$ die Teilbelegungen der Variablen aus $\{X_1, \ldots, X_n\}$, die zu diesen Pfaden gehören.
Im Folgenden wird benutzt, dass der Code C Minimal-Abstand k hat und k-universell ist.
Entsprechend Korollar 2.7 reicht es zu zeigen, dass die Funktionen $f|_{\alpha_1}, \ldots, f|_{\alpha_m}$ linear unabhängig sind. Sei $\{f|_{\alpha_{i_1}}, \ldots, f|_{\alpha_{i_\mu}}\}$ eine nichtleere Teilmenge von $\{f|_{\alpha_1}, \ldots, f|_{\alpha_m}\}$. Ohne Beschränkung der Allgemeinheit werden die Teilbelegungen $\alpha_{i_1}, \ldots, \alpha_{i_\mu}$ im Folgenden mit $\alpha_1, \ldots, \alpha_\mu$ bezeichnet. Dann bleibt zu zeigen $\sum_{i=1}^{\mu} f|_{\alpha_i} \neq 0$.
Sei $D \subseteq \{X_1, \ldots, X_n\}$ die Menge der von α_1 belegten Variablen. Für jede Teilbelegung α_1' der Variablenmenge $\{X_1, \ldots, X_n\} \backslash D$, dem Komplement der von α_1 belegten Variablen, kann ein Vektor $(\alpha_1, \alpha_1') := (b_1, \ldots, b_n) \in \{0,1\}^n$ wie folgt definiert werden. Für $j = 1, \ldots, n$ gilt,

$$b_j := \begin{cases} \alpha_1(X_j) & \text{wenn } \alpha_1(X_j) \text{ definiert ist ;} \\ \alpha_1'(X_j) & \text{wenn } \alpha_1'(X_j) \text{ definiert ist .} \end{cases}$$

Weil C k-universell ist, gibt es ein α_1', sodass $\beta := (\alpha_1, \alpha_1')$ ein Codewort von C ist. Daraus folgt, $f|_{\alpha_1}(\beta) = 1$. Nun wird gezeigt, das für $1 < i \leq \mu$ gilt $f|_{\alpha_i}(\beta) = 0$.
Sei der Vektor $\beta^{(i)} = (b_1^{(i)}, \ldots, b_n^{(i)})$ wie folgt definiert.

$$b_j^{(i)} := \begin{cases} \alpha_i(X_j) & \text{wenn } \alpha_i(X_j) \text{ definiert ist ;} \\ b_j & \text{sonst .} \end{cases}$$

Dann gilt, $f|_{\alpha_i}(\beta) = f(\beta^{(i)})$. Der Abstand zwischen β und $\beta^{(i)}$ ist kleiner gleich k, daraus folgt die Behauptung. □

Eine konkrete untere Schranke liefert, wie in Korollar 2.5, die charakteristische Funktion des binären Reed-Muller-Codes $R(r, \ell)$ mit Ordnung r der Länge $n = 2^\ell$.

Korollar 2.10. *Sei $n = 2^\ell$ und $r = \lfloor n/\ell \rfloor$. Dann ist die Größe jedes graphgesteuerten ⊕BP1s, das die charakteristische Funktion des Codes $R(r, \ell)$ darstellt, nach unten beschränkt durch $2^{\Omega(\sqrt{n})}$.*

2.2.3 Untere Schranke für Permutationsmatrizen

Eine $n \times n$ Matrix über $\{0,1\}$ ist eine Permutationsmatrix, wenn jede Zeile und jede Spalte genau eine 1 enthalten. Die Funktion $\text{PERM}_{n\times n}$ akzeptiert genau die Eingaben

der Länge n^2, die Permutationsmatrizen sind. Um untere Schranken für diese Funktion zu beweisen, werden in diesem Abschnitt Ideen von Krause [Kra88], sowie Krause, Meinel und Waack [KMW91] übernommen.

Satz 2.11. *Die Größe jedes graphgesteuerten ⊕BP1s, das die Funktion* $\mathrm{PERM}_{n\times n}$ *darstellt, ist nach unten beschränkt durch* $\Omega(n^{-1/2}\ 2^n)$.

Beweis. Sei $\mathcal{B}$ ein graphgesteuertes ⊕BP1 mit Graphordnung $\mathcal{G}$, dass die Funktion $f := \mathrm{PERM}_{n\times n}$ auf der Variablenmenge $\mathcal{X} := \{X_{i,j} \mid 1 \leq i,j \leq n\}$ darstellt.
Man betrachtet die $n!$ Permutationsmatrizen $\alpha = (\alpha_{i,j})_{1\leq i,j\leq n}$ und die zugehörigen Pfade in der Graphordnung $\mathcal{G}$. Nachdem genau $n/2$ Variablen mit 1 belegt wurden, werden die Pfade abgeschnitten.
Sei

$$A_1 := \{\alpha_1, \ldots, \alpha_\nu\}$$

die Menge der Teilbelegungen der Variablen aus $\mathcal{X}$, die zu den abgeschnittenen Pfaden gehören.
Für eine Teilbelegung α sei $C(\alpha)$ die Menge der Zeilenindizes i, sodass gilt, $\alpha(X_{i,j}) = 1$ für irgendeinen Spaltenindex j. Analog sei $R(\alpha)$ die Menge der Spaltenindizes j, sodass gilt, $\alpha(X_{i,j}) = 1$ für irgendeinen Zeilenindex i. Aus der Definition folgt, $|C(\alpha)| = |R(\alpha)| = n/2$.
Es werden Teilmengen A von A_1 betrachtet, sodass für zwei unterschiedliche Teilbelegungen $\alpha, \beta \in A$ gilt, $C(\alpha) \neq C(\beta)$ oder $R(\alpha) \neq R(\beta)$. In [Kra88] wird gezeigt, dass es so eine Teilmenge der Größe

$$|A| \geq \frac{n!}{\left(\frac{n}{2}!\right)^2}$$

gibt. Für den Beweis werden zwei Mengen $C, R \subset \{1, \ldots, n\}$ von Zeilen- und Spaltenindizes mit $|C| = |R| = n/2$ festgelegt. Damit liegen auch die Komplementmengen $\overline{C} := \{1, \ldots, n\}\backslash C$ und $\overline{R} := \{1, \ldots, n\}\backslash R$ fest. Sei $\alpha = (\alpha', \alpha'')$ eine Permutationsmatrix, wobei α' und α'' jeweils genau $n/2$ Variablen auf 1 setzen. Gilt $C(\alpha') = C$ und $R(\alpha') = R$ muss auch $C(\alpha'') = \overline{C}$ und $R(\alpha'') = \overline{R}$ gelten. Es gibt genau $(n/2)!$ verschiedene Teilbelegungen α' mit $C(\alpha') = C$ und $R(\alpha') = R$. Ebenso $(n/2)!$ Teilbelegungen α'' für die Komplementmengen. Daraus folgt, jedes Paar C, R wird von $\left(\frac{n}{2}!\right)^2$ der $n!$ möglichen Permutationsmatrizen erzeugt. Das heißt, es gibt $n!/((n/2)!)^2$ viele Permutationsmatrizen, sodass für zwei beliebige (α', α'') und (β', β'') gilt, $C(\alpha') \neq C(\beta')$ oder $R(\alpha') \neq R(\beta'')$, egal in welcher Reihenfolge die Variablen durchlaufen werden.
Sei $A = \{\alpha_1, \ldots, \alpha_m\}$ mit $m \geq n!/\left(\frac{n}{2}!\right)^2$ eine Menge von Teilbelegungen, deren Existenz gerade bewiesen wurde. Die Teilbelegungen in der Menge A erfüllen

die Voraussetzungen von Korollar 2.7, wenn für eine beliebige Teilmenge $A' = \{\beta_1, \ldots, \beta_\lambda\} \subseteq A$ gilt, $\sum_{i=1}^{\lambda} f|_{\beta_i} \neq 0$.
Nach Definition von A gibt es eine Teilbelegung β_1', sodass (β_1, β_1') eine Permutationsmatrix ist. Es gilt, $f|_{\beta_1}(0, \ldots, 0, \beta_1') = 1$, wobei $(0, \ldots, 0, \beta_1')$ wie folgt definiert ist.

$$(0, \ldots, 0, \beta_1')_{k,\ell} := \begin{cases} \beta_1(X_{k,\ell}) & \text{wenn } \beta_1(X_{k,\ell}) \text{ definiert ist ;} \\ 0 & \text{sonst .} \end{cases}$$

Bleibt zu zeigen, $f|_{\beta_i}(0, \ldots, 0, \beta_1') = 0$ für alle $i > 1$.
Beweis durch Widerspruch. Angenommen es gibt einen Index $i > 1$ mit $f|_{\beta_i}(0, \ldots, 0, \beta_1') = 1$. Dann gibt es eine Permutationsmatrix, die nur die Einsen enthält, die von β_i und β_1' gesetzt sind. Aber daraus folgt $C(\beta_i) = C(\beta_1)$ und $R(\beta_i) = R(\beta_1)$, sonst gäbe es in der Matrix $\gamma = (\gamma_{k,\ell})_{1 \leq k,\ell \leq n^2}$, definiert durch

$$\gamma_{k,\ell} := \begin{cases} \beta_i(X_{k,\ell}) & \text{wenn } \beta_i(X_{k,\ell}) \text{ definiert ist ;} \\ (0, \ldots, 0, \beta_1')_{k,\ell} & \text{sonst ,} \end{cases}$$

eine Zeile oder Spalte, die keine 1 enthält. Das ist ein Widerspruch.
Die Behauptung folgt jetzt mit Anwendung von Stirlings Abschätzung auf $|A|$. □

2.2.4 Untere Schranke für $\mathbb{1}^C_{n\times n} \vee \mathbb{1}^{R+}_{n\times n}$

Gesucht wird eine Funktion, die mit ⊕BP1s polynomieller Größe darstellbar ist, aber für die es kein graphgesteuertes ⊕BP1 polynomieller Größe gibt. Die ausgewählte Funktion ist wie $\mathrm{PERM}_{n\times n}$ auf $n \times n$ Matrizen mit Einträgen aus $\{0, 1\}$ definiert und hat Ähnlichkeit mit der Funktion $\mathrm{ROW}_{n\times n} + \mathrm{COL}_{n\times n}$, die von Bollig und Wegener in [BW97] untersucht wird (vergleiche Definition in Abschnitt 2.3). Die Funktion ist definiert durch $\mathbb{1}^C_{n\times n} \vee \mathbb{1}^{R+}_{n\times n}$, wobei

$$\mathbb{1}^C_{n\times n}(\mathcal{X}) := \begin{cases} 1 & \text{wenn jede Spalte von } \mathcal{X} \text{ genau eine 1 enthält ;} \\ 0 & \text{sonst .} \end{cases}$$

$$\mathbb{1}^{R+}_{n\times n}(\mathcal{X}) := \begin{cases} 1 & \text{wenn } (n-1) \text{ Zeilen von } \mathcal{X} \text{ genau eine 1 enthalten} \\ & \text{und eine Zeile enthält genau zwei 1'sen ;} \\ 0 & \text{sonst .} \end{cases}$$

Für beide Funktionen gibt es deterministische OBDDs mit in n quadratischer Größe, allerdings mit unterschiedlichen Variablenordnungen. Durch Verschmelzen der Quellen erhält man ein ⊕BP1 für $\mathbb{1}^C_{n\times n} \vee \mathbb{1}^{R+}_{n\times n}$. Das funktioniert, weil die Eingabemengen, die jeweils von $\mathbb{1}^C_{n\times n}$ und $\mathbb{1}^{R+}_{n\times n}$ akzeptiert werden, disjunkt sind.

Jetzt muss noch gezeigt werden, dass die Funktion nicht mit graphgesteuerten ⊕BP1s polynomieller Größe darstellbar ist.

Satz 2.12. *Die Größe jedes graphgesteuerten ⊕BP1s, das die Funktion* $\mathbb{1}^C_{n\times n} \vee \mathbb{1}^{R+}_{n\times n}$ *darstellt, ist nach unten beschränkt durch* $\Omega(n^{-1/4}\ 2^{n/2})$.

Beweis. Sei $\mathcal{B}$ ein graphgesteuertes ⊕BP1 mit Graphordnung $\mathcal{G}$, dass die Funktion $f := \mathbb{1}^C_{n\times n} \vee \mathbb{1}^{R+}_{n\times n}$ auf der Variablenmenge $\mathcal{X} := \{X_{i,j} \mid 1 \leq i,j \leq n\}$ darstellt. Ohne Beschränkung der Allgemeinheit sei n gerade.
Analog zum Beweis von Satz 2.11 betrachtet man die $n!$ Permutationsmatrizen $\alpha = (\alpha_{i,j})_{1\leq i,j\leq n}$ und die zugehörigen Pfade in der Graphordnung $\mathcal{G}$. Nachdem genau $n/2$ Variablen mit 1 belegt wurden, werden die Pfade abgeschnitten.
Die zu den abgeschnittenen Pfaden gehörigen Teilbelegungen der Variablen aus $\mathcal{X}$, werden in der Menge $A_1 := \{\alpha_1, \ldots, \alpha_\nu\}$ zusammengefasst.
Als Erstes wird die Anzahl der Teilbelegungen in A_1 abgeschätzt. Es gibt $\binom{i+n/2}{n/2}$ viele Teilbelegungen, i Variablen auf 0 und $n/2$ Variablen auf 1 zu setzen. Nur die Teilbelegungen, bei denen die Anzahl der auf 0 gesetzten Variablen höchstens $n^2 - n$ ist, können zu Permutationsmatrizen vervollständigt werden. Daraus folgt,

$$|A_1| = \nu \leq \sum_{i=0}^{n^2-n} \binom{i+n/2}{n/2} \leq n^2 \cdot \binom{n^2-n/2}{n/2} \leq n^2 \cdot (2en)^{n/2} \ .$$

Ohne Beschränkung der Allgemeinheit sei $A_2 := \{\alpha_1, \ldots, \alpha_\mu\}$ die Menge der Teilbelegungen aus A_1, die zu mindestens zwei Permutationsmatrizen vervollständigt werden können. Bei der Abschätzung der Größe von A_2 hilft folgende Überlegung. Für $i = 1, \ldots, (n/2)!$ sei μ_i die Anzahl der Teilbelegungen aus A_1, die zu i Permutationsmatrizen vervollständigt werden können. Mit $\mu = \sum_{i=2}^{(n/2)!} \mu_i$ und $\mu_1 = \nu - \mu$ gilt,

$$n! = \sum_{i=1}^{(n/2)!} i \cdot \mu_i \leq \nu - \mu + \mu \cdot (n/2)! \ .$$

Daraus folgt,

$$|A_2| = \mu \geq \frac{n! - \nu}{(n/2)! - 1} \ .$$

Wie im Beweis der unteren Schranke für Permutationsmatrizen (Satz 2.11), sei $C(\alpha)$ die Menge der Indizes der Zeilen, in denen die Teilbelegung α einen Eintrag mit 1 belegt und $R(\alpha)$ genauso die Menge der entsprechenden Spaltenindizes.
Ohne Beschränkung der Allgemeinheit sei $A := \{\alpha_1, \ldots, \alpha_\kappa\}$ die größte Teilmenge von A_2, sodass für zwei verschiedene Teilbelegungen $\alpha_i, \alpha_j \in A$ gilt, $C(\alpha_i) \neq C(\alpha_j)$ oder $R(\alpha_i) \neq R(\alpha_j)$.

Da höchstens $(n/2)!$ viele Teilbelegungen $\alpha \in A_2$ das gleiche Paar (C, R) haben gilt,

$$|A| = \kappa \geq \frac{n! - \nu}{(n/2)! \cdot (n/2)!} \ .$$

Es gilt, $\nu \leq (n/2)! \cdot (n/2)!$ für hinreichend großes n. Daraus folgt,

$$|A| = \kappa \geq \frac{n!}{(n/2)! \cdot (n/2)!} - o(1) \ .$$

Sei A_C (A_R) die größte Teilmenge von A, die folgende Bedingung erfüllt. Für zwei beliebige ungleiche Teilbelegungen $\alpha, \alpha' \in A_C$ ($\alpha, \alpha' \in A_R$) gilt, $C(\alpha) \neq C(\alpha')$ ($R(\alpha) \neq R(\alpha')$). Durch Abzählen erhält man aus $|A_C| < \sqrt{|A|}$ ($|A_R| < \sqrt{|A|}$) folgt $|A_R| \geq \sqrt{|A|}$ ($|A_C| \geq \sqrt{|A|}$). Diese beiden Fälle werden unterschieden.
Fall $|A_C| \geq \sqrt{|A|}$. Für eine beliebige Teilmenge $A'_C = \{\beta_1, \ldots, \beta_\lambda\}$ von A_C ist zu zeigen

$$\sum_{i=1}^{\lambda} f|_{\beta_i} \neq 0 \ . \tag{2.1}$$

Aufgrund der Auswahl von A gibt es eine Teilbelegung β'_1, sodass (β_1, β'_1) eine Permutationsmatrix ist. Das heißt, $f|_{\beta_1}(0, \ldots, 0, \beta'_1) = 1$, wobei die Matrix $(0, \ldots, 0, \beta'_1)$ genau wie in Satz 2.11 definiert ist

$$(0, \ldots, 0, \beta'_1)_{k,\ell} := \begin{cases} \beta'_1(x_{k,\ell}) & \text{wenn } \beta'_1(X_{k,\ell}) \text{ definiert ist ;} \\ 0 & \text{sonst .} \end{cases}$$

Aber für $i > 1$ gilt $f|_{\beta_i}(0, \ldots, 0, \beta'_1) = 0$, denn nach Definition von A_C ist $C(\beta_i) \neq C(\beta_1)$. Sei $\gamma := (\gamma_{k,\ell})_{1 \leq k,\ell \leq n^2}$ eine Matrix, definiert durch

$$\gamma_{k,\ell} := \begin{cases} \beta_i(X_{k,\ell}) & \text{wenn } \beta_i(X_{k,\ell}) \text{ definiert ist ;} \\ (0, \ldots, 0, \beta'_1)_{k,\ell} & \text{sonst .} \end{cases} \tag{2.2}$$

Dann gibt es eine Zeile in γ, die keine 1 enthält. Daraus folgt, für alle $i > 1$ ist $f|_{\beta_i}(0, \ldots, 0, \beta'_1) = f(\gamma) = 0$ und somit gilt (2.1).
Fall $|A_R| \geq \sqrt{|A|}$. Wie im ersten Fall ist für eine beliebige Teilmenge $A'_R = \{\beta_1, \ldots, \beta_\lambda\}$ von A_R zu zeigen $\sum_{i=1}^{\lambda} f|_{\beta_i} \neq 0$. Jede Teilbelegung β_i aus A'_R kann zu zwei Permutationsmatrizen (β_i, β'_i) und (β_i, β''_i) vervollständigt werden. Das heißt, man kann eine Belegung β^*_i konstruieren, sodass für die Matrix (β_i, β^*_i) gilt, $n - 1$ Zeilen enthalten genau eine 1 und eine Zeile enthält genau zwei 1'sen.
Daraus folgt, $f|_{\beta_1}(0, \ldots, 0, \beta^*_1) = 1$. Für $i > 1$ gilt, $f|_{\beta_1}(0, \ldots, 0, \beta^*_i) = 0$. Analog zum ersten Fall gibt es in der Matrix γ, definiert wie in (2.2), eine Zeile, die keine 1 enthält. Die Behauptung folgt mit Anwendung von Stirlings Abschätzung auf $\sqrt{|A|}$. □

Aus den Überlegungen am Anfang dieses Abschnittes zur Darstellung von $\mathbb{1}^{C}_{n\times n} \vee \mathbb{1}^{R+}_{n\times n}$ mit einem polynomiell großen ⊕BP1 und dem Beweis einer superpolynomiellen unteren Schranke für die Größe von graphgesteuerten ⊕BP1s für diese Funktion folgt,

$$\text{P-graphgesteuertes-}\oplus\text{BP1} \subsetneq \text{P-}\oplus\text{BP1} \ .$$

2.3 Einordnung von graphgesteuerten ⊕BP1s

In Abschnitt 2.1.3 wird gezeigt, dass wohlstrukturierte graphgesteuerte ⊕BP1s eine geringere Darstellungskraft haben als ⊕BPs, bei polynomieller Beschränkung der Größe. Das gilt mit Satz 2.9 auch für graphgesteuerte ⊕BP1s. Krause [Kra88] hat gezeigt, dass sich die Negation der Funktion $\text{PERM}_{n\times n}$ mit ∨OBDDs polynomieller Größe darstellen lässt. Somit ergibt die Argumentation aus Abschnitt 2.1.3, übertragen auf die Funktion $\text{PERM}_{n\times n}$, ebenfalls eine geringere Darstellungskraft von graphgesteuerten ⊕BP1s gegenüber ⊕BPs, bei polynomieller Beschränkung der Größe.
Die Funktion $\mathbb{1}^{C}_{n\times n} \vee \mathbb{1}^{R+}_{n\times n}$ ermöglicht es in Abschnitt 2.2.4 zu zeigen, dass ⊕BP1s polynomieller Größe eine größere Darstellungskraft haben als graphgesteuerte ⊕BP1s polynomieller Größe.
Es ist keine Funktion bekannt, die die Komplexitätsklassen von graphgesteuerten und wohlstrukturierten graphgesteuerten ⊕BP1s trennt. Überlegungen zu diesem Problem finden sich in Abschnitt 2.4.
Die in der Einleitung zu diesem Kapitel gestellte Frage, ob sich die Komplexitätsklassen von graphgesteuerten ⊕BP1s und ⊕OBDDs unterscheiden, kann mit einem Ergebnis von Sieling [Sie99] für die Funktionen MSA_n und INDEX-EQ_n beantwortet werden.
Die Boolesche Funktion MSA_n, mit $n = 2^k$, ist definiert auf der Variablenmenge $\mathcal{X} = (X_0, \ldots, X_{n-1})$. Die Variablenmenge ist zerlegt in k $s \times s$ Matrizen $M_0, \ldots, M_{k-1}$ mit $s = \lfloor \sqrt{n/k} \rfloor$ und die Menge der restlichen Variablen. Die Matrix M_i hat als Einträge $X_{is^2}, \ldots, X_{(i+1)s^2}$. Sei $a_i = 1$ genau dann, wenn M_i eine Zeile enthält, die nur aus Einsen besteht und sei $|a|$ die natürliche Zahl, die von $a = (a_0, \ldots, a_{k-1})$ auf kanonische Weise dargestellt wird.

$$\text{MSA}_n(\mathcal{X}) = \begin{cases} X_0 & \text{wenn } |a| = 0 \ ; \\ X_{|a|} + X_0 & \text{wenn } |a| > 0 \ . \end{cases}$$

Die Funktion INDEX-EQ_n ist definiert auf $n = 3N/2$ Variablen mit $N = 2^k$, wobei k eine Zweierpotenz ist. Die Variablen $X_0, \ldots, X_{N-1}$ werden interpretiert als Speicher und die Variablen $X_N, \ldots, X_{n-1}$ werden interpretiert als $N/(2\log_2 N)$ Zeiger, jeder bestehend aus $\log_2 N$ Bits. Sei $m = N/(4\log_2 N)$ und seien $a(1), \ldots, a(m), b(1), \ldots, b(m)$ Bezeichner für die Werte der Zeiger. Dann gilt,

INDEX-EQ$_n(X_0, \ldots, X_{n-1}) = 1$ genau dann, wenn die folgenden Bedingungen erfüllt sind.

- für alle $i \in \{1, \ldots, m\}$ gilt $X_{a(i)} = X_{b(i)}$;
- $a(1) < \ldots < a(m)$ und $b(1) < \ldots < b(m)$;
- $a(m) < b(1)$ oder $b(m) < a(1)$.

Satz 2.13. *Die Funktion* MSA$_n$ *kann mit polynomiell großen ⊕OBDDs dargestellt werden, aber nur mit exponentiell großen deterministischen BP1s.*
Die Funktion INDEX-EQ$_n$ *kann mit polynomiell großen deterministischen BP1s dargestellt werden, aber nur mit exponentiell großen ⊕OBDDs.*

Sowohl für ein ⊕OBDD als auch für ein deterministisches BP1 kann eine Graphordnung erzeugt werden, sodass die Diagramme wohlstrukturierte graphgesteuerte ⊕BP1s mit dieser Graphordnung sind. Für ⊕OBDDs ist die Graphordnung eine Liste von Knoten, die entsprechend der Variablenordnung markiert sind. Ein deterministisches BP1 ist im Wesentlichen durch sich selbst gesteuert (siehe [SW95]). Das heißt, wohlstrukturierte graphgesteuerte ⊕BP1s verallgemeinern sowohl ⊕OBDDs als auch deterministischen BP1s.
Bevor mit den in diesem Kapitel vorgestellten Techniken untere Schranken für graphgesteuerte ⊕BP1s gezeigt wurden, konnten nur untere Schranken für erheblich stärker eingeschränkte ⊕BP1s bewiesen werden.
Savický und Sieling [SS00] zeigen eine exponentielle untere Schranke für die Größe von $(\oplus, k)$-BPs, die Zeiger-Funktionen darstellen. $(\oplus, k)$-BPs sind nichtdeterministische BP1s mit Parity-Akzeptierungsmodus und genau einem nichtdeterministischen Knoten mit Ausgrad k als Quelle. Dieses Modell ist nur wegen der Vollständigkeit aufgelistet und wird nicht weiter betrachtet.
Bollig [Bol01] beweist die erste untere Schranke für die Größe von eingeschränkten ⊕BP1s mit unbeschränkter Anzahl von nichtdeterministischen Knoten. Das untersuchte Modell sind graphgesteuerte ⊕BP1s mit Baumordnung, genannt baumgesteuerte ⊕BP1s. Eine Graphordnung ist eine Baumordnung, wenn der zugrundeliegende Graph, nach entfernen der Senke und ersetzen aller Mehrfachkanten durch einfache Kanten, ein Baum polynomieller Größe ist. Zum Beispiel ist die Graphordnung in Abbildung 1.3 eine Baumordnung.

Satz 2.14. *Sei $\mathcal{B}$ ein baumgesteuertes ⊕BP1 mit Baumordnung $\mathcal{G}$. Die von $\mathcal{B}$ dargestellte Funktion sei das mittlere Bit der Multiplikation natürlicher Zahlen.*
Dann hat $\mathcal{B}$ die Größe $2^{\Omega(n/\log_2 n)}$.

Die Forderung nach einer Baumordnung ist eine so starke Einschränkung, dass der Beweis für untere Schranken von baumgesteuerten ⊕BP1s auf ⊕OBDDs zurückgeführt

werden kann. Sei $\mathcal{B}$ ein baumgesteuertes ⊕BP1 mit polynomieller Baumordnung $\mathcal{G}$, das die Funktion f darstellt. Die Baumordnung $\mathcal{G}$ hat polynomielle Größe, deshalb kann man einen Pfad π von der Quelle in die Senke finden, sodass unter den n durchlaufen Knoten nur logarithmisch viele nicht den gleichen 0- und 1-Nachfolger haben. Setzt man in $\mathcal{B}$ die Variablen dieser Knoten auf die durch den Pfad π festgelegten Werte, erhält man ein ⊕OBDD für die entsprechende Unterfunktion von f. Daraus folgt unmittelbar, ist jede Unterfunktion von f, für die logarithmisch viele Variablen gesetzt sind, nicht mit ⊕OBDDs polynomieller Größe darstellbar, dann gibt es auch kein baumgesteuerten ⊕BP1 für diese Funktion.
Mit $n = k^2$ sei

$$\text{INDEX-EQ}_n^{\vee}\left((X_{i,j})_{1\le i,j\le k}\right) := \bigvee_{i=1}^{k} \text{INDEX-EQ}(X_{i,1},\ldots,X_{i,k}) \ .$$

Durch Anwendung der beschriebenen Technik aus [Bol01] kann man zeigen, dass jedes baumgesteuerte ⊕BP1 für die Funktion $\text{INDEX-EQ}_n^{\vee}$ superpolynomielle Größe hat.
Einerseits kann die Funktion $\text{INDEX-EQ}_n^{\vee}$ mit deterministischen BP1s polynomieller Größe dargestellt werden. Andererseits kann jedes ⊕OBDD als baumgesteuertes ⊕BP1 betrachtet werden. Somit ist die Funktion MSA_n mit baumgesteuerten ⊕BP1s polynomieller Größe darstellbar. Daraus folgt, die Komplexitätsklassen von baumgesteuerten ⊕BP1s und deterministischen BP1s sind gegenseitig nicht ineinander enthalten.
Zum Beweis für untere Schranken von deterministische OBDDs benutzt Bryant [Bry91] Unterscheidungsmengen (fooling sets). Diese kann man auch für den Beweis von unteren Schranken für ⊕OBDDs verwenden. Sieling [Sie94] konstruiert eine Unterscheidungsmenge der Größe $\Omega(2^k)$ für die Funktion ALU_n mit $n = k^2 + 1$. Sei $\mathcal{X} := (X_{i,j})_{1\le i,j\le k}$, dann ist $\text{ALU}_{k^2+1}(Z,\mathcal{X}) := \text{ite}(Z, \text{ROW}_{k\times k}(\mathcal{X}), \text{COL}_{k\times k}(\mathcal{X}))$, die Funktion, die für $Z = 1$ als Ergebnis $\text{ROW}_{k\times k}(\mathcal{X})$ und sonst $\text{COL}_{k\times k}(\mathcal{X})$ liefert. Wobei gilt,

$$\text{COL}_{n\times n}(\mathcal{X}) := \begin{cases} 1 & \text{wenn eine Spalte von } \mathcal{X} \text{ nur Einsen enthält ;} \\ 0 & \text{sonst .} \end{cases}$$

$$\text{ROW}_{n\times n}(\mathcal{X}) := \begin{cases} 1 & \text{wenn eine Zeile von } \mathcal{X} \text{ nur Einsen enthält ;} \\ 0 & \text{sonst .} \end{cases}$$

Ein ⊕OBDD für ALU_{k^2+1} hat mindestens superpolynomielle Größe. Aber sowohl für $\text{COL}_{k\times k}$ als auch für $\text{ROW}_{k\times k}$ gibt es deterministische OBDDs quadratischer Größe. Werden diese deterministischen OBDDs jeweils 0- und 1-Nachfolger einer mit Z markierten Quelle, erhält man ein baumgesteuertes ⊕BP1 für ALU_{k^2+1}. Daraus folgt, bei polynomiell beschränkter Größe, ist die Darstellungskraft von baumgesteuerten ⊕BP1s größer als die von ⊕OBDDs.

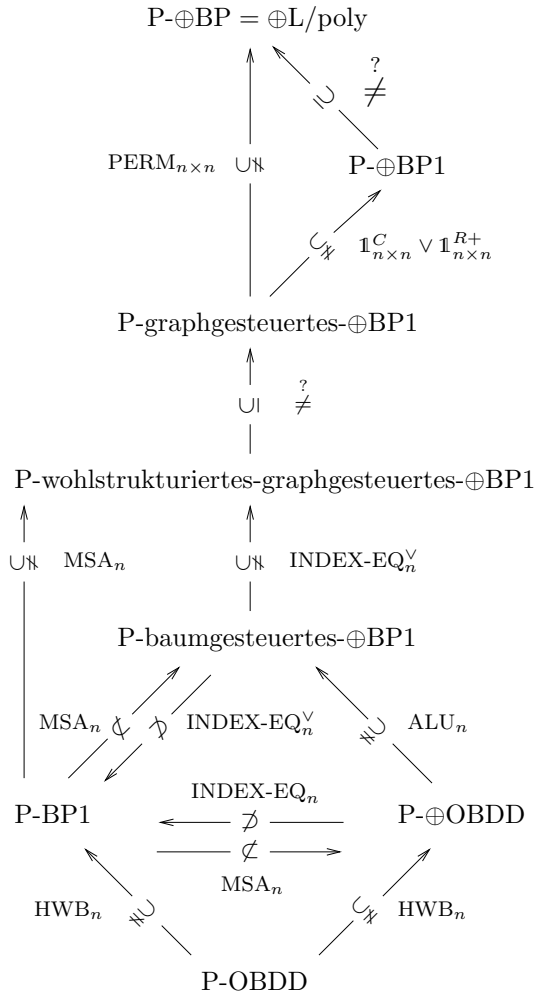

Abbildung 2.1: Hierarchie ausgewählter Komplexitätsklassen.

Bryant [Bry91] untersucht deterministische OBDDs und die Funktion $\mathrm{HWB}_n(X_1, \ldots, X_n) := X_s$ mit $X_0 := 0$ und $s := X_1 + \ldots + X_n$, wobei die Addition über den natürlichen Zahlen durchgeführt wird.

Satz 2.15. *Jedes deterministische OBDD, das die Funktion* HWB_n *darstellt, hat die Größe* $\Omega(2^{n/5})$.

Die Funktion HWB_n kann aber von deterministischen BP1s ([SW95]) und ⊕OBDDs ([GM96]) polynomieller Größe dargestellt werden. Das heißt, sowohl deterministischen BP1s als auch ⊕OBDDs sind eine echte Verallgemeinerung von deterministischen OBDDs.

In Abbildung 2.1 findet sich eine Zusammenstellung der Überlegungen dieses Abschnitts.

2.4 Spezielle Eigenschaften von graphgesteuerten ⊕BP1s

Ein Branchingprogramm kann auf sehr natürliche Weise eingeschränkt werden, indem man fordert, dass die Variablen für eine beliebige Eingabe in einer, für diese Eingabe genau festgelegten, Reihenfolge getestet werden. Die Anwendung dieser Einschränkung auf ⊕BP1s wird im Folgenden genauer beschrieben.
Eine *Variablenordnung* ist eine Anordnung der Elemente der Variablenmenge, das heißt, eine Folge in der jede Variable genau einmal vorkommt. Ein Berechnungspfad unter einer Eingabe ist mit einer Variablenordnung *verträglich*, wenn die Folge der Knotenmarkierungen des Berechnungspfades eine Teilfolge der Variablenordnung ist. Ein ⊕BP1 genügt der Bedingung *eine Variablenordnung pro Eingabe*, wenn für jede Eingabe eine Variablenordnung festgelegt ist und alle Berechnungspfade unter einer beliebigen Eingabe mit der für diese Eingabe festgelegten Variablenordnung verträglich sind.
Eine Graphordnung ist eine Möglichkeit für jede Eingabe eine Variablenordnung festzulegen. Offensichtlich genügt ein graphgesteuertes ⊕BP1 der Bedingung eine Variablenordnung pro Eingabe. Im nachfolgenden Satz wird gezeigt, dass diese Bedingung hinreichend ist, damit zu einem ⊕BP1 eine Graphordnung konstruiert werden kann.

Satz 2.16. *Sei $\mathcal{B}$ ein ⊕BP1 auf der Variablenmenge $\{X_1, \ldots, X_n\}$.*
$\mathcal{B}$ erfüllt die Bedingung eine Variablenordnung pro Eingabe genau dann, wenn es eine Graphordnung $\mathcal{G}$ gibt, sodass $\mathcal{B}$ ein graphgesteuertes ⊕BP1 mit Graphordnung $\mathcal{G}$ ist.

Beweis. Die Bedingung ist offensichtlich notwendig.
Sei $\mathcal{B}$ ein ⊕BP1 für das die Bedingung erfüllt ist. Es wird gezeigt, dass eine Variable X_i ausgewählt werden kann, sodass für jede Eingabe α die Berechnungspfade in $\mathcal{B}$ unter α mit einer Variablenordnung verträglich sind, die mit X_i beginnt. Dann kann eine Graphordnung $\mathcal{G}$ wie folgt konstruiert werden. Die Quelle wird mit X_i markiert. Für $\delta \in \{0, 1\}$ wird das Unterdiagramm $\mathcal{B}|_{X_i=\delta}$ berechnet, indem in $\mathcal{B}$ die Variable X_i auf δ gesetzt wird. Durch das Setzen von X_i wird die Bedingung eine Variablenordnung pro Eingabe in den Unterdiagrammen nicht verletzt. Somit kann für jedes Unterdiagramm wieder eine Variable X_{j_δ} gewählt werden, mit der alle Variablenordnungen beginnen können. Also wird für $\delta \in \{0, 1\}$ der δ-Nachfolger der Quelle von $\mathcal{G}$ mit X_{j_δ} markiert und in dieser Weise iteriert das Verfahren.
Angenommen für jede Variable X_i gibt es eine Eingabe α_i, sodass alle Variablenordnungen, die mit den Berechnungspfaden in $\mathcal{B}$ unter α_i verträglich sind,

nicht mit X_i beginnen. Dann gibt es für jedes X_i eine Eingabe α_i und einen Berechnungspfad p_i unter α_i auf dem eine Variable X_j $(i \neq j)$ vor X_i getestet wird. Durch Umnummerieren erhält man Eingaben $\alpha_1, \ldots, \alpha_\nu$ und Berechnungspfade $p_1, \ldots, p_\nu$, für die gilt,

- die Variable X_ν wird auf p_1 vor X_1 getestet;
- für $i = 2, \ldots, \nu$ ist X_{i-1} die erste Variable auf p_i und X_i wird auch auf p_i getestet.

Die Anzahl ν ist mindestens 2 und höchstens n. Die Folge $X_\nu, X_1, \ldots, X_{\nu-1}, X_\nu$ ist ein *Kreis*, der von den Eingaben $\alpha_1, \ldots, \alpha_\nu$ und den zugehörigen Berechnungspfaden $p_1, \ldots, p_\nu$ erzeugt wird.
Für $i = 1, \ldots, \nu$ sei S_i die Menge der Variablen, die auf p_i vor X_i getestet werden. Die Summe $\sum_{i=1}^{\nu} |S_i|$ ist das *Gewicht* eines Kreises.
Nun wird der Kreis $X_\nu, X_1, \ldots, X_{\nu-1}, X_\nu$ minimalen Gewichts betrachtet, der von den Eingaben $\alpha_1, \ldots, \alpha_\nu$ und den zugehörigen Berechnungspfaden $p_1, \ldots, p_\nu$ erzeugt wird. Aus der Minimalität des Kreises folgt, dass die Mengen $S_1, \ldots, S_\nu$ paarweise disjunkt sind. Somit ist die folgende Eingabe α wohldefiniert.

$$\alpha(X_k) := \begin{cases} \alpha_i(X_k) & \text{wenn } X_k \in S_i \text{ ;} \\ 0 & \text{sonst .} \end{cases}$$

Die Eingabe α erzeugt den Kreis $X_\nu, X_1, \ldots, X_{\nu-1}, X_\nu$. Daraus folgt, für α kann keine Variablenordnung angegeben werden, die mit allen Berechnungspfaden unter α verträglich ist. Das ist ein Widerspruch zu der Bedingung eine Variablenordnung pro Eingabe. □

Mit Satz 2.16 sind ⊕BP1s mit einer Variablenordnung pro Eingabe äquivalent zu graphgesteuerten ⊕BP1s mit beliebiger Graphordnung. Allerdings braucht bei der Angabe der Größe eines ⊕BP1s mit einer Variablenordnung pro Eingabe die Größe der Graphordnung nicht mitberücksichtigt werden. Das bedeutet, graphgesteuerte ⊕BP1s mit einer Graphordnung, deren Größe nicht polynomiell beschränkt ist, sind ein Spezialfall von ⊕BP1s polynomieller Größe mit einer Variablenordnung pro Eingabe. Für wohlstrukturierte graphgesteuerte ⊕BP1s führt eine in der Größe nicht polynomiell beschränkte Graphordnung nicht zu größerer Darstellungskraft. Diese Eigenschaft wird von Bollig, Waack und Woelfel [BWW02] gezeigt.

Lemma 2.17. *Sei $\mathcal{B}$ ein wohlstrukturiertes graphgesteuertes ⊕BP1 mit einer beliebigen Graphordnung.*
Dann kann eine Graphordnung $\mathcal{G}$ konstruiert werden, sodass $\mathcal{B}$ ein wohlstrukturiertes graphgesteuertes ⊕BP1 mit dieser Graphordnung $\mathcal{G}$ ist. Für die Größe der Graphordnung gilt $|\mathcal{G}| \leq 2n|\mathcal{B}|$.

Ein graphgesteuertes ⊕BP1s mit polynomieller Graphordnung kann in ein wohlstrukturiertes graphgesteuertes ⊕BP1 mit gleicher Graphordnung überführt werden. Wie in Abbildung 1.3 illustriert, werden dazu alle Knoten, die nicht eindeutig zu einem Knoten der Graphordnung gehören, auseinandergelegt. Ein Knoten eines graphgesteuerten ⊕BP1s gehört im schlechtesten Fall zu jedem Knoten der Graphordnung, somit ergibt sich folgende Abschätzung.

Lemma 2.18. *Sei $\mathcal{B}$ ein graphgesteuertes ⊕BP1s mit Graphordnung $\mathcal{G}$.*
Dann gibt es ein wohlstrukturiertes graphgesteuertes ⊕BP1 mit der gleichen Graphordnung $\mathcal{G}$ und Größe $\mathcal{O}(|\mathcal{B}| \cdot |\mathcal{G}|)$, das die gleiche Boolesche Funktion darstellt wie $\mathcal{B}$.

Aus Lemma 2.17 und Lemma 2.18 folgt, die Komplexitätsklassen von ⊕BP1s mit einer Variablenordnung pro Eingabe und wohlstrukturierten graphgesteuerten ⊕BP1s sind verschieden, wenn es eine Funktion f mit den folgenden beiden Eigenschaften gibt. Erstens, es gibt ein graphgesteuertes ⊕BP1 polynomieller Größe mit einer Graphordnung nicht polynomiell beschränkter Größe, das f darstellt. Zweitens, jedes graphgesteuerte ⊕BP1 mit einer polynomiellen Graphordnung, das f darstellt, hat nicht polynomielle Größe.
Aus diesem Ansatz ergibt sich folgende grundsätzliche Frage. Gibt es ein graphgesteuertes ⊕BP1 polynomieller Größe für das nur eine exponentiell große Graphordnung existiert? Ein ⊕BP1 mit dem diese Frage positiv beantwortet werden kann, ist in Abbildung 2.2 angegeben.
Sei $\mathcal{B}$ das in Abbildung 2.2 auf der Variablenmenge $\{X_1, \ldots, X_\lambda, Y_1, \ldots, Y_\lambda, Z\}$ definierte ⊕BP1. $\mathcal{B}$ erfüllt die Bedingung eine Variablenordnung pro Eingabe. Sei $\alpha \in \{0,1\}^n$ eine beliebige Eingabe. Sei I die Menge der Indizes der Variablen aus $\{X_1, \ldots, X_\lambda\}$, die von α mit 0 belegt werden und J die Indexmenge der Variablen, die mit 1 belegt werden. Das heißt, $\alpha(X_i) = 0$ für alle $i \in I$ und $\alpha(X_j) = 1$ für alle $j \in J$. Dann ist $(X_1, \ldots, X_\lambda, (Y_i)_{i\in I}, Z, (Y_j)_{j\in J})$ eine Variablenordnung, mit der alle Berechnungspfade unter der Eingabe α verträglich sind.
Nachfolgend wird die Größe jeder Graphordnung $\mathcal{G}$ für $\mathcal{B}$ abgeschätzt. Seien $\alpha, \alpha' \in \{0,1\}^n$ zwei Eingaben, die $X_1, \ldots, X_\lambda$ unterschiedlich belegen und für die gilt, $Y_1 = 1, \ldots, Y_\lambda = 1$ und $Z = 0$. Seien I, I' die Mengen der Indizes der Variablen, die von α, α' mit 0 belegt werden und J, J' die Indexmengen der Variablen, die mit 1 belegt werden. Daraus folgt, in $\mathcal{B}$ gibt es für jedes $i \in I$ einen Berechnungspfad unter α auf dem Y_i vor Z und für jedes $j \in J$ einen auf dem Z vor Y_i durchlaufen wird. Das gilt genauso für α' und I', J'. $\mathcal{B}$ ist ein graphgesteuertes ⊕BP1 mit Graphordnung $\mathcal{G}$. Somit werden auf dem eindeutigen Berechnungspfad in $\mathcal{G}$ unter α die Variablen aus $\{Y_i \mid i \in I\}$ vor und die Variablen aus $\{Y_j \mid j \in J\}$ nach Z durchlaufen. Und auf dem eindeutigen Berechnungspfad unter α' werden die $\{Y_i \mid i \in I'\}$ vor und die $\{Y_j \mid j \in J'\}$ nach Z durchlaufen. Die Eingabe α und α' belegen die $\{X_1, \ldots, X_\lambda\}$ unterschiedlich, das heißt, $I \neq I'$ und $J \neq J'$. Daraus folgt, es gibt in $\mathcal{G}$ zwei verschiedene mit Z

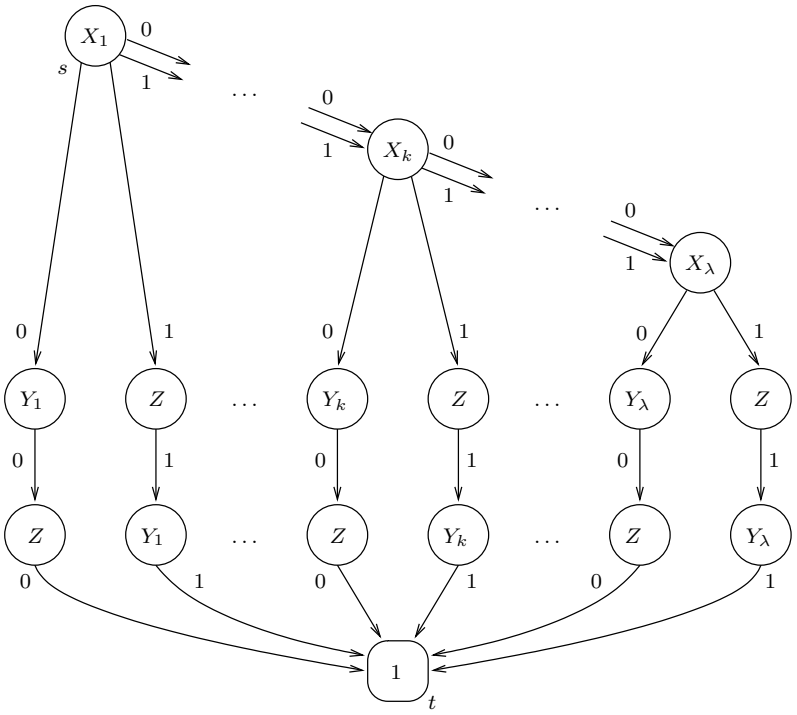

Abbildung 2.2: Graphgesteuertes ⊕BP1 mit exponentieller Graphordnung.

markierte Knoten. Das heißt, in $\mathcal{G}$ muss es für jede mögliche Belegung der $X_1, \ldots, X_\lambda$ paarweise verschiedene mit Z markierte Knoten geben.

Satz 2.19. *Es gibt ⊕BP1s polynomiell beschränkter Größe, die die Bedingung eine Variablenordnung pro Eingabe erfüllen, die, betrachtet als graphgesteuerte ⊕BP1s, eine Graphordnung exponentieller Größe haben müssen.*

Beweis. Sei $\mathcal{B}$ das ⊕BP1 aus Abbildung 2.2 betrachtet als graphgesteuertes ⊕BP1. Die Anzahl der Variablen ist $n = 2\lambda + 1$. Die Größe von $\mathcal{B}$ ist linear und jede Graphordnung für $\mathcal{B}$ hat mindestens die Größe $2^{\frac{n-1}{2}}$. □

Die Funktion $\bigoplus_{i=1}^{\lambda}(\neg X_i \wedge \neg Y_i \wedge \neg Z \oplus X_i \wedge Z \wedge Y_i)$, die von dem ⊕BP1 in Abbildung 2.2 dargestellt wird, kann nicht zur Trennung der Komplexitätsklassen von ⊕BP1s mit einer Variablenordnung pro Eingabe und wohlstrukturierten graphgesteuerten ⊕BP1s verwendet werden. Stellt man die Funktion nach $\neg Z \wedge \bigoplus_{i=1}^{\lambda}(\neg X_i \wedge \neg Y_i) \oplus Z \wedge \bigoplus_{i=1}^{\lambda}(X_i \wedge Y_i)$ um, ist leicht zu sehen, dass sich die Funktion mit der Variablenordnung $(Z, X_1, Y_1, \ldots, X_\lambda, Y_\lambda)$ sogar von einem ⊕OBDD linearer Größe darstellen lässt. Eine Funktion mit der die beiden Modelle getrennt werden können, ist noch nicht bekannt. Es scheint aber aussichtsreich, diese auf den Ansatz in Abbildung 2.2 aufzubauen.

Satz 2.19 liefert nicht nur einen Anhaltspunkt dafür, dass die Darstellungskraft von ⊕BP1s mit einer Variablenordnung pro Eingabe größer als die von wohlstrukturierten graphgesteuerten ⊕BP1s ist. Man gewinnt auch eine Einsicht über die Komplexität des Konsistenztests für graphgesteuerte ⊕BP1s. Der Konsistenztest besteht darin, für ein gegebenes ⊕BP1 zu entscheiden, ob es graphgesteuert ist. Bollig [Bol03] hat gezeigt, dass dieses Problem co-NP-vollständig ist. Dafür wird ein Algorithmus verwendet, der aus einer beliebigen Instanz des 3-SAT Problems ein ⊕BP1 konstruiert. Dieses ⊕BP1 ist genau dann graphgesteuert, wenn die Instanz nicht erfüllbar ist. Viele nicht erfüllbare Instanzen des 3-SAT Problems erzeugen ein graphgesteuertes ⊕BP1, das nur eine exponentiell große Graphordnung haben kann. Allerdings spielt das für den Beweis keine Rolle und findet in [Bol03] auch keine Erwähnung. Trotzdem ist diese Einsicht interessant. Es ist nicht ausgeschlossen, dass die Zeit, die für den Konsistenztest von graphgesteuerten ⊕BP1s benötigt wird, durch die Größe der Graphordnung beschränkt ist. Das heißt, möglicherweise kann der Konsistenztest von graphgesteuerten ⊕BP1s mit polynomieller Graphordnung in polynomieller Zeit durchgeführt werden.

3 Approximation mit $\oplus$OBDDs

In diesem Kapitel werden verschiedene Modelle zur Darstellung von Booleschen Funktionen durch Branchingprogramme mit einmaligen geordneten Tests und Parity-Akzeptierungsmodus ($\oplus$OBDDs) approximiert.
In Abschnitt 3.1 wird der Begriff der Approximation, wie er in diesem Kapitel benutzt wird, und die dazugehörige Notation eingeführt.
Als Hilfsmodell kommen in diesem Kapitel arithmetische OBDDs über verschiedenen Halbringen und Körpern zum Einsatz. Der Akzeptierungsmodus wird immer so gewählt, dass folgende Identifizierungen möglich sind. Sei $\mathbb{F}_2$ der Primkörper mit 2 Elementen. Arithmetische OBDDs über $\mathbb{F}_2$ ($\mathbb{F}_2$-OBDDs) werden als $\oplus$OBDDs aufgefasst. Sei $\mathbf{B}_0$ der Boolesche Halbring mit 2 Elementen. Ein $\mathbf{B}_0$-OBDD kann sowohl als nichtdeterministisches OBDD mit existenziellem ($\vee$OBDD) als auch mit universellem ($\wedge$OBDD) Akzeptierungsmodus aufgefasst werden.
In Abschnitt 3.2 wird gezeigt, wie arithmetische OBDDs über $\mathbb{F}_{2^\kappa}$, einem Oberkörper von $\mathbb{F}_2$ quasipolynomieller Ordnung, mit $\mathbb{F}_2$-OBDDs simuliert werden können. Hierzu werden spezielle Eigenschaften von endlichen Körpern ausgenutzt, unter anderem der sehr nützliche Frobenius-Automorphismus.
Abschnitt 3.3 beschreibt die Approximation von $\mathbf{B}_0$-OBDDs durch $\mathbb{F}_{2^\kappa}$-OBDDs. Dabei werden einige Techniken von Beimel und Gál [BG98] für arithmetische Branchingprogramme (BPs) auf arithmetische OBDDs übertragen.
Mit Hilfe der vorangegangenen Ergebnisse werden in Abschnitt 3.4 $\vee$OBDDs (und $\wedge$OBDDs) polynomieller Größe mit $\oplus$OBDDs quasipolynomieller Größe und umgekehrt quasipolynomiellem Fehler approximiert.
Ob und wie sich die in Abschnitt 3.4 beschriebene Approximation noch verbessern lässt ist Thema von Abschnitt 3.5. Es zeigt sich, dass eine Verbesserung der Größenordnung, sowohl des approximierenden $\oplus$OBDD als auch der Fehlerschranke, nicht möglich ist. Weiterhin kann auch bei Betrachtung der absoluten Größen weder die Qualität der Approximation (Unterabschnitt 3.5.1) noch der Platzbedarf der Simulation (Unterabschnitt 3.5.2) wesentlich verbessert werden.
In Abschnitt 3.6 wird gezeigt, dass sich disjunktive Formen, in deren Unbestimmte $\oplus$OBDDs quasipolynomieller Größe eingesetzt werden, durch $\oplus$OBDDs quasipolynomieller Größe approximieren lassen. Dabei kommt eine Idee von Smolensky [Smo87] für die Approximation der Disjunktion zum Einsatz.
Mit der Frage nach unteren Schranken für approximierende $\oplus$OBDDs beschäftigt sich Abschnitt 3.7. Die Verknüpfung dieses Problems mit der Matrixsteifigkeit wird

aufgezeigt. Daraus folgt, dass zur Zeit keine Technik für den Beweis von unteren Schranken für approximierende ⊕OBDDs zur Verfügung steht.

3.1 Notation

Seien $\mathcal{C}$ und $\mathcal{D}$ arithmetische OBDDs über, möglicherweise verschiedenen, Halbringen auf derselben Variablenmenge $\{X_1, \ldots, X_n\}$.
$\mathcal{C}$ und $\mathcal{D}$ sind *äquivalent* , wenn sie mit $(\neq 0)$- und $(=0)$-Akzeptierungsmodus dieselbe Funktion darstellen. Das ist genau dann der Fall, wenn die Urbilder der 0 der Gewichtsfunktionen von $\mathcal{C}$ und $\mathcal{D}$ übereinstimmen. Das heißt, $\mathcal{C}$ und $\mathcal{D}$ sind äquivalent, wenn für alle Eingaben $\alpha \in \{0,1\}^n$ gilt, $\text{weight}_{\mathcal{C}}(\alpha) = 0$ genau dann, wenn $\text{weight}_{\mathcal{D}}(\alpha) = 0$.
$\mathcal{D}$ *approximiert* $\mathcal{C}$ *mit durch* $0 < \varepsilon \leq 1$ *beschränkten Fehler* , man sagt auch $\mathcal{D}$ ist eine *ε-Approximation* von $\mathcal{C}$, wenn es eine Teilmenge A aller möglichen Eingaben mit Mächtigkeit mindestens $(1-\varepsilon)\,2^n$ gibt, sodass $\mathcal{C}$ und $\mathcal{D}$ für alle Eingaben $\alpha \in A$ äquivalent sind. Das heißt, es gibt $A \subseteq \{0,1\}^n$ mit $|A| \geq (1-\varepsilon)\,2^n$, sodass für alle $\alpha \in A$ gilt, $\text{weight}_{\mathcal{C}}(\alpha) = 0$ genau dann, wenn $\text{weight}_{\mathcal{D}}(\alpha) = 0$.
$\mathcal{D}$ ist eine *einseitige* ε-Approximation von $\mathcal{C}$, wenn $\mathcal{D}$ eine ε-Approximation von $\mathcal{C}$ ist und zusätzlich $\mathcal{C}$ und $\mathcal{D}$ auf dem Urbild der 0 der Gewichtsfunktion von $\mathcal{C}$ gleich sind. Das heißt, für alle α mit $\text{weight}_{\mathcal{C}}(\alpha) = 0$ gilt, $\text{weight}_{\mathcal{D}}(\alpha) = 0$.
Diese Terminologie gilt genauso für zwei Boolesche Funktionen f, g auf der Variablenmenge $\{X_1, \ldots, X_n\}$. Die Funktion g ist eine ε-Approximation der Funktion f, wenn sich g und f höchstens auf $\varepsilon\,2^n$ Eingaben unterscheiden. Und g ist eine einseitige ε-Approximation von f, wenn beide Funktionen auf dem Urbild der 0 von f übereinstimmen. Weiterhin bedeutet, g ist eine *umgekehrt einseitige* ε-Approximation von f, dass beide Funktion auf dem Urbild der 1 von f übereinstimmen.

3.2 Simulation von $\mathbb{F}_{2^K}$-OBDDs mit ⊕OBDDs

Eine wesentliche Erkenntnis für arithmetische OBDDs über endlichen Körpern beschreibt das folgende Lemma.

Lemma 3.1. *Sei $\mathbb{F}_{2^K}$ ein Oberkörper von $\mathbb{F}_2$ der Ordnung 2^K. Sei $\mathcal{C}$ ein $\mathbb{F}_{2^K}$-OBDD der Größe $|\mathcal{C}|$.*
Dann gibt es ein äquivalentes $\mathbb{F}_{2^K}$-OBDD $\mathcal{D}$ der Größe $|\mathcal{D}| = |\mathcal{C}|^K$, dessen Gewichtsfunktion nur Werte aus $\{0,1\} \subseteq \mathbb{F}_{2^K}$ annimmt. Das heißt, für alle $\alpha \in \{0,1\}^n$ gilt,

$$\begin{aligned} \text{weight}_{\mathcal{C}}(\alpha) = 0 \quad &\Longrightarrow \quad \text{weight}_{\mathcal{D}}(\alpha) = 0 \ , \\ \text{weight}_{\mathcal{C}}(\alpha) \neq 0 \quad &\Longrightarrow \quad \text{weight}_{\mathcal{D}}(\alpha) = 1 \ . \end{aligned}$$

Beweis. Der kleine Fermatsche Satz besagt,

$$\omega^{2^K-1} = 1 \quad \text{für alle } \omega \in \mathbb{F}_{2^K} \backslash \{0\} \ .$$

Würde man auf $\mathcal{C}$ einfach $(2^K - 1)$-mal die Produktkonstruktion anwenden, bekäme man ein $\mathbb{F}_{2^K}$-OBDD der exponentiellen Größe $|\mathcal{C}|^{2^K-1}$. Allerdings gibt es einen eleganten Trick um dies zu verhindern.
Anstelle von $2^K - 1$ wird die 2-adische Darstellung benutzt, $2^K - 1 = \sum_{i=0}^{K-1} 2^i$. Ein arithmetisches OBDD über einem Körper der Charakteristik 2 kann leicht mit 2^i potenziert werden, indem man den so genannten Frobenius-Automorphismus benutzt.

$$(\omega_1 + \omega_2)^{2^i} = \omega_1^{2^i} + \omega_2^{2^i} \quad \text{für alle } \omega_1, \omega_2 \in \mathbb{F}_{2^K}$$

Das heißt, dass Potenzieren mit 2^i ist verträglich mit der Addition. Somit kann für $i = 0, \ldots, K-1$ das $\mathbb{F}_{2^K}$-OBDD $\mathcal{C}^{2^i}$ durch Potenzieren aller Kantengewichte von $\mathcal{C}$ mit 2^i erzeugt werden. Anschließend wird $\mathcal{D} := \prod_{i=0}^{K-1} \mathcal{C}^{2^i}$ mit der Produktkonstruktion berechnet. □

Dieses Ergebnis wird nun mit einer Beobachtung von Beimel und Gál [BG98] kombiniert.
Der Körper $\mathbb{F}_{2^K}$ ist isomorph zu $\mathbb{F}_2[\xi]/(Q)$, dem Körper der Polynome $P \in \mathbb{F}_2[\xi]$ reduziert modulo einem irreduziblen Polynom $Q \in \mathbb{F}_2[\xi]$ vom Grad K. Das heißt, der Grad aller Polynome ist kleiner K. Eine Gewichtsfunktion, die in $\mathbb{F}_2[\xi]/(Q)$ abbildet, kann somit durch ein K-Tupel von Gewichtsfunktion, die in $\mathbb{F}_2$ abbilden, simuliert werden. Die i-te Gewichtsfunktion berechnet den Koeffizienten von ξ^i, in der zu simulierenden Gewichtsfunktion.
In [BG98] wird gezeigt, wie diese Überlegung auf ein arithmetisches BP über einem Körper der Ordnung 2^K angewendet wird. Um die Gewichtsfunktion des arithmetischen BPs zu zerlegen, wird die Gewichtsfunktion jedes Knotens zerlegt. Das heißt, um einen Koeffizienten zu simulieren, wird ein arithmetisches BP über einem Körper der Ordnung 2 mit K-facher Größe erzeugt. Diese Konstruktion lässt sich direkt auf arithmetische OBDDs übertragen. Es werden nur Knoten kopiert und, entsprechend dem Distributivgesetz, mit den Kopien der Nachfolger des zu simulierenden Knotens neu verdrahtet. Die Ordnung wird dadurch nicht verändert.

Lemma 3.2. *Sei $\mathbb{F}_{2^K}$ ein Oberkörper von $\mathbb{F}_2$ der Ordnung 2^K und $\mathcal{C}$ ein $\mathbb{F}_{2^K}$-OBDD über $\mathbb{F}_{2^K}$ der Größe $|\mathcal{C}|$.*
Dann gibt es ein äquivalentes $\mathbb{F}_2$-OBDD $\mathcal{D}$ der Größe $|\mathcal{D}| = K \cdot |\mathcal{C}|^K$.

Beweis. $\mathcal{C}$ wird mit Lemma 3.1 durch ein äquivalentes $\mathbb{F}_{2^K}$-OBDD $\mathcal{C}'$ der Größe $|\mathcal{C}|^K$ simuliert, dessen Gewichtsfunktion nur auf $\{0, 1\}$ abbildet. Dieses wiederum wird simuliert durch ein $\mathbb{F}_2$-OBDD $\mathcal{D}$, dessen Gewichtsfunktion den letzten Koeffizienten (das Absolutglied) der Gewichtsfunktion von $\mathcal{C}'$ berechnet. $\mathcal{D}$ ist äquivalent zu $\mathcal{C}$, denn die Gewichtsfunktion von $\mathcal{C}'$ bildet nur nach $\{0, 1\}$ ab. □

Beimel und Gál [BG98] simulieren $\mathbb{F}_{2^K}$-BPs mit $\mathbb{F}_2$-BPs. Für jeden der K Koeffizienten, der Gewichtsfunktion des $\mathbb{F}_{2^K}$-BPs, wird ein $\mathbb{F}_2$-BP konstruiert. Diese werden dann mit einem logischen Oder verknüpft. Das heißt, die Gewichtsfunktion ist 0 genau dann, wenn alle Koeffizienten 0 sind, wie gefordert.
Das logische Oder kann, mit deMorgans Gesetz, durch Komplement und logische Und ersetzt werden. Wendet man diesen Ansatz auf $\mathbb{F}_2$-OBDDs an, ergibt sich ein simulierendes $\mathbb{F}_2$-OBDD der Größe $(K \cdot 2)^K$. Damit erzeugt Lemma 3.2 ein etwas kleineres simulierendes $\mathbb{F}_2$-OBDD als die in [BG98] angewandte Methode.
Umgekehrt führt die Anwendung von Lemma 3.2 auf $\mathbb{F}_2$-BPs zu keiner Verbesserung. Über $\mathbb{F}_2$ ist die Bildung des Komplements einfach und für BPs wird das logische Und durch Hintereinanderstellen erreicht.
Bei Körpern größerer Charakteristik ist der Aufwand für die Bildung des Komplements von der Charakteristik abhängig. Für solche Körper wird in [BG98] eine Konstruktion für das logische Oder angegeben, die effizienter als die Anwendung von deMorgans Gesetz ist. Auch hier führt der Weg in Lemma 3.2 zu einem etwas kleineren simulierenden arithmetischen BP.

3.3 Approximation von $\mathbf{B}_0$-OBDDs durch $\mathbb{F}_{2^K}$-OBDDs

Im folgenden Abschnitt werden Techniken aus [BG98] für arithmetische BPs auf arithmetische OBDDs angewendet.
Jedes $\mathbf{B}_0$-OBDD $\mathcal{C}$ wird zu einem $\mathbb{F}_2$-OBDD $\mathcal{D}$, indem man die Kantengewichte als Werte aus $\mathbb{F}_2$ betrachtet und Addition und Multiplikation in $\mathbb{F}_2$ durchführt. Da die Multiplikation in $\mathbf{B}_0$ und $\mathbb{F}_2$ identisch ist, haben alle Berechnungspfade über beiden Körpern dasselbe Gewicht. Allerdings unterscheiden sich die Ergebnisse der Addition. Zwar gilt für alle Eingaben $\alpha \in \{0,1\}^n$ mit $\text{weight}_{\mathcal{C}}(\alpha) = 0$ auch $\text{weight}_{\mathcal{D}}(\alpha) = 0$. Aber es kann Eingaben α geben für die gilt, $\text{weight}_{\mathcal{C}}(\alpha) = 1$ aber $\text{weight}_{\mathcal{D}}(\alpha) = 0$, wenn unter α eine gerade Anzahl vollständiger Berechnungspfade mit Gewicht 1 in die Senke führt.
Diese Methode macht also keinen Fehler über dem Urbild der 0, aber der Fehler über dem Urbild der 1 kann beliebig groß werden. Um den Fehler einzuschränken, wählt man einen Körper mit Charakteristik 2 aber größerer Ordnung.
Sei $\mathcal{C}$ ein $\mathbf{B}_0$-OBDD der Größe S mit T Kanten $e_1, \ldots, e_T$. Sei $\mathbb{F}_{2^K}[Z_1, \ldots, Z_T]$ der Polynomring über $\mathbb{F}_{2^K}$. In $\mathcal{C}$ wird für $i = 1, \ldots, T$ das Gewicht ω_{e_i} der Kante e_i, durch das Polynom $\omega_{e_i} \cdot Z_i$ ersetzt. So erhält man ein $\mathbb{F}_{2^K}[Z_1, \ldots, Z_T]$-OBDD $\mathcal{C}'$. Aus $\mathcal{C}'$ wird für jede Belegung $r := (r_1, \ldots, r_T) \in \mathbb{F}_{2^K}^T$ ein $\mathbb{F}_{2^K}$-OBDD $\mathcal{D}_r$, indem man das Kantengewicht (Polynom) jeder Kante für die Belegung r auswertet und das Ergebnis jeweils als neues Kantengewicht setzt.

Bleibt, den richtigen Punkt $r \in \mathbb{F}_{2^K}^T$ zu finden. Dazu dient folgende Überlegung. Für alle $\alpha \in \{0,1\}^n$ gilt, $W_\alpha := \text{weight}_{\mathcal{C}'}(\alpha) \in \mathbb{F}_{2^K}[Z_1, \ldots, Z_T]$ ist ein multilineares Polynom. Der Totalgrad von W_α ist nach oben beschränkt durch n, die maximale Länge eines vollständigen Berechnungspfades. Es ist wohlbekannt, dass ein Polynom, das nicht das Nullpolynom ist, nur wenige Nullstellen hat (siehe Anhang B). Zur Abschätzung der Anzahl der Nullstellen eines Polynoms mit bekanntem Totalgrad, wird das folgende Lemma von Schwarz [Sch80] und Zippel [Zip79] benutzt (entspricht Korollar B.2).

Lemma 3.3. *Sei $\mathbb{F}_{2^K}$ ein beliebiger endlicher Körper mit Ordnung 2^K. Sei $P \in \mathbb{F}_{2^K}[Z_1, \ldots, Z_T]$ ein Polynom mit Totalgrad höchstens n, das nicht das Nullpolynom ist. Wähle $(r_1, \ldots, r_T) \in \mathbb{F}_{2^K}^T$ zufällig und gleichverteilt.*
Dann gilt,

$$\Pr[P(r_1, \ldots, r_T) = 0] \leq \frac{n}{2^K} \ .$$

Sei $\varepsilon_{n,K} := n/2^K$. Angewendet auf das Polynom $W_\alpha(Z_1, \ldots, Z_T)$ besagt Lemma 3.3, für jedes $\alpha \in \{0,1\}^n$ mit $\text{weight}_{\mathcal{C}}(\alpha) \neq 0$ gibt es mindestens $(1 - \varepsilon_{n,K})|\mathbb{F}_{2^K}|^T$ Belegungen $r := (r_1, \ldots, r_T) \in \mathbb{F}_{2^K}^T$, sodass gilt, $W_\alpha(r_1, \ldots, r_T) \neq 0$.
Durch doppeltes Abzählen (Schubfachprinzip) über α und r folgt, es gibt ein $r^* \in \mathbb{F}_{2^K}^T$, sodass gilt, $W_\alpha(r^*) \neq 0$ für mindestens $(1 - \varepsilon_{n,K})2^n$ Eingaben α mit $\text{weight}_{\mathcal{C}}(\alpha) \neq 0$.
Also ist das $\mathbb{F}_{2^K}$-OBDD $\mathcal{D} := \mathcal{D}_{r^*}$ eine $\varepsilon_{n,K}$-Approximation des $\mathbf{B}_0$-OBDD $\mathcal{C}$.
Das Ergebnis dieses Abschnittes wird in nachstehendem Lemma zusammengefasst.

Lemma 3.4. *Sei $\varepsilon_{n,K} := n/2^K$ und $\mathbb{F}_{2^K}$ ein Oberkörper von $\mathbb{F}_2$ der Ordnung 2^K. Sei $\mathcal{X}_n := \{X_1, \ldots, X_n\}$ eine n-elementige Variablenmenge.*
Dann gibt es für jedes $\mathbf{B}_0$-OBDD $\mathcal{C}$ auf $\mathcal{X}_n$ ein $\mathbb{F}_{2^K}$-OBDD $\mathcal{D}$ auf $\mathcal{X}_n$ gleicher Größe, dass eine einseitige $\varepsilon_{n,K}$-Approximation von $\mathcal{C}$ ist.

3.4 Approximation von ∨OBDDs durch ⊕OBDDs

In diesem Abschnitt werden die Überlegungen aus Abschnitt 3.2 und Abschnitt 3.3 kombiniert, um ∨OBDDs (und ∧OBDDs) mit ⊕OBDDs zu approximieren.

Satz 3.5. *Für festes k sei $\varepsilon(n) := 1/2^{(\log_2 n)^k}$. Sei $\mathcal{X}_n := \{X_1, \ldots, X_n\}$ eine n-elementige Variablenmenge. Sei $\mathcal{C} = (\mathcal{C}_n)_{n>0}$ eine Familie von $\mathbf{B}_0$-OBDDs auf $\mathcal{X}_n$ mit in n quasipolynomiell beschränkter Größe.*
Dann gibt es eine Familie $\mathcal{D} = (\mathcal{D}_n)_{n>0}$ von $\mathbb{F}_2$-OBDDs auf $\mathcal{X}_n$ mit in n quasipolynomiell beschränkter Größe, sodass für alle n gilt, $\mathcal{D}_n$ ist eine einseitige $\varepsilon(n)$-Approximationen von $\mathcal{C}_n$.

Beweis. Für beliebiges n sei $\mathcal{C}_n$ ein $\mathbf{B}_0$-OBDD der Größe $S_n \leq 2^{(\log_2 n)^\ell}$. Verwendet man $K := \lceil (\log_2 n)^k + \log_2 n \rceil$ in Lemma 3.4 gilt, $\varepsilon(n) = \varepsilon_{n,K}$. Weiterhin gibt es für jedes n ein $\mathbb{F}_{2^K}$-OBDD $\mathcal{C}'_n$ der Größe S_n, das eine einseitige $\varepsilon(n)$-Approximation von $\mathcal{C}_n$ ist. Simuliert man diese Familie von $\mathbb{F}_{2^K}$-OBDDs entsprechend Lemma 3.2, erhält man eine Familie von $\mathbb{F}_2$-OBDDs der Größe $K \cdot S_n^K = 2^{(\log_2 n)^{\ell+k} k \log_2(\log_2 n)} = 2^{(\log_2 n)^{\mathcal{O}(1)}}$. □

Für $* \in \{\vee, \wedge, \oplus\}$ sei Q-$*$OBDD die Klasse der Booleschen Funktionen, die von $*$OBDDs quasipolynomieller Größe dargestellt werden können. Dann ergibt sich aus Satz 3.5 folgendes Korollar.

Korollar 3.6. *Sei $f = (f_n)_{n>0} \in$ Q-$\vee$OBDD und für festes k sei $\varepsilon(n) = 1/2^{(\log_2 n)^k}$. Dann gibt es $g = (g_n)_{n>0} \in$ Q-$\oplus$OBDD, sodass für alle n gilt, g_n ist eine einseitige $\varepsilon(n)$-Approximation von f_n.*

Das Ergebnis dieses Abschnitts lässt sich auch auf $\wedge$OBDDs übertragen. Wird ein $\wedge$OBDD, das die Funktion f darstellt, als $\vee$OBDDs betrachtet, stellt es die komplementäre Funktion $(1 - f)$ dar. Das heißt, Korollar 3.6 gilt auch für Funktionen $f_n \in$ Q-$\wedge$OBDD. Allerdings werden diese mit umgekehrt einseitig beschränktem Fehler approximiert.
Es ist nur die in Satz 3.5 angegebene Richtung der Approximation bekannt. Für die Approximation von $\mathbb{F}_2$-OBDDs durch $\mathbf{B}_0$-OBDDs gibt es keinen erfolgversprechenden Ansatz.

3.5 Verbessern der Approximation von $\vee$OBDDs

Die in Abschnitt 3.4 approximierten nichtdeterministischen OBDDs können Funktionen darstellen, für die es nur $\oplus$OBDDs superpolynomieller Größe gibt. Beispielsweise können $\wedge$OBDDs die Funktion $\mathrm{PERM}_{n\times n}$ und $\vee$OBDDs die komplementäre Funktion $\overline{\mathrm{PERM}}_{n\times n}$ mit polynomieller Größe darstellen (siehe [Kra88]). Wohingegen jedes $\oplus$OBDD für diese Funktionen superpolynomielle Größe hat (vergleiche Satz 2.11). Daraus folgt, dass $\vee$OBDDs und $\wedge$OBDDs polynomieller Größe nicht durch $\oplus$OBDDs derselben Größenordnung simuliert werden können.
Mit demselben Argument lässt sich zeigen, die Größenordnung der Fehlerschranke optimal ist. Angenommen es gibt einen Algorithmus, der jedes $\mathbf{B}_0$-OBDD $\mathcal{C}$ durch ein $\mathbb{F}_2$-OBDD $\mathcal{C}'$ approximiert, mit Fehlern auf nur quasipolynomiell vielen Eingaben. Dann kann in der Größenordnung des Fehlers leicht ein $\mathbb{F}_2$-OBDD $\mathcal{C}''$ konstruiert werden, dessen Gewichtsfunktion genau für die Fehler aus dem Urbild der 0 der Gewichtsfunktion von $\mathcal{C}$ den richtigen Wert 0 liefert und sonst einen Wert ungleich 0. Genauso kann ein $\mathbb{F}_2$-OBDD $\mathcal{C}'''$ konstruiert werden, dessen Gewichtsfunktion für die Fehler, die nicht aus dem Urbild der 0 sind, einen Wert ungleich 0 liefert und sonst 0. Daraus folgt, $\mathcal{D} = \mathcal{C}' \cdot \mathcal{C}'' + \mathcal{C}'''$ simuliert $\mathcal{C}$ und hat quasipolynomielle Größe. Das ist ein Widerspruch.

Trotzdem gibt es zwei mögliche Ansätze Satz 3.5, im Rahmen der festgelegten Größenordnung, zu verbessern. Es könnte eine Verbesserung der Qualität der Approximation oder des Platzbedarfs der Simulation möglich sein. Das heißt, eine mögliche Verkleinerung einerseits der Fehlerschranke oder der Ordnung des Körpers und andererseits des approximierenden $\mathbb{F}_2$-OBDDs. Nachfolgend werden beide Ansätze untersucht.

3.5.1 Qualität der Approximation

Die Fehlerschranke in Satz 3.5 wird bestimmt durch die Abschätzung von Schwarz und Zippel (Lemma 3.3). Diese Abschätzung ist sehr allgemein und funktioniert für die Klasse aller Polynome mit beschränktem Totalgrad. Für diese allgemeine Klasse ist die Abschätzung allerdings optimal (siehe Anhang B). Aber über die Polynome W_α, die in Abschnitt 3.3 betrachtet werden, ist noch mehr bekannt, sie sind *multilinear*. Das heißt, jede Variable kommt höchstens in der ersten Potenz vor. Es ist leicht einzusehen, dass bei der Approximation, für ein gegebenes $\mathbb{F}_{2^K}$-OBDD und eine Eingabe α, jedes multilineare Polynom als W_α vorkommen kann. Das heißt, W_α hat außer dem beschränktem Totalgrad und der Multilinearität keine weiteren Eigenschaften.
Blum, Chandra und Wegman [BCW80] benutzen die Multilinearität für die Abschätzung der Anzahl der Nullstellen eines Polynoms. Kombiniert man diese Überlegung mit denen von Schwarz und Zippel erhält man folgendes Lemma (entspricht Korollar B.5), das hier gleich in der zur Problemstellung passenden Form angegeben wird.

Lemma 3.7. *Sei $\mathbb{F}_{2^K}$ ein Oberkörper von $\mathbb{F}_2$ der Ordnung 2^K. Sei $P \in \mathbb{F}_{2^K}[Z_1, \ldots, Z_T]$ ein multilineares Polynom, nicht das Nullpolynom, mit Totalgrad höchstens n. Sei $(r_1, \ldots, r_T) \in \mathbb{F}_{2^K}^T$ eine zufällige und gleichverteilte Belegung der Variablen $Z_1, \ldots, Z_T$.*
Dann gilt,

$$\Pr[P(r_1, \ldots, r_T) = 0] \leq \delta_{S,K} := 1 - \left(1 - \frac{1}{2^K}\right)^n$$

Wie in Anhang B zu sehen, lässt sich die Schranke aus Lemma 3.7, für multilineare Polynome mit beschränktem Totalgrad, nicht verbessern. Weiterhin gilt, mit $K \geq \log_2 n$,

$$\frac{1}{2^K} \leq 1 - \left(1 - \frac{1}{2^K}\right)^n < \frac{n}{2^K} \ .$$

Das heißt, für eine gegebene Fehlerschranke $1/(\log_2 n)^k$ muss ein arithmetisches OBDD über einem Körper der Ordnung mindestens $2^{(\log_2 n)^k}$ benutzt werden. Dieser Körper ist

etwas kleiner als der in Satz 3.5 benutzte Körper der Ordnung $2^{(\log_2 n)^k + \log_2 n}$. Daraus folgt, die Fehlerschranke kann durch die geringere Körpergröße etwas verbessert werden.

3.5.2 Platzbedarf der Simulation

In Satz 3.5 wird die untere Schranke für die Größe des resultierenden ⊕OBDDs bestimmt durch die Simulation eines $\mathbb{F}_{2^K}$-OBDDs mit einem $\mathbb{F}_2$-OBDD. Nachfolgend wird gezeigt, dass bei solchen Simulationen ein Aufblähen des Diagramms, mit dem Logarithmus der Körpergröße als Exponent, unvermeidbar ist.

Satz 3.8. *Sei $n = 2K$ und $\mathbb{F}_{2^K}$ ein Oberkörper von $\mathbb{F}_2$ der Ordnung 2^K.*
Dann gibt es eine Funktion f auf $\{X_1, \ldots, X_n\}$, die von einem $\mathbb{F}_{2^K}$-OBDD der Größe $2n+1$ mit $(\neq 0)$-Akzeptierungsmodus dargestellt werden kann. Aber jedes ⊕OBDD für f hat die Größe $\Omega(2^{n/2})$.

Beweis. $\mathbb{F}_{2^K}$ ist isomorph zu $\mathbb{F}_2[\xi]/(Q)$, dem Körper der Polynome $P \in \mathbb{F}_2[\xi]$ reduziert modulo einem irreduziblen Polynom $Q \in \mathbb{F}_2[\xi]$ von Grad K. Das heißt, es gibt eine Abbildung φ von $\{0,1\}^{n/2}$ nach $\mathbb{F}_{2^K}$, sodass für alle $\alpha, \beta \in \{0,1\}^{n/2}$ gilt,

$$f(\alpha,\beta) = \begin{cases} 0 & \text{wenn } \varphi(\alpha)\cdot\varphi(\beta) = 1; \\ 1 & \text{sonst.} \end{cases}$$

Die Variablenmenge $\{X_1, \ldots, X_n\}$ wird identifiziert mit $\{Y_0, \ldots, Y_{n/2-1}, Z_0, \ldots, Z_{n/2-1}\}$. Es ist leicht ein $\mathbb{F}_{2^K}$-OBDD $\mathcal{C}$ auf $Y = (Y_0, \ldots, Y_{n/2-1})$ und $Z = (Z_0, \ldots, Z_{n/2-1})$ mit Gewichtsfunktion $\text{weight}_{\mathcal{C}}(Y,Z) = \varphi(Y)\cdot\varphi(Z)$ zu konstruieren. $\mathcal{C}$ besteht aus der Senke t, sowie Verzweigungsknoten $v_0, \ldots, v_{n/2-1}$ und $w_0, \ldots, w_{n/2-1}$, wobei jeweils v_i mit Y_i und w_i mit Z_i markiert ist. Von jedem Knoten w_i führt eine 1-Kante mit Gewicht ξ^i in die Senke. Der Knoten w_0 erhält jedem Knoten $w_1, \ldots, w_{n/2-1}$, über eine 0- und 1-Kante jeweils mit Gewicht 1 als Nachfolger. Offensichtlich gilt, $\text{weight}_{w_0,t}(Y,Z) = \sum_{i=0}^{n/2-1} Z_i\xi^i = \varphi(Z)$. Jeder Knoten v_i erhält w_0 über eine 1-Kante mit Gewicht ξ^i als Nachfolger. Der Knoten $s = v_0$ ist die Quelle von $\mathcal{C}$ und erhält jeden Knoten $v_1, \ldots, v_{n/2-1}$ über eine 0- und eine 1-Kante jeweils mit Gewicht 1 als Nachfolger. Es gilt für $\delta \in \{0,1\}$,

$$\begin{aligned} \text{weight}_{\mathcal{C}}(Y,Z) &= \sum_{i=0}^{n/2-1} \left(Y_i\xi^i \cdot \text{weight}_{w_0,t}(Y,Z)\right) = \left(\sum_{i=0}^{n/2-1} Y_i\xi^i\right)\cdot\varphi(Z) \\ &= \varphi(Y)\cdot\varphi(Z) \ . \end{aligned}$$

Abschließend wird eine 0- und 1-Kante jeweils mit Gewicht (-1) von der Quelle in die Senke hinzugefügt ($(-1) = 1$ über $\mathbb{F}_2[\xi]$). Daraus folgt, für alle $\alpha, \beta \in \{0,1\}^{n/2}$

gilt, $\text{weight}_{s,t_\delta}(\alpha,\beta) = 0$ genau dann, wenn $\varphi(\alpha)\cdot\varphi(\beta) = 1$. Also ist $\mathcal{C}$ ein $\mathbb{F}_{2^K}$-OBDD linearer Größe, das mit $(\neq 0)$-Akzeptierungsmodus die Funktion f darstellt.
Für die untere Schranke wird der Zusammenhang zwischen Kommunikationskomplexität und der Größe von ⊕OBDDs benutzt (siehe [GM96], [Waa01]). Der $\mathbb{F}_2$-Rang einer Kommunikationsmatrix für eine Boolesche Funktion ist eine untere Schranke für jedes ⊕OBDD, das diese Funktion darstellt.
Es wird ein Schnitt durch die Variablenmenge zwischen Y und Z gelegt. Die resultierende Kommunikationsmatrix M wird nachfolgend beschrieben. M ist eine $2^K \times 2^K$ Matrix und hat für jedes $\alpha \in \mathbb{F}_{2^K}$ eine Spalte und für jedes $\beta \in \mathbb{F}_{2^K}$ eine Zeile. Der Eintrag in der Zelle $M_{\alpha,\beta} := f(\alpha,\beta)$ ist der Funktionswert für die Eingabe (α,β). $\mathbb{F}_{2^K}$ ist ein Körper, deshalb gibt es für jedes $\alpha \neq 0$ genau ein β mit $\varphi(\alpha)\cdot\varphi(\beta) = 1$ und umgekehrt. Das heißt, M enthält, mit Ausnahme der Zeile $M_{0,*}$ und der Spalte $M_{*,0}$, pro Zeile und pro Spalte genau eine 0. Das heißt, M hat Rang 2^{K-1}. Daraus folgt, die Größe eines ⊕OBDDs für f ist mindestens $2^{K-1} \in \Omega(2^{n/2})$. □

3.6 Approximation von disjunktiven Formen über ⊕OBDDs mit ⊕OBDDs

Eine Formel der Aussagenlogik über den Unbestimmten $Z_1,\ldots,Z_m$ ist eine disjunktive Form (DF), wenn sie eine Disjunktion (Verknüpfung mit $\vee$) von Konjunktionstermen ist. Ein Konjunktionsterm wird ausschließlich durch die Konjunktion (Verknüpfung mit $\wedge$) von Literalen gebildet. Literale sind dabei nichtnegierte (Z_i) oder negierte ($\neg Z_i$) Unbestimmte. Eine DF hat also die Gestalt

$$\bigvee_{i\in I}\bigwedge_{j\in J_i}(\neg)Z_j \ .$$

Disjunktiven Formen über ⊕OBDDs sind DFs, in deren Unbestimmte ⊕OBDDs über $\{X_1,\ldots,X_n\}$ eingesetzt werden.
Will man nun DFs über ⊕OBDDs mit einem ⊕OBDD simulieren, muss schon zur Berechnung sowohl der Konjunktionsterme aus auch der Disjunktion der Konjunktionsterme die Produktmethode angewendet werden. Das heißt, die Größe des simulierenden ⊕OBDDs ist exponentiell in der Größe und Anzahl der Konjunktionsterme. Damit ist bei vorgegebener, nicht exponentieller, Größenordnung die Klasse der simulierbaren DFs stark eingeschränkt.
Auch hier scheint eine Approximation mit $\mathbb{F}_{2^K}$-OBDDs, die anschließend mit $\mathbb{F}_2$-OBDDs simuliert werden, aussichtsreich. Allerdings benötigt man dafür eine Approximation der Verknüpfungen $\vee$ und $\wedge$. In [Smo87] findet sich ein passender Ansatz, er wird hier unseren Bedürfnissen entsprechend angegeben.

Lemma 3.9. *Sei $\mathbb{F}_{2^K}$ ein Oberkörper von $\mathbb{F}_2$ der Ordnung 2^K. Seien $\mathcal{C}_1, \ldots, \mathcal{C}_m$ $\mathbb{F}_{2^K}$-OBDDs. Sei $\mathcal{C} := \bigvee_{i=1}^m \mathcal{C}_i$, das heißt, für alle $\alpha \in \{0,1\}^n$ gilt,* $\text{weight}_{\mathcal{C}}(\alpha) = 0$ *genau dann, wenn* $\text{weight}_{\mathcal{C}_i}(\alpha) = 0$ *für alle $i = 1, \ldots, m$.*
Dann gibt es $r_1, \ldots, r_m \in \mathbb{F}_{2^K}$, sodass $\mathcal{D} := \sum_{i=1}^m r_i \mathcal{C}_i$ eine 2^{-K}-Approximation von $\mathcal{C}$ ist.

Beweis. Sei $\alpha \in \{0,1\}^n$ beliebig aber fest. Betrachte $P(Z_1, \ldots, Z_m) := \sum_{i=1}^m Z_i \,\text{weight}_{\mathcal{C}_i}(\alpha)$ als Polynom über $\mathbb{F}_{2^K}$. Aus $\text{weight}_{\mathcal{C}}(\alpha) = 0$ folgt $P \equiv 0$. Für $\text{weight}_{\mathcal{C}}(\alpha) \neq 0$ ist P nicht das Nullpolynom, multilinear und hat Totalgrad 1, somit gibt es höchstens $2^{K(m-1)}$ Nullstellen (siehe Korollar B.5). Durch doppeltes Abzählen erhält man Folgendes. Es gibt $r_1, \ldots, r_m \in \mathbb{F}_{2^K}$, sodass $\sum_{i=1}^m r_i \,\text{weight}_{\mathcal{C}_i}(\alpha) = 0$ auf höchstens 2^{n-K} Eingaben $\alpha \in \{0,1\}^n$ mit $\text{weight}_{\mathcal{C}}(\alpha) \neq 0$. □

Lemma 3.9 liefert eine Approximation für $\vee$. Das heißt, mit deMorgans Gesetz (Konjunktion entspricht negierter Disjunktion der negierten Literale) hat man auch eine Approximation für $\wedge$. Allerdings ist die naive Negation eines $\mathbb{F}_{2^K}$-OBDDs $\mathcal{C}$ sehr aufwendig, $\neg\mathcal{C} := \prod_{r \in \mathbb{F}_{2^K} \setminus \{0\}} (\mathcal{C} - r)$. Mit diesem Ansatz gilt für die Größe, $|\neg\mathcal{C}| = |\mathcal{C}|^{2^K - 1}$. Hier ist wieder der Frobenius-Automorphismus und seine Anwendung in Lemma 3.1 nützlich. Man konstruiert zu einem $\mathbb{F}_{2^K}$-OBDD $\mathcal{C}$ ein äquivalentes $\mathbb{F}_{2^K}$-OBDD $\mathcal{D}$, dessen Gewichtsfunktion nur Werte aus $\{0,1\}$ annimmt. Anschließend setzt man $\neg\mathcal{C} = (1 - \mathcal{D})$, dann gilt für die Größe, $|\neg\mathcal{C}| = |\mathcal{D}| = |\mathcal{C}|^K$.
Es muss noch berücksichtigt werden, dass Lemma 3.9 auf $\mathbb{F}_{2^K}$-OBDDs $\mathcal{C}_i'$ angewendet wird, die schon ε-Approximationen von $\mathbb{F}_{2^K}$-OBDDs $\mathcal{C}_i$ sind. Die Disjunktion $\bigvee_{i=1}^m \mathcal{C}_i'$ ist dann eine $(m \cdot \varepsilon)$-Approximation von $\mathcal{C} := \bigvee_{i=1}^m \mathcal{C}_i$ und die Summe $\sum_{i=1}^m r_i \mathcal{C}_i'$ ist eine $(m \cdot \varepsilon + 2^{-K})$-Approximation von $\mathcal{C}$.
Alle diese Vorüberlegungen berücksichtigend, lässt sich folgender Satz formulieren.

Satz 3.10. *Sei F eine DF mit quasipolynomieller beschränkter Anzahl von Unbestimmten und Konjunktionstermen. Für festes k sei $\varepsilon(n) = 1/2^{(\log_2 n)^k}$.*
Dann kann F, angewendet auf quasipolynomiell große $\oplus$OBDDs in n Variablen, durch ein $\oplus$OBDD, mit durch $\varepsilon(n)$ beschränkten Fehler, approximiert werden.

Beweis. Sei t die Anzahl der Konjunktionsterme und m die Anzahl der Unbestimmten von F. Seien $\mathcal{C}_1, \ldots, \mathcal{C}_m$ die $\oplus$OBDDs, auf die F angewendet wird, jedes mit Größe höchstens S. Seien $m, t, S \leq 2^{(\log_2 n)^\ell}$. Wähle $K := \lceil (\log_2 n)^k + \log(t+1) \rceil$ und einen Oberkörper $\mathbb{F}_2^K$ von $\mathbb{F}_2$ der Ordnung 2^K. Für jeden Konjunktionsterm wird ein $\mathbb{F}_{2^K}$-OBDD berechnet, mit deMorgans Gesetz, Lemma 3.9 und anschließender Negation mit Lemma 3.1. Jedes dieser $\mathbb{F}_{2^K}$-OBDDs hat Größe $(mS)^K$ und approximiert seinen Konjunktionsterm mit Fehlerschranke 2^{-K}. Abschließend wird die Disjunktion der $\mathbb{F}_{2^K}$-OBDDs für die Konjunktionsterme durch nochmalige Anwendung von Lemma 3.9 gebildet. Das Ergebnis wird mit Lemma 3.2 durch ein $\mathbb{F}_2$-OBDD simuliert. Die Größe des Ergebnisses ist $K(t(mS)^K)^K = 2^{(\log_2 n)^{\mathcal{O}(1)}}$. Als Fehlerschranke ergibt sich $(t+1)/2^K = 1/2^{(\log_2 n)^k} = \varepsilon(n)$. □

Einige andere interessante Modelle zur Darstellung von Booleschen Funktionen können durch DFs simuliert werden. Dazu gehört zum Beispiel die Schaltkreiskomplexitätsklasse $\mathcal{AC}^0$ über $\oplus$OBDDs. Das sind Schaltkreise polynomieller Größe und konstanter Tiefe, deren Gatter unbeschränkten Eingrad haben. Als Eingabegatter haben solche Schaltkreise Orakelgatter, die Funktionen über $\{X_1, \ldots, X_n\}$ darstellen, für die es quasipolynomiell große $\oplus$OBDDs gibt.

3.7 Nichtapproximierbarkeit

Dieser Abschnitt ist inspiriert durch [Hom03]. Wie beweist man untere Schranken für approximierende $\oplus$OBDDs? Dieses Problem ist verschränkt mit der Steifigkeit von Matrizen. Die Steifigkeitsfunktion $R_M(r)$ einer Matrix M gibt an, wie viele Einträge von M geändert werden müssen, um den Rang von M auf höchstens r zu reduzieren. Das folgende Problem ist grundlegend und schwierig seit es von Valiant [Val77] gestellt wurde. Beweise für eine gegebene $m \times m$ Matrix M, dass $R_M(\varepsilon m) > m^{1+\delta}$ für beliebige Konstanten $\varepsilon, \delta > 0$ ist. Abzählmethoden genügen, um zu zeigen, dass es Matrizen gibt, die diese Eigenschaft haben. Das Problem ist eine entsprechende Matrix und den zugehörigen Beweis anzugeben.
Um den Zusammenhang zwischen Matrixsteifigkeit und approximierenden $\oplus$OBDDs zu beschreiben, wird kurz betrachtet, wie untere Schranken für $\oplus$OBDDs bewiesen werden. Für eine feste Boolesche Funktion wird ein Schnitt durch die Variablenmenge gelegt und die resultierende Kommunikationsmatrix M gebildet. Der $\mathbb{F}_2$-Rang von M ist eine untere Schranke für die Größe jedes $\oplus$OBDDs, dass f darstellt. Diese Methode auf approximierende $\oplus$OBDDs angewendet, ergibt Folgendes. Sei eine Funktion f auf n Variablen und K beliebig aber fest. Jedes $\oplus$OBDD, das f mit Fehler $1/2^K$ approximiert, hat die Größe $\Omega(2^K)$. Dann ergibt die Veränderung der Kommunikationsmatrix M von f, an höchstens 2^{n-K} Stellen, eine Matrix mit Rang $\Omega(2^K)$. Das ist gleichbedeutend mit $r = \Omega(2^K)$, wenn $R_M(r) \leq 2^{n-K}$.
Unglücklicherweise ist die beste bekannte untere Schranke für die Matrixsteifigkeit $R_M(r) = \mathcal{O}(2^{2n}/r \cdot \log_2(2^n/r))$. Das bedeutet, angewendet auf die vorliegende Situation, $R_M(r) = \mathcal{O}(2^{n-K} \cdot (n/2 - K))$. Die letzten Ergebnisse auf diesem Gebiet betrachteten lineare Codes ([Fri93], [SSS97]) und Vandermonde-Matrizen ([Lok95]), die Ergebnisse sind aber zu schwach für die gewünschte Anwendung.
Untere Schranken hinreichender Ordnung für die Steifigkeit implizieren stärkere untere Schranken für viele Varianten von Berechnungsmodellen und sind außer Reichweite der zur Verfügung stehenden Methoden.
Die oben erwähnte Abzählmethode für untere Schranken der Steifigkeit ist auch anwendbar um Existenzbeweise für Funktionen zu führen, die durch $\oplus$OBDDs nicht approximierbar sind. Der Beweis eines solchen Resultats, für eine explizit angegebene Funktion, wäre ein großer Durchbruch.

4 OBDDs mit mehrfachen Tests

Die Forderung nach einfachen geordneten Tests ist eine starke Einschränkung von Branchingprogrammen (BPs). In einem BP mit einmaligen geordneten Tests (OBDD), werden für jede Eingabe die Variablen auf allen Berechnungspfaden in der gleichen Reihenfolge durchlaufen und auf jedem Berechnungspfad wird jede Variable höchstens einmal getestet. Das gilt sowohl für deterministische, nichtdeterministische, randomisierte als auch arithmetische BPs. Wenn im Folgenden nicht ausdrücklich etwas anderes gesagt wird, sind immer alle Modelle gemeint.
In diesem Kapitel wird eine Erweiterung von OBDDs untersucht, OBDDs mit mehrfachen Tests, genauer k-fachen Tests (k-OBDDs). OBDDs entsprechen 1-OBDDs und sind somit ein Spezialfall dieses Modells.
Zu jedem k-OBDDs gehört eine feste Permutation der Variablenindizes, mit deren Hilfe eine Variablenordnung bestimmt wird. Die Variablenordnung wird erzeugt durch k-maliges Hintereinanderstellen der durch die Permutation bestimmten Variablenfolge. Auf jedem Berechnungspfad werden die Variablen, bis auf Auslassungen, in derselben Reihenfolge wie in der Variablenordnung durchlaufen. Nach dieser formlosen Beschreibung wird nachfolgend eine formale Definition von k-OBDDs angegeben.
Sei $\mathcal{B}$ ein k-OBDD auf der Variablenmenge $\{X_1, \ldots, X_n\}$ mit Permutation $\pi : \{1, \ldots, n\} \to \{1, \ldots, n\}$. Sei $\sigma' := (X_{\pi(1)}, \ldots, X_{\pi(n)})$, die von π erzeugte Variablenfolge und seien $\sigma_1, \ldots, \sigma_k := \sigma'$ Kopien dieser Folge. Die Variablenordnung $\sigma := \sigma_1 \cdots \sigma_k$ ergibt sich aus Hintereinanderstellen der k Kopien von σ'. Für eine beliebige Eingabe gilt, die Folge von Knotenmarkierungen jedes Berechnungspfades in $\mathcal{B}$ unter dieser Eingabe ist Teilfolge von σ.
Ein k-OBDD mit Permutation π auf der Variablenmenge $\{X_1, \ldots, X_n\}$ kann in k Ebenen eingeteilt werden, indem man fordert, dass in jeder Ebene die Untervariablenordnung $(X_{\pi(1)}, \ldots, X_{\pi(n)})$ gilt. Jede Ebene wiederum kann in n Level zerlegt werden. Für $i = 1, \ldots, n$ werden alle Knoten, die mit $X_{\pi(i)}$ markiert sind, dem Level i zugeordnet. Die Quelle ist der einzige Knoten im obersten Level der obersten Ebene. Die unterste Ebene enthält einen zusätzlichen Level $n + 1$, der nur die Senke enthält.
Die Größe eines k-OBDDs ist wie üblich die Anzahl der Knoten des zugrunde liegenden Graphen. Die Weite eine k-OBDDs ist das Maximum über die Mächtigkeit aller Levels.
Für deterministische k-OBDDs gibt es eine strenge Hierarchie, die von Bollig, Sauerhoff, Sieling und Wegener [BSSW98] ausgearbeitet wurde. Dazu wird die

Boolesche Funktion $\mathrm{PJ}_{m,k}$ (pointer-jumping) auf $n = \Theta(m \log(m))$ Variablen benutzt. $\mathrm{PJ}_{m,k}$ liefert in einem mit $0, 1$ eingefärbten, gerichteten, bipartiten Graphen mit zwei Knotenmengen der Größe m und einem separaten Startknoten, die Färbung des Endknotens eines eindeutigen Pfades der Länge $2k+1$ zurück. Sei $k \leq (1-\varepsilon) \log \log n$, für eine beliebige Konstante $0 < \varepsilon < 1$. Dann gibt es ein k-OBDD für $\mathrm{PJ}_{m,k}$ der Größe $\mathcal{O}(km^2)$. Aber es gibt kein $(k-1)$-OBDD polynomieller Größe für diese Funktion.
Die unteren Schranken basieren auf der Theorie über die deterministische Kommunikationskomplexität (siehe Monographien von Hromkovič [Hro97] und Kushilevitz und Nisan [KN97]). Einerseits kann man jedem k-OBDD ein Kommunikationsprotokoll mit $2k$ Runden zuordnen. Andererseits haben Nisan und Wigderson [NW91] für explizit gegebene Boolesche Funktionen, bei beschränkter Anzahl der Runden, starke untere Schranken für die deterministische Kommunikationskomplexität und die randomisierte Kommunikationskomplexität mit beschränktem Fehler bewiesen.
Dieser Ansatz für untere Schranken funktioniert nicht für nichtdeterministische k-OBDDs. Unabhängig vom Akzeptierungsmodus kann ein nichtdeterministisches Kommunikationsprotokoll der Länge ℓ, durch ein zwei Runden Protokoll der Länge $\mathcal{O}(\ell)$ simuliert werden (siehe [DKMW04]). In diesem Kapitel wird der Frage nachgegangen, ob sich dieses Simulationsergebnis auf k-OBDDs übertragen lässt.
In Abschnitt 4.1 werden arithmetische k-OBDDs über beliebigen Halbringen $\mathbf{R}$ (k-$\mathbf{R}$-OBDDs) betrachtet. Es wird gezeigt, dass, für konstantes k, k-$\mathbf{R}$-OBDDs polynomieller Größe mit $\mathbf{R}$-OBDDs gleicher Größenordnung simuliert werden können. $\mathbf{R}$-OBDDs können durch die passende Wahl des Halbrings $\mathbf{R}$ und des Akzeptierungsmodus als nichtdeterministische OBDDs mit existenziellem, universellem oder Parity-Akzeptierungsmodus aufgefasst werden. Somit gilt das Ergebnis auch für diese Modelle.
In Abschnitt 4.2 werden die Berechnungen aus Abschnitt 4.1 so angepasst, dass sie auch für arithmetische k-OBDDs mit Wahrscheinlichkeits-Nebenbedingung gelten. Daraus folgt, mit konstantem k ist die Darstellungskraft von randomisierten k-OBDDs und randomisierten OBDDs gleich.
Abschnitt 4.3 fasst die Ergebnisse der vorangegangenen Abschnitte kurz zusammen und zeigt die Grenzen der verwendeten Techniken auf.
Die in diesem Kapitel vorgestellten Ergebnisse sind schon in [BHW06] veröffentlicht.

4.1 Nichtdeterministische OBDDs mit k-fachen Tests

Ohne Einschränkung der Allgemeinheit sei die Permutation, durch die die Variablenordnung der nachfolgend betrachteten k-OBDDs bestimmt wird, die Identität. Jedes dieser k-OBDDs sei *semi-vollständig*. Das hießt, alle Pfade, die an der

Quelle beginnen, haben im obersten Level jeder der k Ebenen einen Knoten. Auch durch diese Festlegung wird die Allgemeinheit nicht eingeschränkt. Ein k-OBDD kann leicht in ein semi-vollständiges k-OBDD überführt werden, ohne die Größenordnung zu ändern. Um die Darstellung zu vereinfachen wird angenommen, dass die Mächtigkeit der obersten Level jeder Ebene der Weite des gesamten Diagramms entspricht, mit Ausnahme des obersten Levels der obersten Ebene, der nur die Quelle enthält.
Sei $\mathcal{C}$ ein semi-vollständiges k-$\mathbf{R}$-OBDD, mit $k \geq 2$, auf der Variablenmenge $\mathcal{X} = \{X_1, \ldots, X_n\}$ über einem beliebigen Halbring $\mathbf{R}$. Sei $|\mathcal{B}|$ die Größe und w die Weite von $\mathcal{C}$. Die Quelle von $\mathcal{C}$ ist s und die Senke t. Mit $\lambda = 2, \ldots, k$ werden für jede Ebene λ die Knoten des obersten Levels mit $v_{\lambda,1}, \ldots, v_{\lambda,w}$ bezeichnet.
Zuerst werden die beiden Spezialfälle, oberste und unterste Ebene, betrachtet. Man definiert zwei Zeilenvektoren der Länge w, die als Einträge Funktionen von $\{0,1\}^n$ nach $\mathbf{R}$ enthalten. Der Vektor der obersten Ebene ist $\mu^{(1)}$ und der Vektor der untersten Ebene ist $\mu^{(k)}$.
Für alle $i = 1, \ldots, w$ setze $\mu_i^{(1)}$ auf die Gewichtsfunktion der Quelle s über $v_{2,i}$, dem i-ten Knoten des obersten Levels der zweiten Ebene.

$$\mu_i^{(1)}(\mathcal{X}) := \text{weight}_{s,v_{2,i}}^{\mathcal{C}}(\mathcal{X})$$

Für $i = 1, \ldots, w$ setze $\mu_i^{(k)}$ auf die Gewichtsfunktion von $v_{k,i}$, dem i-ten Knoten, des obersten Levels der untersten Ebene k, über der Senke t.

$$\mu_i^{(k)}(\mathcal{X}) := \text{weight}_{v_{k,i},t}^{\mathcal{C}}(\mathcal{X})$$

Mit $\lambda = 2, \ldots, k$ wird für jede Ebene λ eine $(w \times w)$ Matrix definiert, die ebenfalls Funktionen von $\{0,1\}^n$ nach $\mathbf{R}$ enthält. Für $i, j = 1, \ldots, w$ entspricht der Eintrag $\mu_{i,j}^{(\lambda)}$ der Gewichtsfunktion des Knotens $v_{\lambda,i}$, dem i-ten Knoten des obersten Levels der Ebene λ, über $v_{\lambda+1,j}$, dem j-ten Knoten des obersten Levels der nächsten Ebene $\lambda+1$.

$$\mu_{i,j}^{(\lambda)}(\mathcal{X}) := \text{weight}_{v_{\lambda,i},\, v_{\lambda+1,j}}^{\mathcal{C}}(\mathcal{X})$$

Mit Hilfe der vorangegangenen Definitionen kann die Gewichtsfunktion von $\mathcal{C}$, also die Gewichtsfunktion der Quelle s über der Senke t, folgendermaßen als Matrixprodukt über dem Halbring $\mathbf{R}$ dargestellt werden.

$$\text{weight}_{s,t}^{\mathcal{C}}(\mathcal{X}) = \mu^{(1)}(\mathcal{X})^T \cdot \mu^{(2)}(\mathcal{X}) \cdot \ldots \cdot \mu^{(k-1)}(\mathcal{X}) \cdot \mu^{(k)}(\mathcal{X})$$

Durch Ausmultiplizieren erhält man

$$\text{weight}_{s,t}^{\mathcal{C}}(\mathcal{X}) = \sum_{i_2,\ldots,i_k \in \{1,\ldots,w\}} \underbrace{\left(\mu_{i_2}^{(1)}(\mathcal{X}) \cdot \mu_{i_2,i_3}^{(2)}(\mathcal{X}) \cdot \ldots \cdot \mu_{i_{k-1},i_k}^{(k-1)}(\mathcal{X}) \cdot \mu_{i_k}^{(k)}(\mathcal{X})\right)}_{=:\, \mu_{i_2,i_3,\ldots,i_{k-1},i_k}(\mathcal{X})} .$$

Diese Gleichung liefert die Konstruktionsmethode für das gesuchte **R**-OBDD. Ausgehend vom k-**R**-OBDD $\mathcal{C}$ ist es einfach für jede in den Vektoren und Matrizen gespeicherte Funktion, ein **R**-OBDD zu konstruieren, dessen Gewichtsfunktion der gespeicherten Funktion entspricht. Jedes dieser **R**-OBDDs hat maximal die gleiche Größe wie $\mathcal{C}$. Durch Anwenden der Produktkonstruktion wird für jede Funktion $\mu_{i_1,\ldots,i_k}$, mit $i_1,\ldots,i_k \in \{1,\ldots,w\}$, ein **R**-OBDD berechnet, mit genau dieser Funktion als Gewichtsfunktion. Die Größe der durch die Produktkonstruktion entstehenden **R**-OBDDs ist beschränkt durch $|\mathcal{C}|^k$. Abschließend kann die Summe dieser w^k **R**-OBDDs durch Verschmelzen jeweils der Quellen, die alle mit derselben Variable markiert sind, und der Senken gebildet werden. Die Größe des resultierenden **R**-OBDDs ist beschränkt durch $w^k|\mathcal{C}|^k$ und die Gewichtsfunktion ist identisch mit der Gewichtsfunktion des k-**R**-OBDDs $\mathcal{C}$. Daraus folgt unmittelbar der nachstehende Satz.

Satz 4.1. *Für jeden Halbring* **R** *und jede konstante natürliche Zahl k gilt,*

$$P\text{-}k\text{-}\mathbf{R}\text{-}(\neq 0)\text{-}OBDD = P\text{-}\mathbf{R}\text{-}(\neq 0)\text{-}OBDD \ ,$$
$$P\text{-}k\text{-}\mathbf{R}\text{-}(= 0)\text{-}OBDD = P\text{-}\mathbf{R}\text{-}(= 0)\text{-}OBDD \ .$$

Beweis. Sei $f \in$ P-k-**R**-$(\neq 0)$-OBDD und $\mathcal{C}$ ein k-**R**-$(\neq 0)$-OBDD polynomieller Größe $|\mathcal{C}|$ und somit auch polynomieller Weite w, das f darstellt. Es gibt ein **R**-$(\neq 0)$-OBDD, das f darstellt, der Größe $w^k|\mathcal{C}|^k$. Die Konstante k hängt nicht von n der Anzahl der Variablen ab, daraus folgt $w^k|\mathcal{C}|^k = n^{\mathcal{O}(1)}$.
Die Beweisführung ist identisch für $f \in$ P-k-**R**-$(= 0)$-OBDD. □

Satz 4.1 gilt für einen beliebigen Halbring und Akzeptierungsmodus, somit folgt das Ergebnis auch für nichtdeterministische OBDDs.

Korollar 4.2. *Für jede konstante natürliche Zahl k gilt,*

$$P\text{-}k\text{-}\vee OBDD = P\text{-}\vee OBDD \ ,$$
$$P\text{-}k\text{-}\wedge OBDD = P\text{-}\wedge OBDD \ ,$$
$$P\text{-}k\text{-}\oplus OBDD = P\text{-}\oplus OBDD \ .$$

4.2 Randomisierte OBDDs mit k-fachen Tests

In diesem Abschnitt wird ein semi-vollständiges arithmetisches k-$\mathbb{Q}$-OBDD $\mathcal{C}$ mit Wahrscheinlichkeits-Nebenbedingung auf der Variablenmenge $\mathcal{X} = \{X_1,\ldots,X_n\}$ über dem Körper der rationalen Zahlen $\mathbb{Q}$ betrachtet. Die Definition der Vektoren und Matrizen, die die Gewichtsfunktionen speichern, ist identisch mit denen in Abschnitt 4.1. Somit ergibt sich für die Gewichtsfunktion der Quelle über der Senke wieder nachfolgende Gleichung.

$$\text{weight}^{\mathcal{C}}_{s,t}(\mathcal{X}) = \sum_{i_2,\ldots,i_k \in \{1,\ldots,w\}} \underbrace{\left(\mu^{(1)}_{i_2}(\mathcal{X}) \cdot \mu^{(2)}_{i_2,i_3}(\mathcal{X}) \cdot \ldots \cdot \mu^{(k-1)}_{i_{k-1},i_k}(\mathcal{X}) \cdot \mu^{(k)}_{i_k}(\mathcal{X})\right)}_{=:\, \mu_{i_2,i_3,\ldots,i_{k-1},i_k}(\mathcal{X})}$$

Analog zum Vorgehen in Abschnitt 4.1 wird die Produktkonstruktion benutzt, um ein $\mathbb{Q}$-OBDD für jede Gewichtsfunktion $\mu_{i_2,i_3,\ldots,i_{k-1},i_k}$ zu konstruieren. Aber die Summe lässt sich nicht durch Verschmelzen der Quellen und Senken bilden, weil dadurch die Wahrscheinlichkeits-Nebenbedingung verletzt würde.
Um die Wahrscheinlichkeits-Nebenbedingung zu erhalten, wird ein aus der Theorie der Turing-Maschinen bekannter Trick benutzt. Jedes der $m = w^k$ $\mathbb{Q}$-OBDDs mit Wahrscheinlichkeits-Nebenbedingung, die für $i_2, \ldots, i_k \in \{1, \ldots, w\}$ die Gewichtsfunktion $\mu_{i_2,\ldots,j_k}$ haben, wird mit $1/2m$ multipliziert. Danach wird durch Verschmelzen jeweils der Quellen und der Senken das $\mathbb{Q}$-OBDD $\mathcal{C}'$ erzeugt. Abschließend wird eine mit 0 und eine mit 1 markierte Kante, beide mit dem Gewicht $1/2((2m-1)/2m)$, von der Quelle zur Senke hinzugefügt.
Daraus folgt für alle $\alpha \in \{0,1\}^n$,

$$\text{weight}^{C'}_{s,t}(\alpha) = \frac{1}{2m} \sum_{i_2,\ldots,i_k \in \{1,\ldots,w\}} \mu_{i_2,\ldots,j_k}(\alpha) + \frac{1}{2}\left(\frac{2m-1}{2m}\right) \ .$$

Diese Konstruktion stellt sicher, dass die Wahrscheinlichkeits-Nebenbedingung erfüllt ist und $\text{weight}^{C'}_{s,t}(\alpha) \geq \frac{1}{2}$ genau dann gilt, wenn $\sum_{i_2,\ldots,i_k \in \{1,\ldots,w\}} \mu_{i_2,\ldots,j_k}(\alpha) \geq \frac{1}{2}$. Das heißt, das k-$\mathbb{Q}$-OBDD $\mathcal{C}$ und das $\mathbb{Q}$-OBDD $\mathcal{C}'$ erfüllen die Wahrscheinlichkeits-Nebenbedingung und stellen mit dem $(\geq 1/2)$-Akzeptierungsmodus die gleiche Boolesche Funktion dar. Daraus folgt für randomisierte OBDDs nachstehender Satz.

Korollar 4.3. *Für jede natürliche Zahl k gilt*

$$\textit{P-k-randomisiertes-OBDD} = \textit{P-randomisiertes-OBDD} \ .$$

4.3 Zusammenfassung

Konstant viele mehrfach geordnete Tests führen weder für nichtdeterministische OBDDs mit existenziellem, universellem oder Parity-Akzeptierungsmodus noch für randomisierte OBDDs zu einer größeren Darstellungskraft.
Die Tatsache, dass bei k-OBDDs in jeder Ebene die gleiche Untervariablenordnung gilt, ermöglicht das Verschmelzen der einzelnen Ebenen. Weicht man von dieser Forderung ab, gilt Satz 4.1 nicht mehr.
Branchingprogramme mit k-fach unterschiedlich geordneten Tests, kurz k-IBDDs (indexed binary decision diagrams), wurden eingeführt von Jain, Abadir, Bitner,

Fussell und Abraham [JAB+92]. Ein k-IBDD auf der Variablenmenge $\{X_1, \ldots, X_n\}$ hat k feste Permutationen $\pi_1, \ldots, \pi_k$. Für $i = 1, \ldots, k$ wird die Untervariablenordnung σ_i der i-ten Ebene durch die Permutation π_i bestimmt, $\sigma_i := \{X_{\pi_i(1)}, \ldots . X_{\pi_i(n)}\}$. Die Variablenordnung σ des ganzen k-IBDD wird durch Hintereinanderstellen der einzelnen Untervariablenordnungen gebildet, $\sigma = \sigma_1 \cdots \sigma_k$. OBDDs sind ein Spezialfall dieses Modells und gleichzusetzen mit 1-IBDDs.
Schon zwei unterschiedlich geordnete Tests erhöhen die Darstellungskraft. Es gilt, nichtdeterministische 2-IBDDs mit Parity-Akzeptierungsmodus (2-$\oplus$IBDDs) haben eine größere Darstellungskraft als nichtdeterministische OBDDs mit Parity-Akzeptierungsmodus ($\oplus$OBDDs). Sei $\text{PERM}_{n\times n}$ die Boolesche Funktion, die eine Eingabe der Länge n^2 genau dann akzeptiert, wenn es sich um eine Permutationsmatrix handelt. Jedes $\oplus$OBDD für die Funktion $\text{PERM}_{n\times n}$ hat superpolynomielle Größe (vergleiche Satz 2.11). Allerdings ist es einfach, ein 2-$\oplus$IBDD linearer Größe für diese Funktion zu konstruieren, wenn in einer Ebene die Variablen spaltenweise und in der anderen zeilenweise durchlaufen werden.
Der in Abschnitt 4.2 benutzte Trick erhält die Wahrscheinlichkeits-Nebenbedingung aber verändert die Gewichtsfunktion. Das Verfahren ist so gestaltet, dass es zum $(\geq 1/2)$-Akzeptierungsmodus passt. Schon eine kleine Änderung des Akzeptierungsmodus führt dazu, dass sich kein entsprechendes Verfahren mehr finden lässt. Sei $\tau < 1/2$, mit $(\geq \tau)$-Akzeptierungsmodus entsprechen k-$\mathbb{Q}$-OBDDs mit Wahrscheinlichkeits-Nebenbedingung randomisierten k-OBDDs mit zweiseitig beschränkten Fehler. Für dieses Modell haben Hromkovič und Sauerhoff [HS03] eine strenge Hierarchie bewiesen.

A Lineare Codes

Eine Einführung in die Verwendung der linearen Algebra und insbesondere linearer Codes in der Kombinatorik findet man in [Juk01].

Sei $\mathbb{F}_2$ der Körper mit zwei Elementen $\{0,1\}$. Ein *linearer Code* C ist ein linearer Unterraum des n-dimensionalen Vektorraums $\mathbb{F}_2^n$ über dem Körper $\mathbb{F}_2$. Die *charakteristische Funktion* eines linearen Codes C ist

$$f_C : \mathbb{F}_2^n \to \{0,1\} \qquad \alpha \mapsto \begin{cases} 1 & \text{wenn } \alpha \in C, \\ 0 & \text{sonst.} \end{cases}$$

Das *Hamming-Gewicht* $w(\alpha)$ eines Codewortes $\alpha \in C$ ist die Anzahl der Einsen in α. Der *Hamming-Abstand* zweier Codewörter $\alpha, \beta \in C$ ist definiert als die Anzahl der Einsen in $\alpha + \beta$, also $w(\alpha + \beta)$. Der *Minimal-Abstand* eines Codes C ist das Minimum über dem Hamming-Abstand aller Paare von verschiedenen Elementen aus C. Da C ein linearer Vektorraum ist, gilt, der Minimal-Abstand von C ist das Minimum über den Hamming-Gewichten aller Elemente aus $C \backslash \{0\}$.
Der *duale Code* $C^\perp$ von C ist die Menge aller Vektoren $\beta \in \mathbb{F}_2^n$, für die gilt $\langle \beta, \alpha \rangle = 0$, für alle $\alpha \in C$ (wobei $\langle \beta, \alpha \rangle = \beta_1\alpha_1 + \cdots + \beta_n\alpha_n$ das übliche Skalarprodukt des Vektorraums $\mathbb{F}_2^n$ ist).
Eine Menge $D \subseteq \mathbb{F}_2^n$ wird *k-universell* genannt, wenn für jede k-elementige Indexmenge $I \subseteq \{1, \ldots, n\}$ gilt, die Projektion der Wörter aus D, auf die durch I festgelegten Koordinaten, ergibt den ganzen Raum $\mathbb{F}_2^k$.
Eine Eigenschaft von linearen Codes ist, dass ihre dualen Codes universell sind. Um diese Eigenschaft zu beweisen, wird eine, aus der linearen Algebra bekannte, Beziehung zwischen Vektorräumen und der Lösungsmenge von homogenen linearen Gleichungen benutzt.

Lemma A.1. *Sei $\mathbb{F}$ ein Körper. Jeder Unterraum U von $\mathbb{F}^n$ der Dimension d ist die Lösungsmenge eines Gleichungssystems $\Lambda X = 0$. Wobei Λ eine $((n-d) \times n)$ Matrix vom Rang $n-d$ und X ein Variablenvektor der Länge n ist.*

Lemma A.2. *Sei C ein linearer Code mit Minimal-Abstand $k+1$, dann ist der duale Code $C^\perp$ k-universell.*

Beweis. Sei $I = \{i_1, \ldots, i_k\} \subseteq \{1, \ldots, n\}$ eine beliebige k-elementige Indexmenge. Die Projektion aller Wörter aus $C^\perp$, auf die durch I festgelegten Koordinaten, ist ein linearer Unterraum U von $\mathbb{F}_2^k$.

Annahme, $\dim(U) = k - \ell$. Mit Lemma A.1 gibt es eine $(\ell \times k)$ Matrix Λ vom Rang ℓ, sodass U von den Lösungen des linearen Gleichungssystems $\Lambda X = 0$ gebildet wird. Die Matrix Λ enthält eine Zeile ungleich dem Nullvektor, somit gibt es eine nicht triviale Gleichung $\lambda_{i_1} X_{i_1} + \cdots + \lambda_{i_k} X_{i_k} = 0$, die von allen $\alpha \in C^\perp$ erfüllt wird. Sei $\xi = (\xi_1, \ldots, \xi_n)$ mit

$$\xi_i := \begin{cases} \lambda_i & \text{wenn } i \in I; \\ 0 & \text{sonst.} \end{cases}$$

Daraus folgt, $\langle \alpha, \xi \rangle = 0$ für alle $\alpha \in C^\perp$, das heißt, $\xi \in C$. Aber $w(\xi) = k$, das ist ein Widerspruch, denn der Minimal-Abstand von C ist $k + 1$. □

B Nullstellen von Polynomen

Die Normalform eines univariaten Polynoms P über einem Körper $\mathbb{F}$ ist $P(X) := \sum_{i=0}^{d} \alpha_i X^i$ mit $\alpha_d, \ldots, \alpha_0 \in \mathbb{F}$ und $\alpha_d \neq 0$. Der Grad des Polynoms ist d und α_d wird als Leitkoeffizient bezeichnet. Ein univariates Polynom vom Grad d hat höchstens d Nullstellen.
Ein multivariates Polynom P in Normalform über einem Körper $\mathbb{F}$ wird definiert durch einen Koeffizientenvektor $A = (\alpha_1, \ldots, \alpha_m)$ über $\mathbb{F}\backslash\{0\}$ und einer $m \times n$ Exponentenmatrix $D = (\delta_{i,j})_{1 \leq i \leq m, 1 \leq j \leq n}$ über $\mathbb{N} \cup \{0\}$,

$$P(X_1, \ldots, X_n) := \sum_{i=0}^{m} \alpha_i \; X_1^{\delta_{i,1}} X_2^{\delta_{i,2}} \cdots X_n^{\delta_{i,n}} \quad .$$

Der Totalgrad von P ist das Maximum über den Summen der Exponenten jedes Terms, also die größte Zeilensumme der Exponentenmatrix.
Ein Polynom heißt multilinear, wenn die Exponentenmatrix nur Einträge aus $\{0, 1\}$ hat. Das heißt, jedes X_i kommt maximal in der ersten Potenz vor. Daraus folgt, der Totalgrad eine Polynoms über n Variablen ist höchstens n.
Ist ein Polynom gegeben, kann die Wahrscheinlichkeit, bei einer zufälligen und gleichverteilten Wahl einer Belegung der Variablen des Polynoms, eine Nullstelle zu treffen, über die Anzahl der Nullstellen abgeschätzt werden.

Lemma B.1. *Sei $\mathbb{F}$ ein beliebiger Körper. Sei $P \in \mathbb{F}[X_1, \ldots, X_n]$ ein Polynom mit Totalgrad d, das nicht das Nullpolynom ist. Sei $S \subseteq \mathbb{F}$, $|S| > 1$ eine beliebige endliche Teilmenge von $\mathbb{F}$.*
Dann hat P in S^n höchstens $d \cdot |S|^{n-1}$ Nullstellen.

Beweis. Beweis durch Induktion über n, die Anzahl der Variablen.
Induktionsanfang, $n = 1$. P ist ein (univariates) Polynom mit (Total)-Grad d und hat somit höchstens d Nullstellen.
Induktionsschritt, die Behauptung gilt für alle Polynome mit bis zu $n - 1$ Variablen. Man betrachtet $P_n := P$ als Polynom in X_n über dem Ring $\mathbb{F}[X_1, \ldots, X_{n-1}]$ in Normalform. Der Grad von P_n ist d_n und P_{n-1} ist der Leitkoeffizient. P_{n-1} ist ein Polynom in $n - 1$ Variablen und Totalgrad d_{n-1}.
Sei $\alpha := (\alpha_1, \ldots, \alpha_{n-1}) \in S^{n-1}$ beliebig aber fest. Entweder ist α eine Nullstelle von P_{n-1}, dann ist $P(\alpha, X_n)$ möglicherweise das Nullpolynom und somit 0 für alle Belegungen $\alpha_n \in S$ von X_n. Oder α ist keine Nullstelle von P_{n-1}, dann ist $P(\alpha, X_n)$

ein Polynom vom Grad d_n und hat höchstens d_n Nullstellen. Das heißt, die Anzahl der Nullstellen von P in S^n ist nach oben beschränkt durch

$$|S| \cdot d_{n-1} \cdot |S|^{n-2} + d_n \cdot |S|^{n-1} = (d_{n-1} + d_n) \cdot |S|^{n-1} \quad .$$

Abschließend bleibt festzustellen, dass die Summe $d_{n-1} + d_n$, aus Totalgrad von P_{n-1} und Grad von P_n, kleiner gleich dem Totalgrad d von P ist. □

Somit folgt für die zufällige Wahl einer Belegung, die aus [Zip79, Sch80] bekannte Abschätzung.

Korollar B.2. *Sei $\mathbb{F}$ ein endlicher Körper der Ordnung $|\mathbb{F}|$. Sei $P \in \mathbb{F}[X_1, \ldots, X_n]$ ein Polynom mit Totalgrad höchstens d, das nicht das Nullpolynom ist. Wähle $(\alpha_1, \ldots, \alpha_n) \in \mathbb{F}^n$ zufällig und gleichverteilt.*
Dann gilt,

$$\Pr[P(\alpha_1, \ldots, \alpha_n) = 0] \leq \frac{d}{|\mathbb{F}|} \quad .$$

In [BCW80] werden multilineare Polynome betrachtet und für die Abschätzung der Nullstellenanzahl eines multilinearen Polynoms wird folgendes Lemma benutzt.

Lemma B.3. *Sei $P \in \mathbb{F}[X_1, \ldots, X_n]$ ein multilineares Polynom über dem Körper $\mathbb{F}$. Sei $S \subseteq \mathbb{F}$, $|S| > 1$ eine beliebige endliche Teilmenge.*
Dann hat P höchstens $|S|^n - (|S| - 1)^n$ viele Nullstellen in S^n.

Wenn zusätzlich noch der Totalgrad des Polynoms bekannt ist, kann mit einem Spezialfall von Lemma B.3 eine bessere Abschätzung erreicht werden.

Lemma B.4. *Sei $P \in \mathbb{F}[X_1, \ldots, X_n]$ ein multilineares Polynom, nicht das Nullpolynom, mit Totalgrad d über dem Körper $\mathbb{F}$. Sei $S \subseteq \mathbb{F}$, $|S| > 1$ eine beliebige endliche Teilmenge.*
Dann hat P höchstens $|S|^n - (|S| - 1)^d |S|^{n-d}$ viele Nullstellen in S^n.

Beweis. Beweis durch Induktion über die Anzahl der Variablen.
Behauptung, P ist für mindestens $(|S| - 1)^d |S|^{n-d}$ Belegungen aus S^n ungleich 0.
Induktionsanfang, $n = 1$. Entweder ist der Totalgrad von P gleich 0, dann ist P konstant und hat keine Nullstellen. Oder P hat Totalgrad 1 und höchstens eine Nullstelle.
Induktionsschritt, die Behauptung gilt für alle Polynome mit bis zu $n - 1$ Variablen.
Sei P ein Polynom in $X_1, \ldots, X_n$ mit Totalgrad d über $\mathbb{F}$. Da P auch als Polynom in X_n über dem Ring $\mathbb{F}[X_1, \ldots, X_{n-1}]$ betrachtet werden kann, ist folgende Darstellung von P möglich

$$P(X_1, \ldots, X_n) = P_0(X_1, \ldots, X_{n-1}) + P_1(X_1, \ldots, X_{n-1}) X_n \quad ,$$

wobei P_0 und P_1 Polynome in $n-1$ Variablen sind.
Fall $P_1 \not\equiv 0$. Der Totalgrad von P_1 ist höchstens $d-1$. Das heißt, P_1 hat $(|S|-1)^{d-1}|S|^{n-d}$ Nicht-Nullstellen. Sei $(\alpha_1, \ldots, \alpha_{n-1}) \in S^{n-1}$ eine Nicht-Nullstelle von P_1. Dann gibt es nur einen Wert $\alpha_n := -P_0(\alpha_1, \ldots, \alpha_{n-1})/P_1(\alpha_1, \ldots, \alpha_{n-1})$ mit $P(\alpha_1, \ldots, \alpha_{n-1}, \alpha_n) = 0$. Daraus folgt, P hat $(|S|-1)^{d-1}|S|^{n-d} \cdot (|S|-1)$ Nicht-Nullstellen, wie behauptet.
Fall $P_1 \equiv 0$. Dann hat P_0 Totalgrad d und $(|S|-1)^d|S|^{n-1-d}$ Nicht-Nullstellen. Da P_0 unabhängig von X_n ist, folgt, P hat $(|S|-1)^d|S|^{n-1-d} \cdot |S|$ Nicht-Nullstellen, wie behauptet. □

Die angegebene Schranke ist scharf. Das Polynom $P(X_1, \ldots, X_n) := X_1 \cdots X_d$ über einem endlichen Körper $\mathbb{F}$ hat genau $|\mathbb{F}|^n - (|\mathbb{F}|-1)^d|\mathbb{F}|^{n-d}$ Nullstellen.
Die Abschätzung der Wahrscheinlichkeit bei zufälliger und gleichverteilter Wahl einer Belegung eine Nullstelle zu treffen, folgt direkt aus dem Lemma.

Korollar B.5. *Sei $\mathbb{F}$ ein endlicher Körper der Ordnung $|\mathbb{F}|$. Sei $P \in \mathbb{F}[X_1, \ldots, X_n]$ ein multilineares Polynom mit Totalgrad höchstens d, das nicht das Nullpolynom ist. Wähle $(\alpha_1, \ldots, \alpha_n) \in \mathbb{F}^n$ zufällig und gleichverteilt.*
Dann gilt,

$$\Pr[P(\alpha_1, \ldots, \alpha_n) = 0] \leq 1 - \left(1 - \frac{1}{|\mathbb{F}|}\right)^d \quad .$$

Für multilineare Polynome über n Variablen mit Totalgrad d liefert Korollar B.5 eine bessere Abschätzung als Korollar B.2, wenn die Ordnung $|\mathbb{F}|$ des endlichen Körpers größer gleich dem Totalgrad d des Polynoms ist. Es gilt, mit $|\mathbb{F}| \geq d$,

$$\frac{1}{|\mathbb{F}|} \leq 1 - \left(1 - \frac{1}{|\mathbb{F}|}\right)^d < \frac{d}{|\mathbb{F}|} \quad .$$

Die untere Schranke ist klar, denn $1 - 1/|\mathbb{F}| < 1$ wird durch potenzieren mit $d > 1$ kleiner.
Die obere Schranke ist etwas aufwendiger.

$$1 - \left(1 - \frac{1}{|\mathbb{F}|}\right)^d = 1 - \sum_{i=0}^{d} \binom{d}{i} \frac{(-1)^i}{|\mathbb{F}|^i} = \frac{d}{|\mathbb{F}|} - \sum_{i=2}^{d} \binom{d}{i} \frac{(-1)^i}{|\mathbb{F}|^i}$$

Bleibt zu zeigen, $\sum_{i=2}^{d} \binom{d}{i} \frac{(-1)^i}{|\mathbb{F}|^i} > 0$. Dazu werden zwei aufeinanderfolgende Summanden betrachtet. Mit i gerade und $2 \leq i \leq d-1$ gilt,

$$\begin{aligned} \binom{d}{i} \frac{1}{|\mathbb{F}|^i} - \binom{d}{i+1} \frac{1}{|\mathbb{F}|^{i+1}} &= \binom{d}{i} \frac{1}{|\mathbb{F}|^i} - \binom{d}{i} \frac{d-i}{i+1} \frac{1}{|\mathbb{F}|^{i+1}} \\ &= \binom{d}{i} \frac{1}{|\mathbb{F}|^i} \left(1 - \frac{d-i}{i+1} \frac{1}{|\mathbb{F}|}\right) \quad . \end{aligned}$$

Es gilt, $\frac{d-i}{i+1} < d \leq |\mathbb{F}|$, das heißt, je zwei aufeinanderfolgende Summanden addieren sich zu einem Wert größer 0 (ist d gerade gibt es noch einen einzelnen Summanden größer 0). Somit ist die Summe positiv und daraus folgt die obere Schranke.

Abbildungsverzeichnis

Literaturverzeichnis

[AK05] S. Aaronson and G. Kuperberg. Complexity zoo. http://qwiki.caltech.edu/wiki/Complexity_Zoo, 2005.

[AM88] N. Alon and W. Maass. Meanders and their applications in lower bounds arguments. *Journal of Computer and System Sciences*, 37(2):118–129, 1988.

[BCW80] M. Blum, A. K. Chandra, and M. N. Wegman. Equivalence of free boolean graphs can be tested in polynomial time. *Information Processing Letters*, 10:80–82, 1980.

[BDHW05] H. Brosenne, C. Damm, M. Homeister, and St. Waack. On approximation by $\oplus$OBDDs. In *Proceedings of the 7th International Symposium on Representations and Methodology of Future Computing Technologies*, 2005.

[BG98] A. Beimel and A. Gál. On arithmetic branching programs. In *Proceedings, 13th IEEE Conference on Computational Complexity*, pages 68–80, 1998.

[BHW01] H. Brosenne, M. Homeister, and St. Waack. Graph–driven free parity BDDs: Algorithms and lower bounds. In *Proceedings, 26th MFCS*, volume 2136 of *Lecture Notes in Computer Science*, pages 212–223. Springer, 2001.

[BHW02] H. Brosenne, M. Homeister, and St. Waack. Characterizing the complexity of Boolean functions represented by well–structured graph–driven parity–FBDDs. *RAIRO – Theoretical Informatics and Applications*, 36(3):229–247, 2002.

[BHW03] H. Brosenne, M. Homeister, and St. Waack. Lower bounds for general graph–driven read–once parity branching programs. In *Proceedings, 28th MFCS*, volume 2747 of *Lecture Notes in Computer Science*, pages 290–299. Springer, 2003.

[BHW06] H. Brosenne, M. Homeister, and St. Waack. Nondeterministic ordered binary decision diagrams with repeated tests and various modes of acceptance. *Information Processing Letters*, 98:6–10, 2006.

[BNS92] L. Babai, N. Nisan, and M. Szegedy. Multiparty protocols, pseudorandom generators for logspace, and time-space trade-offs. *Journal of Computer and System Sciences*, 45:204–232, 1992.

[Bol01] B. Bollig. Restricted nondeterministic read–once branching programs and an exponential lower bound for integer multiplication. *RAIRO – Theoretical Informatics and Applications*, 35:149–162, 2001.

[Bol03] B. Bollig. Complexity theoretical results on nondeterministic graph-driven read-once branching programs. In *Proceedings 20th STACS*, volume 2607 of *Lecture Notes in Computer Science*, pages 295–306. Springer, 2003.

[Bro00] H. Brosenne. Schaltkreisverifikation: Darstellungskraft und algorithmische Handhabbarkeit von Graphgesteuerten Read-Once Parity BDD. Diplomarbeit, Institut für Numerische und Angewandte Mathematik, Georg-August-Universität Göttingen, 2000.

[Bry86] R. E. Bryant. Graph–based algorithms for Boolean function manipulation. *IEEE Transactions on Computers*, 35:677–691, 1986.

[Bry91] R. Bryant. On the complexity of VLSI implementations of Boolean functions with applications to integer multiplication. *IEEE Transactions on Computers*, 40:205–213, 1991.

[BSSW98] B. Bollig, M. Sauerhoff, D. Sieling, and I. Wegener. Hierarchy theorems for k OBDDs and k IBDDs. *Theoretical Computer Science*, 205(1-2):45–60, 1998.

[BW97] B. Bollig and I. Wegener. Complexity theoretical results on partitioned (nondeterministic) binary decision diagrams. In *Proceedings, 22th MFCS*, volume 1295 of *Lecture Notes in Computer Science*, pages 159–168. Springer, 1997.

[BWW02] B. Bollig, St. Waack, and P. Woelfel. Parity graph–driven read–once branching programs and an exponential lower bound for integer multiplication. In *Proceedings, 2nd IFIP International Conference on Theoretical ComputerScience*, 2002.

[DKMW04] C. Damm, M. Krause, Ch. Meinel, and St. Waack. On relations between counting communication complexity classes. *Journal of Computer and System Sciences*, 69(2):259–280, 2004.

[Fri93] J. Friedman. A note on matrix rigidity. *Combinatorica, 13(2)*, pages 235–239, 1993.

[GM93] J. Gergov and Ch. Meinel. Frontiers of feasible and probabilistic feasible Boolean manipulation with branching programs. In *Proceedings, 10th STACS*, volume 665 of *Lecture Notes in Computer Science*, pages 576–585. Springer, 1993.

[GM96] J. Gergov and Ch. Meinel. Mod–2–OBDDs – a data structure that generalizes exor–sum–of–products and ordered binary decision diagrams. *Formal Methods in System Design*, 8:273–282, 1996.

[Hom03] M. Homeister. Personal communication, 2003.

[Hro97] J. Hromkovič. *Communication Complexity and Parallel Computing.* Springer, 1997.

[HS03] J. Hromkovic and M. Sauerhoff. The power of nondeterminism and randomness for oblivious branching programs. *Theory of Computing Systems*, 36(2):159–182, 2003.

[Imm88] N. Immerman. Nondeterministic space is closed under complementation. *SIAM Journal on Computing*, 17(5):935–938, 1988.

[JAB+92] J. Jain, M. Abadir, J. Bitner, D.S. Fussell, and J.A. Abraham. IBDDs: An efficient functional representation for digital circuits. In *Proceedings of the European Conference on Design Automation*, pages 440–446. IEEE Computer Society Press, 1992.

[Juk89] S. Jukna. The effect of null-chains on the complexity of contact schemes. In *Fundamentals of Computation Theory*, volume 380 of *Lecture Notes in Computer Science*, pages 246–256. Springer, 1989.

[Juk95] S. Jukna. A note on read-k times branching programs. *RAIRO – Theoretical Informatics and Applications*, 29(1):75–83, 1995.

[Juk99] S. Jukna. Linear codes are hard for oblivious read–once parity branching programs. *Information Processing Letters*, 69:267–269, 1999.

[Juk01] S. Jukna. *Extremal Combinatorics: With Applications in Computer Science.* Texts in Theoretical Computer Science. An EATCS Series. Springer, 2001.

[KMW91] M. Krause, Ch. Meinel, and St. Waack. Separating the eraser Turing machine classes L_e, NL_e, co–NL_e, and P_e. *Theoretical Computer Science*, 86:267–275, 1991.

[KN97] E. Kushilevitz and Nisan N. *Communication Complexity*. Cambridge University Press, 1997.

[Kra88] M. Krause. Exponential lower bounds on the complexity of local and real–time branching programs. *Journal of Information Processing and Cybernetics (EIK)*, 24:99–110, 1988.

[Kra91] M. Krause. Lower bounds for depth-restricted branching programs. *Information and Computation*, 91(1):1–14, 1991.

[Kra92] Matthias Krause. Separating $\oplus$L from L, NL, co-NL, and AL=P for oblivious turing machines of linear access. *Informatique Théorique et Applications*, 26:507–540, 1992.

[KW91] M. Krause and St. Waack. On oblivious branching programs of linear length. *Information and Computation*, 94(2):232–249, 1991.

[Lok95] S. V. Lokam. Spectral methods for matrix rigidity with applications to size-depth tradeoffs and communication complexity. In *Proceedings, 36th FOCS*, pages 6–15, 1995.

[Mei90] Ch. Meinel. Polynomial size Ω-branching programs and their computational power. *Information and Computation*, 85(2):163–182, 1990.

[MS77] E. J. MacWilliams and N. J. A. Sloane. *The Theory of Error–Correcting Codes*. Elsevier, 1977.

[NW91] N. Nisan and A. Widgerson. Rounds in communication complexity revisited. In *Proceedings, 23th STOC*, pages 419–429, 1991.

[Sch80] J. T. Schwartz. Fast probabilistic algorithms for verification of polynomial identities. *Journal of the ACM*, 27(4):701–717, 1980.

[Sie94] D. Sieling. *Algorithmen und untere Schranken für verallgemeinerte OBDDs*. PhD thesis, Fachbereich Informatik, Universität Dortmund, 1994.

[Sie99] D. Sieling. Lower bounds for linear transformed OBDDs and FBDDs. In *Proceedings, 19th FSTTCS*, number 1738 in Lecture Notes in Computer Science, pages 356–368. Springer, 1999.

[Smo87] R. Smolensky. Algebraic methods in the theory of lower bounds for boolean circuit complexity. In *Proceedings, 19th STOC*, pages 77–82, 1987.

[SS00] P. Savický and D. Sieling. A hierarchy result for read–once branching programs with restricted parity nondeterminism. In *Proceedings, 25th MFCS*, volume 1893 of *Lecture Notes in Computer Science*, pages 650–659. Springer, 2000.

[SSS97] M. A. Shokrollahi, D. A. Spielman, and V. Stemann. A remark on matrix rigidity. *Information Processing Letters*, 64(6):283–285, 1997.

[SW95] D. Sieling and I. Wegener. Graph driven BDDs – a new data structure for Boolean functions. *Theoretical Computer Science*, 141:238–310, 1995.

[Sze87] R. Szelepcsényi. The method of forcing for nondeterministic automata. *Bulletin of the European Association for Theoretical Computer Science*, 33:96–100, 1987.

[Val77] L. G. Valiant. Graph theoretic arguments in low–level complexity. In *Proceedings, 6th MFCS*, volume 53 of *Lecture Notes in Computer Science*, pages 162–176. Springer, 1977.

[Waa01] St. Waack. On the descriptive and algorithmic power of parity ordered binary decision diagrams. *Information and Computation*, 166:61–70, 2001.

[Weg88] Ingo Wegener. On the complexity of branching programs and decision trees for clique functions. *Journal of the ACM*, 35(2):461–471, 1988.

[Wig94] A. Wigderson. NL/poly $\subseteq \oplus$L/poly. In *Proceedings of the 9th Structures in Complexity conference*, pages 59–62, 1994.

[Zák84] S. Zák. An exponential lower bound for one-time-only branching programs. In *Proceeding, 11th MFCS*, volume 176 of *Lecture Notes in Computer Science*, pages 562–566. Springer, 1984.

[Zip79] R. E. Zippel. Probabilistic algorithms for sparse polynomials. In *Proceedings, EUROSAM '79*, volume 72 of *Lecture Notes in Computer Science*, pages 216–226. Springer, 1979.

Index

Lebenslauf

Persönliche Daten

Name	Henrik Brosenne
geboren	23. Januar 1972 in Göttingen
Familienstand	verheiratet
Staatsangehörigkeit	deutsch

Schulausbildung

08.1978 - 06.1982	Grundschule Adelebsen
07.1982 - 07.1984	Orientierungsstufe der Haupt- und Realschule Adelebsen
08.1984 - 05.1991	Otto-Hahn-Gymnasium Göttingen Abschluss: Allgemeine Hochschulreife

Studium

10.1992 - 11.2000	Studium an der Georg-August-Universität zu Göttingen Hauptfach: Mathematik, Nebenfach: Informatik Diplomarbeit: „Schaltkreisverifikation: Darstellungskraft und algorithmische Handhabbarkeit von Graphgesteuerten Read-Once Parity BDD“ Abschluss: Diplom in Mathematik
09.2001 - 04.2006	Promotionsstudium Mathematik

Sonstiges

07.1991 - 06.1992	Wehrdienst im Pionierbataillon 2 in Hann. Münden
11.2000 - 12.2000	Praktikum bei der AB-Consultuing in Berlin
01.2001 - 06.2001	Systemanalytiker bei der BfA in Berlin