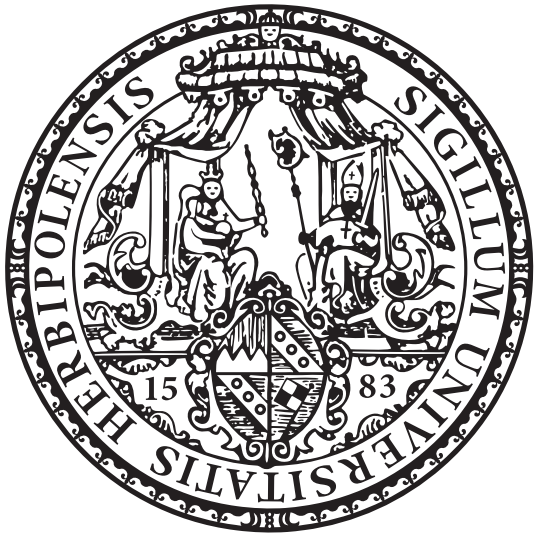
Masterarbeit

Komplexität arithmetischer Schaltkreise über natürlichen Zahlen mit Addition, Multiplikation und Division

Silas Cato Sacher

Abgabedatum: 20. März 2023
Überarbeitet: 30. September 2023
Betreuer: Prof. Dr. Christian Glaßer
MSc. Fabian Egidy



Julius-Maximilians-Universität Würzburg
Lehrstuhl für Informatik I
Algorithmen und Komplexität

Zusammenfassung

In dieser Arbeit wird die Komplexität des Membership-Problems für arithmetische Schaltkreise über natürlichen Zahlen mit Addition, Multiplikation und Division betrachtet. Diese wird in Abhängigkeit von den im Schaltkreis vorkommenden Operationen untersucht. Für Teilmengen von $\{\cup, \cap, ^-, +, \times\}$ gibt es bereits ausführliche Resultate von McKenzie und Wagner (2007). Wir zeigen: Der Fall $\{\cup, \cap, +, \times, /\}$ ist NEXP-vollständig, also genau wie $\{\cup, \cap, +, \times\}$. Die Fälle $\{\cup, \cap, ^-, +, /\}$ und $\{\cup, \cap, ^-, \times, /\}$ sind PSPACE-vollständig, also genau wie die entsprechenden Fälle ohne Division. Polynomial Identity Testing (PIT) kann auf den Fall $\{+, \times, /\}$ reduziert werden, während der Fall ohne Division (also $\{+, \times\}$) P-vollständig ist. Der Fall $\{\cup, /\}$ ist NP-hart ($\{\cup\}$ ist NL-vollständig und $\{\cup, \cap, ^-\}$ ist P-vollständig) und der Fall $\{\times, /\}$ ist PL-hart ($\{\times\}$ ist NL-vollständig). Darüber hinaus werden weitere obere und untere Schranken angegeben.

Inhaltsverzeichnis

1 Einleitung

Stockmeyer und Meyer haben in [SM73] bereits 1973 das Äquivalenzproblem für Formeln natürlicher Zahlen (inklusive Null) definiert und untersucht. Formeln sind dabei Ausdrücke aus natürlichen Zahlen, Mengenoperationen (Schnitt, Vereinigung und Komplement) und Addition. Die ganzen Zahlen werden als einelementige Mengen aufgefasst, welche durch die Operationen kombiniert werden. Die Addition wird dabei elementweise angewendet. Das Äquivalenzproblem ist die Frage, ob zwei solche Formeln die gleiche Menge beschreiben. Die Komplexität des Problems kann in Abhängigkeit davon, welche der oben genannten Operationen in der Formel verwendet wurden, untersucht werden.

Arithmetische Schaltkreise sind eine Verallgemeinerung solcher Formeln. An Stelle von Ausdrücken werden azyklische, gerichtete Graphen zur Beschreibung von Mengen verwendet. Den Knoten des Graphen werden entweder einelementige Mengen zugewiesen oder eine Operation, die beschreibt, wie die Mengen aus den Vorgängerknoten zu einer neuen Menge kombiniert wird. Ein festgelegter Ausgabeknoten beschreibt die Ausgabemenge des Schaltkreises.

Für solche Schaltkreise können verschiedene Entscheidungsprobleme untersucht werden. McKenzie und Wagner haben 2007 in [MW07] das Membership-Problem für arithmetische Schaltkreise über natürlichen Zahlen mit den oben genannten Mengenoperationen sowie Addition und Multiplikation untersucht. Das Membership-Problem ist die Frage, ob eine bestimmte Zahl in der Menge, der durch den Schaltkreis beschriebenen Zahlen, enthalten ist. Es ist unbekannt, ob das allgemeine Membership-Problem für Schaltkreise, die alle der oben genannten Operationen enthalten, entscheidbar ist. Christian Glaßer hat zuerst beobachtet, dass ein Algorithmus für dieses Problem die Frage beantworten würde, ob die Goldbachsche Vermutung wahr ist.

Glaßer et al. haben 2008 in [Gla+08] das Äquivalenzproblem für arithmetische Schaltkreise untersucht und Barth et. al. haben 2017 das Emptiness-Problem (also die Frage, ob ein gegebener Schaltkreis die leere Menge beschreibt) untersucht.

Diese Masterarbeit kann als Erweiterung der Arbeit von McKenzie und Wagner verstanden werden. Wir untersuchen das Membership-Problem für arithmetische Schaltkreise über natürlichen Zahlen, wobei zu den erlaubten Operationen die Division hinzugefügt wird. Bei der Division zweier natürlicher Zahlen ist es im Gegensatz zur Addition und Multiplikation nicht garantiert, dass die Berechnung ein Ergebnis liefert. Die Komplexität des Membership-Problems wird jeweils für verschiedene Teilmengen $\mathcal{O} \subseteq \{\cup, \cap, ^-, +, \times, /\}$ der Operationen untersucht.

Wie ist diese Arbeit aufgebaut? Nachdem wir das Membership-Problem für arithmetische Schaltkreise in Kapitel 2 formal definieren, untersuchen wir in Kapitel 3 obere Schranken für Membership-Probleme mit verschiedenen Operationen. Es gibt je einen Abschnitt über Schaltkreise in denen jeweils eine der Operationen Komplement, Addition und Multiplikation fehlt. Für das Membership-Problem mit allen Operationen außer Komplement werden wir feststellen, dass es NEXP-vollständig ist. Das Membership-Problem ohne Addition bzw. Multiplikation ist jeweils PSPACE-vollständig. Alle drei Probleme sind jeweils äquivalent zu dem entsprechenden Problem ohne Divisi-

on. Wir werden außerdem sehen, dass die Menge der Primzahlen sich auch mit Division nicht ohne Multiplikation ausdrücken lässt. In Kapitel 4 werden wir untere Schranken für Membership-Probleme mit verschiedenen Operationen zeigen. Hier werden wir sehen, dass es Mengen von Operationen $\mathcal{O} \subseteq \{\cup, \cap, ^{-}, +, \times, /\}$ gibt, sodass die Komplexität des Membership-Problems durch das Hinzufügen der Division erhöht würde, sollten bestimmte komplexitätstheoretische Vermutungen, wie z.B. das NL $\neq$ PL ist, wahr sein.

2 Grundlagen

2.1 Arithmetische Schaltkreise

In diesem Kapitel werden arithmetische Schaltkreise und ihre Ergebnismengen definiert. Wir werden außerdem das Membership-Problem definieren und eine Codierung für Schaltkreise formulieren, damit wir diese als Eingabe von Algorithmen verwenden können.

2.1.1 Definition arithmetischer Schaltkreise

Definition
Sei $\emptyset \neq \mathcal{O} \subseteq \{\cup, \cap, ^-, +, \times, /\}$ eine Menge von Operationen auf Mengen natürlicher Zahlen. (In dieser Arbeit gehen wir von $0 \in \mathbb{N}$ aus.) Die Operationen $\cup$, $\cap$ und $^-$ bezeichnen die Vereinigung, den Schnitt und das Komplement für Mengen natürlicher Zahlen. Die Multiplikation auf den natürlichen Zahlen schreiben wir als $\times$. Die arithmetischen Operationen $\sigma \in \{+, \times, /\}$ werden für Mengen $A, B \subseteq \mathbb{N}$ wie folgt erweitert:

$$A\sigma B =_{def} \{c \in \mathbb{N} \mid \text{es existieren } a \in A \text{ und } b \in B \text{ mit } c = a\sigma b\}$$

*Ein **(arithmetischer) $\mathcal{O}$-Schaltkreis** $C = (V, E, g_c, \alpha)$ ist ein azyklischer, gerichteter Multigraph $G = (V, E)$, dessen Knoten jeweils Eingangsgrad 0, 1 oder 2 haben. Es sei $V \subseteq \mathbb{N}$. (Das heißt, es gibt eine totale Ordnung auf der Knotenmenge.) $E = (E', m)$ sei eine Multimenge mit $E' \subseteq V^2$ und einer Abbildung $m : E' \to \{1, 2\}$, die angibt, wie häufig eine Kante $e \in E'$ im Graph G vorkommt. Die Knoten von G werden als **Gatter** bezeichnet. Gatter mit Eingangsgrad 0 heißen **Eingabegatter**. Das festgelegte Gatter $g_c \in V$ wird als **Ausgabegatter** bezeichnet. Die Gatter werden mit der Funktion $\alpha : V \to \mathcal{O} \cup \mathbb{N}$ beschriftet. Dabei gilt:*

1. *Für jedes Eingabegatter g ist $\alpha(g) \in \mathbb{N}$.*
2. *Für jedes Gatter g mit Eingangsgrad 1 ist $\alpha(g) = ^-$.*
3. *Für jedes Gatter g mit Eingangsgrad 2 ist $\alpha(g) \in \{\cup, \cap, +, \times, /\}$.*

*Ein Gatter g mit Beschriftung $\alpha = \alpha(g)$ wird auch als **α-Gatter** bezeichnet.*

*Für jedes Gatter $g \in V$ wird die **Ergebnismenge** $I(g) \subseteq \mathbb{N}$ berechnet. Diese ist wie folgt induktiv definiert:*

1. *Für Eingabegatter g mit Beschriftung $\alpha(g) \in \mathbb{N}$ ist $I(g) =_{def} \{\alpha(g)\}$.*
2. *Für ein Gatter g mit Beschriftung $\alpha(g) = ^-$ und Vorgänger p ist $I(g) =_{def} \mathbb{N} \setminus I(p)$. Wir definieren für $^-$-Gatter die Vorgängerfunktion $p(g) =_{def} p$.*
3. *Für ein Gatter g mit Beschriftung $\alpha(g) = \sigma \in \{\cup, \cap, +, \times, /\}$ und Vorgängern g_1 und g_2 mit $g_1 \leq g_2$ ist $I(g) =_{def} I(g_1)\sigma I(g_2)$. Wir nennen g_1 den ersten und g_2 den zweiten Vorgänger von g und definieren die Vorgängerfunktionen $p_1(g) =_{def} g_1$ und $p_2(g) =_{def} g_2$.*

$I(C) =_{def} I(g_c)$ *ist die* ***Ergebnismenge des Schaltkreises*** C*. Falls wir mehrere arithmetische Schaltkreise gleichzeitig betrachten, schreiben wir zur Differenzierung manchmal* I_C *statt* I*.*

Bemerkung *1. Im Fall* $\alpha(g) = \sigma \in \{\cup, \cap, +, \times\}$ *für ein Gatter* g *gilt wegen der Kommutativität der Operation* σ*:*

$$I(p_1(g))\sigma I(p_2(g)) = I(p_2(g))\sigma I(p_1(g))$$

Die Reihenfolge der Vorgänger spielt daher keine Rolle für die Ergebnismenge des Gatters.

2. *Die Festlegung der Reihenfolge der Vorgänger der /-Gatter durch die Nummerierung der Gatter scheint zunächst eine Einschränkung zu sein, denn so ist es nicht möglich einen Schaltkreis zu konstruieren, in dem es zwei /-Gatter gibt, deren Vorgänger wechselseitig vertauscht sind. Dieses Problem lässt sich durch die Einführung eines „Dummy-Gatters“ lösen: Wenn wir ein Gatter* g *beschreiben wollen, dessen gewünschter zweiter Vorgänger* g_2 *kleiner als der erste Vorgänger* g_1 *ist, so führen wir ein neues /-Gatter* g_d *mit* $g_d > g_1$ *ein. Der erste Vorgänger von* g_d *ist* g_2 *und der zweite Vorgänger von* g_d *ist ein neues Eingabegatter* e_1 *mit* $\alpha(e_1) = 1$*. Dann ist* $I(g_d) = I(g_2)/I(e_1) = I(g_2)/\{1\} = I(g_2)$*. Wir verwenden nun* g_d *an Stelle von* g_2*. Die Größe des Schaltkreises ändert sich dadurch nur unwesentlich. Damit können wir o.B.d.A. davon ausgehen, dass wir jeden* $\mathcal{O}$*-Schaltkreise, den wir zeichnen können, auch formal beschreiben können.*

3. *Wenn ein Gatter* $g \in V$ *mit* $\alpha(g) = /$ *als Eingangskante eine Kante* $(p, g) \in V^2$ *mit* $m((p, g)) = 2$ *hat, so ist* $p_1(g) = p = p_2(g)$*.*

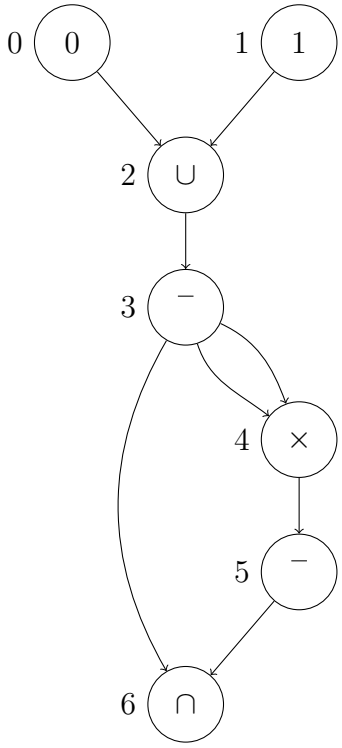
McKenzie und Wagner beschreiben in [MW07] (Beispiel 2.1) wie die Menge der Primzahlen $\mathbb{P}$ als Schaltkreis dargestellt werden kann. Formal können wir diesen Schaltkreis wie folgt darstellen:

Beispiel 2.1
Sei $C = (V, (E', m), \alpha, 6)$ *ein arithmetischer Schaltkreis mit Gattern* $V = \{0, 1, \ldots, 6\}$*, Kanten* $E' = \{(0, 2), (1, 2), (2, 3), (3, 6), (3, 4), (4, 5), (5, 6)\}$ *mit Vielfachheit*

$$m(e) = \begin{cases} 2 & \text{falls } e = (3, 4) \\ 1 & \text{sonst} \end{cases}$$

und Gatter-Beschriftungen $\alpha : 0 \mapsto 0, 1 \mapsto 1, 2 \mapsto \cup, 3 \mapsto {}^-, 4 \mapsto \times, 5 \mapsto {}^-, 6 \mapsto \cap$*.*

Dies ist eine Zeichnung des Schaltkreises:



Es ergeben sich die folgenden Ergebnismengen:

$$
\begin{aligned}
&I(0) = \{0\} \\
&I(1) = \{1\} \\
&I(2) = I(0) \cup I(1) = \{0, 1\} \\
&I(3) = \overline{I(2)} = \overline{\{0, 1\}} \\
&I(4) = I(3) \times I(3) = \{xy \mid x, y \geq 2\} \\
&I(5) = \overline{I(4)} = \mathbb{P} \cup \{0, 1\} \\
&I(6) = I(3) \cap I(5) = \overline{\{0, 1\}} \cap (\mathbb{P} \cup \{0, 1\}) = \mathbb{P}
\end{aligned}
$$

$I(C)$ ist also die Menge der Primzahlen.

2.1.2 Das Membership-Problem

Für arithmetische Schaltkreise können verschiedene Entscheidungsprobleme untersucht werden. Diese Arbeit beschäftigt sich hauptsächlich mit dem Membership-Problem.

Definition
Für $\mathcal{O} \subseteq \{\cup, \cap, ^-, +, \times, /\}$ wird

$$MC(\mathcal{O}) =_{def} \{(C, b) \mid C \text{ ist ein } \mathcal{O}\text{-Schaltkreis und } b \in \mathbb{N}, \text{ sodass } b \in I(C)\}$$

*als das **Membership-Problem für $\mathcal{O}$-Schaltkreise** bezeichnet. Wir verwenden die Kurzschreibweise $MC(\sigma_1, \ldots, \sigma_n)$ für $MC(\{\sigma_1, \ldots, \sigma_n\})$.*

An einigen Stellen werden wir auch das Emptiness-Problem für Schaltkreise benötigen. Dieses wurde für Teilmengen der Operationen $\{\cup, \cap, ^-, +, \times\}$ ausführlich in [Bar+17] untersucht.

Definition
Für $\mathcal{O} \subseteq \{\cup, \cap, ^-, +, \times, /\}$ wird

$$EC(\mathcal{O}) =_{def} \{C \mid C \text{ ist ein } \mathcal{O}\text{-Schaltkreis und } I(C) = \emptyset\}$$

*als das **Emptiness-Problem für $\mathcal{O}$-Schaltkreise** bezeichnet. Wir verwenden die Kurzschreibweise $EC(\sigma_0, \ldots, \sigma_n)$ für $EC(\{\sigma_1, \ldots, \sigma_n\})$.*

Um die Entscheidungsprobleme auf ihre Komplexität zu untersuchen, müssen wir sie als Eingaben von Turingmaschinen verstehen. Dafür codieren wir einen Schaltkreis $C = (V, E, g_c, \alpha)$ wie folgt:

Definition
Der Schaltkreis wird als Liste seiner Gatter in topologischer Reihenfolge dargestellt. Jedes Gatter $g \in V$ hat dabei die Darstellung:

$$(g, \alpha(g), l(g))$$

Dabei ist $l(g)$ die Liste der Nachfolger von g. Falls g eine ausgehende Doppelkante hat, so wird der entsprechende Vorgänger zweimal aufgezählt. Diese Liste der Gatter codieren wir in Bits, also über dem Alphabet $\{0, 1\}$. Die Länge der Codierung in Bits schreiben wir als $|C|$.

Beispiel 2.2
Eine Codierung für den Schaltkreis aus dem obigen Beispiel ist:

$$(0, 0, <2>), (1, 1, <2>), (2, \cup, <3>),$$
$$(3, ^-, <4, 4, 6>), (4, \times, <5>), (5, ^-, <6>), (6, \cap, <>)$$

Bemerkung
Der Wert $|C|$ hängt sowohl von der Größe des zu Grunde liegenden Multigraphen als auch von den Beschriftungen an den Eingabegattern ab.

Bemerkung
Die topologische Sortierung der Kanten ist möglich, weil der Graph azyklisch ist. Es ist möglich in logarithmischem Raum deterministisch zu überprüfen, ob die angegebene Sortierung topologisch ist. Damit kann in logarithmischem Raum deterministisch geprüft werden, ob die Eingabe eine gültige Codierung eines Schaltkreises ist.

2.2 Komplexitätstheorie

Dieser Abschnitt enthält die Definitionen der Komplexitätsklassen und weiterer Begriffe aus der Komplexitätstheorie, die in der Arbeit vorkommen. Es wird davon ausgegangen, dass Lesende mit der Landau-Notation und mit Turingmaschinen mit mehreren Bändern (k-Band-Turingmaschinen) vertraut sind. Wir werden außerdem einfache Aussagen über arithmetische Schaltkreise treffen, die in Kapitel 3 und 4 an mehreren Stellen Verwendung finden.

2.3 Komplexitätsklassen

Zeitklassen DTIME(f) ist die Klasse aller Sprachen, die von einer deterministischen k-Band-Turing-Maschine erkannt werden, die der Zeitschranke $f : \mathbb{N} \to \mathbb{N}$ genügt. Dabei muss $f : \mathbb{N} \to \mathbb{N}$ eine totale Funktion sein, für die $f(n) > n$ für alle $n \in \mathbb{N}$ gilt. Die Maschine M genügt der Zeitschranke f, falls für jede Eingabe w über dem Eingabealphabet von M, jeder Rechenweg aus maximal $f(|w|)$ Schritten besteht. (Für Zeitschranken $f : \mathbb{N} \to \mathbb{N}$, sodass es $n \in \mathbb{N}$ mit $f(n) \leq n$ gibt, wäre die Definition nicht sinnvoll, denn die Maschine hätte nicht mal genug Zeit, um die Eingabe vollständig zu lesen und festzustellen, dass das Ende der Eingabe erreicht ist.) NTIME(f) wird analog für nichtdeterministische Turingmaschinen definiert. In dieser Arbeit werden die folgenden Zeitklassen verwendet:

$$\begin{aligned} \mathrm{P} &= \bigcup_{k \geq 1} \mathrm{DTIME}\left(O(n^k)\right) \\ \mathrm{NP} &= \bigcup_{k \geq 1} \mathrm{NTIME}\left(O(n^k)\right) \\ \mathrm{E} &= \mathrm{DTIME}\left(2^{O(n)}\right) \\ \mathrm{NEXP} &= \bigcup_{k \geq 1} \mathrm{NTIME}\left(2^{O(n^k)}\right) \end{aligned}$$

Raumklassen DSPACE(f) ist die Klasse aller Sprachen, die von einer deterministischen k-Band-Turing-Maschine erkannt werden, die der Raumschranke $f : \mathbb{N} \to \mathbb{N}$ genügt. Dabei muss $f : \mathbb{N} \to \mathbb{N}$ eine totale Funktion sein. Die Maschine M genügt der Raumschranke f, falls für jede Eingabe w über dem Eingabealphabet von M maximal der Raum $f(|w|)$ verwendet wird. Der verwendete Raum bezieht sich dabei darauf, wie viele Felder die Turingmaschine bei Einlesen von w auf den Arbeitsbändern verwendet. NSPACE(f) wird analog für nichtdeterministische Turingmaschinen definiert. In dieser Arbeit werden die folgenden Raumklassen verwendet:

$$\begin{aligned} \mathrm{L} &= \mathrm{DSPACE}\left(O(\log(n))\right) \\ \mathrm{NL} &= \mathrm{NSPACE}\left(O(\log(n))\right) \\ \mathrm{PSPACE} &= \bigcup_{k \geq 1} \mathrm{DSPACE}\left(O(n^k)\right) \end{aligned}$$

Dabei definieren wir $\log : \mathbb{N} \to \mathbb{N}$ als $\log(n) = \lfloor \log_2(n) \rfloor$ für alle $n > 0$ und $\log(0) = 0$.

Probabilistische Komplexitätsklassen PSPACE(s) ist die Klasse aller Sprachen, die von einer probabilistischen Turingmaschine M mit einem Eingabeband und einem Arbeitsband im Raum $s : \mathbb{N} \to \mathbb{N}$ erkannt werden. Eine probabilistische Turingmaschine ist dabei wie eine nichtdeterministische Turingmaschine definiert, für die aber immer

nur ein Weg verfolgt wird. Es gibt an jeder Verzweigung jeweils zwei Möglichkeiten, die jeweils mit der Wahrscheinlichkeit $\frac{1}{2}$ gewählt werden. Ein Rechenweg α mit n Verzweigungen hat damit die Wahrscheinlichkeit $prob(\alpha) = (\frac{1}{2})^n$. Die Wahrscheinlichkeit $prob_M(w)$, dass eine Eingabe w akzeptiert wird, ist die Summe der Wahrscheinlichkeiten aller Rechenwege auf denen w akzeptiert wird. Die von der probabilistischen Turingmaschine M erkannte Sprache ist die Menge aller Wörter w über dem Eingabealphabet, die mit einer Wahrscheinlichkeit $prob_M(w) > \frac{1}{2}$ akzeptiert werden. In dieser Arbeit wird ausschließlich die probabilistische Raumklasse PL verwendet. Diese ist wie folgt definiert:

$$\mathrm{PL} = \mathrm{PSPACE}\,(\log(n))$$

Zählklassen Für ein Alphabet Σ ist #L die Klasse aller Funktionen $f : \Sigma^* \to \mathbb{N}$, sodass es eine nichtdeterministische Turingmaschine M gibt, die Eingaben $w \in \Sigma^*$ in logarithmischem Raum erkennt und dabei genau $f(w)$ akzeptierende Rechenwege hat. C$_=$L ist die Klasse aller Sprachen A für die es Funktionen $f_0, f_1 \in$ #L gibt, sodass w genau dann in A ist, wenn $f_0(w) = f_1(w)$ ist.

Polynomial Identity Testing PIT (Polynomial Identity Testing) ist das folgende Problem: Wir betrachten die Menge der $\{+, \times\}$-Schaltkreise C über den ganzen Zahlen mit unbelegten Eingängen $u_1, \ldots, u_n$. Wir assoziieren den Schaltkreis C mit einem Polynom mit den Variablen $x_1, \ldots, x_n$ aus $\mathbb{Z}$. C ist genau dann in PIT, wenn $I(C(x_1, \ldots, x_n)) = \{0\}$ für alle Belegungen $x_1, \ldots, x_n \in \mathbb{Z}$ ist. $\mathcal{PIT}$ ist die Klasse der Probleme, die $\leq_m^{\log}$-reduzierbar auf PIT sind.

2.4 Reduzierbarkeit

FL ist die Klasse aller Funktionen $f : \Sigma_1^* \to \Sigma_2^*$, die von einer deterministischen k-Band-Turingmaschine mit Ausgabe im Raum $O(\log(n))$ berechnet werden. Eine Sprache $A_1 \subseteq \Sigma_1^*$ ist auf eine Sprache $A_2 \subseteq \Sigma_2^*$ in logarithmische Raum many-one-reduzierbar, falls es eine Funktion $f \in$ FL gibt, sodass für alle $w \in \Sigma_1^*$ gilt:

$$w \in A_1 \Leftrightarrow f(w) \in A_2$$

Das bedeutet, wir können überprüfen, ob $w \in A_1$ ist, indem wir $f(w)$ berechnen und prüfen, ob $f(w)$ in A_2 ist. Wir schreiben dafür $A_1 \leq_m^{\log} A_2$. Falls $A_1 \leq_m^{\log} A_2$ und $A_2 \leq_m^{\log} A_1$ gilt, so schreiben wir $A_1 \equiv_m^{\log} A_2$. Es ist bekannt, dass FL unter Komposition abgeschlossen ist, und dass $\leq_m^{\log}$ reflexiv und transitiv ist.

Alle im obigen Abschnitt genannten Raum- und Zeitklassen außer E sind unter $\leq_m^{\log}$ abgeschlossen. Das bedeutet, wenn wenn A_2 in der Klasse $\mathcal{C}$ ist und $A_1 \leq_m^{\log} A_2$ gilt, dann ist auch A_1 in $\mathcal{C}$.

Wir sagen, dass A_2 $\leq_m^{\log}$-hart für eine Komplexitätsklasse $\mathcal{C}$ ist, falls $A_1 \leq_m^{\log} A_2$ für alle $A_1 \in \mathcal{C}$ gilt. Falls außerdem $A_2 \in \mathcal{C}$ ist, so sagen wir, dass A_2 $\leq_m^{\log}$-vollständig für $\mathcal{C}$ ist. Ein solches $\leq_m^{\log}$-vollständiges Problem, können wir uns als „eines der schwierigsten Probleme“ in $\mathcal{C}$ vorstellen, denn wir könne alle anderen Probleme aus $\mathcal{C}$ darauf reduzieren.

Analog zu $\leq_m^{\log}$ definieren wir $\leq_m^{\mathrm{P}}$ für polynomielle Zeit und $\leq_m^{\mathrm{PSPACE}}$ für polynomiellen Raum. Falls klar ist, über welche Reduzierbarkeit $\leq$ gesprochen wird, so sagen wir, dass ein Problem A $\mathcal{C}$-hart (bzw. $\mathcal{C}$-vollständig) ist, falls A $\leq$-hart (bzw. $\leq$-vollständig) für $\mathcal{C}$ ist.

2.5 Einfache Aussagen für Schaltkreise

Für $\emptyset \neq \mathcal{O}_1 \subseteq \mathcal{O}_2 \subseteq \{\cup, \cap, ^-, +, \times, /\}$ können wir jeden $\mathcal{O}_1$-Schaltkreis als einen $\mathcal{O}_2$-Schaltkreis auffassen. Daher erhalten wir folgendes Lemma, mit dem sich Aussagen über die Schwierigkeit von $MC(\mathcal{O}_1)$ und $MC(\mathcal{O}_2)$ aufeinander übertragen lassen:

Lemma 2.3
Seien $\emptyset \neq \mathcal{O}_1 \subseteq \mathcal{O}_2 \subseteq \{\cup, \cap, ^-, +, \times, /\}$ und $\mathcal{C}$ eine Komplexitätsklasse. Dann gilt:

1. $MC(\mathcal{O}_1) \leq_m^{\log} MC(\mathcal{O}_2)$
2. *Wenn $MC(\mathcal{O}_1)$ $\leq_m^{\log}$-hart für $\mathcal{C}$ ist, so ist auch $MC(\mathcal{O}_2)$ $\leq_m^{\log}$-hart für $\mathcal{C}$.*
3. *Wenn $MC(\mathcal{O}_2) \in \mathcal{C}$ und $\mathcal{C}$ unter $\leq_m^{\log}$ abgeschlossen ist, so ist auch $MC(\mathcal{O}_1) \in \mathcal{C}$.*

Beweis.
1. $MC(\mathcal{O}_1) \subseteq MC(\mathcal{O}_2)$. Das heißt, $MC(\mathcal{O}_1)$ kann mit der Identitätsfunktion auf $MC(\mathcal{O}_2)$ reduziert werden.
2. Sei $A \in \mathcal{C}$. Dann ist $A \leq_m^{\log} MC(\mathcal{O}_1)$ Es gilt $MC(\mathcal{O}_1) \leq_m^{\log} MC(\mathcal{O}_2)$. Da $\leq_m^{\log}$ transitiv ist, ist damit auch $A \leq_m^{\log} MC(\mathcal{O}_2)$.
3. Es sei $MC(\mathcal{O}_2) \in \mathcal{C}$. Wegen $MC(\mathcal{O}_1) \leq_m^{\log} MC(\mathcal{O}_2)$ und der Abgeschlossenheit von $\mathcal{C}$ unter $\leq_m^{\log}$, ist damit auch $MC(\mathcal{O}_1) \in \mathcal{C}$.

□

2.6 Alternierende Maschinen

Alternierende Maschinen sind ein Modell für Parallelrechner. Wir werden an einigen Stellen in Kapitel 3 alternierende Maschinen verwenden, um das Membership-Problem für verschiedene Mengen von Operationen zu lösen. Dabei werden wir die Frage, ob eine natürliche Zahl x Element der Ergebnismenge eines Gatters g ist beantworten, indem wir raten aus welchen Zahlen x_1 und x_2 berechnet wurde und parallel überprüfen, ob x_1 in der Ergebnismenge von $p_1(g)$ und x_2 in der Ergebnismenge von $p_2(g)$ liegt.

Definition
*Formal ist eine **alternierende Maschine** eine nichtdeterministische Turingmaschine, deren Zustände entweder als akzeptierend, ablehnend, existenziell oder universell gekennzeichnet sind.*

Zu einer alternierenden Maschine M und einer Eingabe x wird der **Berechnungsbaum** $\beta_M(x)$ definiert, dessen Knoten Konfigurationen von M sind. Die Konfigurationen enthalten jeweils die Bandinhalte, Kopfpositionen und den Zustand von M in einem

Takt. Die Wurzel von $\beta_M(x)$ ist die Startkonfiguration von M bei Eingabe x. Jede Konfiguration in $\beta_M(x)$ ist über gerichtete Kanten mit ihren Nachfolgekonfigurationen verbunden. Konfigurationen werden dabei entsprechend ihrem Zustand als akzeptierend, ablehnend, existenziell oder universell bezeichnet. Ein **akzeptierender Teilbaum** β von $\beta_M(x)$ enthält

- die Wurzel von $\beta_M(x)$,
- für jede existenzielle Konfiguration in β genau eine Nachfolgekonfiguration,
- für jede universelle Konfiguration in β alle Nachfolgekonfigurationen

und jedes Blatt von β ist eine akzeptierende Konfiguration.

M akzeptiert x genau dann, wenn $\beta_M(x)$ einen akzeptierenden Teilbaum hat. Die Menge der von M akzeptierten Wörter schreiben wir als $L(M)$.

Die akzeptierenden, ablehnenden und existenziellen Zustände entsprechen also den Zuständen einer nichtdeterministischen Turingmaschine. Die universellen Zustände erweitern das Modell.

M **akzeptiert** die Sprache $L(M)$ in Zeit $t : \mathbb{N} \to \mathbb{N}$, falls es für jedes $x \in L(M)$ einen akzeptierenden Teilbaum von $\beta_M(x)$ mit Tiefe $\leq t(|x|)$ gibt.

Die Komplexitätsklasse ATIME(t) ist die Menge aller Sprachen L, die von einer alternierenden k-Band-Turingmaschine in Zeit $t : \mathbb{N} \to \mathbb{N}$ akzeptiert werden.

Eine Funktion $s : \mathbb{N} \to \mathbb{N}$ heißt raumkonstruierbar, falls es eine deterministische k-Band-Turingmaschine gibt, die auf allen Eingaben der Form 1^n hält und dabei genau den Raum $s(n)$ verwendet. Durch Simulation aller Pfade der Turingmaschine lässt sich beweisen:

Satz 2.4
Für raumkonstruierbare $s : \mathbb{N} \to \mathbb{N}$ mit $s(n) \geq n$ für alle $n \in \mathbb{N}$ gilt:

$$\text{ATIME}(s) \subseteq \text{DSPACE}(s)$$

3 Obere Schranken

In diesem Kapitel untersuchen wir obere Schranke für Membership-Probleme mit verschiedenen Operationen. Dabei betrachten wir in je einem Abschnitt Membership-Probleme ohne Addition, ohne Komplement und ohne Multiplikation. Dadurch, dass wir jeweils eine Operation weglassen, wird das Membership-Problem jeweils einfach genug, um es durch einen Algorithmus zu lösen.

3.1 Schwierigkeit bei allgemeinen Schaltkreisen

Das allgemeine Problem $MC\{\cup, \cap, ^-, +, \times, /\}$ lösen wir in dieser Arbeit nicht. Wir können nach Lemma 2.3 $MC\{\cup, \cap, ^-, +, \times\}$ in logarithmischem Raum darauf many-one-reduzieren. McKenzie und Wagner haben in [MW07] (Beispiel 2.2) bewiesen, dass sich die Goldbachsche Vermutung darauf zurückführen lässt. Diese Vermutung ist eine sehr bekannte, unbewiesene Aussage aus der Zahlentheorie.

Definition
Die Goldbachsche Vermutung besagt, dass jede gerade Zahl größer als 2 sich als Summe zweier Primzahlen darstellen lässt.

Der von McKenzie und Wagner konstruierte Schaltkreise berechnet die Menge

$$M_{Goldbach} =_{def} (\mathbb{N}^{\geq 2} \times \{2\}) \cap (\overline{\mathbb{P} + \mathbb{P}})$$

wobei $\mathbb{N}^{\geq 2}$ die Menge der natürlichen Zahlen größer 2 sei. Wir haben in Beispiel 2.1 bereits gesehen, wie $\mathbb{N}^{\geq 2}$ und $\mathbb{P}$ durch Schaltkreise dargestellt werden können. ($\mathbb{N}^{\geq 2}$ war in dem Beispiel die Ergebnismenge des Gatters 3.) Die Goldbachsche Vermutung ist genau dann wahr, wenn $M_{Goldbach}$ leer ist. Dies ist genau dann der Fall, wenn $0 \in \overline{\{0\} \times M_{Goldbach}}$ und das lässt sich als Instanz von $MC\{\cup, \cap, ^-, +, \times\}$ formulieren. Ein Algorithmus für $MC\{\cup, \cap, ^-, +, \times\}$ könnte also insbesondere verwendet werden, um zu berechnen, ob die Goldbachsche Vermutung wahr ist.

3.2 Schaltkreise ohne Multiplikation

In diesem Abschnitt werden wir sehen, wie sich das Membership-Problem für arithmetische Schaltkreise ohne Multiplikation auf alternierenden Maschinen lösen lässt. Dafür benötigen wir zunächst zwei technische Hilfsaussagen.

Glaßer et al. stellen in Proposition 1 aus [Gla+08] fest, dass es für $\{\cup, \cap, ^-, +\}$-Schaltkreise eine Schranke gibt, ab der entweder alle Werte durch den Schaltkreis beschrieben werden oder keine. Diese Eigenschaft bleibt erhalten, wenn wir auch /-Gatter erlauben. Formal erhalten wir folgendes Lemma:

Lemma 3.1
Sei C ein $\mathcal{O}$-Schaltkreis mit $\emptyset \neq \mathcal{O} \subseteq \{\cup, \cap, ^-, +, /\}$. Dann gibt es ein $n \leq 2^{|C|} + 1$, sodass für alle $z \geq n$ gilt:

$$z \in I(C) \Leftrightarrow n \in I(C)$$

Beweis. Wir führen eine Induktion über die Struktur von C. Sei dabei C_g der Schaltkreis, der aus C entsteht, wenn man g als Ausgabegatter verwendet und alle Gatter, von denen aus g nicht erreichbar ist, entfernt. Wir zeigen, dass es für alle Gatter g ein $n_g \leq 2^{|C_g|}+1$ gibt, sodass für alle $z \geq n_g$ gilt:

$$z \in I(g) \;\Leftrightarrow\; n_g \in I(g)$$

Das Lemma folgt dann, indem man diese Aussage auf das Ausgabegatter von C anwendet.

IA: Sei g ein Eingabegatter. Dann enthält C_g nur das Gatter g und es gilt $I(g) = \{\alpha(g)\}$. Sei $n_g = 2^{|C_g|}+1$. Für alle $z \geq n_g$ gilt dann $z \geq 2^{|C_g|}+1 \geq 2^{\log(\alpha(g))+1}+1 > \alpha(g)$ und $z \notin I(g)$.

IV: Sei g ein Gatter aus C mit Vorgängern. Die Aussage sei für $p(g)$ bzw. $p_1(g)$ und $p_2(g)$ bereits bewiesen.

IS: Wir zeigen zunächst die folgende **Hilfsaussage**: Für alle Gatter g, sodass $\alpha(g)$ eine zweistellige Operation ist, gilt $n_{p_1(g)} + n_{p_2(g)} \leq 2^{|C_g|} + 1$.
Beweis der Hilfsaussage:

$$\begin{aligned}
n_{p_1(g)} + n_{p_2(g)} &\leq (2^{|C_{p_1(g)}|} + 1) + (2^{|C_{p_2(g)}|} + 1) \text{ (nach IV)} \\
&= 2^{|C_{p_1(g)}|} + 2^{|C_{p_2(g)}|} + 2 \\
&\leq 2 \cdot 2^{max\{|C_{p_1(g)}|,|C_{p_2(g)}|\}} + 2 \\
&= 2^{max\{|C_{p_1(g)}|,|C_{p_2(g)}|\}+1} + 2 \\
&= 2^{max\{|C_{p_1(g)}|,|C_{p_2(g)}|\}+2}/2 + 2 \\
&\leq 2^{|C_g|}/2 + 2 \text{ (siehe unten)} \\
&\leq 2^{|C_g|} + 1 \text{ (wegen } |C_g| \geq 1)
\end{aligned}$$

Die vorletzte Ungleichung gilt, weil zur Konstruktion von C_g aus $C_{p_1(g)}$ und $C_{p_2(g)}$ zwei Kanten (oder eine Doppelkante) hinzugefügt werden müssen. Dadurch hat die Codierung von C_g mindestens zwei Zeichen mehr als die Codierungen von $C_{p_1(g)}$ und $C_{p_2(g)}$. Damit ist die Hilfsaussage bewiesen.
Fall 1: Es gilt $\alpha(g) = \cup$.
Sei $n_g =_{def} max\{n_{p_1(g)}, n_{p_2(g)}\}$ und $z \geq n_g$.

$$\begin{aligned}
z \in I(g) &\Leftrightarrow z \in I(p_1(g)) \vee z \in I(p_2(g)) \\
&\Leftrightarrow n_{p_1(g)} \in I(p_1(g)) \vee n_{p_2(g)} \in I(p_2(g)) \text{ (nach IV)} \\
&\Leftrightarrow n_g \in I(p_1(g)) \vee n_g \in I(p_2(g)) \text{ (nach IV)} \\
&\Leftrightarrow n_g \in I(p_1(g)) \cup I(p_2(g)) \\
&\Leftrightarrow n_g \in I(g)
\end{aligned}$$

Nach der Hilfsaussage gilt: $n_g = max\{n_{p_1(g)}, n_{p_2(g)}\} \leq n_{p_1(g)} + n_{p_2(g)} \leq 2^{|C_g|} + 1$.

Fall 2: Es gilt $\alpha(g) = \cap$. (analog zu Fall 1)

Fall 3: Es gilt $\alpha(g) = ^-$.
Sei $n_g =_{def} n_{p(g)}$ und $z \geq n_g$. Dann gilt:

$$z \in I(g) \Leftrightarrow z \notin I(p(g)) \overset{\text{IV}}{\Leftrightarrow} n_g \notin I(p(g)) \Leftrightarrow n_g \in \overline{I(p(g))} \Leftrightarrow n_g \in I(g)$$

Es gilt: $n_g = n_{p(g)} \leq 2^{|C_{p(g)}|} + 1 \leq 2^{|C_g|} + 1$.
Fall 4: Es gilt $\alpha(g) = +$.
Sei $n_g =_{def} n_{p_1(g)} + n_{p_2(g)}$ und $z \geq n_g$. Nach der Hilfsaussage gilt:

$$n_g = n_{p_1(g)} + n_{p_2(g)} \leq 2^{|C_g|} + 1$$

Fall 4 a: Es sei $n_{p_1(g)} \notin I(p_1(g))$ und $n_{p_2(g)} \notin I(p_2(g))$.
Dann gilt: $I(p_1(g)) \subseteq [0, n_{p_1(g)})$ und $I(p_2(g)) \subseteq [0, n_{p_2(g)})$. Also ist

$$I(g) = I(p_1(g)) + I(p_2(g)) \subseteq [0, n_{p_1(g)} + n_{p_2(g)}) = [0, n_g)$$

Das heißt, für alle $z \geq n_g$ ist $z \notin I(g)$.

Fall 4 b: Es sei $n_{p_1(g)} \in I(p_1(g))$ und $n_{p_2(g)} \notin I(p_2(g))$.
Falls $I(p_2(g))$ leer ist, dann gilt die Aussage offensichtlich. Sei nun $I(p_2(g))$ nicht leer. Dann existiert $a \in I(p_2(g))$. Nach IV gilt $a < n_{p_2(g)}$. Also gilt für alle $z \geq n_g$:

$$\begin{aligned} &(z-a) > (n_{p_1(g)} + n_{p_2(g)}) - n_{p_2(g)} = n_{p_1(g)} \\ \Rightarrow\ &(z-a) \in I(p_1(g)) \text{ (nach IV)} \\ \Rightarrow\ &z = (z-a) + a \in I(p_1(g)) + I(p_2(g)) = I(g) \end{aligned}$$

Fall 4 c: Es sei $n_{p_1(g)} \notin I(p_1(g))$ und $n_{p_2(g)} \in I(p_2(g))$. (analog zu Fall 4 b)

Fall 4 d: Es sei $n_{p_1(g)} \in I(p_1(g))$ und $n_{p_2(g)} \in I(p_2(g))$.
Sei $z \geq n_g$. Dann ist $(z - n_{p_1(g)}) \geq n_g - n_{p_1(g)} = n_{p_2(g)}$ und damit $z = n_{p_1(g)} + (z - n_{p_1(g)}) \in I(p_1(g)) + I(p_2(g)) = I(g)$ nach IV.

Fall 5: Es gilt $\alpha(g) = /$.
Sei $n_g =_{def} n_{p_1(g)}$. Nach der Hilfsaussage gilt:

$$n_g = n_{p_1(g)} \leq n_{p_1(g)} + n_{p_2(g)} \leq 2^{|C_g|} + 1$$

Fall 5 a: Es sei $n_{p_1(g)} \notin I(p_1(g))$.
Dann gilt $I(p_1(g)) \subseteq [0, n_{p_1(g)})$ nach IV und damit:

$$I(g) = I(p_1(g))/I(p_2(g)) \subseteq [0, n_{p_1(g)}) = [0, n_g)$$

Also ist $z \notin I(g)$ für alle $z \geq n_g$.

Fall 5 b: Es sei $n_{p_1(g)} \in I(p_1(g))$.
Falls $I(p_2(g)) \subseteq \{0\}$ ist, dann ist $I(g)$ leer und die Aussage gilt offensichtlich. Sei nun $I(p_2(g)) \not\subseteq \{0\}$. Dann gibt es $a \in I(p_2(g))$ mit $a > 0$. Für alle $z \geq n_g$ ist $za \geq n_g = n_{p_1(g)}$. Damit ist $za \in I(p_1(g))$ nach IV und $z = za/a \in I(p_1(g))/I(p_2(g)) = I(g)$.

□

Folgerung 3.2 *Sei C ein $\mathcal{O}$-Schaltkreis mit $\emptyset \neq \mathcal{O} \subseteq \{\cup, \cap, ^-, +, /\}$. Dann gilt für alle $z \geq 2^{|C|} + 1$:*

$$z \in I(C) \Leftrightarrow 2^{|C|} + 1 \in I(C)$$

Beweis. Sei n wie in Lemma 3.1 und $z \geq 2^{|C|} + 1 \geq n$. Dann gilt:

$$z \in I(C) \Leftrightarrow n \in I(C) \Leftrightarrow 2^{|C|} + 1 \in I(C)$$

□

Damit kann das Membership-Problem für $\{\cup, \cap, ^-, +, /\}$-Schaltkreise auf alternierenden Maschinen gelöst werden. Wir werden für eine Eingabe (C, b) prüfen, ob es Werte $x \in I(g)$ für die Gatter g gibt, die $b \in I(C)$ beweisen. Dabei verwenden wir Folgerung 3.2, um die Größe der Werte x auf $O(|C|)$ Bits zu beschränken: Falls ein Wert x zu groß ist, können wir $2^{|C|} + 1 \in I(g)$ statt $x \in I(g)$ prüfen. Damit erhalten wir einen Algorithmus, der in polynomieller, alternierender Zeit arbeitet. Für Schaltkreise, die auch die $\times$-Gatter enthalten, funktioniert dieser Ansatz nicht, da die Werte in den Ergebnismengen zu groß werden können.

Satz 3.3
Für alle $\emptyset \neq \mathcal{O} \subseteq \{\cup, \cap, ^-, +, /\}$ gilt $MC(\mathcal{O}) \in$ PSPACE.

Beweis. Nach De Morgan gilt:

$$MC(\cup, \cap, ^-, +, /) \equiv_m^{\log} MC(\cup, ^-, +, /)$$

Algorithmus 1 löst das Problem $MC(\cup, ^-, +, /)$ für gültige Eingaben (C, b) mittels einer alternierenden Maschine.

Algorithm 1 Algorithmus für Schaltkreise mit Vereinigung, Komplement, Addition und Division

Eingabe: (C, b)

Sei $m = 2^{|C|} + 1$, $g = g_C$, $x = b$ und $t = 1$.
while $\alpha(g) \notin \mathbb{N}$ **do**
  if $x > m$ **then**
    Sei $x = m$.
  end if
  // Falls $t == 1$ teste $x \in I(g)$ und falls $t == 0$ teste $x \notin I(g)$.
  if $\alpha(g) == ^-$ **then**
    Sei $g = p(g)$ und $t = 1 - t$.
  else if $\alpha(g) == \cup$ und $t == 0$ **then**
    Setze mittels universellem Zustand parallel $g = p_1(g)$ und $g = p_2(g)$.
  else if $\alpha(g) == \cup$ und $t == 1$ **then**
    Rate $i \in \{1, 2\}$.
    Sei $g = p_i(g)$.
  else if $\alpha(g) == +$ und $t == 1$ **then**

15: Rate bitweise ein binär codiertes $a \in [0, x] \cap \mathbb{N}$.
16: Setze mittels universellem Zustand parallel $(x, g) = (x - a, p_1(g))$ und $(x, g) = (a, p_2(g))$.
17: **else if** $\alpha(g) == +$ und $t == 0$ **then**
18: Setze mittels universeller Zustände a parallel auf jeden Wert aus $[0, x] \cap \mathbb{N}$.
19: Rate $i \in \{0, 1\}$.
20: **if** $i == 0$ **then**
21: Sei $(x, g) = (x - a, p_1(g))$
22: **else**
23: Sei $(x, g) = (a, p_2(g))$.
24: **end if**
25: **else if** $\alpha(g) == /$ und $t == 1$ **then**
26: Rate bitweise ein binär codiertes $a \in [1, m] \cap \mathbb{N}$.
27: Setze mittels universellem Zustand parallel $(x, g) = (a \cdot x, p_1(g))$ und $(x, g) = (a, p_2(g))$.
28: **else if** $\alpha(g) == /$ und $t == 0$ **then**
29: Setze mittels universeller Zustände a parallel auf jeden Wert aus $[1, m] \cap \mathbb{N}$.
30: Rate $i \in \{1, 2\}$.
31: **if** $i == 0$ **then**
32: Sei $(x, g) = (a \cdot x, p_1(g))$
33: **else**
34: Sei $(x, g) = (a, p_2(g))$.
35: **end if**
36: **end if**
37: **end while**
38: // g ist ein Eingabegatter.
39: **if** $t == 1$ **then**
40: Akzeptiere, falls $x == \alpha(g)$ gilt. Andernfalls lehne ab.
41: **else**
42: Lehne ab falls $x == \alpha(g)$ gilt. Andernfalls akzeptiere.
43: **end if**

Wir wollen zunächst die Korrektheit von Algorithmus 1 zeigen. Sei C ein $\{\cup, \bar{}, +, /\}$-Schaltkreis und $b \in \mathbb{N}$. Sei $\beta_M(C, b)$ der Berechnungsbaum des Algorithmus bei Eingabe (C, b). Wir wollen zeigen, dass $\beta_M(C, b)$ genau dann einen akzeptierenden Teilbaum hat, wenn $b \in I(C)$ gilt. Für ein Gatter g' aus C, $x' \in \mathbb{N}$ und $t' \in \{0, 1\}$ sei $\beta(g', x', t')$ ein Teilbaum von $\beta_M(C, b)$ ab einem Zeitpunkt zu dem die Variablen g, x und t die Werte g', x' und t' angenommen haben. $\beta(g', x', t')$ kann mehrfach in $\beta_M(C, b)$ vorkommen. $\beta(g', x', t')$ ist dennoch eindeutig bestimmt, da alle Vorkommen identisch sind. Sei au-

ßerdem:

$$A(g', x', t') =_{def} \begin{cases} 1 & \text{falls } x' \in I(g') \text{ und } t' = 1 \\ 1 & \text{falls } x' \notin I(g') \text{ und } t' = 0 \\ 0 & \textit{sonst} \end{cases}$$

Wir zeigen induktiv über die Struktur von C, dass $A(g', x', t')$ genau dann 1 ist, wenn $\beta(g', x', t')$ einen akzeptierenden Teilbaum hat.

IA: Sei g' ein Eingabegatter. Dann wird der Algorithmus Zeile 38 erreichen und akzeptiert genau unter den Bedingungen unter denen auch $A(g', x', t') = 1$ gilt.

IV: Sei g' ein Gatter aus C mit Vorgängern, sodass die Aussage für alle $x \in \mathbb{N}$ und für alle $t \in \{0, 1\}$ und $p(g')$ bzw. $p_1(g')$ und $p_2(g')$ gilt.

IS: Sei $x' \in \mathbb{N}$ und $t \in \{0, 1\}$. O.B.d.A. sei $x' \leq m = 2^{|C|} + 1$. Sonst wird in Zeile 4 x der Wert m zugewiesen. Nach Folgerung 3.2 gilt:

$$x' \in I(g') \Leftrightarrow m \in I(g')$$

Falls also $x' > m$ gilt, ist $A(g', x', t') = A(g', m, t')$ nach Folgerung 3.2. Der Algorithmus macht in dieser Zeile keine Verzweigungen. Daher ist $\beta(g', m, t')$ ein Teilbaum von $\beta(g', x', t')$, der genau dann einen akzeptierenden Teilbaum hat, wenn $\beta(g', x', t')$ einen hat.

Fall 1: Es gilt $\alpha(g') = \bar{}$. Der Algorithmus setzt in Zeile 8 ohne Verzweigungen $(g, x, t) = (p(g'), x', 1 - t')$. Daher hat $\beta(g', x', t')$ genau dann einen akzeptierenden Teilbaum, wenn $\beta(p(g'), x', 1 - t')$ einen akzeptierenden Teilbaum hat. Es gilt:

$$\begin{aligned} A(g', x', t') &= \begin{cases} 1 & \text{falls } x' \in I(g') \text{ und } t' = 1 \\ 1 & \text{falls } x' \notin I(g') \text{ und } t' = 0 \\ 0 & \textit{sonst} \end{cases} \\ &= \begin{cases} 1 & \text{falls } x' \notin I(p(g')) \text{ und } 1 - t' = 0 \\ 1 & \text{falls } x' \in I(p(g')) \text{ und } 1 - t' = 1 \\ 0 & \textit{sonst} \end{cases} \\ &= A(p(g'), x', 1 - t') \end{aligned}$$

Nach IV hat $\beta(p(g'), x', t')$ genau dann einen akzeptierenden Teilbaum, wenn $A(p(g'), x', t') = 1$ gilt. Daher hat $\beta(g', x', t')$ genau dann einen akzeptierenden Teilbaum, wenn $A(g', x', t') = 1$ gilt.

Fall 2: Es gilt $\alpha(g') == \cup$ und $t' = 0$. Der Algorithmus erreicht Zeile 10 und ist in einem universellen Zustand. (g, x, t) wird parallel von (g', x', t') auf $(p_1(g'), x', t')$

und $(p_2(g'), x', t')$ gesetzt. $\beta(g', x', t')$ hat also genau dann einen akzeptierenden Teilbaum, wenn $\beta(p_1(g'), x', t')$ und $\beta(p_2(g'), x', t')$ einen akzeptierenden Teilbaum haben. Es gilt:

$$\begin{aligned} A(g', x', t') &= \begin{cases} 1 & \text{falls } x' \notin I(g') \\ 0 & \mathit{sonst} \end{cases} \\ &= \begin{cases} 1 & \text{falls } x' \notin I(p_1(g')) \text{ und } x' \notin I(p_2(g')) \\ 0 & \mathit{sonst} \end{cases} \\ &= \begin{cases} 1 & \text{falls } A(p_1(g'), x', t') = 1 \text{ und } A(p_2(g')', x', t') = 1 \\ 0 & \mathit{sonst} \end{cases} \end{aligned}$$

Damit folgt nach IV, dass $\beta(g', x', t')$ genau dann einen akzeptierenden Teilbaum hat, wenn $A(g', x', t') = 1$ gilt.

Fall 3: Es gilt $\alpha(g') == \cup$ und $t' = 1$. Der Algorithmus erreicht Zeile 12 und ist in einem existenziellen Zustand. (g, x, t) wird parallel von (g', x', t') auf $(p_1(g'), x', t')$ und $(p_2(g'), x', t')$ gesetzt. $\beta(g', x', t')$ hat genau dann einen akzeptierenden Teilbaum, wenn $\beta(p_1(g'), x', t')$ oder $\beta(p_2(g'), x', t')$ einen akzeptierenden Teilbaum haben. Es gilt:

$$\begin{aligned} A(g', x', t') &= \begin{cases} 1 & \text{falls } x' \in I(g') \\ 0 & \mathit{sonst} \end{cases} \\ &= \begin{cases} 1 & \text{falls } x' \in I(p_1(g')) \text{ oder } x' \in I(p_2(g')) \\ 0 & \mathit{sonst} \end{cases} \\ &= \begin{cases} 1 & \text{falls } A(p_1(g'), x', t') = 1 \text{ oder } A(p_2(g')', x', t') = 1 \\ 0 & \mathit{sonst} \end{cases} \end{aligned}$$

Damit folgt nach IV, dass $\beta(g', x', t')$ genau dann einen akzeptierenden Teilbaum hat, wenn $A(g', x', t') = 1$ gilt.

Fall 4: Es gilt $\alpha(g') == +$ und $t' = 1$. Der Algorithmus erreicht Zeile 15. $\beta(g', x', t')$ hat für jedes $a \in [0, x'] \cap \mathbb{N}$ einen Teilbaum $\beta(p_1(g'), x' - a, t')$ und einen Teilbaum $\beta(p_2(g'), a, t')$, welche durch eine Kette von existenziellen Zuständen und einen universellen Zustand erreicht werden. $\beta(g', x', t')$ hat also genau dann einen akzeptierenden Teilbaum, wenn es ein $a \in [0, x'] \cap \mathbb{N}$ gibt, sodass $\beta(p_1(g'), x' - a, t')$ und $\beta(p_2(g'), a, t')$ einen akzeptierenden Teilbaum haben.

$$A(g', x', t') = \begin{cases} 1 & \text{falls } x' \in I(g') \\ 0 & \textit{sonst} \end{cases}$$
$$= \begin{cases} 1 & \text{falls } \exists a \in \mathbb{N} : x' - a \in I(p_1(g')) \text{ und } a \in I(p_2(g')) \\ 0 & \textit{sonst} \end{cases}$$
$$= \begin{cases} 1 & \text{falls } \exists a \in [0, x'] \cap \mathbb{N} : x' - a \in I(p_1(g')) \text{ und } a \in I(p_2(g')) \\ 0 & \textit{sonst} \end{cases}$$
$$= \begin{cases} 1 & \text{falls } \exists a \in [0, x'] \cap \mathbb{N} : \\ & A(p_1(g'), x' - a, t') = 1 \text{ und } A(p_2(g'), a, t') = 1 \\ 0 & \textit{sonst} \end{cases}$$

Nach IV hat $\beta(g', x', t')$ daher genau dann einen akzeptierenden Teilbaum hat, wenn $A(g', x', t') = 1$ gilt.

Fall 5: Es gilt $\alpha(g') == +$ und $t' = 0$. Die Aussage lässt sich analog zu Fall 4 zeigen. Dabei sind universelle und existenzielle Zustände, der Quantor und „und“ und „oder“ jeweils vertauscht.

Fall 6: Es gilt $\alpha(g') == /$ und $t' = 1$. Der Algorithmus erreicht Zeile 26. $\beta(g', x', t')$ hat für jedes $a \in [1, m] \cap \mathbb{N}$ einen Teilbaum $\beta(p_1(g'), a \cdot x, t')$ und einen Teilbaum $\beta(p_2(g'), a, t')$, welche durch eine Kette von existenziellen Zuständen und einen universellen Zustand erreicht werden. $\beta(g', x', t')$ hat also genau dann einen akzeptierenden Teilbaum, wenn es ein $a \in [1, m] \cap \mathbb{N}$ gibt, sodass $\beta(p_1(g'), a \cdot x', t')$ und $\beta(p_2(g'), a, t')$ einen akzeptierenden Teilbaum haben.

$$A(g', x', t') = \begin{cases} 1 & \text{falls } x' \in I(g') \\ 0 & \textit{sonst} \end{cases}$$
$$= \begin{cases} 1 & \text{falls } \exists a \in \mathbb{N}^+ : a \cdot x' \in I(p_1(g')) \text{ und } a \in I(p_2(g')) \\ 0 & \textit{sonst} \end{cases}$$

Nach Folgerung 3.2 gilt für $a > m$:

$$a \cdot x' \in I(p_1(g')) \Leftrightarrow m \cdot x' \in I(p_1(g'))$$

und

$$a \in I(p_2(g')) \Leftrightarrow m \in I(p_2(g'))$$

Daher ist:

$$A(g', x', t') = \begin{cases} 1 & \text{falls } \exists a \in [1, m] \cap \mathbb{N} : a \cdot x' \in I(p_1(g')) \text{ und } a \in I(p_2(g')) \\ 0 & \textit{sonst} \end{cases}$$
$$= \begin{cases} 1 & \text{falls } \exists a \in [1, m] \cap \mathbb{N} : \\ & A(g', a \cdot x, t') = 1 \text{ und } A(g', a, t') = 1 \\ 0 & \textit{sonst} \end{cases}$$

Nach IV hat $\beta(g', x', t')$ daher genau dann einen akzeptierenden Teilbaum hat, wenn $A(g', x', t') = 1$ gilt.

Fall 7: Es gilt $\alpha(g') == /$ und $t' = 0$. Die Aussage lässt sich analog zu Fall 6 zeigen.

Das heißt, der Berechnungsbaum $\beta(g_C, b, 1)$ hat genau dann einen akzeptierenden Teilbaum, wenn $A(g_C, b, 1) = 1$. Das ist genau dann der Fall, wenn $b \in I(g_C) = I(C)$. Damit haben wir bewiesen, dass der Algorithmus korrekt ist.

Es verbleibt zu zeigen, dass der alternierende Algorithmus nur polynomielle Zeit benötigt. Die Variable g nimmt auf jedem Pfad von der Wurzel von $\beta_M(C, b)$ zu einem Blatt den Wert jedes Gatters aus C maximal einmal an, weil C azyklisch ist. In jedem Schleifendurchlauf ändert sich g. Daher liegt die Anzahl der Schleifendurchläufe in $O(|C|)$. Die Laufzeit eines Schleifendurchlaufs liegt in $O(|C|^2)$. Daher liegt die Laufzeit von Algorithmus 1 in $O(n^3)$, wobei n die Größe der Eingabe (C, b) ist. Nach Satz 2.4 ist $\text{ATIME}(n^3) \subseteq \text{DSPACE}(n^3)$. Die Gültigkeit der Eingabe kann in deterministischem logarithmischem Raum überprüft werden.

$$\Rightarrow MC(\cup, ^-, +, /) \in \text{DSPACE}(n^3) \subseteq \text{PSPACE}$$

□

Korollar 3.4
$MC(\cup, \cap, ^-, +, /)$ *und* $MC(\cup, \cap, +, /)$ *sind* $\leq_m^{\log}$*-vollständig für* PSPACE.

Beweis. McKenzie und Wagner erklären im Beweis von Lemma 5.1 aus [MW07] wie das Erfüllbarkeitsproblem für quantifizierte boolesche Formeln (QBF) in logarithmischem Raum auf $MC(\cup, \cap, +)$ many-one-reduziert werden kann. QBF ist bekanntlich $\leq_m^{\log}$-hart für PSPACE. Mit Lemma 2.3 erhalten wir:

$$\text{QBF} \leq_m^{\log} MC(\cup, \cap, +) \leq_m^{\log} MC(\cup, \cap, +, /) \leq_m^{\log} MC(\cup, \cap, ^-, +, /)$$

Damit sind $MC(\cup, \cap, ^-, +, /)$ und $MC(\cup, \cap, +, /)$ $\leq_m^{\log}$-hart für PSPACE. Mit Satz 3.3 erhalten wir die PSPACE-Vollständigkeit. □

Damit reihen sich $MC(\cup, \cap, ^-, +, /)$ und $MC(\cup, \cap, +, /)$ in die Liste der PSPACE-vollständigen Membership-Probleme ein. McKenzie und Wagner haben in [MW07] die PSPACE-Vollständigkeit von $MC(\cup, +, \times)$, $MC(\cup, \cap, ^-, +)$, $MC(\cup, \cap, +)$, $MC(\cup, \cap, ^-, \times)$ und $MC(\cup, \cap, \times)$ bewiesen. Wir werden im Laufe dieser Arbeit (siehe Abschnitt 3.7 und 4.1) noch weitere Membership-Probleme mit Division zu dieser Liste hinzufügen.

3.3 Keine Primzahlen ohne Multiplikation

Aus Folgerung 3.2 erhalten wir noch ein weiteres interessantes Resultat: Wie wir bereits wissen haben McKenzie und Wagner in [MW07] beschreiben, wie sich die Menge der Primzahlen $\mathbb{P}$ durch einen $\{\cup, \cap, ^-, \times\}$-Schaltkreis beschreiben lässt und konnten damit auch die Goldbachsche Vermutung als Membership-Problem ausdrücken. Wir können nun beweisen, dass $\{\cup, \cap, ^-, +, /\}$-Schaltkreise nicht geeignet sind, um die Primzahlen auszudrücken. Dafür treffen wir zunächst folgende Feststellung über die Struktur der von $\{\cup, \cap, ^-, +, /\}$-Schaltkreisen beschriebenen Mengen:

Korollar 3.5
Sei $A \subseteq \mathbb{N}$ mit A und $\overline{A}$ unendlich. Dann gibt es keinen $\{\cup, \cap, ^-, +, /\}$-Schaltkreis C mit $I(C) = A$.

Beweis. Angenommen es gibt einen solchen $\{\cup, \cap, ^-, +, /\}$-Schaltkreis C. A und $\overline{A}$ sind unendlich. Also gibt es $a \in A$ mit $a \geq 2^{|C|} + 1$ und $b \in \overline{A}$ mit $b \geq 2^{|C|} + 1$. Durch doppeltes Anwenden von Folgerung 3.2 erhalten wir:

$$a \in A = I(C) \Rightarrow 2^{|C|} + 1 \in I(C) \Rightarrow b \in I(C) = A$$

b ist aber in $\overline{A}$. Dies ist ein Widerspruch. Also kann ein solcher Schaltkreis C nicht existieren. □

Da sowohl $\mathbb{P}$ als auch $\mathbb{P} \setminus \mathbb{N}$ unendlich sind, folgt:

Folgerung 3.6 *Es gibt keinen $\{\cup, \cap, ^-, +, /\}$-Schaltkreis C mit $I(C) = \mathbb{P}$.*

3.4 Schaltkreise ohne Komplement

Zum Lösen des Membership-Problems für Schaltkreise ohne Komplement, machen wir uns zu nutze, dass die vom Schaltkreis berechneten Mengen endlich und nach oben beschränkt sind. Das folgende Lemma beschreibt die oberen Schranken:

Lemma 3.7
Sei C ein $\mathcal{O}$-Schaltkreis mit $\mathcal{O} \neq \emptyset$.

1. $\mathcal{O} \subseteq \{\cup, \cap, +, /\} \Rightarrow I(C) \subseteq \left[0, 2^{|C|}\right]$
2. $\mathcal{O} \subseteq \{\cup, \cap, +, \times, /\} \Rightarrow I(C) \subseteq \left[0, 2^{2^{|C|}}\right]$

Beweis. 1. $I(C)$ ist endlich. Damit folgt die Behauptung direkt aus Folgerung 3.2.

2. Sei C ein $\{\cup, \cap, +, \times, /\}$-Schaltkreis. Wir zeigen $I(C) \subseteq \left[0, 2^{2^{|C|}}\right]$ per Induktion über den Aufbau von C. Sei wieder C_g der Schaltkreis, der aus C entsteht, wenn man g als Ausgabegatter verwendet und alle Gatter, von denen aus g nicht erreichbar ist, entfernt.

   IA: Sei g ein Eingabegatter. Dann gilt $I(g) = I(\alpha(g))$ und $\log(\alpha(g)) \leq |C_g| \leq 2^{2^{|C_g|}}$.

   IV: Sei g ein Gatter mit Vorgängern, sodass $I(p_i(g)) \subseteq \left[0, 2^{2^{|C_{p_i(g)}|}}\right]$ für alle $i \in \{1, 2\}$ gilt.

IS: Ohne Beschränkung der Allgemeinheit sei $max(p_1(g)) \geq max(p_2(g)) \geq 2$. Dann gilt

$$\begin{aligned} max(I(g)) &\leq (max(p_1(g)))^2 \\ &\leq \left(2^{2^{|C_{p_1(g)}|}}\right)^2 \text{ (nach IV)} \\ &= 2^{\left(2^{|C_{p_1(g)}|}+2^{|C_{p_1(g)}|}\right)} \\ &= 2^{2^{\left(|C_{p_1(g)}|+1\right)}} \\ &\leq 2^{2^{|C_{p(g)}|}} \end{aligned}$$

und damit $I(g) \subseteq \left[0, 2^{2^{|C_g|}}\right]$.

□

3.5 Schaltkreise ohne Komplement und Vereinigung

In Schaltkreisen ohne Komplement und Vereinigung sind die zu den Gattern berechneten Mengen immer höchstens einelementig. Wir können die Mengen zu den Gattern daher einfach ausrechnen.

Satz 3.8 *1. Für alle $\mathcal{O} \subseteq \{\cap, +, /\}$ gilt $MC(\mathcal{O}) \in$ P.*

2. Für alle $\mathcal{O} \subseteq \{\cap, +, \times, /\}$ gilt $MC(\mathcal{O}) \in$ E.

Beweis. 1. Sei C ein $\mathcal{O}$-Schaltkreis und $b \in \mathbb{N}$. Wir wollen $(C, b) \in MC(\mathcal{O})$ prüfen. Für jedes Gatter g aus C gilt $\#I(g) \leq 1$ und für alle $n \in I(g)$ gilt $n \leq 2^{|C|}$ gemäß Lemma 3.7 (1). Also kann $I(g)$ in $|C|$ Bits gespeichert werden. Berechne $I(C)$ iterativ durch Berechnung von $I(g)$ für jedes Gatter g. Suche in jeder Iteration ein Gatter g mit Vorgängern g_0 und g_1, sodass $I(g_0)$ und $I(g_1)$ bereits berechnet sind und berechne daraus $I(g)$. $I(g)$ lässt sich in Polynomialzeit berechnen. Wiederhole dies so lange, bis $I(g_C)$ für das Ausgabegatter g_C berechnet wurde. Da ist Anzahl der Gatter durch $|C|$ nach oben beschränkt ist, wird insgesamt nur polynomielle Zeit zur Berechnung der Menge $I(g_C)$ benötigt. Prüfe nun $b \in I(g_C) = I(C)$.

2. Analog zu (1) mit Lemma 3.7 (2).

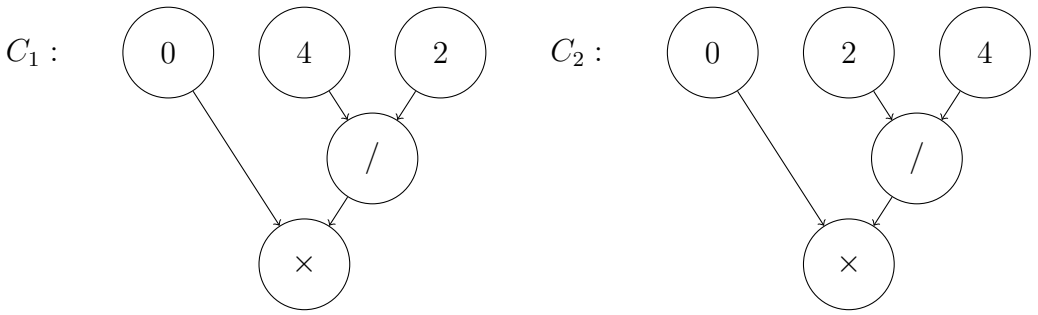
□

Überlegungen zu $MC(\cap, +, \times, /)$ mittels alternierender Maschinen Bei der Erstellung dieser Arbeit gab es den Ansatz $MC(\cap, +, \times, /)$ in alternierender polynomieller Zeit zu lösen, der nicht zum Erfolg geführt hat: Mit einer alternierenden Maschine ist es möglich für einen $\{\cap, +, \times, /\}$-Schaltkreis C Primzahlen p bis zu einer

Größe von $O(2^{2|C|})$ zu raten. Die Ergebnismengen der Gatter werden jeweils modulo p gespeichert. Dafür sind $O(2|C|)$ Bits pro Gatter ausreichend. Die Maschine soll $(f \bmod p) \in \{e \bmod p \mid e \in I(g)\}$ statt $f \in I(g)$ für alle solchen Primzahlen parallel prüfen und nur akzeptieren, wenn die Tests für alle solchen Primzahlen erfolgreich sind. Falls das möglich ist, kann mit dem Chinesischen Restsatz bewiesen werden, dass es nur eine Zahl $b \leq 2^{2^{|C|}}$ gibt, für die der Algorithmus (C, b) akzeptiert. Dieser Ansatz hat für /-Gatter zwei Probleme:

1. Die alternierende Maschine kann modulo p nicht überprüfen, ob für zwei Zahlen x_1 und x_2, die beide durch p teilbar sind, x_1 durch x_2 teilbar ist. Das Ergebnis von x_1/x_2 modulo p ist in diesem Fall nicht definiert, denn $x_1 = x_2 \times a$ gilt modulo p für alle natürlichen Zahlen a. Dieses Problem lässt sich lösen, indem für alle Primzahlen, die im Schaltkreis als Primfaktoren vorkommen, akzeptiert wird. Mit dem Chinesischen Restsatz lässt sich zeigen, dass die Eindeutigkeit von b mit $I(C) = \{b\}$ garantiert ist, falls b modulo p für $2^{|C|}$ viele verschiedene Primzahlen gegeben ist. Dies ist garantiert der Fall, weil wir Primzahlen bis zur Größe $O(2^{2|C|})$ raten.

2. Die ganzen Zahlen modulo p entsprechend dem endlichen Körper $\mathbb{F}_p$. In diesem Körper hat jede Division, bei der nicht durch die Null des Körpers geteilt wird, ein Ergebnis. Wir betrachten folgendes Beispiel:



   Es gilt $I(C_1) = \{0\}$ und $I(C_2) = \emptyset$. Die alternierende Maschine kann für jede Primzahl $p > 2$ eine Zahl x_p finden, sodass im Körper $\mathbb{F}_p$ die Gleichung $2/4 = x_p$ gilt. Für alle solchen p ist $0 \times x_p = 0$. Daher würde die alternierende Maschine $(C_2, 0)$ akzeptieren, obwohl $I(C_2) = \emptyset$ ist.

3.6 Schaltkreise ohne Komplement (mit Vereinigung)

Nehmen wir die Vereinigung wieder hinzu, so können die Mengen zu den Gattern jeweils mehr als nur ein Element beinhalten. Wir können die Repräsentanten, die $(C, b) \in MC(\cup, \cap, +, \times, /)$ beweisen, durch Nichtdeterminismus raten. Dabei ist allerdings zu beachten, dass zur Berechnung von b im Ausgabegatter aus den Vorgängergattern jeweils möglicherweise mehrere verschiedene Werte benötigt werden.

Satz 3.9
$MC(\cup, \cap, +, \times, /) \in$ NEXP

Beweis. Sei für diesen Beweis $\mathcal{O} =_{def} \{\cup, \cap, +, \times, /\}$. Eine $\mathcal{O}$-Formel ist ein $\mathcal{O}$-Schaltkreis, in dem die Gatter maximal den Ausgangsgrad 1 haben. Wir betrachten zunächst

$$MF(\mathcal{O}) =_{def} \{(F, b) \mid F \text{ ist eine } \mathcal{O}\text{-Formel und } b \in I(F)\}$$

Wir wollen zunächst zeigen, dass $MF(\mathcal{O}) \in$ NP. Sei $F = (G, E, g_F, \alpha)$ eine $\mathcal{O}$-Formel und $b \in \mathbb{N}$. Sei

$$I = \{\alpha(v) \mid v \text{ ist ein Eingabegatter von } F\}$$

Dann ist $m =_{def} \Pi_{a \in I}(a + 1)$ eine obere Schranke für alle $I(g)$ für Gatter g aus F. Rate für jedes Gatter g aus F eine Zahl $b_g \leq m$ und prüfe, ob die geratenen Zahlen $b \in I(g_F)$ beweisen. Es gilt:

$$\log(b_g) \leq \log(m) = \log(\Pi_{a \in I}(a + 1)) = \Sigma_{a \in I}(\log(a + 1)) \leq |F|$$

Daher kann dies in nichtdeterministischer Polynomialzeit bewerkstelligt werden.

Sei nun C ein $\mathcal{O}$-Schaltkreis und $b \in \mathbb{N}$. Um $(C, b) \in MC(\mathcal{O})$ zu entscheiden, entfalten wir C zu einer äquivalenten $\mathcal{O}$-Formel F_C. (Diese Idee stammt aus [MW07]. McKenzie und Wagner verwenden sie z.B. im Beweis von Theorem 6.2.) Dann testen wir mit dem obigen NP-Algorithmus, ob $(F, b) \in MF(\mathcal{O})$. Da F maximal exponentiell größer ist als C, liefert dies einen NEXP-Algorithmus für $MC(\mathcal{O})$. □

Korollar 3.10 *1.* $MC(\cup, \cap, +, \times, /)$ *ist* $\leq_m^{\log}$*-vollständig für* NEXP*.*

2. $MC(\cup, \cap, +, \times, /) \equiv_m^{\log} MC(\cup, \cap, +, \times)$

Beweis. McKenzie und Wagner beweisen in [MW07] (Lemma 6.1 und Theorem 6.2), dass $MC(\cup, \cap, +, \times)$ $\leq_m^{\log}$-vollständig für NEXP ist. NEXP ist bekanntlich abgeschlossen unter $\leq_m^{\log}$. Damit ist gemäß Lemma 2.3 auch $MC(\cup, \cap, +, \times, /)$ $\leq_m^{\log}$-hart für NEXP. Zusammen mit Satz 3.9 ergibt sich die NEXP-Vollständigkeit. Da $MC(\cup, \cap, +, \times, /)$ und $MC(\cup, \cap, +, \times)$ beide $\leq_m^{\log}$-vollständig für NEXP sind, sind sie insbesondere auch $\equiv_m^{\log}$-äquivalent. □

3.7 Schaltkreise ohne Addition

Der folgende Abschnitt basiert auf Abschnitt 3 aus [MW07]. Wir werden $MC(\cap, \times, /) \in$ P und $MC(\cup, \cap, ^-, \times, /) \in$ PSPACE beweisen, indem wir die dort geführten Beweise für $MC(\cap, \times) \in$ P und $MC(\cup, \cap, ^-, \times) \in$ PSPACE um Division bzw. Subtraktion erweitern. Dafür benötigen wir zunächst zwei Definitionen:

Definition
Seien $a_1, \ldots, a_n$ *positive natürliche Zahlen. Dann gibt es* $k \geq 1$*,* $q_1, \ldots q_k \geq 2$ *und* $e_{1,1}, \ldots e_{n,k} \geq 0$*, sodass* $ggT(q_i, q_j) = 1$ *für alle* $i, j \in \{1, \ldots, k\}$ *mit* $i \neq j$ *und* $a_i = \prod_{j=1}^{k} q_j^{e_{i,j}}$ *für alle* $i \in \{1, \ldots, n\}$*.* $\{q_1, \ldots q_k\}$ *heißt* ggT-freie Basis *von* $\{a_1, \ldots, a_n\}$*.*

Die folgende Proposition übernehmen wir aus Abschnitt 3 aus [MW07]:

Proposition 3.11
ggT-freie Basen können in Polynomialzeit berechnet werden.

Definition
Sei $\emptyset \neq \mathcal{O} \subseteq \{\cup, \cap, ^-, +, -, \}$. $MC^*(\mathcal{O})$ *bezeichnet das Membership-Problem für* $\mathcal{O}$*-Schaltkreise über* $\mathbb{N}^k \cup \{\infty\}$ *für* $k \geq 1$. *Dabei seien die arithmetischen Operation* $\circ \in \{+, -\}$ *auf* $\mathbb{N}^k$ *komponentenweise definiert. Dabei ist* $(a_1, \ldots, a_k) - (b_1, \ldots, b_k)$ *definiert, falls* $a_i - b_i$ *für alle* $i \in \{1, \ldots, k\}$ *eine natürliche Zahl ist. Außerdem sei für alle* $m \in \mathbb{N}^k$*:*

1. $\infty + m = \infty = m + \infty$ *und* $\infty + \infty = \infty$
2. $\infty - m = \infty$ *und* $m - \infty$ *sowie* $\infty - \infty$ *sind nicht definiert*

Bemerkung
Für alle $k \geq 1$ *ist* $(\mathbb{N}^k \cup \{\infty\}, +, (0, \ldots, 0))$ *ein Monoid.*

Satz 3.12 *(i) Für alle* $\mathcal{O} \subseteq \{\cup, \cap\}$ *gilt:* $MC(\mathcal{O} \cup \{\times, /\}) \leq_m^{\mathrm{P}} MC^*(\mathcal{O} \cup \{+, -\})$

(ii) Für alle $\mathcal{O} \subseteq \{\cup, \cap, ^-\}$ *gilt:* $MC(\mathcal{O} \cup \{\times, /\}) \leq_m^{\mathrm{PSPACE}} MC^*(\mathcal{O} \cup \{+, -\})$

Beweis. (i) Sei $\mathcal{O} \subseteq \{\cup, \cap\}$ und C ein $\mathcal{O} \cup \{\times, /\}$-Schaltkreis mit Eingabegatter $g_1, ..., g_n$ mit Labeln $a_1, \ldots, a_n \in \mathbb{N}$ und sei $b \in \mathbb{N}$. Berechne in Polynomialzeit eine ggT-freie Basis $\{q_1, \ldots, q_k\}$ von $\{a_1, \ldots, a_n, b\}$. (Siehe Proposition 3.11.)

Sei $M =_{def} \left\{ \prod_{j=1}^k q_j^{d_j} \;\middle|\; d_1, \ldots, d_k \in \mathbb{N} \right\}$. Dann ist $M \subseteq \mathbb{N}^+$ und $M \cup \{0\}$ ist abgeschlossen unter Multiplikation. Es ist außerdem $1 = \prod_{j=1}^k q_j^0 \in M$. Also ist $(M \cup \{0\}, \times, 1)$ ein Monoid, wobei $\times$ die Multiplikation der natürlichen Zahlen eingeschränkt auf $M \cup \{0\}$ ist.

Definiere

$$\sigma : \begin{array}{ll} M \cup \{0\} & \to \mathbb{N}^k \cup \{\infty\} \\ \prod_{j=1}^k q_j^{d_j} & \mapsto (d_1, \ldots, d_k) \\ 0 & \mapsto \infty \end{array}$$

Die Primfaktorzerlegung von q_j ist für alle $j \in \{1, \ldots, k\}$ eindeutig. Da die q_j paarweise teilerfremd sind, kommt keine Primzahl in der Primfaktorzerlegung von q_j und q_i für $i \neq j$ vor. Daher ist die Darstellung einer Zahl $x \in M$ als Produkt der Form $x = \prod_{j=1}^k q_j^{d_j}$ ebenfalls eindeutig. Also ist σ wohldefiniert.

Wir wollen nun zeigen, dass σ ein Monoid-Homomorphismus ist. Es ist $\sigma(1) = \sigma(\prod_{j=1}^{k} q_j^0) = (0, \ldots, 0)$. Seien $\prod_{j=1}^{k} q_j^{d_j}, \prod_{j=1}^{k} q_j^{e_j} \in M$. Dann gilt:

$$\begin{aligned}\sigma\left(\prod_{j=1}^{k} q_j^{d_j} \times \prod_{j=1}^{k} q_j^{e_j}\right) &= \sigma\left(\prod_{j=1}^{k} q_j^{d_j+e_j}\right)\\ &= (d_1 + e_1, \ldots, d_k + e_k)\\ &= (d_1, \ldots, d_k) + (e_1, \ldots, e_k)\\ &= \sigma\left(\prod_{j=1}^{k} q_j^{d_j}\right) + \sigma\left(\prod_{j=1}^{k} q_j^{e_j}\right)\end{aligned}$$

Für alle $m \in M \cup \{0\}$ gilt

$$\sigma(0 \times m) = \sigma(0) = \infty = \infty + \sigma(m)$$

und analog:

$$\sigma(m \times 0) = \sigma(m) + \infty$$

Also ist σ ein Homomorphismus. Seien $x_1, x_2 \in M$ mit $x_1 \neq x_2$. Dann gibt es $d_1, \ldots d_k \in \mathbb{N}$ mit $x_1 = \prod_{j=1}^{k} q_j^{d_j}$ und $e_1, \ldots e_k \in \mathbb{N}$ mit $x_2 = \prod_{j=1}^{k} q_j^{e_j}$. Wegen $x_1 \neq x_2$ gibt es ein $j \in \{1, \ldots, k\}$ mit $d_j \neq e_j$. Also gilt

$$\sigma\left(\prod_{j=1}^{k} q_j^{d_j}\right) = (d_1, \ldots, d_k) \neq (e_1, \ldots, e_k) = \sigma\left(\prod_{j=1}^{k} q_j^{e_j}\right)$$

und σ ist injektiv. Da σ offensichtlich auch surjektiv ist, ist σ eine Monoid-Isomorphismus. Für alle $S \subseteq M \cup \{0\}$ sei $\sigma(S)$ das Bild von S unter σ, das heißt:

$$\sigma(S) =_{def} \{\sigma(s) \mid s \in S\}$$

Wir zeigen zunächst, dass für alle $S_1, S_2 \subseteq M \cup \{0\}$ gilt:

$$\sigma(S_1 \times S_2) = \sigma(S_1) + \sigma(S_2) \tag{1}$$
$$\sigma(S_1/S_2) = \sigma(S_1) - \sigma(S_2) \tag{2}$$
$$\sigma(S_1 \cup S_2) = \sigma(S_1) \cup \sigma(S_2) \tag{3}$$
$$\sigma(S_1 \cap S_2) = \sigma(S_1) \cap \sigma(S_2) \tag{4}$$

(1) Für alle $y \in \mathbb{N}^k \cup \{\infty\}$ gilt:

$$\begin{aligned}y \in \sigma(S_1 \times S_2) &\Leftrightarrow \exists x \in (S_1 \times S_2) : y = \sigma(x)\\ &\Leftrightarrow \exists x_1 \in S_1\ \exists x_2 \in S_2 : y = \sigma(x_1 \times x_2)\\ &\Leftrightarrow \exists x_1 \in S_1\ \exists x_2 \in S_2 : y = \sigma(x_1) + \sigma(x_2)\ (\sigma \text{ ist Homo.})\\ &\Leftrightarrow \exists y_1 \in \sigma(S_1)\ \exists y_2 \in \sigma(S_2) : y = y_1 + y_2\\ &\Leftrightarrow y \in \sigma(S_1) + \sigma(S_2)\end{aligned}$$

Damit ist (1) bewiesen.

(2) Sei $y \in \sigma(S_1/S_2)$. Dann gibt es $x \in (S_1/S_2)$ mit $y = \sigma(x)$ und damit $x_1 \in S_1$ und $x_2 \in S_2$, sodass $x = x_1/x_2$. Insbesondere ist x_1 teilbar durch x_2 und $x_2 \neq 0$. Seien $e_1, \ldots, e_k \in \mathbb{N}$ mit $x_2 = \prod_{j=1}^{k} q_j^{e_j}$. Also ist $\sigma(x_2) = (e_1, \ldots e_k) \in \mathbb{N}^k$. Falls $x_1 = 0$ ist, so gilt:

$$\begin{aligned} y = \sigma(x) &= \sigma(x_1/x_2) = \sigma(0/x_2) = \sigma(0) \\ &= \infty = \infty - \sigma(x_2) = \sigma(0) - \sigma(x_2) \\ &= \sigma(x_1) - \sigma(x_2) \in \sigma(S_1) - \sigma(S_2) \end{aligned}$$

Falls $x_1 \neq 0$ ist, so gibt es $d_1, \ldots, d_k \in \mathbb{N}$ mit $x_1 = \prod_{j=1}^{k} q_j^{d_j}$. Sei $I = \{j \in \{1, \ldots, k\} \mid d_j \geq e_j\}$ und $I' = \{1, \ldots, k\} \setminus I$. Dann gilt:

$$\begin{aligned} x_1/x_2 &= \prod_{j=1}^{k} q_j^{d_j - e_j} \\ &= \prod_{j \in I} q_j^{d_j - e_j} \cdot \prod_{i \in I'} q_i^{d_i - e_i} \\ &= \frac{\prod_{j \in I} q_j^{d_j - e_j}}{\prod_{i \in I'} q_i^{e_i - d_i}} \in \mathbb{N} \end{aligned}$$

Angenommen es gibt ein $l \in I'$. Sei p ein Primfaktor von q_l. Dann ist p ein Teiler von $\prod_{j \in I} q_j^{d_j - e_j}$. Also gibt es ein $j \in I$ mit $p | q_j$ und daher ist $ggT(q_l, q_j) \geq p > 1$. Dies ist ein Widerspruch. Also ist $I' = \emptyset$, $x_1/x_2 = \prod_{j \in I} q_j^{d_j - e_j} \in M$ und es gilt:

$$\begin{aligned} y &= \sigma(x_1/x_2) \\ &= (d_1 - e_1, \ldots, d_k - e_k) \\ &= (d_1, \ldots, d_k) - (e_1, \ldots, e_k) \\ &= \sigma(x_1) - \sigma(x_2) \in \sigma(S_1) - \sigma(S_2) \end{aligned}$$

Sei nun $y \in \sigma(S_1) - \sigma(S_2)$. Dann gibt es $y_1 \in \sigma(S_1)$ und $y_2 \in \sigma(S_2)$ mit $y = y_1 - y_2$. Insbesondere ist $y_1 - y_2$ definiert. Also ist $y_2 \in \mathbb{N}^k$ und damit gibt es $e_1, \ldots, e_k \in \mathbb{N}$ mit $y_2 = (e_1, \ldots, e_k)$. Also ist $x_2 =_{def} \prod_{j=1}^{k} q_j^{e_j} \in S_2$. Falls $y_1 = \infty$, so ist $0 \in S_1$ und es so gilt:

$$\begin{aligned} y &= \infty - (e_1, \ldots e_k) \\ &= \infty \\ &= \sigma(0) \\ &= \sigma(0/x_2) \in \sigma(S_1/S_2) \end{aligned}$$

Andernfalls ist $y_1 \in \mathbb{N}^k$ und es gibt $d_1, \ldots, d_k \in \mathbb{N}$ mit $y_1 = (d_1, \ldots, d_k)$. Da $y = y_1 - y_2 = (d_1, \ldots d_k) - (e_1, \ldots e_k)$ definiert ist, ist $d_j \geq e_j$ für alle $j \in \{1, \ldots, k\}$. Also ist $x_1 =_{def} \prod_{j=1}^{k} q_j^{d_j} \in S_1$ und x_2 teilt x_1. Das heißt:

$$\begin{aligned} y &= (d_1, \ldots d_k) - (e_1, \ldots e_k) \\ &= (d_1 - e_1, \ldots, d_k - e_k) \\ &= \sigma\left(\prod_{j=1}^{k} q_j^{d_j - e_j}\right) \\ &= \sigma\left(\frac{\prod_{j=1}^{k} q_j^{d_j}}{\prod_{j=1}^{k} q_j^{e_j}}\right) \\ &= \sigma(x_1/x_2) \in \sigma(S_1/S_2) \end{aligned}$$

Damit ist (2) bewiesen.

(3) Diese Gleichung gilt für alle Abbildungen.

(4) Die Gleichung gilt, weil σ injektiv ist.

Wir zeigen nun induktiv, über die Struktur von C und C', dass für alle Gatter g aus C bzw. C' gilt:

$$I_{C'}(g) = \sigma(I_C(g))$$

IA Wir betrachten ein Eingabegatter g. Das heißt, $\alpha(g) \in \{a_1, \ldots, a_n\} \subseteq M \cup \{0\}$. Also ist $I_C(g) = \{\alpha(g)\}$ und $\sigma(\alpha(g))$ definiert. Wir erhalten:

$$I_{C'}(g) = \{\sigma(\alpha(g))\} = \sigma(\{\alpha(g)\}) = \sigma(I_C(g))$$

IV Sei g ein Gatter mit Vorgängern. Die Aussagen $I_{C'}(p_1(g)) = \sigma(I_C(p_1(g)))$ und $I_{C'}(p_2(g)) = \sigma(I_C(p_2(g)))$ seien bereits bewiesen.

IS Es gilt $\alpha(g) \in \{\cup, \cap, \times, /\}$. Wir definieren:

$$\alpha' =_{def} \begin{cases} + & \text{falls } \alpha(g) = \times \\ - & \text{falls } \alpha(g) = / \\ \alpha(g) & \text{sonst} \end{cases}$$

Dann gilt:

$$\begin{aligned} I_{C'}(g) =& I_{C'}(p_1(g))\ \alpha'\ I_{C'}(p_1(g)) && \text{(nach Konstruktion)} \\ =& \sigma(I_C(p_1(g)))\ \alpha'\ \sigma(I_C(p_2(g))) && \text{(nach IV)} \\ =& \sigma\left(I_C(p_1(g))\ \alpha(g)\ I_C(p_2(g))\right) && \text{(nach den obigen Gleichungen)} \\ =& \sigma(I_C(g)) \end{aligned}$$

Also gilt für alle $a \in M \cup \{0\}$:

$$a \in I_C(g) \Leftrightarrow \sigma(a) \in \sigma(I_C(g)) \Leftrightarrow \sigma(a) \in I_{C'}(g)$$

Da $b \in M \cup \{0\}$ ist, erhalten wir:

$$(C, b) \in MC(\mathcal{O} \cup \{\times, /\}) \Leftrightarrow (C', \sigma(b)) \in MC^*(\mathcal{O} \cup \{+, -\})$$

(ii) Wegen De Morgan können wir davon ausgehen, dass $\cap \notin \mathcal{O}$. (Andernfalls können wir $\cap$-Gatter durch $\cup$- und $^-$-Gatter nachbauen, ohne die Größe es Schaltkreises wesentlich zu erhöhen.) Durch die $^-$-Gatter kann es sein, dass im Schaltkreis natürliche Zahlen berechnet werden, die sich nicht mehr bezüglich der ggT-freien Basis $q_1, \dots q_k$ darstellen lassen. Daher wollen wir die Konstruktion aus (i) mit der Primfaktorzerlegung (statt der Zerlegung bezüglich einer ggT-freien Basis) verwenden. Damit sich die Zahlen dennoch als endliche Vektor darstellen lassen, fassen wir alle Potenzen von Primfaktoren, die größer als die Primfaktoren in $a_1, \dots, a_n$ und b sind, zu einer einzigen Zahl zusammen. Sei $q_1, q_2, \dots$ die (aufsteigend geordnete) Folge der Primzahlen. Sei $k \in \mathbb{N}$ ausreichend groß, sodass alle Primfaktoren von $a_1, \dots, a_n$ und b in $\{q_1, \dots q_k\}$ enthalten sind.

Definiere

$$\sigma : \begin{array}{ll} \mathbb{N} & \to \mathbb{N}^{k+1} \cup \{\infty\} \\ 0 & \mapsto \infty \\ \prod_{j=1}^{\infty} q_j^{d_j} & \mapsto \left(d_1, \dots, d_k, \sum_{j=k+1}^{\infty} d_j\right) \end{array}$$

Für die Reduktion konvertieren wir C wie in (i) in einen $\mathcal{O} \cup \{+, -\}$-Schaltkreis C' über $\mathbb{N}^k \cup \{\infty\}$. Die Zerlegung der $a_1, \dots, a_n$ und von b in Primfaktoren ist in PSPACE möglich. Daher kann C' in PSPACE konstruiert werden.

Wir wollen nun zeigen, dass σ ein Monoid-Homomorphismus zwischen $(\mathbb{N}, \times, 1)$ und $(\mathbb{N}^{k+1} \cup \{\infty\}, +, (0, \dots 0))$ ist. Es ist $\sigma(1) = \sigma(\prod_{j=1}^{\infty} q_j^0) = (0, \dots, \sum_{j=k+1}^{\infty} 0) = (0, \dots, 0)$. Seien $\prod_{j=1}^{\infty} q_j^{d_j}, \prod_{j=1}^{\infty} q_j^{e_j} \in M$. Dann gilt:

$$\begin{aligned} \sigma\left(\prod_{j=1}^{\infty} q_j^{d_j} \times \prod_{j=1}^{\infty} q_j^{e_j}\right) &= \sigma\left(\prod_{j=1}^{\infty} q_j^{d_j+e_j}\right) \\ &= \left(d_1 + e_1, \dots, d_k + e_k, \sum_{j=k+1}^{\infty} d_j + e_j\right) \\ &= \left(d_1, \dots, d_k, \sum_{j=k+1}^{\infty} d_j\right) + \left(e_1, \dots, e_k, \sum_{j=k+1}^{\infty} e_j\right) \\ &= \sigma\left(\prod_{j=1}^{\infty} q_j^{d_j}\right) + \sigma\left(\prod_{j=1}^{\infty} q_j^{e_j}\right) \end{aligned}$$

Für alle $m \in M \cup \{0\}$ gilt

$$\sigma(0 \times m) = \sigma(0) = \infty = \infty + \sigma(m)$$

und analog:

$$\sigma(m \times 0) = \sigma(m) + \infty$$

Also ist σ ein Homomorphismus.

Im Gegensatz zur Konstruktion aus (i) ist σ allerdings nicht injektiv. Wir wollen zeigen, dass dennoch für alle Gatter g aus C und $a \in \mathbb{N}$ **Hilfsaussage 1** gilt:

$$a \in I_C(g) \Leftrightarrow \sigma(a) \in I_{C'}(g)$$

Dafür definieren wir:

$$M =_{def} \left\{ \prod_{j=1}^{k} q_j^{d_j} \;\middle|\; d_1, \ldots, d_k \in \mathbb{N} \right\}$$

$$K_i =_{def} \left\{ \prod_{j=k+1}^{\infty} q_j^{d_j} \;\middle|\; d_{k+1}, d_{k+2}, \cdots \in \mathbb{N} \text{ und } \sum_{j=k+1}^{\infty} d_j = i \right\} \text{ für alle } i \in \mathbb{N}$$

Dann ist $M = M \times \{1\} = M \times K_0$ und $\mathbb{N} \setminus \{0\} = \bigcup_{i=0}^{\infty} M \times K_i$. Mit $\times$ ist dabei die Multiplikation der Mengen (und nicht deren kartesische Produkt) gemeint.

Betrachte nun $m = \prod_{j=1}^{\infty} q_j^{d_j} \in \mathbb{N} \setminus \{0\}$. Dann gilt für alle $i \in \mathbb{N}$:

$$\begin{aligned} m \in M \times K_i \Leftrightarrow & \sum_{j=k+1}^{\infty} d_j = i \\ \Leftrightarrow & \sigma(m) = \left(d_1, \ldots, d_k, \sum_{j=k+1}^{\infty} d_j \right) \in \mathbb{N}^k \otimes \{i\} \end{aligned}$$

Dabei sei $\otimes$ das kartesische Produkt. Also ist $\sigma(M \times K_i) = \mathbb{N}^k \otimes \{i\}$ für alle $i \in \mathbb{N}$. Wir zeigen nun für alle $S_1, S_2 \subseteq M$:

$$\sigma(S_1 \times S_2) = \sigma(S_1) + \sigma(S_2) \tag{5}$$

$$\sigma(S_1 / S_2) = \sigma(S_1) - \sigma(S_2) \tag{6}$$

$$\sigma(S_1 \cup S_2) = \sigma(S_1) \cup \sigma(S_2) \tag{7}$$

$$\sigma(M \setminus S_1) = \sigma(M) \setminus \sigma(S_1) \tag{8}$$

Die Gleichungen (5) und (6) können analog zu den Gleichungen (1) und (2) aus (i) bewiesen werden. Da die Abbildung σ beschränkt auf M injektiv ist, folgen die Gleichungen (7) und (8) sofort. Sie gelten für alle injektiven Abbildungen.

Sei $e_i =_{def} (0, \ldots, 0, i) \in \mathbb{N}^{k+1}$. Wir beweisen folgende **Hilfsaussage 2**: Für jedes Gatter g in C gibt es $T \subseteq \{0\}$ und $S_0, S_1, \cdots \subseteq M$, sodass:

$$I_C(g) = T \cup \bigcup_{i \in \mathbb{N}} (S_i \times K_i)$$

$$I_{C'}(g) = \sigma(T) \cup \bigcup_{i \in \mathbb{N}} (\sigma(S_i) + \{e_i\})$$

Den Beweis führen wir über die Struktur von C.

IA: Sei g ein Eingabegatter. Dann ist $\alpha(g) = a_l$ für ein $1 \leq l \leq n$. Wir unterscheiden zwei Fälle:

(I) Sei $a_l = 0$. Sei $T = \{0\}$ und $S_0, S_1, S_2, \cdots =_{def} \emptyset$. Dann gilt:

$$I_C(g) = T = T \cup \bigcup_{i \in \mathbb{N}} (S_i \times K_i)$$
$$I_{C'}(g) = \{\sigma(0)\} = \sigma(T) = \sigma(T) \cup \bigcup_{i \in \mathbb{N}} (\sigma(S_i) + e_i)$$

(II) Sei $a_l > 0$. Dann ist $a_i \in M$. Sei $S_0 =_{def} \{a_l\}$ und $T, S_1, S_2, \cdots =_{def} \emptyset$. Dann gilt:

$$I_C(g) = S_0 = S_0 \times K_0 = T \cup \bigcup_{i \in \mathbb{N}} (S_i \times K_i)$$
$$I_{C'}(g) = \{\sigma(a_l)\} = \sigma(S_0) = S_0 + e_0 = \sigma(T) \cup \bigcup_{i \in \mathbb{N}} (\sigma(S_i) + e_i)$$

IS: Sei g ein Gatter aus C mit Vorgängern. Sei $g_1 =_{def} p_1(g)$ und $g_2 =_{def} p_2(g)$ bzw. $g_1 =_{def} p(g)$. Nach IV gibt es es $T_r \subseteq \{0\}$ und $S_0^r, S_1^r, \cdots \subseteq M$ mit $I_C(g_r) = T_r \cup \bigcup_{i \in \mathbb{N}} (S_i^r \times K_i)$ und $I_{C'}(g_r) = \sigma(T_r) \cup \bigcup_{i \in \mathbb{N}} (\sigma(S_i^r) + \{e_i\})$ für alle $r \in \{1, 2\}$ bzw. für $r = 1$.

(a) Sei g ein $\cup$-Gatter. Dann gilt:

$$\begin{aligned} I_C(g) &= I_C(g_1) \cup I_C(g_2) \\ &= \left(T_1 \cup \bigcup_{i \in \mathbb{N}} \left(S_i^1 \times K_i\right)\right) \cup \left(T_2 \cup \bigcup_{i \in \mathbb{N}} \left(S_i^2 \times K_i\right)\right) \\ &= (T_1 \cup T_2) \cup \bigcup_{i \in \mathbb{N}} ((S_i^1 \cup S_i^2) \times K_i) \\ I_{C'}(g) &= I_{C'}(g_1) \cup I_{C'}(g_2) \\ &= \sigma(T_1 \cup \bigcup_{i \in \mathbb{N}} (S_i^1 \times K_i)) \cup \sigma(T_2 \cup \bigcup_{i \in \mathbb{N}} (S_i^2 \times K_i)) \\ &= \sigma(T_1 \cup T_2) \cup \bigcup_{i \in \mathbb{N}} (\sigma(S_i^1 \cup S_i^2) + \{e_i\}) \end{aligned}$$

(b) Sei g ein $^-$-Gatter. Dann gilt:

$$\begin{aligned}
I_C(g) &= \overline{I_C(g_1)} \\
&= \overline{T_1 \cup \bigcup_{i\in\mathbb{N}} (S_i^1 \times K_i)} \\
&= (\{0\} \setminus T_1) \cup \bigcup_{i\in\mathbb{N}} ((M \setminus S_i^1) \times K_i) \\
I_{C'}(g) &= \overline{I_{C'}(g_1)} \\
&= \overline{\sigma(T_1) \cup \bigcup_{i\in\mathbb{N}} (\sigma(S_i^1) + \{e_i\})} \\
&= (\{\infty\} \setminus \sigma(T_1)) \cup \bigcup_{i\in\mathbb{N}} ((\mathbb{N}^k \otimes \{i\}) \setminus (\sigma(S_i^1) + \{e_i\})) \\
&= \sigma(\{0\} \setminus T_1) \cup \bigcup_{i\in\mathbb{N}} (((\mathbb{N}^k \otimes \{0\}) \setminus \sigma(S_i^1)) + \{e_i\}) \\
&= \sigma(\{0\} \setminus T_1) \cup \bigcup_{i\in\mathbb{N}} ((\sigma(M) \setminus \sigma(S_i^1)) + \{e_i\}) \\
&= \sigma(\{0\} \setminus T_1) \cup \bigcup_{i\in\mathbb{N}} (\sigma(M \setminus S_i^1) + \{e_i\})
\end{aligned}$$

(c) Sei g ein $\times$-Gatter. Für alle $i_1, i_2 \in \mathbb{N}$ gilt:

$$\begin{aligned}
K_{i_1} \times K_{i_2} &= \left\{ \prod_{j=k+1}^{\infty} q_j^{e_j} \;\middle|\; \sum_{j=k+1}^{\infty} e_j = i_1 \right\} \times \left\{ \prod_{j=k+1}^{\infty} q_j^{d_j} \;\middle|\; \sum_{j=k+1}^{\infty} d_j = i_2 \right\} \\
&= \left\{ \prod_{j=k+1}^{\infty} q_j^{d_j+e_j} \;\middle|\; \sum_{j=k+1}^{\infty} d_j = i_1 \wedge \sum_{j=k+1}^{\infty} e_j = i_2 \right\} \\
&= \left\{ \prod_{j=k+1}^{\infty} q_j^{f_j} \;\middle|\; \sum_{j=k+1}^{\infty} f_j = i_1 + i_2 \right\} \\
&= K_{i_1+i_2}
\end{aligned}$$

Damit gilt

$$\begin{aligned}
I_C(g) &= I_C(g_1) \times I_C(g_2) \\
&= \left(T_1 \cup \bigcup_{i\in\mathbb{N}} (S_i^1 \times K_i) \right) \times \left(T_2 \cup \bigcup_{i\in\mathbb{N}} (S_i^2 \times K_i) \right) \\
&= T \cup \bigcup_{i\in\mathbb{N}} \left(\left(\bigcup_{j=0}^{i} (S_j^1 \times S_{i-j}^2) \right) \times K_i \right) \\
&= T \cup \bigcup_{i\in\mathbb{N}} (S_i \times K_i)
\end{aligned}$$

mit folgenden Definitionen

$$T =_{def} (T_1 \times T_2) \cup \left(T_1 \times \bigcup_{i \in \mathbb{N}} (S_i^2 \times K_i) \right) \cup \left(T_2 \times \bigcup_{i \in \mathbb{N}} (S_i^1 \times K_i) \right)$$

$$S_i =_{def} \bigcup_{j=0}^{i} (S_j^1 \times S_{i-j}^2) \text{ für alle } i \in \mathbb{N}$$

Außerdem:

$$\begin{aligned}
I_{C'}(g) =& I_{C'}(g_1) + I_{C'}(g_2) \\
=& \left(\sigma(T_1) \cup \bigcup_{i \in \mathbb{N}} (\sigma(S_i^1) + \{e_i\}) \right) + \left(\sigma(T_2) \cup \bigcup_{i \in \mathbb{N}} (\sigma(S_i^2) + \{e_i\}) \right) \\
=& (\sigma(T_1) + \sigma(T_2)) \cup \left(\sigma(T_1) + \bigcup_{i \in \mathbb{N}} (\sigma(S_i^2) + \{e_i\}) \right) \\
& \cup \left(\sigma(T_2) + \bigcup_{i \in \mathbb{N}} (\sigma(S_i^1) + \{e_i\}) \right) \\
& \cup \left(\left(\bigcup_{i \in \mathbb{N}} (\sigma(S_i^1) + \{e_i\}) \right) + \left(\bigcup_{i \in \mathbb{N}} (\sigma(S_i^2) + \{e_i\}) \right) \right) \\
=& \sigma(T) \cup \bigcup_{i \in \mathbb{N}} \bigcup_{j=0}^{i} (\sigma(S_j^1) + \sigma(S_{i-j}^2) + \{e_i\}) \\
=& \sigma(T) \cup \bigcup_{i \in \mathbb{N}} \left(\left(\bigcup_{j=0}^{i} (\sigma(S_j^1) + \sigma(S_{i-j}^2)) \right) + \{e_i\} \right) \\
=& \sigma(T) \cup \bigcup_{i \in \mathbb{N}} \left(\left(\bigcup_{j=0}^{i} \sigma(S_j^1 \times S_{i-j}^2) \right) + \{e_i\} \right) \\
=& \sigma(T) \cup \bigcup_{i \in \mathbb{N}} (\sigma(S_i) + \{e_i\})
\end{aligned}$$

(d) Sei g ein /-Gatter. Es gilt:

$$\begin{aligned}
I(g) =& I(g_1)/I(g_2)\\
=& I(g_1)/\left(T_2 \cup \bigcup_{i\in\mathbb{N}}(S_i^2 \times K_i)\right)\\
=& (I(g_1)/T_2) \cup \left(I(g_1)/\bigcup_{i\in\mathbb{N}}(S_i^2 \times K_i)\right)\\
=& \left(I(g_1)/\bigcup_{i\in\mathbb{N}}(S_i^2 \times K_i)\right) \text{ (denn } T_2 \subseteq \{0\})\\
=& \left(T_1 \cup \bigcup_{i\in\mathbb{N}}(S_i^1 \times K_i)\right) / \bigcup_{i\in\mathbb{N}}(S_i^2 \times K_i)\\
=& T \cup \left(\left(\bigcup_{i\in\mathbb{N}}(S_i^1 \times K_i)\right) / \left(\bigcup_{i\in\mathbb{N}}(S_i^2 \times K_i)\right)\right)\\
=& T \cup \bigcup_{i\in\mathbb{N}}(S_i \times K_i) \text{ (nach (*))}
\end{aligned}$$

mit

$$\begin{aligned}
T =_{def} & T_1/\left(\bigcup_{i\in\mathbb{N}}(S_i^2 \times K_i)\right)\\
S_i =_{def} & \bigcup_{j=i}^{\infty}(S_j^1/S_{j-i}^2) \text{ für alle } i \in \mathbb{N}
\end{aligned}$$

Wir zeigen (*): Sei $x \in (\bigcup_{i\in\mathbb{N}}(S_i^1 \times K_i))/(\bigcup_{i\in\mathbb{N}}(S_i^2 \times K_i))$. Dann gibt es $i_1 \in \mathbb{N}$, $s_1 \in S_{i_1}^1$ und $k_1 \in K_{i_1}$, sowie $i_2 \in \mathbb{N}$, $s_2 \in S_{i_2}^2$ und $k_2 \in K_{i_2}$ mit $x = (s_1k_1)/(s_2k_2)$. s_2 teilt s_1 und k_2 teilt k_1. Insbesondere ist $i_1 \geq i_2$. Also ist $x = (s_1/s_2)(k_1/k_2) \in (S_{i_1}^1/S_{i_2}^2) \times K_{i_1-i_2}$ und $S_{i_1}^1/S_{i_2}^2 \subseteq S_{i_1-i_2}$.
Sei andersherum $x \in \bigcup_{i\in\mathbb{N}}((\bigcup_{j=i}^{\infty}(S_j^1/S_{j-i}^2)) \times K_i)$. Dann gibt es $i \in \mathbb{N}$, $j \geq i$, $s_1 \in S_j^1$, $s_2 \in S_{j-i}^2$ und $l \in K_i$ mit $x = (s_1/s_2)\ l = (s_1 l q_{k+1}^{j-i})/(s_2 q_{k+1}^{j-i}) \in (S_j^1 \times K_j)/(S_{j-i}^2 \times K_{j-i})$.

Außerdem:

$$\begin{aligned}
I_{C'}(g) =& I_{C'}(g_1) - I_{C'}(g_2) \\
=& I_{C'}(g_1) - \left(\sigma(T_2) \cup \bigcup_{i\in\mathbb{N}} (\sigma(S_i^2) + \{e_i\})\right) \\
=& (I_{C'}(g_1) - \sigma(T_2)) \cup \left(I_{C'}(g_1) - \bigcup_{i\in\mathbb{N}} (\sigma(S_i^2) + \{e_i\})\right) \\
=& I_{C'}(g_1) - \left(\bigcup_{i\in\mathbb{N}} (\sigma(S_i^2) + \{e_i\})\right) \text{ (denn } \sigma(T_2) \subseteq \{\infty\}) \\
=& \left(\sigma(T_1) \cup \bigcup_{i\in\mathbb{N}} (\sigma(S_i^1) + \{e_i\})\right) - \left(\bigcup_{i\in\mathbb{N}} (\sigma(S_i^2) + \{e_i\})\right) \\
=& \left(\sigma(T_1) - \left(\bigcup_{i\in\mathbb{N}} (\sigma(S_i^2) + \{e_i\})\right)\right) \\
& \cup \left(\left(\bigcup_{i\in\mathbb{N}} (\sigma(S_i^1) + \{e_i\})\right) - \left(\bigcup_{i\in\mathbb{N}} (\sigma(S_i^2) + \{e_i\})\right)\right) \\
=& \sigma(T) \cup \sigma\left(\left(\bigcup_{i\in\mathbb{N}} (S_i^1 \times K_i)\right) / \left(\bigcup_{i\in\mathbb{N}} (S_i^2 \times K_i)\right)\right) \\
=& \sigma(T) \cup \sigma\left(\bigcup_{i\in\mathbb{N}} (S_i \times K_i)\right) \\
=& \sigma(T) \cup \bigcup_{i\in\mathbb{N}} (\sigma(S_i) + \{e_i\})
\end{aligned}$$

Damit können wir nun Hilfsaussage 1 beweisen. Sei g ein Gatter aus C und $a \in \mathbb{N}$. Nach Hilfsaussage 2 gibt es $T \subseteq \{0\}$ und $S_0, S_1, \cdots \subseteq M$ mit $I_C(g) = T \cup \bigcup_{i\in\mathbb{N}} (S_i \times K_i)$ und $I_{C'}(g) = \sigma(T) \cup \bigcup_{i\in\mathbb{N}} (\sigma(S_i) + \{e_i\})$. Wir unterscheiden zwei Fälle:

(I) Es sei $a = 0$. Dann gilt

$$0 \in I_C(g) \Leftrightarrow T = \{0\} \Leftrightarrow \sigma(T) = \{\infty\} \Leftrightarrow \infty \in I_{C'}(g) \Leftrightarrow \sigma(0) \in I_{C'}(g)$$

(II) Es sei $a \neq 0$. Also gibt es $d_1, d_2, \cdots \in \mathbb{N}$ mit $a = \prod_{j=1}^{\infty} q_j^{d_j}$.
Sei $d =_{def} \sum_{j=k+1}^{\infty} d_j$. Es gilt:

$$\begin{aligned}
\prod_{j=1}^{\infty} q_j^{d_j} \in I_C(g) &\Leftrightarrow \prod_{j=1}^{\infty} q_j^{d_j} \in \bigcup_{i\in\mathbb{N}} (S_i \times K_i) \\
&\Leftrightarrow \prod_{j=1}^{\infty} q_j^{d_j} \in (S_d \times K_d) \\
&\Leftrightarrow \prod_{j=1}^{k} q_j^{d_j} \in S_d \\
&\Leftrightarrow \sigma\left(\prod_{j=1}^{k} q_j^{d_j}\right) \in \sigma(S_d) \text{ (da } \sigma \text{ beschränkt auf } M \text{ injektiv ist)} \\
&\Leftrightarrow \sigma\left(\prod_{j=1}^{k} q_j^{d_j}\right) + \{e_d\} \in \sigma(S_d) + \{e_d\} \\
&\Leftrightarrow \sigma\left(\prod_{j=1}^{k} q_j^{d_j}\right) + \{e_d\} \in \bigcup_{i\in\mathbb{N}} (\sigma(S_i) + \{e_i\}) \\
&\Leftrightarrow \sigma\left(\prod_{j=1}^{\infty} q_j^{d_j}\right) \in I_{C'}(g)
\end{aligned}$$

Insbesondere gilt für alle $a \in \mathbb{N}$:

$$a \in I(C) \Leftrightarrow \sigma(a) \in I(C')$$

□

Analog zu Satz 3.8 (1) zeigt man:

Satz 3.13
$MC^*(\cap, +, -) \in \mathrm{P}$

Folgerung 3.14 $MC(\cap, \times, /), MC(\times, /) \in \mathrm{P}$

Wir wollen nun $MC^*(\cup, \cap, ^-, +, -)$ auf alternierenden Maschinen lösen. Dabei gehen wir analog zu Algorithmus 1 vor. Um die Größe der Elemente, mit denen die Maschine pro Knoten rechnet, beschränken zu können, verallgemeinern wir Lemma 3.1:

Lemma 3.15 *Sei $C = (V, E, g_C, \alpha)$ ein $\mathcal{O}$-Schaltkreis mit $\emptyset \neq \mathcal{O} \subseteq \{\cup, \cap, ^-, +, -\}$ über $\mathbb{N}^m \cup \{\infty\}$ mit $m \geq 1$. Sei $n_C =_{def} (2^{|C|}+1, \ldots, 2^{|C|}+1) \in \mathbb{N}^m$. Dann gilt für alle $g \in V$ und für alle $z = (z_1, \ldots, z_m) \in \mathbb{N}^m$, für die es ein $1 \leq i \leq m$ mit $z_i \geq 2^{|C|}+1$ gibt:*

$$z \in I(g) \Leftrightarrow n_C \in I(g)$$

Beweis. Sei C_g der Schaltkreis, der aus C entsteht, wenn man $g \in V$ als Ausgabegatter verwendet und alle Gatter, von denen aus g nicht erreichbar ist, entfernt. Sei außerdem $n_g =_{def} (2^{|C_g|}+1, \ldots, 2^{|C_g|}+1) \in \mathbb{N}^m$.
Wir zeigen die folgende **Hilfsaussage**: Für alle $g \in V$ und für alle $z = (z_1, \ldots, z_m) \in \mathbb{N}^m$, für die ein $1 \leq i \leq m$ mit $z_i \geq 2^{|C_g|}+1$ existiert, gilt:

$$z \in I(g) \;\Leftrightarrow\; n_g \in I(g)$$

Induktion über die Struktur von C:

IA: Sei g ein Eingabegatter und $\alpha(g) = (y_1, \ldots, y_m)$. Damit ist $I(g) = \{(y_1, \ldots, y_m)\}$. Da die Binärdarstellung von y_i in der Codierung von C_g enthalten sein muss, ist $\log(y_i)+1 \leq |C_g|$ und damit $y_i \leq 2^{\log(y_i)+1} < 2^{|C_g|}+1$ für alle $1 \leq i \leq m$. Daher gilt für alle $z = (z_1, \ldots, z_m) \in \mathbb{N}^m$, für die es ein $1 \leq i \leq m$ mit $z_i \geq 2^{|C_g|}+1$ gibt, schon $z \notin I(g)$. Insbesondere ist $n_g \notin I(g)$.

IV: Sei g ein Gatter aus C mit Vorgängern. Die Hilfsaussage sei für $p(g)$ bzw. $p_1(g)$ und $p_2(g)$ bereits bewiesen.

IS: Sei $z = (z_1, \ldots, z_m) \in \mathbb{N}^m$ und $z_i \geq 2^{|C_g|}+1$ für ein $1 \leq i \leq m$.
Fall 1: Sei $\alpha(g) = \cup$. Es ist $z_i \geq 2^{|C_g|}+1 \geq 2^{|C_{p_j(g)}|}+1$ für alle $j \in \{1,2\}$. Daraus folgt:

$$\begin{aligned} z \in I(g) &\Leftrightarrow z \in I(p_1(g)) \vee z \in I(p_2(g)) \\ &\Leftrightarrow n_{p_1(g)} \in I(p_1(g)) \vee n_{p_2(g)} \in I(p_2(g)) \text{ (nach IV)} \\ &\Leftrightarrow n_g \in I(p_1(g)) \vee n_g \in I(p_2(g)) \text{ (nach IV)} \\ &\Leftrightarrow n_g \in I(g) \end{aligned}$$

Fall 2: Sei $\alpha(g) = \cap$. Wir zeigen $(z \in I(g) \Leftrightarrow n_g \in I(g))$ analog zu Fall 1.
Fall 3: Sei $\alpha(g) = {}^-$. Es ist $z_i \geq 2^{|C_g|}+1 \geq 2^{|C_{p(g)}|}+1$. Daraus folgt:

$$\begin{aligned} z \in I(g) &\Leftrightarrow z \notin I(p(g)) \\ &\Leftrightarrow n_{p(g)} \notin I(p(g)) \text{ (nach IV)} \\ &\Leftrightarrow n_g \notin I(p(g)) \text{ (nach IV)} \\ &\Leftrightarrow n_g \in I(g) \end{aligned}$$

Fall 4: Sei $\alpha(g) = +$.
Fall 4 a: Sei $n_{p_1(g)} \notin I(p_1(g))$ und $n_{p_2(g)} \notin I(p_2(g))$. Nach IV gilt damit $I(p_1(g)) \subseteq [0, 2^{|C_{p_1(g)}|}]^m$ und $I(p_2(g)) \subseteq [0, 2^{|C_{p_2(g)}|}]^m$. Daher gilt:

$$I(g) = I(p_1(g)) + I(p_2(g)) \subseteq [0, 2^{|C_{p_1(g)}|} + 2^{|C_{p_2(g)}|}]^m \subset [0, 2^{|C_g|}]^m$$

Die letzte Inklusion ergibt sich mit $(2^{|C_{p_1(g)}|}+1)+(2^{|C_{p_2(g)}|}+1) \leq (2^{|C_g|}+1)$. Die Aussage nennen wir (1) und sie lässt sich analog zu der entsprechenden Aussage aus dem Beweis von Lemma 3.1 zeigen. Also ist $z \notin I(g)$ und $n_g \notin I(g)$.
Fall 4 b: Sei $n_{p_1(g)} \in I(p_1(g))$ und $n_{p_2(g)} \notin I(p_2(g))$. Für alle $y = (y_1, \ldots, y_m) \in I(p_2(g))$ gilt $y_i < 2^{|C_{p_2(g)}|}+1$ und damit auch (2):

$$z_i - y_i \geq (2^{|C_g|}+1) - y_i > (2^{|C_g|}+1) - (2^{|C_{p_2(g)}|}+1) \geq 2^{|C_{p_1(g)}|}+1$$

Die letzte Ungleichung ergibt sich mit der Aussage (1).
Also gilt:

$$\begin{aligned} z \in I(g) &\Leftrightarrow \exists y \in \mathbb{N}^m : (z-y) \in I(p_1(g)) \wedge y \in I(p_2(g)) \\ &\Leftrightarrow \exists y \in \mathbb{N}^m : n_{p_1(g)} \in I(p_1(g)) \wedge y \in I(p_2(g)) \text{ (nach (2) und IV)} \\ &\Leftrightarrow \exists y \in \mathbb{N}^m : (n_g - y) \in I(p_1(g)) \wedge y \in I(p_2(g)) \text{ (nach (2) und IV)} \\ &\Leftrightarrow n_g \in I(g) \end{aligned}$$

Fall 4 c: Sei $n_{p_1(g)} \notin I(p_1(g))$ und $n_{p_2(g)} \in I(p_2(g))$. Wir zeigen ($z \in I(g) \Leftrightarrow n_g \in I(g)$) analog zu Fall 4 b.
Fall 4 d: Sei $n_{p_1(g)} \in I(p_1(g))$ und $n_{p_2(g)} \in I(p_2(g))$. Nach (1) gilt (3):

$$z_i - (2^{|C_{p_2(g)}|}+1) \geq (2^{|C_g|}+1) - (2^{|C_{p_2(g)}|}+1) \geq 2^{|C_{p_1(g)}|}+1$$

Definiere:

$$\begin{aligned} a =_{def} (&z_1 &,\ldots &,z_{i-1} &,z_i - (2^{|C_{p_2(g)}|}+1) &,z_{i+1} &,\ldots &,z_m &) \\ b =_{def} (&0 &,\ldots &,0 &,2^{|C_{p_2(g)}|}+1 &,0 &,\ldots &,0 &) \end{aligned}$$

Dann ist $a \in I(p_1(g))$ nach (3) und IV, $b \in I(p_2(g))$ nach IV und damit $a + b = z \in I(g)$. Definiere außerdem:

$$\begin{aligned} c &=_{def} \left(\left((2^{|C_g|}+1) - (2^{|C_{p_2(g)}|}+1)\right), \ldots, \left((2^{|C_g|}+1) - (2^{|C_{p_2(g)}|}+1)\right)\right) \in \mathbb{N}^m \\ d &=_{def} (2^{|C_{p_2(g)}|}+1, \ldots, 2^{|C_{p_2(g)}|}+1) \in \mathbb{N}^m \end{aligned}$$

Dann ist $c \in I(p_1(g))$ nach (3) und IV, $d \in I(p_2(g))$ nach IV und damit $c + d = n_g \in I(g)$.
Fall 5: Sei $\alpha(g) = -$. Für alle $y = (y_1, \ldots, y_m) \in \mathbb{N}^m$ gilt (4):

$$z_i + y_i \geq (2^{|C_g|}+1) + y_i \geq (2^{|C_g|}+1) \geq 2^{|C_{p_1(g)}|}+1$$

$$\begin{aligned} z \in I(g) &\Leftrightarrow \exists y \in \mathbb{N}^m : z + y \in I(p_1(g)) \wedge y \in I(p_2(g)) \\ &\Leftrightarrow \exists y \in \mathbb{N}^m : n_{p_1(g)} \in I(p_1(g)) \wedge y \in I(p_2(g)) \text{ (nach (4) und IV)} \\ &\Leftrightarrow \exists y \in \mathbb{N}^m : (n_g + y) \in I(p_1(g)) \wedge y \in I(p_2(g)) \text{ (nach (4) und IV)} \\ &\Leftrightarrow n_g \in I(g) \end{aligned}$$

Das Lemma folgt aus der Hilfsaussage, weil $|C_g| \leq |C|$ für alle $g \in V$ ist. Damit gilt für alle $z = (z_1, \ldots, z_m) \in \mathbb{N}^m$, für die ein $1 \leq i \leq m$ mit $z_i \geq 2^{|C|} + 1$ existiert:

$$\begin{aligned} z \in I(g) &\Leftrightarrow n_g \in I(g) \\ &\Leftrightarrow n_C \in I(g) \end{aligned}$$

□

Satz 3.16
$MC^*(\cap, \cup, ^-, +, -) \in$ PSPACE

Beweis. Nach De Morgan gilt:

$$MC^*(\cup, \cap, ^-, +, -) \equiv_m^{\log} MC^*(\cup, ^-, +, -)$$

Sei $m \in \mathbb{N}^+$. Algorithmus 2 löst das Problem $MC^*(\cup, ^-, +, -)$, für gültige Codierungen von $\{\cap, \cup, ^-, +, -\}$-Schaltkreisen C über $\mathbb{N}^m \cup \{\infty\}$ und Zahlen $b \in \mathbb{N}^m \cup \{\infty\}$ auf einer alternierenden Maschine.

Algorithm 2 Algorithmus für Schaltkreise mit Vereinigung, Komplement, Addition und Subtraktion über $\mathbb{N}^m \cup \{\infty\}$

Eingabe: (C, b, m)

Sei $n = 2^{|C|} + 1$, $g = g_C$, $x = b$ und $t = 1$.
while $\alpha(g) \notin \mathbb{N}^m \cup \{\infty\}$ **do**
  if $x \neq \infty$ **then**
    Sei $(x_1, \ldots, x_m) = x$.
    if $\exists i \in \{1, \ldots, m\} : x_i \geq n$ **then**
      Sei $x = (n, \ldots, n) \in \mathbb{N}^m$.
    end if
  end if
  // Falls $t == 1$ teste $x \in I(g)$ und falls $t == 0$ teste $x \notin I(g)$.
  if $\alpha(g) == \cup$ **then**
    // Vereinigung
    if $\alpha(g) == \cup$ und $t == 0$ **then**
      Setze mittels universellem Zustand parallel $g = p_1(g)$ und $g = p_2(g)$.
    else if $\alpha(g) == \cup$ und $t == 1$ **then**
      Rate $i \in \{1, 2\}$.
      Sei $g = p_i(g)$.
    end if
  else if $\alpha(g) == ^-$ **then**
    // Komplement
    Sei $g = p(g)$ und $t = 1 - t$.
  else if $\alpha(g) == +$ **then**
    // Addition
    if $t == 1$ **then**

24: // Rate $a \in [0,n]^m \cup \{\infty\}$.
25: Rate $j \in \{0,1\}$.
26: Sei $a = \infty$.
27: **if** $j == 1$ **then**
28: **for all** $i \in \{1,\dots,m\}$ **do**
29: Rate $a_i \in [0,n] \cap \mathbb{N}$.
30: **end for**
31: Sei $a = (a_1,\dots,a_m)$.
32: **end if**
33: // Rate $b \in [0,n]^m \cup \{\infty\}$.
34: Rate $k \in \{0,1\}$.
35: Sei $b = \infty$.
36: **if** $k == 1$ **then**
37: **for all** $i \in \{1,\dots,m\}$ **do**
38: Rate $b_i \in [0,n] \cap \mathbb{N}$.
39: **end for**
40: Sei $b = (b_1,\dots,b_m)$.
41: **end if**
42: // Stelle $a + b == x$ sicher.
43: **if** $a + b \neq x$ **then**
44: Sei $a = x$ und $b = (0,\dots,0) \in \mathbb{N}^m$.
45: **end if**
46: Setze mittels universellem Zustand parallel $(x,g) = (a, p_1(g))$ und $(x,g) = (b, p_2(g))$.
47: **else**
48: // $t == 0$
49: // Setze a parallel auf jeden Wert aus $[0,n]^m \cup \{\infty\}$.
50: Setze j mittels universellem Zustand parallel auf jeden Wert aus $\{0,1\}$.
51: Sei $a = \infty$.
52: **if** $j == 1$ **then**
53: **for all** $i \in \{1,\dots,m\}$ **do**
54: Setze a_i mittels universellem Zustand parallel auf jeden Wert aus $[0,n] \cap \mathbb{N}$.
55: **end for**
56: Sei $a = (a_1,\dots,a_m)$.
57: **end if**
58: // Setze b parallel auf jeden Wert aus $[0,n]^m \cup \{\infty\}$.
59: Setze k mittels universellem Zustand parallel auf jeden Wert aus $\{0,1\}$.
60: Sei $b = \infty$.
61: **if** $k == 1$ **then**
62: **for all** $i \in \{1,\dots,m\}$ **do**
63: Setze b_i mittels universellem Zustand parallel auf jeden Wert aus $[0,n] \cap \mathbb{N}$.
64: **end for**

65: Sei $b = (b_1, \ldots, b_m)$.
66: **end if**
67: // Stelle $a + b == x$ sicher.
68: **if** $a + b \neq x$ **then**
69: Sei $a = x$ und $b = (0, \ldots, 0) \in \mathbb{N}^m$.
70: **end if**
71: Rate $i \in \{1, 2\}$.
72: **if** $k == 1$ **then**
73: $(x, g) = (a, p_1(g))$.
74: **else**
75: $(x, g) = (b, p_2(g))$.
76: **end if**
77: **end if**
78: **else**
79: // Subtraktion
80: **if** $t == 1$ **then**
81: **for all** $i \in \{1, \ldots, m\}$ **do**
82: Rate bitweise ein binär codiertes $a_i \in [0, n] \cap \mathbb{N}$.
83: **end for**
84: Sei $a = (a_1, \ldots, a_m)$.
85: Setze mittels universellem Zustand parallel $(x, g) = (x + a, p_1(g))$ und $(x, g) = (a, p_2(g))$.
86: // Es gilt: $x + a == \infty$, falls $x == \infty$
87: **else**
88: // $t == 0$
89: **for all** $i \in \{1, \ldots, m\}$ **do**
90: Setze a_i mittels universellem Zustand parallel auf jeden Wert aus $[0, n] \cap \mathbb{N}$.
91: **end for**
92: Sei $a = (a_1, \ldots, a_m)$.
93: Rate $i \in \{1, 2\}$.
94: **if** $i == 1$ **then**
95: Sei $(x, g) = (a + x, p_1(g))$
96: // Es gilt: $x + a == \infty$, falls $x == \infty$
97: **else**
98: Sei $(x, g) = (a, p_2(g))$.
99: **end if**
100: **end if**
101: **end if**
102: **end while**
103: // g ist ein Eingabegatter.
104: **if** $t == 1$ **then**
105: Akzeptiere, falls $x == \alpha(g)$ gilt. Andernfalls lehne ab.
106: **else**

107: Lehne ab falls $x == \alpha(g)$ gilt. Andernfalls akzeptiere.
108: **end if**

Korrektheit: Wir zeigen die Korrektheit von Algorithmus 2 analog zur Korrektheit von Algorithmus 1 aus dem Beweis von Satz 3.3. Sei x', g' und t' wieder die aktuelle Variablenbelegung von x, g und t und $\beta(g', x', t')$ und $A(g', x', t')$ definiert wie in dem Beweis.

Zur Erinnerung: $\beta(g', x', t')$ war der Teilbaum des Berechnungsbaums, ab dem Zeitpunkt zu dem x, g und t zum ersten Mal die Werte x', g' und t' annehmen und $A(g', x', t')$ war ein Wahrheitswert, der anzeigt, ob der Algorithmus sich für x', g' und t' korrekt verhält. Wir hatten per Induktion gezeigt, dass $A(g', x', t')$ genau dann 1 ist, wenn $\beta(g', x', t')$ eine akzeptierenden Teilbaum hat.

In Algorithmus 2 ist $x' \in \mathbb{N}^m \cup \{\infty\}$ statt $x' \in \mathbb{N}$. Wir können im Induktionsschluss o.B.d.A. annehmen, dass $x' \in [0, 2^{|C|}+1]^m \cup \{\infty\}$ ist. Falls dies zu Beginn eines Durchlaufs der while-Schleife nicht der Fall ist, so wird x' in Zeile 6 der Wert $n' =_{def} (n, \ldots, n) \in \mathbb{N}^m$ zugewiesen. Diese Zuweisung ändert gemäß Lemma 3.15 am Wahrheitswert von $x \in I(g)$ nichts.

In den Fällen $\alpha(g') = {}^-$ und $\alpha(g') = \cup$ verhält sich Algorithmus 2 genau wie Algorithmus 1. Wir wollen nun nachrechnen, dass sich Algorithmus 2 auch in den anderen beiden Fällen $\alpha(g') = +$ und $\alpha(g') = -$ korrekt verhält.

Wir unterscheiden die Fälle $t = 1$ und $t = 0$:

(I) Sei $t = 1$. Wir unterscheiden die Fälle $\alpha(g') = +$ und $\alpha(g') = -$:

  (a) Sei $\alpha(g') = +$. Es gilt $x' \in [0, 2^{|C|}+1]^m \cup \{\infty\}$. Der Algorithmus erreicht Zeile 25. $\beta(g', x', t')$ hat für alle $a, b \in [0, 2^{|C|}+1]^m \cup \{\infty\}$ mit $a + b = x'$ Teilbäume $\beta(p_1(g), a, t')$ und $\beta(p_2(g), b, t')$, welche durch eine Kette von existenziellen Zuständen und einem universellen Zustand erreicht werden. $\beta(g', x', t')$ hat also genau dann einen akzeptierenden Teilbaum, wenn es $a, b \in [0, 2^{|C|} +$

$1]^m \cup \{\infty\}$ mit $a + b = x'$ gibt, sodass $\beta(g_0, a, t')$ und $\beta(g_1, b, t')$ jeweils einen akzeptierenden Teilbaum haben. Es gilt:

$$
\begin{aligned}
A(g', x', t') &= \begin{cases} 1 & \text{falls } x' \in I(g') \\ 0 & \textit{sonst} \end{cases} \\
&= \begin{cases} 1 & \text{falls } \exists a, b \in \mathbb{N}^m \cup \{\infty\} \text{ mit } a + b = x' : \\ & a \in I(p_1(g)) \wedge b \in I(p_2(g)) \\ 0 & \textit{sonst} \end{cases} \\
&= \begin{cases} 1 & \text{falls } \exists a, b \in [0, 2^{|C|} + 1]^m \cup \{\infty\} \text{ mit } a + b = x' : \\ & a \in I(p_1(g)) \wedge b \in I(p_2(g)) \\ 0 & \textit{sonst} \end{cases} \\
&\quad \text{(weil } x' \in [0, 2^{|C|} + 1]^m \cup \{\infty\}\text{)} \\
&= \begin{cases} 1 & \text{falls } \exists a, b \in [0, 2^{|C|} + 1]^m \cup \{\infty\} \text{ mit } a + b = x' : \\ & A(g_0, a, t') = 1 \wedge A(g_1, b, t') = 1 \\ 0 & \textit{sonst} \end{cases}
\end{aligned}
$$

(b) Sei $\alpha(g') = -$. Es gilt $x' \in [0, 2^{|C|} + 1]^m \cup \{\infty\}$. Der Algorithmus erreicht Zeile 81. $\beta(g', x', t')$ hat für jedes $a \in [0, 2^{|C|} + 1]^m$ Teilbäume $\beta(g_0, x' + a, t')$ und $\beta(g_1, a, t')$, welche durch eine Kette von existenziellen Zuständen und einem universellen Zustand erreicht werden. $\beta(g', x', t')$ hat also genau dann einen akzeptierenden Teilbaum, wenn es ein $a \in [0, 2^{|C|} + 1]^m$ gibt, sodass $\beta(p_1(g), x' + a, t')$ und $\beta(p_2(g), a, t')$ jeweils einen akzeptierenden Teilbaum haben. Es gilt:

$$
\begin{aligned}
A(g', x', t') &= \begin{cases} 1 & \text{falls } x' \in I(g') \\ 0 & \textit{sonst} \end{cases} \\
&= \begin{cases} 1 & \text{falls } \exists a \in \mathbb{N}^m : x' + a \in I(p_1(g)) \text{ und } a \in I(p_2(g)) \\ 0 & \textit{sonst} \end{cases} \\
&\quad \text{(denn } b - \infty \text{ ist für alle } b \in \mathbb{N}^m \cup \{\infty\} \text{ nicht definiert)} \\
&= \begin{cases} 1 & \text{falls } \exists a \in [0, 2^{|C|} + 1]^m : x' + a \in I(p_1(g)) \\ & \text{und } a \in I(p_2(g)) \\ 0 & \textit{sonst} \end{cases} \\
&\quad \text{(nach Lemma 3.15)} \\
&= \begin{cases} 1 & \text{falls } \exists a \in [0, 2^{|C|} + 1]^m : A(g_0, x' + a, t') = 1 \\ & \text{und } A(g_1, a, t') = 1 \\ 0 & \textit{sonst} \end{cases}
\end{aligned}
$$

In beiden Fällen hat $\beta(g', x', t')$ nach IV also genau dann einen akzeptierenden Teilbaum, wenn $A(g', x', t') = 1$ gilt.

(II) Sei $t = 0$. Die Aussage lässt sich analog zu (I) zeigen. Dabei sind universelle und existenzielle Zustände, $\in$ und $\notin$, der Quantor und „und“ und „oder“ jeweils vertauscht.

Laufzeit: C ist azyklisch. Daher nimmt die Variable g auf jedem Pfad von der Wurzel von $\beta_M(C, b, m)$ zu einem Blatt den Wert eines Gatters aus C maximal einmal an. In jedem Schleifendurchlauf ändert sich g. Daher liegt die Anzahl der Schleifendurchläufe in $O(|C|)$. Die Werte von x werden durch jeweils $O(m \cdot |C|)$ Zeichen beschreiben. Alle Operationen, die pro Schleifendurchlauf durchgeführt werden, sind linear in der Größe von x. Die Laufzeit pro Schleifendurchlauf ist daher in $O(m \cdot |C|)$. Die Laufzeit des Codes vor und nach der while-Schleife ist ebenfalls in $O(m \cdot |C|)$. Es ist $m \in O(|C|)$, weil C mindestens ein Eingabegatter hat. Daher liegt die Laufzeit von Algorithmus 2 in $O(n^3)$, wobei n die Eingabegröße beschreibt. Nach Satz 2.4 ist $\text{ATIME}(n^3) \subseteq \text{DSPACE}(n^3)$. Die Gültigkeit der Eingabe kann in deterministischem logarithmischem Raum überprüft werden. Also ist:

$$\Rightarrow MC^*(\cup, \cap, ^-, +, -) \equiv_m^{\log} MC^*(\cup, ^-, +, -) \in \text{DSPACE}(n^3) \subseteq \text{PSPACE}$$

□

Aus Satz 3.16 und Satz 3.12 (2) folgt:

Folgerung 3.17 *Für alle $\mathcal{O}$ mit $\emptyset \neq \mathcal{O} \subseteq \{\cup, \cap, ^-, \times, /\}$ ist $MC(\mathcal{O}) \in$ PSPACE.*

Korollar 3.18
Für alle $\mathcal{O} \in \{\{\cup, \cap, ^-, \times\}, \{^-, \times\}, \{\cup, \cap, \times\}\}$ gilt:

1. *$MC(\mathcal{O} \cup \{/\})$ ist $\leq_m^{\log}$-vollständig für* PSPACE.
2. *$MC(\mathcal{O}) \equiv_m^{\log} MC(\mathcal{O} \cup \{/\})$*

Beweis. 1. Wie im Beweis von Korollar 3.4 reduzieren wir das PSPACE-harte Problem QBF auf $MC(\cup, \cap, \times)$. Dafür verwenden wir wieder den Beweis von Lemma 5.1 aus [MW07] von McKanzie und Wagner. Zusammen mit Lemma 2.3 erhalten wir:

$$\text{QBF} \leq_m^{\log} MC(\cup, \cap, \times) \leq_m^{\log} MC(\cup, \cap, \times, /) \leq_m^{\log} MC(\cup, \cap, ^-, \times, /)$$

Barth et al. haben in Korollar 43 von [Bar+17] gezeigt, dass $MC(^-, \times) \leq_m^{\log}$-hart für PSPACE ist. Aus Lemma 2.3 erhalten wir:

$$MC(^-, \times) \leq_m^{\log} MC(^-, \times, /)$$

Mit Folgerung 3.17 folgt, dass $MC(\cup, \cap, ^-, \times, /)$, $MC(^-, \times, /)$ und $MC(\cup, \cap, \times, /)$ PSPACE-vollständig sind.

2. Wir haben oben bereits gesehen, dass $MC(\cup, \cap, ^{-}, \times)$ und $MC(\cup, \cap, \times)$ $\leq_m^{\log}$-hart für PSPACE sind. McKanzie und Wagner haben ebenfalls gezeigt, dass beide Probleme in PSPACE sind. Damit folgen die beiden Äquivalenzen $MC(\cup, \cap, ^{-}, \times) \equiv_m^{\log} MC(\cup, \cap, ^{-}, \times, /)$ und $MC(\cup, \cap, \times) \equiv_m^{\log} MC(\cup, \cap, \times, /)$.

   Barth et al. haben in Lemma 2 (2) von [Bar+17] $\overline{EC(^{-}, +)} \leq_m^{\log} MC(^{-}, +)$ bewiesen. Theorem 42 des selben Reports besagt, dass $EC(^{-}, +) \leq_m^{\log}$-hart für PSPACE ist. PSPACE ist bekanntlich unter Komplement abgeschlossen. Daher ist $MC(^{-}, +)$ ebenfalls $\leq_m^{\log}$-hart für PSPACE. Damit ist $MC(^{-}, +) \leq_m^{\log} MC(\cup, \cap, ^{-}, \times)$ nach Lemma 2.3 und es gilt auch $MC(^{-}, +) \in$ PSPACE. Dies zeigt, dass $MC(^{-}, +)$ $\leq_m^{\log}$-vollständig für PSPACE ist und daraus folgt die Äquivalenz $MC(^{-}, \times) \equiv_m^{\log} MC(^{-}, \times, /)$.

   □

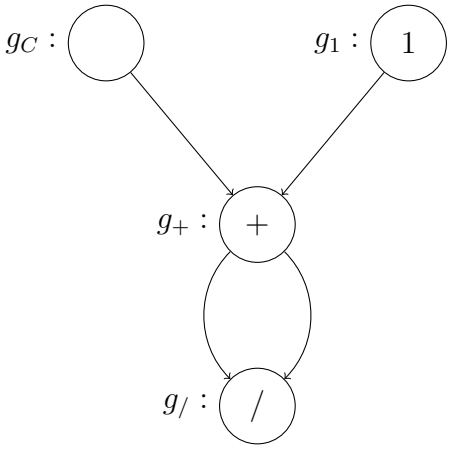
4 Untere Schranken

4.1 Schaltkreise mit Komplement, Addition und Division

Wir wollen in diesem Abschnitt die PSPACE-Vollständigkeit von $MC(^-, +, /)$ beweisen. Dafür werden wir die Addition und die Division verwenden, um zu überprüfen, ob ein Schaltkreis an seinem Ausgabegatter eine leere Menge liefert. So können wir das Komplement des Emptiness-Problems für Schaltkreise mit Addition auf das entsprechende Memebership-Problem mit Addition und Division reduzieren.

Satz 4.1
Sei $\{+\} \subseteq \mathcal{O} \subseteq \{\cup, \cap, ^-, +, \times, /\}$. *Dann gilt* $\overline{EC(\mathcal{O})} \leq_m^{\log} MC(\mathcal{O} \cup \{/\})$.

Beweis. Sei C ein $\mathcal{O}$-Schaltkreis mit Ausgabegatter g_C. Füge drei neue Gatter g_1, g_+ und $g_/$, so in C ein, dass g_C und g_1 die Vorgänger von g_+ sind und g_+ der erste und der zweite Vorgänger von $g_/$ ist. Es sei $\alpha(g_1) = 1$, $\alpha(g_+) = +$ und $\alpha(g_/) = /$. Den so konstruierten Schaltkreis mit Ausgabegatter $g_/$ nennen wir C'. Insbesondere ist $I_{C'}(g_1) = \{1\}$ und $I_{C'}(g_+) = I_C(g_C) + \{1\}$.



Dann gilt:

$$\begin{aligned} & I(C) \neq \emptyset \\ \Leftrightarrow & \exists b \in \mathbb{N} : b \in I_C(g_C) \\ \Leftrightarrow & \exists b \in \mathbb{N} : b + 1 \in I_{C'}(g_+) \\ \Leftrightarrow & 1 \in I(C') \end{aligned}$$

Also:

$$C \in \overline{EC(\mathcal{O})} \Leftrightarrow (C', 1) \in MC(\mathcal{O} \cup \{/\})$$

Daraus folgt die Behauptung, weil die Konstruktion von C' aus C in logarithmischen Raum durchgeführt werden kann. □

Folgerung 4.2 *$MC(^-, +, /)$ ist $\leq_m^{\log}$-vollständig für* PSPACE.

Beweis. Sei $A \in$ PSPACE. Wir wollen A auf $MC(^-, +, /)$ reduzieren. PSPACE ist unter Komplementbildung abgeschlossen. Das heißt, es ist auch $\overline{A} \in$ PSPACE. In Theorem 42 aus [Bar+17] wurde gezeigt, dass $EC(^-, +) \leq_m^{\log}$-hart für PSPACE ist. Also gilt $\overline{A} \leq_m^{\log} EC(^-, +)$ und damit auch $A \leq_m^{\log} \overline{EC(^-, +)}$. Mit Satz 4.1 erhalten wir $A \leq_m^{\log} MC(^-, +, /)$, weil $\leq_m^{\log}$ transitiv ist. Also ist $MC(^-, +, /)$ auch $\leq_m^{\log}$-hart für PSPACE. Mit Satz 3.3 erhalten wir die PSPACE-Vollständigkeit. □

4.2 Schaltkreise mit Schnitt, Addition, Multiplikation und Division

In diesem Abschnitt wollen wir zeigen, dass $MC(+, \times, /)$ $\mathcal{PIT}$-hart ist. Dafür zeigen wir zunächst, dass $MC(\cap, +, \times, /)$ $\mathcal{PIT}$-hart ist und zeigen dann, dass wir $MC(\cap, +, \times, /)$ auf $MC(+, \times, /)$ reduzieren können, indem wir die $\cap$-Gatter mit Multiplikation und Division nachbauen.

Folgerung 4.3 *$MC(\cap, +, \times, /)$ ist $\leq_m^{\log}$-hart für $\mathcal{PIT}$.*

Beweis. In Theorem 58 aus [Bar+17] wurde $MC(\cap, +, \times) \equiv_m^{\log}$ PIT bewiesen. Daher gilt für alle $A \in \mathcal{PIT}$:

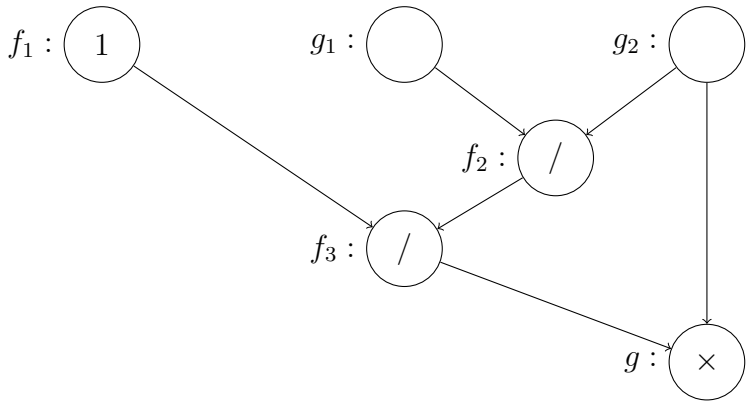
$$A \leq_m^{\log} \text{PIT} \leq_m^{\log} MC(\cap, +, \times) \leq_m^{\log} MC(\cap, +, \times, /)$$

□

Lemma 4.4
Sei $\emptyset \subseteq \mathcal{O} \subseteq \{+\}$. Dann ist $MC(\{\cap, \times, /\} \cup \mathcal{O}) \equiv_m^{\log} MC(\{\times, /\} \cup \mathcal{O})$.

Beweis. Wir zeigen $MC(\{\cap, \times, /\} \cup \mathcal{O}) \leq_m^{\log} MC(\{\times, /\} \cup \mathcal{O})$. Dafür bauen wir ein $\cap$-Gatter mit den anderen Gattern nach.

Sei C ein $(\{\cap, \times, /\} \cup \mathcal{O})$-Schaltkreis. Dann sei C' der $(\{\times, /\} \cup \mathcal{O})$-Schaltkreis, in dem wir jedes $\cap$-Gatter g aus C wie folgt umbauen:
Seien $g_i =_{def} p_i(g)$ für $i \in \{1, 2\}$ die Vorgänger von g in C. Wir fügen neue Gatter f_1, f_2 und f_3 in C' ein. Dabei sei $\alpha_{C'}(f_1) = 1$ und $\alpha_{C'}(f_2) = \alpha_{C'}(f_3) = /$. Außerdem sei $\alpha_{C'}(g) = \times$. In C' sei g_1 der erste und g_2 der zweite Vorgänger von f_2, f_1 der erste und f_2 der zweite Vorgänger von f_3 und f_3 und g_2 die Vorgänger von g. Alle anderen Gatter werden inklusive ihrer Vorgänger unverändert aus C nach C' übernommen.



Induktiv zeigen wir: $I_C(g) = I_{C'}(g)$ für alle Gatter g aus C.

IA: Sei g ein Eingabegatter in C. Dann gilt $I_C(g) = \{\alpha_C(g)\} = \{\alpha_{C'}(g)\} = I_{C'}(g)$.

IS: Sei g ein $\cap$-Gatter aus C mit Vorgängern g_1 und g_2. Es gilt $\#I_C(g_1) \leq 1$ und $\#I_C(g_2) \leq 1$. Falls $I_C(g_1) = \emptyset$ oder $I_C(g_1) = \emptyset$, so ist $I_{C'}(f_2) = I_{C'}(f_3) = I_{C'}(g) = \emptyset = I_C(g)$.

Andernfalls gibt es $x_1, x_2 \in \mathbb{N}$ mit $I_C(g_1) = \{x_1\}$ und $I_C(g_2) = \{x_2\}$. Nach Induktionsvoraussetzung sind dann auch $I_{C'}(g_1) = \{x_1\}$ und $I_{C'}(g_2) = \{x_2\}$. Es gilt:

$$
\begin{aligned}
I_{C'}(f_2) &= \begin{cases} \{x_1/x_2\} & \text{falls } x_2 | x_1 \\ \emptyset & \textit{sonst} \end{cases} \\
I_{C'}(f_3) &= \begin{cases} \{1\} & \text{falls } x_2 | x_1 \text{ und } x_1/x_2 = 1 \\ \emptyset & \textit{sonst} \end{cases} \\
&= \begin{cases} \{1\} & \text{falls } x_1 = x_2 \\ \emptyset & \textit{sonst} \end{cases} \\
I_{C'}(g) &= \begin{cases} \{x_2\} & \text{falls } x_1 = x_2 \\ \emptyset & \textit{sonst} \end{cases} \\
&= I_C(g)
\end{aligned}
$$

Sei g ein Gatter aus C mit $\alpha(g) \neq \cap$. Dann ist $\alpha_C(g) = \alpha_{C'}(g)$ und die Vorgänger von g wurden von C nach C' übernommen. Also ist $I_C(g) = I_{C'}(g)$.

Sei g_C das Ausgabegatter von C und C'. Dann gilt $I(C) = I_C(g_C) = I_{C'}(g_C) = I(C')$. C' kann in logarithmischem Raum aus C konstruiert werden. Daher gilt:

$$MC(\{\cap, \times, /\} \cup \mathcal{O}) \leq_m^{\log} MC(\{\times, /\} \cup \mathcal{O})$$

Die andere Richtung folgt aus Lemma 2.3. □

Folgerung 4.5 *$MC(+, \times, /)$ ist $\leq_m^{\log}$-hart für $\mathcal{PIT}$.*

4.3 Schaltkreise nur mit Division

In diesem Abschnitt zeigen wir, dass $MC(/)$ NL-hart ist. Dafür reduzieren wir das NL-vollständige Graph-Erreichbarkeitsproblem auf das Komplement von $MC(/)$.

Definition

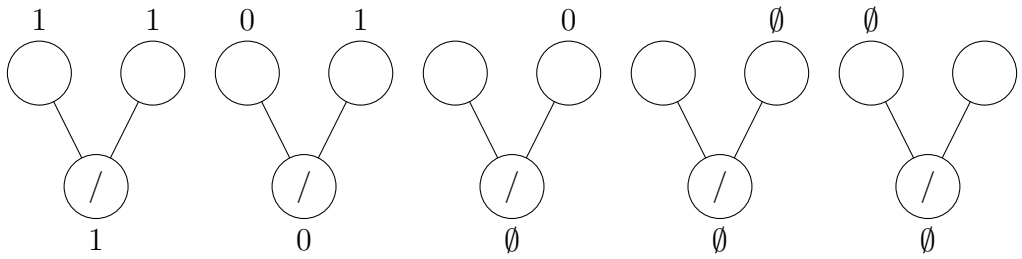
$$\text{GAP} =_{def} \{(G,s,t) \mid G=(V,E) \textit{ ist ein gerichteter, azyklischer Graph,} \\ s,t \in V \textit{ und in } G \textit{ ist } t \textit{ von } s \textit{ aus erreichbar}\}$$

ist das Graph-Erreichbarkeitsproblem (Graph Accessibility Problem).

Satz 4.6
$MC(/)$ *ist* $\leq_m^{\log}$*-hart für* NL.

Beweis. Wir wollen GAP auf $\overline{MC(/)}$ reduzieren. Sei $G=(V,E)$ ein gerichteter azyklischer Graph mit $s,t \in V$. Wir können G in logarithmischem Raum so umbauen, dass s eine Quelle ist, t eine Senke ist und jeder Knoten $v \in V$ Eingangsgrad 0 oder 2 hat. Wir gehen außerdem von $s \neq t$ aus. Für jeden Knoten v mit $v \neq s$ und Eingangsgrad 0 sei $\alpha(v)=1$ und für alle Knoten w mit Eingangsgrad 2 sei $\alpha(w)=/$. Sei außerdem $\alpha(s)=0$. Dann ist $C=(G,E,t,\alpha)$ ein $\{/\}$-Schaltkreis. Die Reihenfolge der Vorgänger der Knoten mit Eingangsgrad 2 kann zufällig gewählt werden. Induktiv kann man für alle Knoten $v \in V$ zeigen:

1. $I(v) \in \{\{0\},\{1\},\emptyset\}$
2. Ein Knoten w ist genau dann von s aus erreichbar, wenn $I(w) \in \{\emptyset,\{0\}\}$ ist.



Dann gilt:

$$\begin{aligned}(G,s,t) \in \text{GAP} &\Leftrightarrow \text{Es gibt einen s-t-Pfad in G.} \\ &\Leftrightarrow I(t) \in \{\emptyset,\{0\}\} \\ &\Leftrightarrow 1 \notin I(t) = I(C) \\ &\Leftrightarrow (C,1) \notin MC(/) \\ &\Leftrightarrow (C,1) \in \overline{MC(/)}\end{aligned}$$

GAP ist $\leq_m^{\log}$-vollständig für NL. Wegen NL = coNL ist damit $MC(/)$ $\leq_m^{\log}$-hart für NL. □

4.4 Schaltkreise mit Komplement und Division

Wir werden nun zeigen, dass $MC(^{-}, /)$ P-hart ist. Dafür verwenden wir das Schaltkreisauswertungsproblem, welches P-vollständig ist.

Definition

$$\text{CVP} =_{def} \{(C, x_1, \ldots, x_n) \mid C \textit{ ist ein Schaltkreis mit } \{\vee, \wedge, \neg\}\textit{-Gattern, } n \textit{ Eingängen und einem Ausgang, sodass bei Belegung der Eingänge mit } x_1, \ldots, x_n \in \{0,1\} \textit{ am Ausgang der Wert } 1 \textit{ entsteht}\}$$

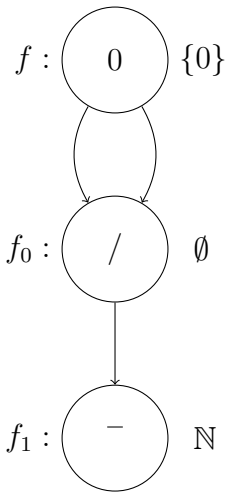
ist das Schaltkreisauswertungsproblem (Circuit Value Problem).

Satz 4.7
$MC(^{-}, /)$ *ist* $\leq_m^{\log}$*-hart für* P.

Beweis. Sei ein $\{\vee, \wedge, \neg\}$-Schaltkreis C mit einem Ausgangsgatter g_C, Eingabegattern $g_1, \ldots, g_n$ und Eingaben $x_1, \ldots, x_n \in \{0,1\}$ gegeben. Wir gehen davon aus, dass C keine $\vee$-Gatter enthält. (Diese können andernfalls nach den de-morganschen Regeln durch $\wedge$- und $\neg$-Gatter nachgebaut werden.)

Wir konstruieren einen algebraischen $\{^{-}, /\}$-Schaltkreis C' aus C. Dabei wollen wir die Werte 0 und 1 durch die leere Menge und ihr Komplement (die Menge der natürlichen Zahlen) darstellen.

Seien f, f_0 und f_1 Gatter, die nicht in C vorkommen. Sei $\alpha(f) = 0$, $\alpha(f_0) = /$ und $\alpha(f_1) = {}^{-}$. Sei f der erste und der zweite Vorgänger von f_0 und sei f_0 der Vorgänger von f_1. Dann gilt $I(f_0) = \{0\}/\{0\} = \emptyset$ und $I(f_1) = \overline{\emptyset} = \mathbb{N}$.



Wir kopieren alle Gatter außer die Eingabegatter aus C nach C'. Dabei ersetzen wir $\wedge$-Gatter durch /-Gatter und $\neg$-Gatter durch $^{-}$-Gatter. (Die Reihenfolge der Vorgänger der /-Gatter darf zufällig gewählt werden.) Wenn ein Gatter g in C ein Eingabegatter g_i als Vorgänger hatte, so verbinden wir g an dessen Stelle mit f_{x_i}. Mit den natürlichen Zahlen als Grundmenge gelten die Gleichungen:

1. $\emptyset/\emptyset = \emptyset/\mathbb{N} = \mathbb{N}/\emptyset = \emptyset$ und $\mathbb{N}/\mathbb{N} = \mathbb{N}$
2. $\overline{\emptyset} = \mathbb{N}$ und $\overline{\mathbb{N}} = \emptyset$

Dies entspricht den folgenden Gleichungen für die Wahrheitswerte 0 und 1:

1. $0 \wedge 0 = 0 \wedge 1 = 1 \wedge 0 = 0$ und $1 \wedge 1 = 1$
2. $\neg 0 = 1$ und $\neg 1 = 0$

Damit lässt sich per Induktion zeigen, dass für alle Gatter g aus C' mit $g \notin \{f, f_0, f_1\}$ gilt:

(i) $I(g) \in \{\emptyset, \mathbb{N}\}$

(ii) $I(g) = \mathbb{N}$ genau dann, wenn in C bei Belegung der Eingänge mit den Werten $x_1, \ldots, x_n$ am Gatter g der Wert 1 steht.

Also gilt:

$$\begin{aligned}(C, x_1, \ldots, x_n) \in \text{CVP} &\Leftrightarrow \text{In } C \text{ steht an } g_C \text{ bei Eingaben } x_1, \ldots, x_n \text{ eine 1.} \\ &\Leftrightarrow I(g_C) = \mathbb{N} \text{ (wegen (ii))} \\ &\Leftrightarrow 1 \in I(g_C) \text{ (wegen (i))} \\ &\Leftrightarrow (C', 1) \in MC(^-, /)\end{aligned}$$

C' kann in logarithmischen Raum aus $(C, x_1, \ldots, x_n)$ konstruiert werden. Daher ist CVP $\leq_m^{\log} MC(^-, /)$. □

4.5 Schaltkreise mit Multiplikation und Division

Satz 4.8
$MC(\times, /)$ ist $\leq_m^{\log}$-hart für $\text{C}_=\text{L}$.

Beweis. Sei $A \in \text{C}_=\text{L}$ eine Sprache über den Alphabet Σ. Das heißt es existieren Funktionen $f_0, f_1 \in \#\text{L}$ mit $(x \in A \Leftrightarrow f_0(x) = f_1(x))$. Für $i \in \{0, 1\}$ sei M_i die nichtdeterministische Turingmaschine, die in logarithmischem Raum arbeitet und für alle $x \in \Sigma^*$ genau $f_i(x)$ akzeptierende Pfade hat. Wir betrachten den Berechnungsbaum von $M_i(x)$. Dieser hat $f_i(x)$ akzeptierende Pfade. Sei C_i^x der $\{\times\}$-Schaltkreis, der aus dem Berechnungsbaum entsteht, wenn wir an das Ende jedes Pfades ein Eingabegatter setzen und die Wurzel als Ausgabegatter verwenden. Für alle Eingabegatter g sei $\alpha(g) = 2$, falls der Pfad akzeptiert und $\alpha(g) = 1$ sonst. Für alle anderen Gatter g' in C_i^x sei $\alpha(g') = \times$. Dann gilt $I(C_i^x) = \{2^{f_i(x)}\}$. Sei C^x der Schaltkreis, der entsteht, wenn wir die Schaltkreise C_0^x und C_1^x kombinieren und um ein neues Ausgabegatter g_{C^x} ergänzen. Die Vorgänger von g_{C^x} seien die Ausgabegatter von C_0^x und C_1^x. g_{C^x} sei ein /-Gatter. Also:

$$\begin{aligned}x \in A &\Leftrightarrow f_0(x) = f_1(x) \\ &\Leftrightarrow I(C_0^x) = I(C_1^x) \\ &\Leftrightarrow 1 \in I(C_0^x)/I(C_1^x) \\ &\Leftrightarrow 1 \in I(C^x) \\ &\Leftrightarrow (C^x, 1) \in MC(\times, /)\end{aligned}$$

C^x kann in logarithmischem Raum aus x konstruiert werden. Damit ist die Behauptung bewiesen. □

Satz 4.9
$MC(\times, /)$ ist $\leq_m^{\log}$-hart für PL.

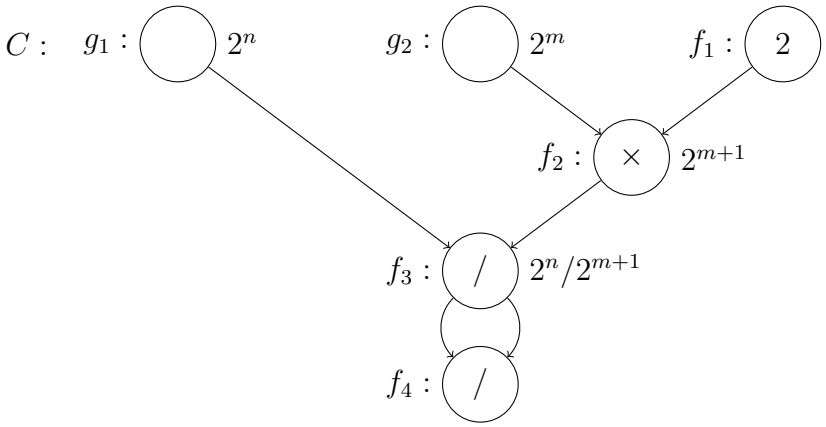
Beweis. Sei $A \in$ PL. Das heißt es existiert eine probabilistische Turingmaschine M, die A in logarithmischem Raum erkennt. Es gilt also für alle x:

$$x \in A \Leftrightarrow prob_M(x) > \frac{1}{2}$$

O.B.d.A. haben alle Pfade von M die gleiche Länge. Wir konstruieren nun zwei $\{\times, /\}$-Schaltkreise C_1^x und C_2^x aus dem Berechnungsbaum von $M(x)$ für eine Eingabe x. Wir ersetzen dabei jede Verzweigung des Berechnungsbaums durch ein $\times$-Gatter und jedes Pfadende durch ein Eingabegatter. In C_1^x wird jeder akzeptierende Pfad durch eine 2 am Eingabegatter am Pfadende dargestellt und jeder ablehnende Pfad durch eine 1. In C_2^x beschriften wir die Eingabegatter andersherum. Sei n die Anzahl der akzeptierenden Pfade des Berechnungsbaums und m die Anzahl der ablehnenden Pfade. Dann ist $I(C_1^x) = 2^n$ und $I(C_2^x) = 2^m$. Es gilt:

$$\begin{aligned} x \in A &\Leftrightarrow n > m \\ &\Leftrightarrow n \geq m + 1 \\ &\Leftrightarrow 2^n \geq 2^{m+1} \\ &\Leftrightarrow 2^n / 2^{m+1} \in \mathbb{N} \end{aligned}$$

Die letzte Bedingung können wir durch das Membership-Problem eines $\{\times, /\}$-Schaltkreises C^x darstellen. Wir übernehmen alle Gatter und deren Beschriftungen und Verbindungen aus C_1^x und C_2^x nach C^x. Seien g_1 und g_2 die Ausgabegatter von C_1^x bzw. C_2^x. Wir fügen vier neue Gatter $f_1, \ldots, f_4$ zu C hinzu. Sei $\alpha(f_1) = 2, \alpha(f_2) = \times$ und $\alpha(f_3) = \alpha(f_4) = /$. Seien g_2 und f_1 die Vorgänger von f_2, g_1 der erste und f_2 der zweite Vorgänger von f_3 und f_3 der erste und zweite Vorgänger von f_4. Sei f_4 das Ausgabegatter von C^x.



Dann gilt:

$$\begin{aligned} x \in A &\Leftrightarrow 2^n/2^{m+1} \in \mathbb{N} \\ &\Leftrightarrow 1 \in I(C^x) \end{aligned}$$

C^x kann in logarithmischem Raum konstruiert werden. Daher ist $A \leq_m^{\log} MC(\times, /)$. □

4.6 Schaltkreise mit Vereinigung und Division

Wir wollen zeigen, dass $MC(\cup, /)$ NP-hart ist, indem wir $EXACT - COVER$ darauf reduzieren.

Definition
Sei X eine Menge und S eine Menge von nichtleeren Teilmengen von X. Eine ***exakte Überdeckung*** *von X ist eine Teilmenge $U \subseteq S$, sodass die Elemente aus U paarweise disjunkt sind und $\bigcup_{A \in U} A = X$. Das heißt in einer exakten Überdeckung ist jedes Element von X in genau einem Element von U enthalten. $EXACT - COVER$ ist das Entscheidungsproblem, ob gegeben S und X eine solche exakte Überdeckung existiert, also*

$$EXACT - COVER =_{def} \{(S, X) \mid \exists\, U \subseteq S\text{: } U \text{ ist eine exakte Überdeckung von } X\}$$

Wir werden das folgende Lemma benötigen:

Lemma 4.10 *Sei p_n die n-te Primzahl für alle $n \in \mathbb{N}$. Für $n \geq 27$ lässt sich p_n mit $2\log(n) + 1$ Bits darstellen.*

Beweis. Wir benutzen folgende Aussage: Für $n \geq 4$ gibt es mindestens $\frac{n}{4\log_e(n)}$ Primzahlen $\leq n$.

Das heißt, für $n \geq 2$ gibt es mindestens $\frac{n^2}{8\log_e(n)}$ Primzahlen $\leq n^2$. Die Binärdarstellung der Zahl n^2 hat $2\log(n) + 1$ Bits. Für $n \geq 27$ ist $\frac{n^2}{8\log_e(n)} \geq n$. Das heißt, für $n \geq 27$ gibt es mindestens n Primzahlen, die sich in $2\log(n) + 1$ Bits darstellen lassen. Insbesondere kann p_n mit $2\log(n) + 1$ Bits dargestellt werden. □

Satz 4.11
$MC(\cup, /)$ ist $\leq_m^{\log}$-hart für NP.

Beweis. Wir zeigen den Satz, indem wir $EXACT - COVER$ auf $MC(\cup, /)$ reduzieren. Sei $X = \{a_1, \ldots, a_m\}$ eine endliche Menge und $S \subseteq \mathcal{P}(X)$ mit $\emptyset \notin S$. Sei $p : X \to \mathbb{P}$ die Abbildung die a_i auf die i-te Primzahl p_i abbildet. Gemäß Lemma 4.10 ist $p \in$ FL. Sei

$$f : \begin{array}{ll} \mathcal{P}(X) & \to \mathbb{N} \\ A & \mapsto \Pi_{a \in A} p(a) \end{array}$$

Immerman und Landau zeigen in Theorem 2.2 aus *The Complexity of Iterated Multiplication* [IL95], dass die Berechnung von Produkten, wie dem aus der Abbildung f, in

logarithmischem Raum möglich ist. Es ist bekannt, dass die Klasse FL unter Komposition abgeschlossen ist. Daher ist $f \in \text{FL}$.

Da p injektiv ist und die Primfaktorzerlegung eindeutig ist, ist f ebenfalls injektiv. Es gilt für alle $A, B \in S$:

$$A \subseteq B \Leftrightarrow f(A) | f(B) \Leftrightarrow \frac{f(B)}{f(A)} \in \mathbb{N}$$

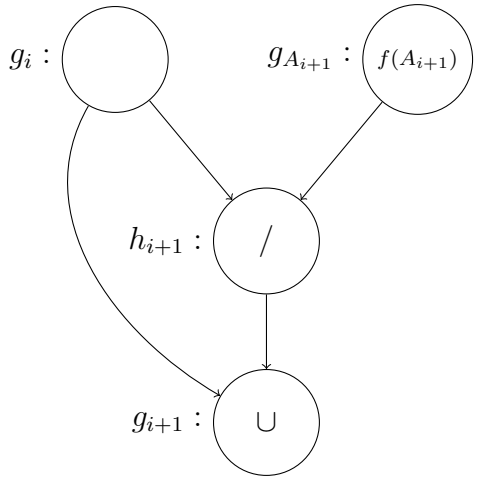
und falls $A \subseteq B$ so ist $\frac{f(B)}{f(A)} = f(B \setminus A)$. Diese Aussage nennen wir (*). Sei $S = \{A_1, \ldots, A_k\}$ für ein geeignetes k. Wir wollen einen $\{\cup, /\}$-Schaltkreis C konstruieren, sodass

$$(S, X) \in EXACT - COVER \Leftrightarrow 1 \in I(C)$$

Für jedes $A_i \in S$ sei g_{A_i} ein Eingabegatter mit $I(g_{A_i}) = \{f(A_i)\}$. Sei außerdem g_0 ein Eingabegatter mit $I(g_0) = \{f(X)\}$. Wir wollen nun für alle $1 \leq i \leq k$ ein Gatter g_i mit

$$I(g_i) = \left\{ f\left(X \setminus \bigcup_{j \in I} A_j\right) \;\middle|\; I \subseteq \{1, \ldots, i\} \text{ und die } A_j \text{ mit } j \in I \text{ sind paarweise disjunkt} \right\}$$

konstruieren. Sei g_i für ein $0 \leq i < k$ schon konstruiert. Sei h_{i+1} ein neues Gatter mit erstem Vorgänger g_i, zweitem Vorgänger $g_{A_{i+1}}$ und $\alpha(h_{i+1}) = /$. Sei g_{i+1} ein neues Gatter mit Vorgängern g_i und h_{i+1} und $\alpha(g_{i+1}) = \cup$.



Dann ist

$$\begin{aligned}
I(h_{i+1}) &= \left\{ \frac{x}{f(A_{i+1})} \;\middle|\; x \in I(g_i) \text{ und } f(A_{i+1}) \text{ teilt } x \right\} \\
&= \left\{ \frac{f\left(X \setminus \bigcup_{j \in I} A_j\right)}{f(A_{i+1})} \;\middle|\; \begin{array}{l} I \subseteq \{1, \ldots, i\}, \text{ die } A_j \text{ mit } j \in I \text{ sind paarweise} \\ \text{disjunkt und } f(A_{i+1}) \text{ teilt } f\left(X \setminus \bigcup_{j \in I} A_j\right) \end{array} \right\} \\
&\overset{(*)}{=} \left\{ f\left(X \setminus \bigcup_{j \in I} A_j\right) \;\middle|\; \begin{array}{l} \{i+1\} \subseteq I \subseteq \{1, \ldots, i+1\} \text{ und die } A_j \\ \text{mit } j \in I \text{ sind paarweise disjunkt} \end{array} \right\}
\end{aligned}$$

und $I(g_{i+1})$ ist wie gewünscht. Sei g_k das Ausgabegatter von C.
Dann gilt:

1. Falls die Elemente von $U \subseteq S$ paarweise disjunkt sind, so ist $f(X \setminus \bigcup_{A \in U} A) \in I(C)$.
2. Für alle $z \in I(C)$ gibt es ein $U \subseteq S$, sodass die Elemente von U paarweise disjunkt sind und $z = f(X \setminus \bigcup_{A \in U} A)$.
3. $f(X \setminus \bigcup_{A \in U} A) = 1$ genau dann, wenn $\bigcup_{A \in U} A = X$.

Also gibt es genau dann eine exakte Überdeckung $U \subseteq S$ von X, wenn $1 \in I(C)$. C kann in logarithmischem Raum konstruiert werden. Also ist $EXACT - COVER \leq_m^{\log} MC(\cup, /)$. $EXACT - COVER$ ist NP-vollständig. Damit ist $MC(\cup, /) \leq_m^{\log}$-hart für NP. □

5 Übersicht

Die nachfolgende Tabelle stellt die Ergebnisse dieser Arbeit dar. Für $MC(\mathcal{O})$ wird jeweils als untere Schranke eine Klasse $\mathcal{C}_1$ angegeben, sodass das Problem $\mathcal{C}_1$-hart ist. Eine obere Schranke im Sinne der Tabelle ist eine Klasse $\mathcal{C}_2$, sodass $MC(\mathcal{O}) \in \mathcal{C}_2$. Wenn eine untere Schranke in der Tabelle mit einer externen Quelle belegt wurde, so handelt es sich um eine Quelle für die $\mathcal{C}_1$-Härte von $MC(\mathcal{O}')$ mit $\mathcal{O}' \subseteq \mathcal{O} \setminus \{/\}$. Daraus folgt die $\mathcal{C}_1$-Härte von $MC(\mathcal{O})$ mit Lemma 2.3. In den Zeilen 2 und 14 wird sich dabei jeweils auf Korollar 43 aus [Bar+17] bezogen. Die von McKenzie und Wagner übernommenen Schranken können der Tabelle am Ende von [MW07] entnommen werden.

	$\mathcal{O}$	Untere Schranke	Quelle	Obere Schranke	Quelle
1	$\cup, \cap, ^-, +, \times, /$	NEXP	[MW07]	?	
2	$^-, +, \times, /$	PSPACE	[Bar+17]	?	
3	$\cup, \cap, +, \times, /$	NEXP	[MW07]	NEXP	3.9
4	$\cup, +, \times, /$	PSPACE	[MW07]	NEXP	3.9
5	$\cap, +, \times, /$	$\mathcal{PIT}$	4.3	E	3.8
6	$+, \times, /$	$\mathcal{PIT}$	4.5	E	3.8
7	$\cup, \cap, ^-, +, /$	PSPACE	[MW07]	PSPACE	3.3
8	$^-, +, /$	PSPACE	4.2	PSPACE	3.3
9	$\cup, \cap, +, /$	PSPACE	[MW07]	PSPACE	3.3
10	$\cup, +, /$	NP	[MW07]	PSPACE	3.3
11	$\cap, +, /$	$C_=L$	[MW07]	P	3.8
12	$+, /$	$C_=L$	[MW07]	P	3.8
13	$\cup, \cap, ^-, \times, /$	PSPACE	[MW07]	PSPACE	3.17
14	$^-, \times, /$	PSPACE	[Bar+17]	PSPACE	3.17
15	$\cup, \cap, \times, /$	PSPACE	[MW07]	PSPACE	3.17
16	$\cup, \times, /$	NP	[MW07]	PSPACE	3.17
17	$\cap, \times, /$	PL	4.9	P	3.14
18	$\times, /$	PL	4.9	P	3.14
19	$\cup, \cap, ^-, /$	NP	4.11	PSPACE	3.3
20	$^-, /$	P	4.7	PSPACE	3.3
21	$\cup, \cap, /$	NP	4.11	PSPACE	3.3
22	$\cup, /$	NP	4.11	PSPACE	3.3
23	$\cap, /$	NL	[MW07]	P	3.8
24	$/$	NL	4.6	P	3.8

6 Fazit

Wir haben bewiesen, dass $MC(\cup, \cap, +, \times, /)$ äquivalent zu $MC(\cup, \cap, +, \times)$ und NEXP-vollständig ist. In [MW07] haben McKenzie und Wagner für einige Membership-Probleme bewiesen, dass diese PSPACE-vollständig sind. Wir konnten die Probleme $MC(\mathcal{O} \cup \{+, /\})$ und $MC(\mathcal{O} \cup \{\times, /\})$ für alle $\mathcal{O} \in \{\{^-\}, \{\cup, ^-\}, \{\cap, ^-\}, \{\cup, \cap, ^-\}, \{\cup, \cap\}\}$ zu dieser Liste hinzufügen. Für diese $\mathcal{O}$ gilt jeweils $MC(\mathcal{O}) \equiv_m^{\log} MC(\mathcal{O} \cup \{/\})$. Das heißt, das Hinzufügen der Division hat die Schwierigkeit dieser Membership-Probleme nicht wesentlich verändert.

Die Frage, ob $MC(\cup, \cap, ^-, +, \times, /)$ und $MC(^-, +, \times, /)$ entscheidbar sind bleiben offen. Eine obere Schranke für $MC(\cup, \cap, ^-, +, \times, /)$ wäre zugleich auch eine obere Schranke für $MC(\cup, \cap, ^-, +, \times)$. Aus [MW07] wissen wir, dass ein Algorithmus für dieses Problem die Goldbachsche Vermutung beantworten würde. Daher wäre ein Algorithmus für dieses Problem ein äußerst bemerkenswertes Resultat. Wir konnten auch einige untere Schranken zeigen, die im Vergleich zu den Resultaten aus [MW07] interessant sind:

- $MC(\cup)$ ist NL-vollständig. $MC(\cup, /)$ ist NP-hart.
- $MC(\cup, \cap)$ und $MC(\cup, \cap, ^-)$ sind P-vollständig. $MC(\cup, \cap, /)$ und $MC(\cup, \cap, ^-, /)$ sind NP-hart.
- $MC(\times)$ ist NL-vollständig. $MC(\times, /)$ ist PL-hart.
- $MC(+, \times)$ ist P-hart. $MC(+, \times, /)$ ist $\mathcal{PIT}$-hart. (Das heißt, wir können PIT darauf reduzieren.)

Diese Resultate zeigen, dass falls die jeweiligen Klassen nicht zusammen fallen, diese Membership-Probleme durch das Hinzufügen der Division echt schwerer geworden sind.

Literatur

[Bar+17] Dominik Barth u. a. “Emptiness Problems for Integer Circuits”. In: *Electronic Colloquium on Computational Complexity* 12 (Jan. 2017). ISSN: 1433-8092.

[Gla+08] Christian Glaßer u. a. “Equivalence Problems for Circuits over Sets of Natural Numbers”. In: *Theory of Computing Systems* 46.1 (Sep. 2008), S. 80. ISSN: 1433-0490. DOI: 10.1007/s00224-008-9144-8. URL: https://doi.org/10.1007/s00224-008-9144-8.

[IL95] Neil Immerman und Susan Landau. “The Complexity of Iterated Multiplication”. In: *Information and Computation 116(1)* (1995), S. 103–116.

[MW07] Pierre McKenzie und Klaus W. Wagner. “The Complexity of Membership Problems for Circuits Over Sets of Natural Numbers”. In: *computational complexity* 16.3 (Okt. 2007), S. 211–244. ISSN: 1420-8954. DOI: 10.1007/s00037-007-0229-6. URL: https://doi.org/10.1007/s00037-007-0229-6.

[SM73] L. J. Stockmeyer und A. R. Meyer. “Word Problems Requiring Exponential Time(Preliminary Report)”. In: *Proceedings of the Fifth Annual ACM Symposium on Theory of Computing*. STOC ’73. Austin, Texas, USA: Association for Computing Machinery, 1973, S. 1–9. ISBN: 9781450374309. DOI: 10.1145/800125.804029. URL: https://doi.org/10.1145/800125.804029.