Entscheidbare Fälle des Postschen Korrespondenzproblems

zur Erlangung des akademischen Grades eines

Doktors der Naturwissenschaften

der Fakultät für Informatik
der Universität Fridericiana zu Karlsruhe (TH)

genehmigte

Dissertation

von

Mirko Rahn

aus Leipzig

Mündliche Prüfung: 12. Dezember 2008

Erster Gutachter: Prof. Roland Vollmar

Zweiter Gutachter: Prof. Peter Sanders

Präambel

Diese Arbeit entstand während meiner Beschäftigung als wissenschaftlicher Mitarbeiter am Lehrstuhl „Informatik für Ingenieure und Naturwissenschaftler“ von Prof. Roland Vollmar an der Fakultät für Informatik der Universität Karlsruhe (TH).

Ich bedanke mich bei Herrn Prof. Vollmar für die Freiheiten, die er mir gewährte und für die Unterstützung bei Reisen zu Konferenzen und bei meinem Besuch am mathematischen Institut von Prof. Juhani Karhumäki an der Universität Turku in Finnland.

Die Professoren Peter Sanders und Roland Vollmar haben das Referat übernommen. Vielen Dank dafür.

Meinen Kollegen Matthias Schulz und Dr. Thomas Worsch danke ich für das Lesen von Teilen dieser Arbeit und für Interesse an einigen meiner Resultate. Thomas Worsch hat außerdem seine Kenntnisse und Vorstellungen über Textsatz mit mit geteilt und diskutiert.

Zusammenfassung

Das Postsche Korrespondenzproblem fragt nach nichtleeren Elementen in der Gleichheitsmenge $\mathrm{Eq}(h,g) = \{u \mid h(u) = g(u)\}$ zweier Morphismen $h, g : A^* \to B^*$. Es handelt sich um ein unentscheidbares Problem, das in vielen Bereichen der Informatik als Referenzproblem bei Unentscheidbarkeitsbeweisen dient. Trotz der einfachen Struktur gibt es nur wenige Ansatzpunkte, um konkrete Instanzen zu lösen oder ihre Unlösbarkeit zu erkennen.

Wir untersuchen, wie Instanzen klassifiziert werden können. Wir stellen Basistechniken vor, angefangen bei Argumenten über Balancen oder inkompatible Morphismen bis zum Beschneiden von Suchgraphen. Wir verallgemeinern das Suchen nach einer Lösung und konstruieren einen Transducer, der die Koinzidenzmenge $\mathbb{C}(h,g) = \{(u,v) \mid h(u) = g(v)\}$ akzeptiert. Darauf aufbauend entwickeln wir die Theorie der Nachfolger neu und erstmals umfassend. Ein Nachfolger einer Instanz spiegelt dabei die Grundblöcke der Koinzidenzmenge wider und die Lösbarkeiten von Instanz und Nachfolger hängen eng zusammen. Wir führen einen neuen und klar strukturierten Beweis der Entscheidbarkeit binärer Instanzen. Unsere Darstellung bietet einen einheitlichen Rahmen, in dem alle bekannten Techniken gefasst werden können und der es darüber hinaus erlaubt, echt mehr Instanzen zu klassifizieren.

Es stellt sich heraus, dass sämtliche Methoden zur Identifikation unlösbarer Instanzen auf einem der beiden Prinzipien Balancieren und Suchen beruhen. Eine genaue Untersuchung des Prinzips Suchen gibt Einblicke in die selbstreferentielle Natur unentscheidbarer Probleme: Wenn wir unseren Standpunkt verändern um Vereinfachung zu erzielen, müssen wir schließlich feststellen, dass wir wieder dem Originalproblem gegenüber stehen.

Wir geben viele Beispielinstanzen an, die teilweise aus großen Kandidatenmengen unter erheblichem Aufwand ausgewählt wurden. Wir stellen dar, wie die entsprechenden Programme erstellt wurden, die die leistungsfähigsten weltweit sind. Praktisch alle bekannten schweren Instanzen wurden mit unseren Programmen gefunden. Auch parallele Verfahren wurden eingesetzt. Damit wurde es erstmals möglich, auf dem universitären Parallelrechner HP XC6000 Suchgraphen mit mehr als 10^{15} Knoten zu bearbeiten.

Inhalt

Einleitung

Die Frage jeder Einleitung lautet: Wo beginnen? Wie weit den Bogen spannen? Ist Aristoteles als Begründer der formalen Logik ein angemessener Anfangspunkt? Oder Wilhelm Leibniz, der die Philosophie zu einer Rechendisziplin machen wollte? Was ist mit David Hilbert, der ganz im Geiste von Leibniz eine algorithmisierte Mathematik anstrebte? Oder reicht es, in die 30er Jahre des letzten Jahrhunderts zurück zu blicken, zu Kurt Gödels Unvollständigkeitssatz, zum λ-Kalkül von Stephen Kleene, zur Maschine von Alan Turing, zu den Tag-Systemen von Emil Post? Zu all diesen Themen gibt es zahlreiche und ausführliche Literatur, die Rolle von Post wurde kürzlich sehr umfangreich von Liesbeth de Mol dargestellt. (de Mol [30])

Für diese Arbeit soll genügen, dass Emil Leon Post (1897–1954) sich, angeregt von Hilberts Programm, auch mit der vollständigen Beweisbarkeit logischer Systeme beschäftigte. Er zeigte in seiner Dissertation Konsistenz und Vollständigkeit der Aussagenlogik in der *Principia Mathematica* von Bertrand Russel und Alfred Whitehead, beschäftigte sich mit Beweistheorie und hatte bereits 1921 das Gödelsche Resultat, er entwickelte die Theorie rekursiver Funktionen mit, er entwarf 1936 eine Maschine, die die von Turing vorwegnahm, kurzum, er war ein innovativer Denker seiner Zeit.

Das abstrakte Beweismodell von Post bezeichnen wir heute als Ableitungssystem. Ein Beweis ist darin eine Folge von Ableitungen, beginnend bei einem Axiom. Eine Ableitung ist das Ersetzen eines Teilwortes durch ein anderes gemäß einer Regel aus einer vorgegebenen Menge von Regeln. Die allgemeinste von Post betrachtete Form ist die des Tag-Systems, bei dem in einem Ableitungsschritt die ersten $\nu > 0$ Symbole entfernt und ein vom ersten Symbol abhängiges Wort angehängt werden. Diese Systeme sind in der Lage, arithmetisierte Turingmaschinen zu simulieren. (Minsky [27])

Post gelangt schließlich zu kanonischen Tag-Systemen in Normalform, das sind Systeme mit Regeln der Form (l, r), die es erlauben, aus dem Wort lu das Wort ur abzuleiten. (Post [34]) Er zeigt, dass die Erreichbarkeit für solche Systeme unentscheidbar ist, dass es also keinen Algorithmus gibt, der feststellen kann, ob ein Wort aus einem Axiom in endlich vielen Schritten

ableitbar ist. Von dort ist es nur noch ein kleiner Schritt zum *Postschen Korrespondenzproblem*, das von Post definiert wird als die Frage, ob für eine endliche Menge $\{(g_1, g'_1), (g_2, g'_2), \ldots, (g_\mu, g'_\mu)\}$ von Paaren nichtleerer Wörter die Gleichung $g_{i_1} g_{i_2} \cdots g_{i_n} = g'_{i_1} g'_{i_2} \cdots g'_{i_n}$ mit $n > 0$ und $\{i_1, i_2, \ldots, i_n\} \subseteq \{1, 2, \ldots, \mu\}$ eine Lösung hat. (Post [33]) In dieser Form wird es oft noch heute präsentiert. Eine alternative Formulierung fragt, ob die Gleichheitsmenge $\mathrm{Eq}(h, g) = \{u \mid h(u) = g(u)\}$ zweier Morphismen $h, g : A^* \to B^*$ nichtleere Elemente enthält und diese Form wird die bestimmende in dieser Arbeit sein. Der Morphismus h beschreibt die ersten und der Morphismus g die zweiten Komponenten der Paare $\{(g_1, g'_1), (g_2, g'_2), \ldots, (g_\mu, g'_\mu)\}$. Die Wörter g_i und g'_i sind für $i \in \{1, \ldots, \mu\}$ in der Menge B^* enthalten und die Menge A vergibt Namen an die einzelnen Paare, in der Originalformulierung von Post gilt $A = \{1, \ldots, \mu\}$.

Die Einfachheit des Postschen Korrespondenzproblems erlaubt die Annäherung an die Grenze zwischen Entscheidbarkeit und Unentscheidbarkeit und damit an die Natur von Berechnung. (Margenstern [23]) Es handelt sich um ein generisches Problem, das seine Nützlichkeit in verschiedenen Bereichen bewiesen hat. Es lässt sich leicht auf andere Probleme reduzieren und ist besonders in der Theorie formaler Sprachen *das* Werkzeug, um die Unentscheidbarkeit bestimmter Probleme zu zeigen. Beispielhaft seien hier das Problem genannt, zu entscheiden, ob eine kontextfreie Grammatik mehrdeutig ist und das Problem, zu entscheiden, ob der Schnitt zweier kontextfreier Sprachen leer ist.

Das Postsche Korrespondenzproblem wird viel eingesetzt, aber es ist nicht sehr viel darüber bekannt: Von oben ist es mit 7 Paaren unentscheidbar via Unentscheidbarkeit des Wortproblems in Halbgruppen. (Matiyasevich, Sénizergues [24, 25]) Von unten ist es mit einem Paar trivial entscheidbar. Um die Entscheidbarkeit von Instanzen mit zwei Paaren zu zeigen, wurden recht komplizierte Techniken entwickelt: Algorithmen für periodische Instanzen und für markierte Instanzen, das sind Instanzen, bei denen die Bilder unterschiedlicher Symbole der Morphismen h und g mit unterschiedlichen Symbolen beginnen. Es handelt sich um eine echte Teilmenge der Menge der injektiven Morphismen. Der erste Beweis der Entscheidbarkeit von Instanzen mit zwei Paaren von Karhumäki et al. ist mehr als 20 Seiten lang und führt aufwändige Fallunterscheidungen durch, die die eigentlichen Prinzipien eher verdecken als offenlegen. (Ehrenfeucht, Karhumäki, Rozenberg [5]) Später hat Vesa Halava schrittweise für mehr Klarheit gesorgt und den Beweis sowohl für markierte Instanzen verallgemeinert als auch vereinfacht. (Halava et al. [8, 11, 12]) Die fortgeschrittenste Technik konstruiert Graphen von Nachfolgerinstanzen, das sind Instanzen, deren Paare den Grundblöcken entsprechen, die bei einer Zerlegung der Koinzidenzmenge $\mathbb{C}(h, g) = \{(u, v) \mid h(u) = g(v)\}$ auftreten.

Die Lösbarkeiten von Instanz und Nachfolger hängen eng zusammen und die Nachfolgertechnik erlaubt die Klassifizierung aller Instanzen mit zwei Paaren und von markierten Instanzen. Mit Hilfe einer verbesserten Nachfolgertechnik kann sogar eine echte Obermenge der markierten Instanzen entschieden werden, nämlich die Menge aller Instanzen mit markierten Nachfolgern. In dieser Menge sind auch Instanzen enthalten, deren Morphismen nicht injektiv sind. (Halava et al. [10])

Der Status von Instanzen mit 3, 4, 5 oder 6 Paaren ist ungeklärt. Mit 6 Regeln lassen sich Familien konstruieren, die lösbar sind, aber bei denen die Länge der kürzesten Lösung exponentiell wächst, mit 3 Regeln ist kubisches Wachstum bekannt. (Schmidt [39]) Klar ist, dass sich die Techniken, die für binäre Instanzen entwickelt wurden, nicht auf drei oder mehr Regeln übertragen lassen. Die meisten Forscher halten sich mit Vermutungen über die ungeklärten Klassen zurück, einzig Stephen Wolfram „strongly suspects“, dass das Problem bereits mit 3 Regeln unentscheidbar ist. (Wolfram [43])

Ziele

a

Diese Arbeit soll eine einheitliche und umfassende Darstellung der Möglichkeiten geben, Instanzen zu entscheiden. Es sollen möglichst wenige Basistechniken identifiziert und Zusammenhänge hergestellt werden. Alle bekannten Techniken sollen erfasst werden und deren Grenzen ausgelotet. Neue Techniken, Kombinationen und Erweiterungen sollen entwickelt und bewertet werden. Die Grenze zwischen Entscheidbarkeit und Unentscheidbarkeit kann dann hoffentlich genauer als bisher beschrieben werden.

Darüber hinaus sollen die gesammelten Erkenntnisse angewendet werden. Dazu sollen Programme entwickelt werden, die schnell große Mengen von Instanzen klassifizieren können. Als Beweis der Leistungsfähigkeit sollen neue Rekordinstanzen gefunden werden, dass sind einfache Instanzen mit besonders langen kürzesten Lösungen.

Dieses Dokument

b

Es gibt nur eine Art der Nummerierung, die für *alles* benutzt wird und schnelle Navigation erlauben soll: Jede Seite hat eine Seitennummer, die im Fuß angegeben wird, und pro Seite wird ein lokaler Zähler fortlaufend erhöht. Dieser lokale Zähler wird mit kleinen Buchstaben im Rand dargestellt. Referenzen bestehen aus der Seitennummer und dem Buchstaben der referenzierten Stelle. Mit 3b wird also auf die auf Seite 3 mit b markierte Stelle

verwiesen, diesen Abschnitt. Referenzen auf Gleichungen werden traditionell in Klammern angegeben, zum Beispiel steht die Gleichung (20b) auf Seite 20 an der Stelle, wo im Rand der Buchstabe b steht.

Der Schreibstil ist so angelegt, dass einerseits die notwendige Genauigkeit gegeben ist, andererseits aber nicht die totale Symbolwüste entsteht. So finden sich Beweise oft informell im Fließtext und die vielen Beispiele sollen die Aussagen eines Abschnitts verdeutlichen. Es gibt einige Beispielinstanzen, die sich durch das gesamte Dokument ziehen. Auf Seite 133 findet sich eine Übersicht, die für diese Instanzen die Vorkommen auflistet. Sämtliche Beispiele wurden sorgfältig ausgewählt, teilweise aus großen Mengen von Kandidaten und unter erheblichem Aufwand. Zur weiteren Erleichterung finden sich am Ende des Dokuments ein Verzeichnis verwendeter Symbole und ein Stichwortverzeichnis. Das Dokument wurde unter Annahme linearer Lesereihenfolge erstellt und teilt sich in die beiden Hauptteile Theorie und Praxis:

a **Theorie** Im theoretischen Teil werden zunächst in 7a die notwendigen Begriffe wie Automat oder Morphismus definiert. Wir verwenden weder ungewöhnliche noch komplizierte Konzepte, klären lediglich die Notation. Anschließend definieren wir in Abschnitt 12b das verallgemeinerte Postsche Korrespondenzproblem und geben Basisresultate und Basisbeispiele an.

Das Vereinfachen in 18c beschreibt triviale Verfahren, die Instanzen zum Beispiel wegen falscher Balancen oder inkompatibler Morphismen als unlösbar erkennen. Diese Verfahren sind trotz ihrer Einfachheit recht effizient und sind perfekt zur schnellen Filterung einfacher Instanzen geeignet. Sie reichen bis zur Entscheidbarkeit periodischer Instanzen, die in Essenz deshalb entscheidbar sind, weil es in w^* höchstens ein Wort der Länge k gibt.

Nachdem die besonders einfachen Instanzen behandelt wurden, widmen wir uns in 28a dem Suchen nach einer Lösung. Das Postsche Korrespondenzproblem ist semientscheidbar in dem Sinn, dass wir alle Lösungen einer Instanz aufzählen können und wir definieren einen natürlichen Suchgraphen. Wir stellen Methoden dar, diesen Suchgraphen zu beschneiden. Dabei werden Pfade sowohl anhand von Pfadbeschriftungen als auch anhand von Knoteninhalten abgeschnitten. Es handelt sich bereits um recht komplexe Verfahren, nach Vereinfachen und Suchen sind zum Beispiel in der Klasse $\mathcal{P}(3,3)$ der Instanzen mit höchstens 3 Regeln, deren längstes Bild höchstens die Länge 3 hat, von 253 976 Isomorphieklassen nur 18 nicht entschieden.

Haben wir bis zu diesem Punkt hauptsächlich Techniken präsentiert, die zumindest im Ansatz bereits an anderer Stelle zu finden sind, gehen wir in 45a deutlich über das Bekannte hinaus. Dieser Abschnitt enthält die Hauptre-

sultate dieser Arbeit. Wir betrachten eine naheliegende Verallgemeinerung der Suchrelation, die wir als allgemeine Suchrelation bezeichnen und von der verschiedene weitere Suchrelationen abgeleitet werden. In 54a haben wir es schließlich mit einer endlichen Relation zu tun, mit deren Hilfe wir die Koinzidenzmenge beschreiben können. Die Koinzidenzmenge $\mathbb{C}(h, g)$ zweier Morphismen $h, g : A^* \to B^*$ ist die Menge $\mathbb{C}(h, g) = \{(u, v) \mid h(u) = g(v)\}$. Das erlaubt die Formulierung von Satz 56a, in dessen Beweis ein Transducer konstruiert wird, der die Koinzidenzmenge akzeptiert. Für eine Reihe älterer Resultate über bedingte Koinzidenzmengen geben wir in 58a einfache Beweise innerhalb des neuen Rahmens.

Als Folgerung aus Satz 56a schließt sich in 63b eine neue und allgemeine Darstellung der Theorie der Nachfolger an. Erstmals wird der Begriff *Nachfolger* umfassend definiert und über die Zerlegung von Koinzidenzen anstelle von Lösungen motiviert. Wichtige Eigenschaften von Nachfolgern, die Entscheidbarkeit markierter Instanzen und als Folgerung die Entscheidbarkeit von binären Instanzen werden gezeigt. Unter dem Strich ergibt sich ein neuer Beweis der Entscheidbarkeit von verallgemeinerten Instanzen mit zwei Paaren und der Entscheidbarkeit markierter verallgemeinerter Instanzen, der erste mit klarer Struktur. Darüber hinaus lassen sich nicht nur alle bekannten Techniken in dem neuen Rahmen fassen, sondern die allgemeine Darstellung erlaubt die Klassifikation von Instanzen, die nicht mit einer der bekannten Techniken klassifiziert werden können.

Praxis Das prinzipielle Ziel hier lautet: Erzeuge schnelle und gute Programme. Der Testfall ist das Auffinden kleiner Instanzen mit möglichst langen kürzesten Lösungen. Dieses Feld wurde bereits von anderen Autoren beackert und verlangt sowohl sehr gute Filterung unlösbarer Instanzen als auch sehr schnelle Programme zum Finden von langen Lösungen. *a*

Wir stellen dar, wie unlösbare Instanzen möglichst schnell identifiziert werden können. Hier spielt unter anderem die Reihenfolge eine Rolle, in der die einzelnen Techniken zur Anwendung kommen. Werden Instanzen nicht als unlösbar erkannt, dann muss irgendwann die Suche nach einer Lösung erfolgen. Prinzipielle Überlegungen deuten auf eine iterative Tiefensuche hin, das heißt auf eine beschränkte Tiefensuche, bei der die maximale Suchtiefe schrittweise wächst.

Zunächst leiten wir in 112a ein sequentielles Programm ab, das automatisch individuell auf eine Instanz zugeschnitten wird. Es handelt sich um ein Programm, das unter Vermeidung jeder Indirektion und mit konstantem und kleinem Speicherbedarf gut auf aktueller Technik funktioniert. In 116b stellen wir kurz zwei Möglichkeiten der Parallelisierung vor, zufälliges Anfragen und

Aufteilen in Meister und Arbeiter. Beide Varianten haben Vor- und Nachteile und beide erreichen auf großen homogenen Parallelrechnern praktisch optimale Beschleunigung.

Schließlich belegen wir die Leistungsfähigkeit unserer Programme durch Angabe einiger neuer Rekordinstanzen, die teilweise um mehrere Größenordnungen komplizierter sind, als die bisherigen Rekordinstanzen. Insgesamt kann festgehalten werden, dass praktisch alle schwierigen Instanzen vom Autor gefunden wurden.

Unter der Adresse `ftp://ftp.ira.uka.de/pub/pcp` stehen Listen mit Rekordinstanzen und sämtliche Quellen bereit, die zur Erstellung dieser Listen und dieses Dokuments benutzt wurden.

a Das Verhältnis der Länge der Teile Theorie und Praxis gibt nicht deren Bedeutung wieder. Vielmehr ist der theoretische Teil mit Beispielen durchsetzt, die nur durch große praktische Anstrengungen gefunden werden konnten. Darüber hinaus haben immer wieder Programme geholfen, Instanzen zu finden, die bestimmte Vermutungen bestätigen oder widerlegen und nicht zuletzt sind Programme für aufwändige Rechnungen und für Visualisierungen nahezu unersetzlich.

b Abschließend noch eine Bemerkung zur Verwendung des Begriffs *entscheidbar*: Bekanntlich heißt eine Teilmenge $A \subseteq M$ einer Menge M entscheidbar, wenn ihre charakteristische Funktion $\chi_A : M \to \{0, 1\}$ berechenbar ist. Es gibt dann einen Algorithmus, der zu jedem Element aus M feststellt, ob es zu A gehört oder nicht. Insofern müsste der Titel dieser Arbeit eigentlich lauten „Entscheidbare *Mengen von Instanzen* des Postschen Korrespondenzproblems“. Wir wollen hier etwas lockerer mit diesem Begriff umgehen und sprechen auch von entscheidbaren Instanzen und entscheidbaren Fragen. Entscheidbare Instanzen sind Instanzen, deren Lösbarkeit oder Unlösbarkeit algorithmisch festgestellt werden kann. Eine Instanz nennen wir *entschieden* oder *klassifiziert*, wenn deren Lösbarkeit oder Unlösbarkeit festgestellt wurde. Eine Frage ist entscheidbar, wenn es einen Algorithmus gibt, der sie in endlicher Zeit beantwortet.

Theorie

Definitionen *a*

Wir schreiben kurz $a \not\equiv b$ statt $\neg(a \iff b)$, wenn also a und b nicht den gleichen Wahrheitswert haben. Häufig schreiben wir $a \implies b \implies c$ für gewisse a, b und c. Dabei ist die Implikation wie üblich rechtsassoziativ, $a \implies b \implies c$ bezeichnet also die Aussage $a \implies (b \implies c)$. Diese Aussage ist logisch äquivalent zu der Aussage $(a \wedge b) \implies c$, hat aber den Vorteil, dass die Prämissen eine Gewichtung haben.

Die Komposition $f \cdot g$ der Funktionen $f : A \to B$ und $g : B \to C$ erfolgt von links nach rechts, also $(f \cdot g)(x) = g(f(x))$.

Für Paare $p \in A \times B$ sind die *Projektionen* $\uparrow : A \times B \to A$ und $\downarrow : A \times B \to B$ gegeben durch $p = (p_\uparrow, p_\downarrow)$. Wir benutzen die Postfixnotation im unteren Index, $p_\uparrow$ bezeichnet also die erste Komponente eines Paares und $p_\downarrow$ die zweite. Die Symbole $\uparrow$ und $\downarrow$ werden auch in anderen Bedeutungen benutzt, anhand des Kontextes und anhand der Position der Symbole ist für jedes Auftreten die Bedeutung eindeutig. Die *Vertauschung* des Paares $p \in A \times B$ ist das Paar $\overline{p} = (p_\downarrow, p_\uparrow)$. Es gilt $\overline{\overline{p}} = p$. Wir schreiben machmal $\left[\begin{smallmatrix} x \\ y \end{smallmatrix}\right]$ statt (x, y).

Für eine Menge V von Vektoren aus einem Vektorraum ist $\operatorname{span} V$ der von V aufgespannte Raum, also die Menge aller Linearkombinationen von Vektoren aus V. Für zwei Vektoren $x = (x_0, \ldots, x_{n-1})$ und $y = (y_0 \ldots, y_{n-1})$ eines n-dimensionalen Vektorraums bezeichnet $\langle x|y\rangle$ das Standard-Skalarprodukt $\sum_{i=0}^{n-1} x_i y_i$ von x und y.

Relationen *b*

Eine *ternäre (gewichtete) Relation* R ist eine Teilmenge $R \subseteq M \times W \times M$. Wir schreiben genau dann $x \xrightarrow{a} y$, wenn $(x, a, y) \in \longrightarrow$ gilt und $\longrightarrow$ eine gewichtete Relation ist. Jede *binäre (ungewichtete) Relation* $B \subseteq M \times M$ kann als gewichtete Relation $R \subseteq M \times \{\emptyset\} \times M$ mit $(x, \emptyset, y) \in R \iff (x, y) \in B$ betrachtet werden. Wir unterscheiden nicht streng zwischen binären und

ternären Relationen. Operationen, die für ternäre Relationen definiert sind, werden ohne weitere Hervorhebung auch auf binäre Relationen angewendet.

Wenn $S \subseteq R$ eine Teilrelation einer gewichteten Relation R ist, dann sind die *Knoten von* S gegeben durch $\mathrm{N}(S) = \bigcup\{\{x, y\} \mid (x, a, y) \in S\}$.

Die *Hülle* $R^\star(L)$ *(von* L*) (über* R*)* für Teilmengen $L \subseteq M$ von M ist die kleinste Teilrelation von R, deren Knoten L enthalten und die transitiv abgeschlossen ist. Es gilt $R \cap (L \times W \times M) \subseteq R^\star(L)$ und $(x, a, y) \in R^\star(L) \implies (y, b, z) \in R \implies (y, b, z) \in R^\star(L)$.

Die Menge $\mathrm{id}(X) = \{(x, x) \mid x \in X\}$ ist die *Identität auf* X.

a

Monoide und Morphismen

Ein *Monoid* ist ein Tripel $(M, \circ, e)$, wobei M eine beliebige Menge ist, $\circ : M \times M \to M$ eine assoziative Operation auf M und $e \in M$ das *Einselement* von $\circ$ über M. Es gelten $u \circ (v \circ w) = (u \circ v) \circ w$ und $e \circ u = u \circ e = u$.

Die Operation $\circ$ wird meist durch den Kontext festgelegt. Wir schreiben dann kurz uv statt $u \circ v$. Ist auch das Einselement e durch den Kontext bestimmt, schreiben wir kurz M anstelle von $(M, \circ, e)$. Die Elemente der Menge $M \setminus \{e\}$ nennen wir häufig *echt*. Abkürzend schreiben wir M° statt $M \setminus \{e\}$.

Wir schreiben $v^{-1}u = w$ und $uw^{-1} = v$, falls $u = vw$ gilt.

Die Operation $\circ$ wird auf Teilmengen $H \subseteq M$ und $G \subseteq M$ mittels $HG = \{hg \mid (h, g) \in H \times G\}$ ausgedehnt. Wir schreiben kurz Mx statt $M\{x\}$ und xM statt $\{x\}M$.

Die *Potenzen von* H sind induktiv definiert durch $H^0 = \{e\}$ und $H^{k+1} = H^k H$. Die *(reflexive) Hülle* H^* *von* H ist die Menge $H^* = \bigcup_{i \geq 0} H^i$, die *positive Hülle* H^+ die Menge $H^+ = \bigcup_{i > 0} H^i$. Wir verwenden auch die Notationen $H^{\leq k}$ und $H^{>j}$ und bezeichnen damit die Mengen $H^{\leq k} = \bigcup_{0 \leq i \leq k} H^i$ und $H^{>j} = \bigcup_{i > j} H^i$.

Wenn $X^* = M$ für ein $X \subseteq M$ gilt, dann ist X eine *Basis* von M und X *erzeugt* M. Jedes echte Element von M hat dann mindestens eine Zerlegung in Elemente aus X, das heißt, wenn $m \in M$, dann gibt es mindestens eine Folge $(x_0, x_1, \ldots, x_{k-1})$ mit $\{x_0, x_1, \ldots, x_{k-1}\} \subseteq X$ und $m = x_0 x_1 \cdots x_{k-1}$. Unterscheiden sich alle solche Zerlegungen paarweise nur durch eine Permutation, dann wird M von X *frei kommutativ erzeugt*. Gibt es genau eine solche Zerlegung, dann wird M von X *frei erzeugt*. Wenn M von X frei (kommutativ) erzeugt wird, dann ist das Einselement nicht in X enthalten.

Für eine Teilmenge $A \subseteq M$ von M definieren wir die Abkürzungen $A^\uparrow = A \times \{e\}$, $A^\downarrow = \{e\} \times A$ und $A^\updownarrow = A^\uparrow \cup A^\downarrow$.

Für $v \in M$ sind die Mengen $\mathrm{pref}(v) = \{u \in M \mid v \in uM\}$, $\mathrm{suff}(v) = \{u \in M \mid v \in Mu\}$ und $\mathrm{inf}(v) = \{u \in M \mid v \in MuM\}$ die Mengen aller *Präfixe,*

Suffixe und Infixe von v. Wir schreiben genau dann $u \preceq_\mathrm{p} v$ wenn $u \in \mathrm{pref}(v)$ gilt und genau dann $u \preceq_\mathrm{s} v$, wenn $u \in \mathrm{suff}(v)$ gilt. Zwei Elemente u und v heißen *(präfix-)vergleichbar*, wenn $u \preceq_\mathrm{p} v$ oder $v \preceq_\mathrm{p} u$, also mindestens eines der Elemente ein Präfix des anderen ist. Wir schreiben dann $u \prec_\mathrm{p} v$. Analog schreiben wir $u \prec_\mathrm{s} v$, wenn mindestens ein Element ein Suffix des anderen ist. Solche Elemente heißen *suffixvergleichbar*.

Sind zwei Elemente u und v vergleichbar, dann gibt es ein x, so dass $ux = v$ oder $u = vx$ gilt. Wir nennen x den *Überhang von* (u, v).

Seien $(A, \circ_A, e_A)$ und $(B, \circ_B, e_B)$ zwei Monoide. Eine Funktion $m : A \to B$ heißt genau dann *(Monoid-) Morphismus (von A nach B)*, wenn $m(e_A) = e_B$ und für $\{x, y\} \subseteq A$ stets $m(x \circ_A y) = m(x) \circ_B m(y)$ gelten. Wenn M frei von X erzeugt wird, dann ist m eindeutig durch die Bilder für Elemente aus X festgelegt.

Ein Morphismus $m : A \to B$ heißt genau dann *periodisch*, wenn es ein $b \in B$ gibt, so dass $h(A) \subseteq b^*$, genau dann *löschend*, wenn $\{a \in A^\circ \mid m(a) = e_B\} \neq \emptyset$ und genau dann *markiert*, wenn m nicht löschend ist und $\{x, y\} \subseteq A \implies \mathrm{pref}(m(x)) \cap \mathrm{pref}(m(y)) \neq \{e_B\} \implies x = y$, das heißt, wenn Bilder unterschiedlicher Symbole mit unterschiedlichen Symbolen beginnen. Markierte Morphismen bilden eine echte Unterklasse der injektiven Morphismen.

Für Teilmengen $A' \subseteq A$ und $B' \subseteq B$ bezeichnen wir mit $m(A') \subseteq B$ die Menge $m(A') = \{m(x) \mid x \in A'\}$ und mit $m^{-1}(B') \subseteq A$ die Menge $m^{-1}(B') = \{x \mid m(x) \in B'\}$. Wir sprechen auch von der *Menge der Bilder von m über A'* und der *Menge der Urbilder von m über B'*.

Ein Monoid $(M, \circ, e)$ ist genau dann ein *Monoid mit Länge*, wenn es einen Morphismus $l : M \to \mathbb{N}$ zwischen $(M, \circ, e)$ und $(\mathbb{N}, +, 0)$ mit $l^{-1}(0) = \{e\}$ gibt. Wir schreiben $|u|$ statt $l(u)$.

Wenn $(M, \circ, e)$ ein Monoid mit Länge ist, dann ist die Menge $X = M^\circ \setminus (M^\circ M^\circ)$ die eindeutige und bezüglich Inklusion minimale Basis von M: Zunächst können wir aus X keine Elemente streichen, da in X kein Element aus anderen Elementen zusammengesetzt ist. Bleibt zu zeigen, dass X eine Basis von M ist. Die Inklusion $X^* \subseteq M$ gilt offensichtlich. Die Inklusion $M \subseteq X^*$ folgt per Induktion über die Länge eines Elements. Wenn nämlich $y \in M$ die Länge $|y| = n + 1$ hat, dann lässt sich y entweder als uv schreiben mit kürzeren u und v, die nach Induktionsvoraussetzung in X^* enthalten sind, oder y lässt sich nicht als uv schreiben und ist dann nach Definition in X enthalten. (Choffrut, Karhumäki [2])

Wenn $(M, \circ, e)$ ein Monoid ist, dann ist durch $(M \times M, \bullet, (e, e))$ mit $(x, y) \bullet (x', y') = (x \circ x', y \circ y')$ ebenfalls ein Monoid definiert, das *Produkt von M*. Wenn M ein Monoid mit Länge ist, dann ist das Produkt von M mit ebenfalls ein Monoid mit Länge, wir setzen einfach $|(x, y)| = |x| + |y|$.

Die Menge aller Suffixe von Bildern von m über einer Basis X von A wird mit S_m^X bezeichnet. Es gilt $\mathrm{S}_m^X = \bigcup_{x \in X} \mathrm{suff}(m(x))$. Wenn die Basis X durch den Kontext festgelegt ist, schreiben wir kurz S_m.

a

Alphabete, Wörter und Sprachen

Ein *Alphabet* ist eine Menge von unterscheidbaren *Symbolen*. Wir wollen uns hier zunächst weder auf endliche noch auf nichtleere Alphabete festlegen. Ein *Wort* ist die Konkatenation von Symbolen. Mit ε bezeichnen wir das *leere Wort*, also die leere Konkatenation. Die Menge aller Wörter über einem Alphabet A wird mit A^* bezeichnet. Die Konkatenation $\circ$ ist eine assoziative Operation auf der Menge aller Wörter. Das Tripel $(A^*, \circ, \varepsilon)$ ist ein Monoid mit Länge, das frei von A erzeugt wird. Das Produkt $A^* \times A^*$ von A^* wird von $A^{\updownarrow}$ frei kommutativ erzeugt, aber nicht frei. Es ist ein Monoid mit Länge und dem Einselement $(\varepsilon, \varepsilon)$.

Das *gespiegelte Wort u^{R} von u* ist induktiv eindeutig gegeben durch: $\varepsilon^{\mathrm{R}} = \varepsilon$ und $(uv)^{\mathrm{R}} = v^{\mathrm{R}}u^{\mathrm{R}}$. Gäbe es mehr als ein gespiegeltes Wort von u, dann gäbe es mehr als eine Zerlegung von u in Elemente aus A. Wir erweitern R auf das Produkt $A^* \times A^*$ mittels $(u, v)^{\mathrm{R}} = (u^{\mathrm{R}}, v^{\mathrm{R}})$. Die Eindeutigkeit überträgt sich auf das Produkt und es gilt $\overline{x^{\mathrm{R}}} = \overline{x}^{\mathrm{R}}$, die Operationen Vertauschen und Spiegeln kommutieren.

Sei $Z = (a_0, a_1, \dots)$ eine feste Aufzählung von A. Das *kommutative (oder Parikh-) Bild $\psi_A^Z(u)$ von u (über A) (bezüglich Z)* ist definiert als der Vektor $\psi_A^Z(u) = (|u|_{a_0}, |u|_{a_1}, \dots)$, wobei $|u|_{a_i}$ die Anzahl der Vorkommen des Symbols a_i in u bezeichnet. Formal dürfen A und/oder u auch unendlich groß sein, im Anwendungsfall werden kommutative Bilder entweder nur miteinander verglichen oder sowohl A als auch u sind endlich. Wenn die Aufzählung Z oder das Alphabet A aus dem Kontext hervorgehen, schreiben wir auch kurz $\psi_A(u)$ oder auch $\psi(u)$.

Eine Menge $L \subseteq A^*$ von Wörtern über A heißt *Sprache (über A)*. Für Sprachen $X \subseteq A^*$ und $Y \subseteq A^*$ sind die *links- und die rechtsseitigen Quotienten von X und Y* die Sprachen $X \lhd Y = \{z \mid \exists y \in Y : yz \in X\}$ und $X \rhd Y = \{z \mid \exists y \in Y : zy \in X\}$. Der linksseitige Quotient von X und Y entsteht durch Löschen aller Präfixe in Wörtern aus X, die in Y enthalten sind. Es verbleiben Suffixe von Wörtern aus X, die durch Voranstellen eines Wortes aus Y zu einem Wort aus X werden.

Schließlich definieren wir noch für Morphismen $m : A^* \to B^*$ den *gespiegelten Morphismus von m* durch $m^{\mathrm{R}}(u) = (m(u^{\mathrm{R}}))^{\mathrm{R}}$. Geneigte Leserinnen und Leser mögen überprüfen, dass es sich um einen Morphismus handelt.

Wenn $m : A^* \to A^*$ ein Morphismus ist und in der Folge $(w_0, w_1, \dots)$ mit $w_{i+1} = m(w_i)$ stets w_i Präfix von w_{i+1} ist, dann ist das unendliche Wort

$m^\omega(w_0) = \lim_{i\to\infty} w_i$ wohl definiert und heißt *Fixpunkt von m (über w_0)*. Ein *unendliches Wort* ist dabei eine Abbildung von $\mathbb{N}$ nach A.

Beispiel: (Prouhet 1851, Thue 1906, Morse 1921, Choffrut, Karhumäki [2]) *a*
Sei $A = \{0,1\}$ und $\tau : A^* \to A^*$ der durch $\{0 \mapsto 01, 1 \mapsto 10\}$ gegebene Morphismus. Dann gilt für die Folge $(w_0, w_1, \dots)$ mit $w_0 = 0$ und $w_{i+1} = \tau(w_i)$ stets $w_i \preceq_\mathrm{p} w_{i+1}$. Tatsächlich gilt $w_0 = 0 \preceq_\mathrm{p} 01 = w_1$ und induktiv folgt aus $w_{i+1} = w_i v_i$ auch $w_{i+1} \preceq_\mathrm{p} w_{i+1}\tau(v_i) = \tau(w_i v_i) = w_{i+2}$. Der Fixpunkt $\tau^\omega(0)$ ist das *Thue-Morse Wort* $01101001100101101001011001101001\cdots$, das keinen Faktor der Form $uvuvu$ mit $\{u, v\} \subset A^+$ enthält. ⌟

Automaten und Transducer *b*

Ein (M, X, e)-*Automat* T ist ein Tupel $T = (Q, X, S, F, \delta)$, mit einer endlichen Menge Q von *Zuständen*, einem Monoid M mit der Basis X und dem Einselement e, einer Menge $S \subseteq Q$ von *Startzuständen*, einer Menge $F \subseteq Q$ von *Finalzuständen* und einer Überführungsrelation $\delta \subseteq Q \times X \times Q$. Die Relation δ wird transitiv zu $\delta' \subseteq Q \times M \times Q$ erweitert, das heißt $\delta \cup (Q \times \{e\} \times Q) \subseteq \delta'$ und $\{(p,u,t),(t,v,q)\} \subseteq \delta' \implies (p, uv, q) \in \delta'$. Die *von T akzeptierte Sprache* ist die Menge $L(T) = \{x \in M \mid \exists (s,f) \in S \times F : (s,x,f) \in \delta'\}$.

In dieser Arbeit interessieren wir uns für zwei Klassen von Automaten: Für (A^*, A, ε)-Automaten, die genau die Klasse der *endlichen Automaten* bilden, und für $(A^* \times A^*, A^\updownarrow, (\varepsilon,\varepsilon))$-Automaten, die *Mehrbandautomaten* oder *Transducer* genannt werden. (Berstel [1]) Die von endlichen Automaten akzeptierten Sprachen sind genau die *regulären Sprachen* und die von Transducern akzeptierten Sprachen sind genau die *rationalen Relationen*. Für beide Klassen gilt das

Lemma. (Pumplemma) *Sei $T = (Q, X, S, F, \delta)$ ein (A^*, A, ε)- oder ein $(A^* \times A^*, A^\updownarrow, (\varepsilon,\varepsilon))$-Automat. Dann gibt es ein $n \in \mathbb{N}$, so dass sich jedes $z \in L(T)$ mit $|z| \geq n$ in der Form $z = uvw$ schreiben lässt, wobei $|uv| \leq n$, $|v| > 0$ und $\{uv^i w \mid i \in \mathbb{N}\} \subseteq L(T)$.* *c*

Beweis: Sei $n = |Q|$ und $z \in L(T)$ mit $|z| = k \geq n$. Da $z \in L(T)$, gibt es einen Pfad in T von einem Startzustand s zu einem Endzustand f, der mit einer Zerlegung $z = x_0 x_1 \cdots x_{k-1}$ von z in k Elemente aus X beschriftet ist. Dieser Pfad verläuft durch $k + 1 > n$ Zustände. Die ersten $n + 1$ dieser Zustände können nach Schubfachprinzip nicht alle voneinander verschieden sein. Sei p einer der ersten $n + 1$ Zustände, der mehr als einmal vorkommt, dann ist $\{(s,u,p),(p,v,p),(p,w,f)\} \subseteq \delta'$ wobei $|v| > 0$, $|uv| \leq n$ und $\{uv^i w \mid i \in \mathbb{N}\} \subseteq L(T)$. □ *d*

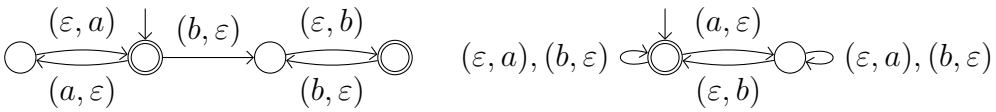
Aus dem Pumplemma folgt, dass es für endliche Automaten und für Transducer entscheidbar ist, ob die akzeptierte Sprache leer, endlich oder unendlich ist: Wenn n die Konstante aus dem Pumplemma ist, dann ist $L(T)$ genau dann leer, wenn kein Pfad der Länge höchstens n ein akzeptierender Pfad ist und genau dann unendlich, wenn es einen akzeptierenden Pfad mit einer Länge zwischen n und $2n$ gibt.

Die Klasse der endlichen Automaten ist abgeschlossen gegenüber Vereinigung, Konkatenation, Hülle, Substitution, Homomorphismus, inversem Homomorphismus, links- und rechtsseitigem Quotient, Spiegeln, Komplement und Schnitt. Die Klasse der Transducer ist abgeschlossen gegenüber Vereinigung und Konkatenation, jedoch nicht gegen Schnitt und damit auch nicht gegen Komplement.

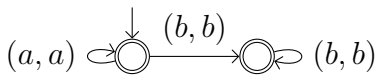
Die Sprache, die ein endlicher Automat oder ein Transducer akzeptiert, kann effektiv aufgezählt werden. Die Aufzählung kann so vorgenommen werden, dass kürzere Elemente vor längeren Elementen genannt werden. Das entspricht einer Breitensuche durch δ.

Bei der Darstellung von (M, X, e)-Automaten gehen wir meist liberal vor und lassen statt nur Beschriftungen aus X auch Beschriftungen aus M zu.

a **Beispiel:** Für die Transducer T_1 und T_2



gilt $L(T_1) = (a, a)^*(b, b)^*$ und $L(T_2) = \{(u, v) \mid |u|_a = |v|_b\}$. Es gibt keinen Transducer, der den Schnitt $L(T_1) \cap L(T_2) = \{(a, a)^n (b, b)^n \mid n \in \mathbb{N}\}$ akzeptiert. Eine liberalisierte Darstellung von T_1 ist



Eine Aufzählung der Elemente von $L(T_1)$ ist $((\varepsilon, \varepsilon), (b, b), (a, a), (bb, bb), (ab, ab), (aa, aa), (bbb, bbb), (abb, abb), (aab, aab), (aaa, aaa), \dots)$. Dabei spielt es keine Rolle, ob von der normalen oder der liberalisierten Darstellung ausgegangen wird. ⌟

b

Postsches Korrespondenzproblem

Ein Tupel $I = (A, B, s, (h, g), f)$ heißt genau dann *Instanz des verallgemeinerten Postschen Korrespondenzproblems (GPCP)*, wenn A und B Alphabete,

s und f Paare aus $B^{*\updownarrow}$ und $h : A^* \to B^*$ und $g : A^* \to B^*$ Morphismen sind. Die Paare s und f heißen auch *Startpaar* und *Endpaar*. Für $a \in A$ nennen wir das Paar $(h(a), g(a))$ auch *Regel*. Wir können davon ausgehen, dass jede Regel nur ein Mal vorkommt.

Die *Größe von* I ist $|I| = |A|$, die Anzahl der Symbole in A und die *Weite von* I ist $\mathrm{wd}(I) = \max\{|s|, |f|, \max\{\bigcup_{a\in A}\{|h(a)|, |g(a)|\}\}\}$, das längste an der Beschreibung von I beteiligte Wort.

Die Menge aller Instanzen des GPCP wird mit $\mathcal{G}$ bezeichnet. Wenn $s = f = (\varepsilon, \varepsilon)$, dann handelt es sich um das (herkömmliche) Postsche Korrespondenzproblem (PCP). Die Menge aller Instanzen des PCP wird mit $\mathcal{P}$ bezeichnet. Die Menge $\mathcal{G}(n)$ enthält alle Instanzen, die n Regeln haben und die Menge $\mathcal{G}(n, m)$ alle Instanzen, die n Regeln haben bei denen kein Bild von h oder g über A länger als m ist, das heißt $\mathcal{G}(n) = \{I \in \mathcal{G} \mid |I| \leq n\}$ und $\mathcal{G}(n, m) = \{I \in \mathcal{G}(n) \mid \mathrm{wd}(I) \leq m\}$. Für herkömmliche Instanzen definieren wir analog $\mathcal{P}(n) = \mathcal{P} \cap \mathcal{G}(n)$ und $\mathcal{P}(n, m) = \mathcal{P} \cap \mathcal{G}(n, m)$. Die Instanzen in $\mathcal{G}(1)$ heißen *unär* und die Instanzen in $\mathcal{G}(2)$ heißen *binär*.

Die für I zu entscheidende Frage lautet, ob es Wörter $u \in A^*$ gibt, für die $s_\uparrow h(u) f_\uparrow = s_\downarrow g(u) f_\downarrow$ gilt. Solche Wörter heißen *Lösung von* I und die Menge aller Lösungen ist die *Gleichheitsmenge von* I und wird mit $\mathrm{Eq}(I)$ bezeichnet. Auch wenn es sich bei $\mathrm{Eq}(I)$ nicht zwingend um ein Monoid handelt, schreiben wir abkürzend $\mathrm{Eq}^\circ(I)$ statt $\mathrm{Eq}(I) \setminus \{\varepsilon\}$. Gleichheitsmengen sind immer kontextsensitive Mengen: Eine Turingmaschine kann in $2(n + 2)\,\mathrm{wd}(I)$ Platz die h- und g-Bilder einer Eingabe der Länge n auf Gleichheit testen.

Herkömmliche Instanzen haben entweder nur die triviale oder unendlich viele Lösungen. Es gilt $I \in \mathcal{P} \implies \mathrm{Eq}(I) = \{\varepsilon\} \not\equiv |\,\mathrm{Eq}(I)| = \infty$. Die Länge der kürzesten echten Lösung bezeichnen wir mit $\mu(I)$, das heißt $\mu(I) = \min\{|u| \mid u \in \mathrm{Eq}^\circ(I)\}$, wobei $\min \emptyset$ per Definition gleich 0 sei.

Eine Teilmenge $\mathcal{I} \subseteq \mathcal{G}$ heißt genau dann *entscheidbar*, falls es einen Algorithmus gibt, der für jede der Instanzen aus $\mathcal{I}$ in endlicher Zeit feststellt, ob sie echte Lösungen hat.

Emil Post hat 1946 gezeigt:

Satz. (Post [33]) *$\mathcal{P}$ ist nicht entscheidbar.* *a*

Im Originalbeweis simuliert Post mit einer Instanz des PCP die Arbeit eines Tag-Systems. Dabei handelt es sich um einen Satz von Regeln (l, r), die nach dem Muster „Lösche links, konkateniere rechts“ verwendet werden, das heißt ur ist aus lu ableitbar. Die Unentscheidbarkeit folgt dann aus der Existenz universeller Tag-Systeme.

Heute finden wir meist einen Beweis, der die Berechnung einer Turingmaschine simuliert. Dazu erzeugen beide Morphismen ein Wort, das die Folge

von Konfigurationen kodiert, die bei einer bestimmten Berechnung der Turingmaschine auftreten kann. In 16d geben wir ein Beispiel einer solchen Konstruktion, die Details sind nicht kompliziert. (Hopcroft, Ullmann [16])

Es wird auch über *unendliche Lösungen* gesprochen, das sind unendliche Wörter $\omega = a_0 a_1 \cdots$ mit $a_i \in A$, so dass für jedes endliche Präfix $w \preceq_{\mathrm{p}} \omega$ von ω die Wörter $s_\uparrow h(w)$ und $s_\downarrow g(w)$ vergleichbar sind. Die Menge aller unendlichen Lösungen von I wird mit $\mathrm{Eq}^\omega(I)$ bezeichnet. Durch Voranstellen von ω wollen wir kennzeichnen, dass wir nach unendlichen Lösungen fragen. Es gilt:

a **Satz.** (Ruohonen [36]) *$\omega\,\mathcal{P}$ ist nicht entscheidbar.*

Dabei dürfen die Morphismen h und g sogar die Bifixeigenschaft haben, das heißt, kein Bild ist Präfix oder Suffix eines anderen Bildes. Keijo Ruohonen konstruiert eine universelle reversible Turingmaschine derart, dass zwei Morphismen mit Bifixeigenschaft gemeinsam deren Arbeit simulieren können. Die Morphismen gehören sogar zu einer noch kleineren Klasse, es handelt sich um total synchronisierte Morphismen, das sind injektive Morphismen $m : A^* \to B^*$, für die gilt $uvw \in m(A^*) \implies v \in m(A) \implies \{u, w\} \subseteq m(A^*)$. Wir bezeichnen mit $\mathcal{S}$ die Klasse aller Instanzen mit zwei total synchronisierten Morphismen. Es gilt:

b **Satz.** (Ruohonen [36]) *$\mathcal{S}$ und $\omega\,\mathcal{S}$ sind nicht entscheidbar.*

Die Konstruktion von Ruohonen ist darüber hinaus so angelegt, dass die Gleichheitsmenge gleich w^* für ein Wort w ist, das eine haltende Berechnung einer Turingmaschine kodiert. Wir wissen also, dass die Gleichheitsmenge regulär ist, kennen sogar die genaue Struktur, aber es ist im Allgemeinen trotzdem nicht möglich, das Wort w effektiv zu konstruieren.

Die *gespiegelte und die transponierte Instanz I^{R} und $\overline{I}$ von I* sind die Instanzen $I^{\mathrm{R}} = (A, B, f^{\mathrm{R}}, (h^{\mathrm{R}}, g^{\mathrm{R}}), s^{\mathrm{R}})$ und $\overline{I} = (A, B, \overline{s}, (g, h), \overline{f})$. Für Bijektionen $\alpha : A \to C$, die Regeln neue Namen geben, und deren ebenfalls mit α bezeichnete induzierte Isomorphismen $\alpha : A^* \to C^*$ sei das *α-Bild $\alpha(I)$ von I* die Instanz $\alpha(I) = (C, B, s, (\alpha^{-1}h, \alpha^{-1}g), f)$. Das *$\beta$-Bild von I* ist die Instanz $\beta(I) = (A, C, (\beta(s_\uparrow), \beta(s_\downarrow)), (h\beta, g\beta), (\beta(f_\uparrow), \beta(f_\downarrow)))$, wobei $\beta : B^* \to C^*$ ein injektiver Morphismus ist. Die Gleichheitsmengen der so abgeleiteten Instanzen sind strukturell gleich. Genauer gelten

c
$$\mathrm{Eq}(I) = \mathrm{Eq}(\overline{I}) = \alpha^{-1}(\mathrm{Eq}(\alpha(I))) = \mathrm{Eq}(\beta(I)) = \mathrm{Eq}(I^{\mathrm{R}})^{\mathrm{R}}$$

und

d
$$\mathrm{Eq}^\omega(I) = \mathrm{Eq}^\omega(\overline{I}) = \alpha^{-1}(\mathrm{Eq}^\omega(\alpha(I)) = \mathrm{Eq}^\omega(\beta(I))).$$

Die Gleichheitsmenge und die unendliche Gleichheitsmenge bleiben strukturell erhalten, wenn die Morphismen getauscht werden, wenn die Regeln umgeordnet werden oder ein injektiver Morphismus angewendet wird. Das Spiegeln erhält allerdings nur die Gleichheitsmenge.

Beweis: Zunächst gelten $\mathrm{Eq}(I) = \mathrm{Eq}(\overline{I})$ und $\mathrm{Eq}^\omega(I) = \mathrm{Eq}^\omega(\overline{I})$ wegen der Symmetrie von Gleichheit und von Vergleichbarkeit. Nun gilt, da α ein Isomorphismus ist, $u \in \mathrm{Eq}(I) \iff s_\uparrow h(u) f_\uparrow = s_\downarrow g(u) f_\downarrow \iff s_\uparrow h(\alpha^{-1}\alpha u) f_\uparrow = s_\downarrow g(\alpha^{-1}\alpha u) f_\downarrow \iff s_\uparrow (\alpha^{-1}h)(\alpha u) f_\uparrow = s_\downarrow (\alpha^{-1}g)(\alpha u) f_\downarrow \iff \alpha(u) \in \mathrm{Eq}(\alpha(I))$. Daraus folgt $\mathrm{Eq}(I) = \alpha^{-1}(\mathrm{Eq}(\alpha(I)))$ und analog auch $\mathrm{Eq}^\omega(I) = \alpha^{-1}(\mathrm{Eq}^\omega(\alpha(I)))$. Für einen injektiven Morphismus $\beta : B^* \to C^*$ gilt $u \in \mathrm{Eq}(I) \iff s_\uparrow h(u) f_\uparrow = s_\downarrow g(u) f_\downarrow \iff \beta(s_\uparrow h(u) f_\uparrow) = \beta(s_\downarrow g(u) f_\downarrow) \iff \beta(s_\uparrow)\beta(h(u))\beta(f_\uparrow) = \beta(s_\downarrow)\beta(g(u))\beta(f_\downarrow) \iff u \in \mathrm{Eq}(\beta(I))$ und damit $\mathrm{Eq}(I) = \mathrm{Eq}(\beta(I))$ und auch hier folgt analog $\mathrm{Eq}^\omega(I) = \mathrm{Eq}^\omega(\beta(I))$. Schließlich gilt $u \in \mathrm{Eq}(I) \iff s_\uparrow h(u) f_\uparrow = s_\downarrow g(u) f_\downarrow \iff (s_\uparrow h(u) f_\uparrow)^\mathrm{R} = (s_\downarrow g(u) f_\downarrow)^\mathrm{R} \iff f_\uparrow^\mathrm{R} h^\mathrm{R}(u^\mathrm{R}) s_\uparrow^\mathrm{R} = f_\downarrow^\mathrm{R} g^\mathrm{R}(u^\mathrm{R}) s_\downarrow^\mathrm{R} \iff u^\mathrm{R} \in \mathrm{Eq}(I^\mathrm{R})$ und damit $\mathrm{Eq}(I) = \mathrm{Eq}(I^\mathrm{R})^\mathrm{R}$. □ *a*

Mit $\mathcal{E}^k$ wird die Menge der Instanzen des GPCP bezeichnet, bei denen k Morphismen löschend sind. Die Menge $\{\mathcal{E}^0, \mathcal{E}^1, \mathcal{E}^2\}$ ist eine Partition von $\mathcal{G}$. Mit $\mathcal{E}^{\leq 1}$ bezeichnen wir die Menge $\mathcal{E}^0 \cup \mathcal{E}^1$ und mit $\mathcal{E}^{\geq 1}$ die Menge $\mathcal{E}^1 \cup \mathcal{E}^2$.

Die Menge aller Instanzen des GPCP, bei denen beide Morphismen markiert sind, wird mit $\mathcal{M}$ bezeichnet, ist nur ein Morphismus markiert schreiben wir auch $\mathcal{M}^1$.

Folgerung. $\mathcal{M} \subset \mathcal{E}^0$. *b*

Schließlich bezeichnen wir mit $\mathcal{D}$ die Menge aller Instanzen des GPCP, bei denen mindestens einer der Morphismen periodisch ist.

Beispiel: Sei $I_{15c} = (\{a, b\}, \{0, 1\}, (101, \varepsilon), (h, g), (\varepsilon, 1010))$ die Instanz des GPCP mit $h(a) = 0$, $h(b) = 01$, $g(a) = 10$ und $g(b) = \varepsilon$. Wir stellen Instanzen meist in der Form $I_{15c} = |\,{}^{101}_{\varepsilon}\,|\,{}^{0\ 01}_{10\ \varepsilon}\,|\,{}^{\varepsilon}_{1010}\,|$ dar. Bis auf das Alphabet A sind alle Informationen über I_{15c} immer noch ersichtlich. Üblicherweise gehen wir von A als Anfangsstück des Alphabets der Kleinbuchstaben aus. Wir unterdrücken dann eine explizite Angabe von A. Es ist $I_{15c}^\mathrm{R} = |\,{}^{\varepsilon}_{0101}\,|\,{}^{0\ 10}_{01\ \varepsilon}\,|\,{}^{101}_{\varepsilon}\,|$ und $\overline{I_{15c}} = |\,{}^{\varepsilon}_{101}\,|\,{}^{10\ \varepsilon}_{0\ 01}\,|\,{}^{1010}_{\varepsilon}\,|$ und es gilt $\mathrm{Eq}(I_{15c}) = \mathrm{Eq}(\overline{I_{15c}}) = \{ba\}$ und $\mathrm{Eq}(I_{15c}^\mathrm{R}) = \{ab\}$. Alle drei Instanzen sind Elemente von $\mathcal{E}^1 \cap \mathcal{D}$. ⌟ *c*

Beispiel: (Halava [12]) Die Instanz $I_{15d} = \|\,{}^{abb\ \ bb\ \ cbbb}_{a\ \ bbb\ \ c}\,\|$ hat die Gleichheitsmenge $\mathrm{Eq}(I_{15d}) = \{abb, cbbb\}^*$ und die Menge unendlicher Lösungen $\mathrm{Eq}^\omega(I_{15d}) = \{abb, cbbb\}^\omega \cup \{abb, cbbb\}^* b^\omega$. Sei $\alpha : \{a, b, c\}^* \to \{a, b, c\}^*$ der *d*

Isomorphismus, der durch die Permutation (b, c, a) induziert wird. Dann gilt $\alpha(I_{15d}) = \| \begin{smallmatrix} cbbb & abb & bb \\ c & a & bbb \end{smallmatrix} \|$ und $\mathrm{Eq}(\alpha(I_{15d})) = \{bcc, accc\}^*$. Mit dem injektiven Morphismus $\beta : \{a, b, c\}^* \to \{0, 1\}^*$, der durch $\beta(a) = 00$, $\beta(b) = 1$ und $\beta(c) = 01$ induziert wird, ergibt sich $\beta(I_{15d}) = \| \begin{smallmatrix} 0011 & 11 & 01111 \\ 00 & 111 & 01 \end{smallmatrix} \|$ mit $\mathrm{Eq}(\beta(I_{15d})) = \mathrm{Eq}(I_{15d})$. Alle drei Instanzen sind Elemente von $\mathcal{E}^0$, aber im Gegensatz zu I_{15d} und $\alpha(I_{15d})$ ist $\beta(I_{15d})$ kein Element von $\mathcal{M}$. ⌟

a **Beispiel:** Für die Instanz $I_{16a} = \| \begin{smallmatrix} 0 & 1 \\ 01 & 10 \end{smallmatrix} \|$ gilt $\mathrm{Eq}(I_{16a}) = \mathrm{Eq}^\omega(I^{\mathrm{R}}_{16a}) = \emptyset$ und $\mathrm{Eq}^\omega(I_{16a}) = \{\tau^\omega(0), \tau^\omega(1)\}$, wobei τ der Morphismus aus Beispiel 11a ist. ⌟

b **Beispiel:** Für die Instanz $I_{16b} = \| \begin{smallmatrix} 0 & 001 \\ 001 & 1 \end{smallmatrix} \| \begin{smallmatrix} 1 \\ 0 \end{smallmatrix} \|$ gilt $\mu(I_{16b}) = 75$. Die beiden kürzesten Lösungen *aacaacabbabccaaccaaaacbaabbaacbacbbccbbacbaccbcbacbb acbaccbacbbbacccbabbccb* und *aacaacaaacbabbaacacbccbbabacbcaaccbacabbbbab ccccbcaababaaccbcbbbacccbabbccb* wurden durch erschöpfende Suche gefunden. Die Instanz $J_{16b} = \| \begin{smallmatrix} 0 & 001 \\ 001 & 1 \end{smallmatrix} \| \begin{smallmatrix} 10 \\ 0 \end{smallmatrix} \|$ hat hingegen keine echte Lösung. Der Beweis dafür war nicht leicht zu finden. Tatsächlich handelt es sich bei den Instanzen I_{16b} und J_{16b} um die beiden schwierigsten aus der Menge $\mathcal{P}(3, 3) \cap \mathcal{E}^0$: Es gibt keine Instanz in $\mathcal{P}(3, 3) \cap \mathcal{E}^0$ mit längerer kürzester Lösung als I_{16b} und J_{16b} war die letzte Instanz aus $\mathcal{P}(3, 3) \cap \mathcal{E}^0$, für die offen war, ob es echte Lösungen gibt oder nicht. Mittlerweile liegt der maschinell überprüfbare (und überprüfte) Beweis aus 42a vor, der jedoch nicht vollständig von einer Maschine generiert wurde. ⌟

c **Beispiel:** (Schmidt [39]) Mario Schmidt zeigt in seiner Diplomarbeit, dass es das Semi-Thue-System R mit den Regeln $\{01 \mapsto 110, 21 \mapsto 2\}$ erlaubt, das Wort 20^k aus dem Wort 20^k1 abzuleiten und dass dafür genau $2^{k+1} - 1$ Schritte benötigt werden. Daraus folgt unmittelbar, dass die Familie

$$I_{16c}(k) = \left| \begin{array}{c|cccccc|c} \varepsilon & 0 & 1 & 2 & \# & 01 & 21 & 20^k\# \\ 20^k1\# & 0 & 1 & 2 & \# & 110 & 2 & \varepsilon \end{array} \right|$$

eine minimale Lösungslänge hat, die exponentiell bezüglich der Weite wächst. Mit der Konstruktion von Volker Claus, die von François Nicolas kürzlich detailliert dargestellt wurde, können wir aus R Familien mit exponentiellem Wachstum in $\mathcal{G}(4)$ und $\mathcal{P}(6)$ konstruieren. (Claus [3], Nicolas [32]) ⌟

d **Beispiel:** Die Länge der minimalen Lösung kann auch dann exponentiell bezüglich der Weite einer Familie wachsen, wenn für alle $a \in A$ stets $|h(a)| = |g(a)|$ gilt. Die Familie

$$I_{16d}(k) = \left| \begin{array}{c|cccccccccc|c} \varepsilon & a & r & t & \# & qa & qr & rq\# & tq\# & rqt & tqt & tqa^k\# \\ qaa^k\# & a & r & t & \# & rq & tq & qr\# & qt\# & qra & qta & \varepsilon \end{array} \right|$$

simuliert einen Teil der Berechnung der deterministischen Turingmaschine mit dem einzigen Zustand q und den Transitionen $a \mapsto +r$, $r \mapsto +t$, $\# \mapsto -\#$ und $t \mapsto -a$. Hier geben die Vorzeichen die Richtung der Kopfbewegung an. Es lässt sich zeigen, dass die Turingmaschine die Konfiguration $qaa^k\#$ in genau $2^{k+2} - 1$ Schritten in die Konfiguration $tqa^k\#$ überführt: Die Maschine simuliert das Traversieren eines vollständigen binären Baums, das Symbol a steht für „noch nicht betrachtet“, das Symbol r für die Wurzel eines Teilbaums und das Symbol t für einen Teilbaum, dessen eine Hälfte bereits vollständig traversiert wurde.

Die Instanz $I_{16d}(k)$ ist offensichtlich entscheidbar, denn nach spätestens 5^{k+3} Schritten wird ein Überhang wiederholt. Genauer sogar nach spätestens $(k+2)(k+3)3^{k+1}$, da es in jedem Überhang genau ein Symbol q und genau ein Symbol # gibt. Die Instanzen $I_{16d}(0)$ bis $I_{16d}(10)$ haben kürzeste Lösungen der Längen 5, 18, 53, 140, 347, 826, 1913, 4344, 9719, 21 494 und 47 093. ⌟

a

Beispiel: Bereits mit 5 Regeln lässt sich das Collatz-Problem kodieren. Sei $f : \mathbb{N} \to \mathbb{N}$ gegeben durch $f(n) = n/2$, falls n gerade und $f(n) = 3n + 1$, falls n ungerade. Dann besteht das Problem darin, zu ermitteln, ob es für jede natürliche Zahl $n \in \mathbb{N}$ ein $k \in \mathbb{N}$ gibt, so dass $f^k(n) = 1$ gilt. Trotz zahlreicher Bemühungen ist diese Frage unbeantwortet. Verwenden wir wie Maurice Margenstern statt f die Funktion $f' : \mathbb{N} \to \mathbb{N}$ mit $f'(2m) = m$ und $f'(2m + 1) = (3(2m + 1) + 1)/2 = 3m + 2$ und bauen auf dem Tag-System von Liesbeth de Mol auf, dann erhalten wir die Familie $I_{17a}(n) = \left|\begin{smallmatrix}\varepsilon\\ a^n\end{smallmatrix}\right| \begin{smallmatrix}aa & ac & cy & ya & yc\\ cy & cy & a & aaa & aaa\end{smallmatrix} \left|\begin{smallmatrix}a\\ \varepsilon\end{smallmatrix}\right|$. Die Instanz $I_{17a}(n)$ hat genau dann eine Lösung, wenn es ein k gibt, so dass $f^k(n) = 1$ ist. (Margenstern [23], de Mol [29]) ⌟

b

Beispiel: Gleichheitsmengen können Mengen sein, die nicht kontextfrei sind: Für die Instanz $I_{17b} = \left\|\begin{smallmatrix}00 & 1 & 1\\ 0 & 0 & 11\end{smallmatrix}\right\|$ gilt $\mathrm{Eq}(I_{17b}) \cap a^*b^*c^* = \{a^nb^nc^n \mid n \in \mathbb{N}\}$. ⌟

c

Beispiel: Die Anzahl kürzester Lösungen kann hoch sein. Die Instanz $I_{17c} = \left\|\begin{smallmatrix}0 & 0000 & 0101 & 11\\ 00 & 1111 & 1 & 10\end{smallmatrix}\right\|$ hat 4998 verschiedene kürzeste Lösungen der Länge 216, die alle das kommutative Bild $(96, 24, 32, 64)$ haben, alle mit a^{25} beginnen und alle mit $bd^8c^4d^2c$ enden. ⌟

d

Beispiel: Die Instanz $I_{17d} = \left\|\begin{smallmatrix}0 & 0 & 01 & 1111\\ 000 & 0101 & 1 & 10\end{smallmatrix}\right\|$ hat die kürzeste Lösung *aaaa aabbbbbbbbbaaaaccccccccccccccccccabbbbaabbbbbadddbbbccccccccaaaaaaccccccccc bbccccbcccccccddaaaaaaaaaaaaaaaaaaaddccccdcaddcbbbbbbbbbbbbbbbbbbbbbbbbbbb bbbbbbbbbbbbbbbbbbbbbbbbbbbbcccdcaaccccccccccccccccccccccccccccccccccccccc ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc dcaaaaaaddddddddddddddddddddddddddddddcbbbbbbbbbbbbbbbbbcccccccccccccccc ccccccccccccccccccccccccccccccccccccccccccccccccddddddddddddddddcccccccccc cccccccddddccccdc* der Länge 451, die interessant ist, weil sie einen Block von 111 aufeinanderfolgenden Symbolen c enthält. ⌟

a **Beispiel:** Die Instanz $I_{18a} = \| \begin{smallmatrix} 0 \\ 00010 \end{smallmatrix} \begin{smallmatrix} 010 \\ 01 \end{smallmatrix} \begin{smallmatrix} 100 \\ 0 \end{smallmatrix} \|$ hat die kürzeste Lösung *babacba babacbbababcbacbbacbabcbababbacbacbbcbababababccbbabacbacbcbbababababbcba cbbabacbbccbabcbacbabcbabbacbababbacbbcbababacbbcbabccbbabbababccbacbcb bcbacbbababababbccbabcbacbacbbcbabbacbababababccbbcbababbabacbbabccbbabbcba cbabcbacbcbbccbababbacbabbcbabacbbcbababbccbacbabccbacbbcbababababbabab abccbbabacbbcbacbbabacbcbbababccbabcbacbabbcbacbbababbacbababbccbabcb acbbcbababacbbcbabbacbabccbbabbababccbbcbababacbcbbcbacbbabccbbabbaba bbccbabcbacbcbbcbacbbcbabbacbabbccbabccbbcbababbcbabbabccbbabbccbbcbac bcbbcbccbabbccbabbcbabbccbbcbcc* der Länge 528, die keinen Block von drei aufeinanderfolgenden gleichen Symbolen enthält. ⌟

b **Beispiel:** Manchmal haben ganze Familien von Instanzen gemeinsame Lösungen: Sei I_{18b} die Familie mit $I_{18b}(k) = \| \begin{smallmatrix} 0 \\ 000 \end{smallmatrix} \begin{smallmatrix} 0 \\ 01^k \end{smallmatrix} \begin{smallmatrix} 0011 \\ 0 \end{smallmatrix} \begin{smallmatrix} 1^k0 \\ 11 \end{smallmatrix} \|$ mit $k \in \mathbb{N}$. Es gilt $0 < \mu(I_{18b}(k)) \leq 516$, denn alle Mitglieder der Familie haben die Lösung *c cdcbdadbcbacdcccbdadccdccdbcbadcbdadcbdacdcbadbcbadbabdcccdcbcadcbab adccdcbdadadbadabadcbdadbcbcbccbabadbcbcadadadccdabcadadbcbcbadcbda ccdbcbcadadcbadbacdcbdacdadbcbadcbacbdadbabdcbacdcbcccdcadbcbabadcc cbdadadccdcbdadbacdabadccdccdbcbcbadcbdadbcbacbdabadcbdadcbdacdadcb adbcbcadacdbabadbcbadbabdcbadcbcadadbabdcabacdcbabadcbadcccdbcbabad cabdadabadcbadccdcbdacdabadcbcbabadcbadcbdadbabdabadadadabadcbadbcbab abcbabadcbacdacadabadabdcadbabccbadbaadccdcbabadcbdadabadbcbabcadac cdbcadcbdacdbadbabdccbabadccdabadcbdabadbabaa* der Länge 516. Manche Instanzen der Familie haben auch kürzere Lösungen, zum Beispiel hat $I_{18b}(4)$ die Lösung *ddba* und allgemein hat die Instanz $I_{18b}(4(k+1))$ die Lösung $d^{2k+2}b^{2k+1}a^{k+1}$ der Länge $5k+4$. Die Instanz $I_{18b}(4(k+1)+2)$ hat die Lösung $a^{k+2}b^{2k+4}ad^{2k+4}cd^{2k+3}b^{2k+1}a^{k+1}$ der Länge $10k+17$. ⌟

c

Vereinfachen

Wir stellen Basistechniken vor, die es erlauben, Instanzen zu entscheiden oder zu reduzieren. Dazu betrachten wir in 19a mögliche Bilder von Lösungen und leiten daraus reguläre Obermengen der Gleichheitsmenge ab. In 19e benutzen wir einen algebraischen Ansatz, der aus Anzahlen von Symbolen in Regeln ein Gleichungssystem konstruiert, mit dessen Lösung wir die Gleichheitsmenge einschränken können. In 23b und 24a werden diese beiden Techniken erweitert und kombiniert und in 26a schließlich benutzt, um zu zeigen, wie periodische Instanzen entschieden werden können. Zu den periodischen Instanzen zählen neben den unären auch Instanzen, deren Zielalphabet nur ein Symbol enthält und auch binäre Instanzen, bei denen mindestens ein Morphismus löschend ist. In allen diesen Fällen folgt aus Lemma 26b, dass die Gleichheitsmenge kontextfrei ist und effektiv berechnet werden kann.

Vorgeben

a

Wir konstruieren einfache Obermengen der Gleichheitsmengen. Falls diese Obermengen endlich sind, oder nicht alle Regeln in Wörtern dieser Obermengen auftreten, dann kann die entsprechende Instanz entschieden oder zumindest reduziert werden.

Dazu definieren wir die *Start- und Endmenge von I*, die alle gültigen ersten und letzten Regeln einer echten Lösung beschreiben, durch $\mathrm{St}(I) = \{a \in A \mid s_\uparrow h(a) \asymp_\mathrm{p} s_\downarrow g(a)\}A^*$ und $\mathrm{Fl}(I) = A^*\{a \in A \mid h(a)f_\uparrow \asymp_\mathrm{s} g(a)f_\downarrow\}$. Beide Mengen sind regulär und es gelten $\mathrm{Eq}^\circ(I) \subseteq \mathrm{St}(I)$ und $\mathrm{Eq}^\circ(I) \subseteq \mathrm{Fl}(I)$.

Wenn wir uns überlegen, welche Wörter überhaupt als Lösung einer Instanz I in Frage kommen, dann stellen wir schnell fest, dass es sich bei den Bildern einer Lösung sowohl um Bilder von h als auch um Bilder von g handeln muss. Die Menge

$$\mathrm{M}(I) = s_\uparrow h(A^*)f_\uparrow \cap s_\downarrow g(A^*)f_\downarrow \qquad b$$

beschreibt eine Obermenge von in Frage kommenden Bildern und es gilt

$$\mathrm{Eq}(I) \subseteq h^{-1}(s_\uparrow^{-1}\,\mathrm{M}(I)f_\uparrow^{-1}) \cap g^{-1}(s_\downarrow^{-1}\,\mathrm{M}(I)f_\downarrow^{-1}) \qquad c$$

Diese Obermenge der Gleichheitsmenge nennen wir *Vorgabemenge von I* und bezeichnen sie mit $\mathrm{R}(I)$.

Beispiel: Für die Instanz $I_{19d} = \left\| \begin{smallmatrix} 0 & 00 & 101 & 101 \\ 00 & 101 & 0 & 11 \end{smallmatrix} \right\|$ mit $A = \{a, b, c, d\}$ gilt $\mathrm{St}(I_{19d}) = aA^*$, $\mathrm{Fl}(I_{19d}) = A^*a$ und $\mathrm{R}(I_{19d}) = \{a, b, c\}^*$. Die Regel d kann gelöscht werden, da sie in keiner Lösung verwendet wird. ⌟ *d*

Zählen

e

In diesem Abschnitt stellen wir einfache algebraische Überlegungen an, mit dem Ziel, die Menge der möglichen Lösungen einer Instanz einzuschränken. Wir konstruieren ein lineares Gleichungssystem, so dass das kommutative Bild der Gleichheitsmenge eine Menge von Lösungen des Gleichungssystems ist. Die Struktur der Lösungsmenge des Gleichungssystems erlaubt dann manchmal Aussagen über das Nichtvorkommen von Regeln in Lösungen von I. Darüber hinaus kann mit Hilfe des ersten Schrittes des Simplexalgorithmus entschieden werden, ob es Lösungen von I gibt, in denen alle Regeln mindestens ein Mal verwendet werden. Ist das nicht der Fall, dann kann die Lösbarkeit von I auf die Lösbarkeit einer Menge von Instanzen zurückgeführt werden, die jeweils weniger Regeln als I enthalten.

Sei also $u \in \mathrm{Eq}(I)$ eine Lösung von I. Dann gilt $s_\uparrow h(u)f_\uparrow = s_\downarrow g(u)f_\downarrow$ und es folgt sofort, dass auch die kommutativen Bilder der beiden Seiten gleich sind, also $\psi_B(s_\uparrow h(u)f_\uparrow) = \psi_B(s_\downarrow g(u)f_\downarrow)$ und damit ergibt sich

a $$\psi_B(s_\uparrow f_\uparrow) + \psi_B(h(u)) = \psi_B(s_\downarrow f_\downarrow) + \psi_B(g(u))$$

mit Vektoren auf beiden Seiten und komponentenweiser Gleichheit. Also gilt $\forall x \in B : |s_\uparrow f_\uparrow|_x + |h(u)|_x = |s_\downarrow f_\downarrow|_x + |g(u)|_x$ und damit

b $$\forall x \in B : \textstyle\sum_{a\in A} |u|_a(|h(a)|_x - |g(a)|_x) = |s_\downarrow f_\downarrow|_x - |s_\uparrow f_\uparrow|_x.$$

Bei (20b) handelt es sich um nichts anderes als um ein Gleichungssystem mit $|B|$ Gleichungen für die $|A|$ Unbekannten im Vektor $\psi_A(u)$.

c **Folgerung.** *Wenn $u \in \mathrm{Eq}(I)$ eine Lösung von I ist, dann ist $\psi_A(u)$ eine Lösung des Gleichungssystems in (20b).*

Damit können Instanzen direkt ausgeschlossen werden, für die (20b) keine Lösung besitzt. Instanzen, für die (20b) endlich viele Lösungen besitzt, können entschieden werden, da es nur endlich viele Wörter mit einem bestimmten kommutativen Bild gibt.

Auch Instanzen, für die (20b) unendlich viele Lösungen besitzt, können manchmal als unlösbar erkannt werden, oder es können Regeln identifiziert werden, die in keiner Lösung vorkommen können. Genauer gilt es festzustellen, ob $\operatorname{span} V \cap (\mathbb{N} \setminus \{0\})^{|I|}$ leer ist oder nicht, wobei V eine Basis des Nullraums von (20b) ist. Das entspricht genau dem ersten Schritt des Simplexalgorithmus, ist also entscheidbar.

d **Beispiel:** Für die Instanz $I_{20d} = |\,{}^{0}_{\varepsilon}\,|\,{}^{01}_{0}\;{}^{0}_{10}\,|\,{}^{\varepsilon}_{1}\,|$ ist (20b) das Gleichungssystem

$$\begin{aligned} |u|_a(1-1) + |u|_b(1-1) &= 0 - 1 && \text{(mit } x = 0\text{)} \\ |u|_a(1-0) + |u|_b(0-1) &= 1 - 0 && \text{(mit } x = 1\text{)} \end{aligned}$$

das offensichtlich keine Lösung hat. Daraus folgt $\mathrm{Eq}(I_{20d}) = \emptyset$. ⌟

e **Beispiel:** Für die Instanz $I_{15c} = |\,{}^{101}_{\varepsilon}\,|\,{}^{0}_{10}\;{}^{01}_{\varepsilon}\,|\,{}^{\varepsilon}_{1010}\,|$ ist die erweiterte Koeffizientenmatrix von 20b die Matrix $\left(\begin{smallmatrix} -1 & 1 & 0 \\ 0 & 1 & 1 \end{smallmatrix}\right)$, mit der einzigen Lösung $\psi_A(u) = (1, 1)$. Damit kommen nur die Wörter ab und ba als Lösungen in Frage. ⌟

f **Beispiel:** Für die Instanz $I_{20f} = \|\,{}^{0}_{00}\;{}^{0}_{1}\;{}^{000}_{11}\;{}^{1}_{000}\,\|$ folgt aus 20c zunächst, dass $\psi_A(\mathrm{Eq}(I_{20f})) \subseteq \operatorname{span}\{(1, -2, 1, 0), (-2, 1, 0, 1)\}$ und daraus, dass für jede Lösung u von I_{20f} gelten muss $\sum \psi_A(u) = 0$. Also gilt $\mathrm{Eq}(I_{20f}) = \{\varepsilon\}$. ⌟

g **Beispiel:** Für die Instanz $I_{20g} = \|\,{}^{0}_{00}\;{}^{0}_{010}\;{}^{001}_{0}\;{}^{101}_{1}\,\|$ folgt aus 20c zunächst, dass $\psi_A(\mathrm{Eq}(I_{20g})) \subseteq \operatorname{span}\{(0, 1, 1, 0), (0, 1, 0, 1)\}$ und daraus, dass die Regel a in keiner Lösung von I_{20g} vorkommt. Es gilt $\mathrm{Eq}(I_{20g}) = \{cb, bd, db\}^*$. ⌟

h **Beispiel:** Für die Instanz $I_{20h} = \|\,{}^{0}_{001}\;{}^{1}_{10}\;{}^{101}_{1}\;{}^{11}_{01}\,\|$ folgt aus 20c zunächst, dass $\psi_A(\mathrm{Eq}(I_{20h})) \subseteq \operatorname{span}\{(1, 0, 1, 0), (1, -2, 0, 1)\}$ und daraus, dass für Lösungen u von I_{20h} sowohl $|u|_b$ als auch $|u|_d$ gleich 0 sein müssen. Die Instanz I_{20h} ist demnach genau dann lösbar, wenn die Instanz lösbar ist, die durch Streichen von b und d entsteht. Diese Instanz ist unlösbar, also gilt $\mathrm{Eq}(I_{20h}) = \{\varepsilon\}$. ⌟

Teilwörter zählen *a*

Matthias Schulz hat in privater Kommunikation mit einer Technik erfolgreich einige herkömmliche Instanzen als unlösbar erkannt, bei der nicht mehr die Anzahl einzelner Symbole, sondern ganzer Teilwörter gezählt wird. Wir können das System (20b) auch für Teilwörter aufstellen, wenn wir uns u als zyklisch vorstellen. Das dürfen wir dann, wenn die Gleichheitsmenge ein Monoid ist, also zum Beispiel bei herkömmlichen Instanzen. Wir zählen dann auch zusammengesetztes Auftreten korrekt. Beim Zählen von einzelnen Symbolen gibt es kein zusammengesetztes Auftreten, es ist dann egal, ob wir uns u als zyklisch vorstellen oder nicht. Für ein gegebenes Teilwort w überlegen wir uns, in welchen minimalen zusammengesetzten Bildern w vorkommen kann und erhalten so ein Gleichungssystem, das als Unbekannte die Anzahlen von *Teilwörtern* von Lösungen hat. Das ist eine Verallgemeinerung vom Vorgehen in 19e, wo die Unbekannten die Anzahlen einzelner *Symbole* aus Lösungen sind, die wir als Teilwörter der Länge 1 auffassen können.

Das verallgemeinerte System lässt manchmal Rückschlüsse auf nicht in einer Lösung vorkommende Teilwörter zu. Diese Information kann beim Suchen nach Lösungen wie in 28a als Zusatzinformation zum Beschneiden des Suchgraphen genutzt werden.

Matthias Schulz verwendet sie auch, um neue Instanzen zu konstruieren, die genau dann lösbar sind, wenn die Originalinstanz lösbar ist. Dazu sucht er nach Informationen der Art „vor jedem a steht ein b“ und geht dann, falls a keine Startregel ist, zur Instanz über, die statt $(h(a), g(a))$ die Regel $(h(ba), g(ba))$ enthält. Diese Schlussweise ist leider weder leicht noch umfassend zu implementieren. Statt dessen können immer nur Teilwörter bis zu einer gewissen Länge betrachtet und nur Zusatzinformation mit vorab bekannter oder in ihrer Komplexität begrenzter Struktur gesucht werden.

Beispiel: Sei I_{21b} die Instanz $I_{21b} = \| \begin{smallmatrix} 0 \\ 00 \end{smallmatrix} \begin{smallmatrix} 01001 \\ 1 \end{smallmatrix} \begin{smallmatrix} 1 \\ 10011 \end{smallmatrix} \|$. Wir stellen das System *b*

	a	b	c	aa	ab	ac	ba	bb	bc	ca	cb	cc
0	-1	3	-2	0	0	0	0	0	0	0	0	0
1	0	1	-2	0	0	0	0	0	0	0	0	0
00	-1	1	-1	0	1	0	0	0	0	0	0	0
$a = aa + ab + ac$	-1	0	0	1	1	1	0	0	0	0	0	0
$b = ba + bb + bc$	0	-1	0	0	0	0	1	1	1	0	0	0
$c = ca + cb + cc$	0	0	-1	0	0	0	0	0	0	1	1	1
$a = aa + ba + ca$	-1	0	0	1	0	0	1	0	0	1	0	0
$b = ab + bb + cb$	0	-1	0	0	1	0	0	1	0	0	1	0
$c = ac + bc + cc$	0	0	-1	0	0	1	0	0	1	0	0	1

auf. Hier stehen die ersten 3 Zeilen für die Differenzen der Anzahlen der Vorkommen der Teilwörter 0, 1 und 00, wobei bei den Spalten für die Kombinationen nur die überlappenden Vorkommen gezählt werden, also die neu hinzu gekommenen. Die letzten 6 Zeilen reflektieren, dass in einer als zyklisch betrachteten Lösung die Anzahl zum Beispiel des Symbols a gleich der Anzahl der a ist, die von a, b oder c gefolgt werden. Lösen wir dieses System, erhalten wir die Basis des Nullraums

a	b	c	aa	ab	ac	ba	bb	bc	ca	cb	cc
0	0	0	1	0	−1	−1	0	1	0	0	0
4	2	1	0	3	1	3	−1	0	1	0	0
4	2	1	0	3	1	4	−2	0	0	1	0
4	2	1	1	3	0	3	−1	0	0	0	1

aus dem wir folgern, dass bei positiver Länge einer Lösung die Anzahl der Teilwörter bb negativ sein muss: Um eine positive Länge zu erhalten, muss einer der Vektoren der letzten drei Zeilen verwendet werden. Also hat I_{21b} keine echte Lösung. ⌟

a **Beispiel:** Analog ergibt sich für die Instanz $I_{22a} = \| \begin{smallmatrix} 0 \\ 00 \end{smallmatrix} \begin{smallmatrix} 01 \\ 1110 \end{smallmatrix} \begin{smallmatrix} 0110 \\ 1 \end{smallmatrix} \|$ das System

	a	b	c	aa	ab	ac	ba	bb	bc	ca	cb	cc
0	−1	0	2	0	0	0	0	0	0	0	0	0
1	0	−2	1	0	0	0	0	0	0	0	0	0
00	−1	0	0	0	1	1	−1	0	0	1	1	1
$a = aa + ab + ac$	−1	0	0	1	1	1	0	0	0	0	0	0
$b = ba + bb + bc$	0	−1	0	0	0	0	1	1	1	0	0	0
$c = ca + cb + cc$	0	0	−1	0	0	0	0	0	0	1	1	1
$a = aa + ba + ca$	−1	0	0	1	0	0	1	0	0	1	0	0
$b = ab + bb + cb$	0	−1	0	0	1	0	0	1	0	0	1	0
$c = ac + bc + cc$	0	0	−1	0	0	1	0	0	1	0	0	1

aus dessen Nullraumbasis

a	b	c	aa	ab	ac	ba	bb	bc	ca	cb	cc
0	0	0	2	−2	0	−2	2	0	0	0	0
0	0	0	2	0	−2	−2	0	2	0	0	0
4	1	2	1	1	2	1	0	0	2	0	0
0	0	0	0	2	−2	0	0	0	0	−2	2

sofort $c = ca$ geschlossen werden kann. Das heißt, nach jedem c folgt ein a und da c nicht letztes Symbol in einer Lösung sein kann, können wir zur Instanz

$\| \begin{smallmatrix} 0 \\ 00 \end{smallmatrix} \begin{smallmatrix} 01 \\ 1110 \end{smallmatrix} \begin{smallmatrix} 01100 \\ 100 \end{smallmatrix} \|$ übergehen, in der statt $(h(c), g(c))$ die Regel $(h(ca), g(ca))$ verwendet wird. Diese Instanz ist unlösbar wie zum Beispiel mit der Technik des Maskierens aus 42b ermittelt werden kann. ⌟

Beispiel: Zählen wir für die Instanz $I_{23a} = \| \begin{smallmatrix} 0 \\ 0010 \end{smallmatrix} \begin{smallmatrix} 1 \\ 1011 \end{smallmatrix} \begin{smallmatrix} 1010 \\ 10 \end{smallmatrix} \|$ das Teilwort 00, dann erhalten wir für eine Lösung u die Gleichung $|u|_{aa} + |u|_{ca} = |u|_a + |u|_{aa} + |u|_{ca}$ und damit sofort $|u|_a = 0$. Da $\psi_A(u) \in \operatorname{span}\{(1, 1, 3)\}$ gelten muss, folgt auch $|u|_b = |u|_c = 0$ und damit $\operatorname{Eq}(I_{23a}) = \{\varepsilon\}$. ⌟ *a*

Balancieren *b*

In 19e werden die möglichen kommutativen Bilder von Lösungen einer Instanz eingeschränkt: Es sind nur solche Bilder erlaubt, die Lösung eines spezifischen Gleichungssystems sind. Diese Methode versagt aber bei Instanzen wie $\| \begin{smallmatrix} 11 \\ 00 \end{smallmatrix} \begin{smallmatrix} 0 \\ 1 \end{smallmatrix} \|$ bei der aus (20b) für eine Lösung u lediglich $\psi_A(u) \in \operatorname{span}\{(1, 2)\}$ folgt. Die erweiterte Koeffizientenmatrix von (20b) dieser Instanz ist die Matrix $\left(\begin{smallmatrix} 2 & -1 & 0 \\ -2 & 1 & 0 \end{smallmatrix} \right)$, die Summe aller Gleichungen ergibt $0 = 0$.

Allgemeiner sprechen wir von einer *unbalancierten* Instanz, wenn es eine Linearkombination von Gleichungen in (20b) gibt, bei der alle Koeffizienten der linken Seite das gleiche Vorzeichen haben oder gleich Null sind. Dann muss für eine Lösung u die Gleichung $\sum_{a \in A} z_a |u|_a = c$ gelten, wobei die z_i sämtlich das selbe Vorzeichen haben oder gleich Null sind und c eine Konstante ist. In den meisten Fällen hat nun (20b) nur endlich viele Lösungen oder einige Komponenten einer Lösung können nicht positiv sein. Diese Fälle werden bereits durch die Techniken aus 19e erkannt und behandelt. Es verbleibt lediglich der Fall wie oben, dass alle h-Bilder die selbe Länge wie die korrespondierenden g-Bilder haben.

Bei einer solchen Menge von Regeln gilt stets $||s_\uparrow h(u)| - |s_\downarrow g(u)|| = ||s_\uparrow| - |s_\downarrow||$, also wird sich der Überhang in $(s_\uparrow h(u), s_\downarrow g(u))$ nach spätestens $|B|^d$ Schritten wiederholen, wobei $d = ||s_\uparrow| - |s_\downarrow||$ sein soll. Daraus folgt aber, dass eine kürzeste Lösung nicht länger als $|B|^d$ sein kann.

Folgerung. *Es ist entscheidbar, ob Instanzen mit $a \in A \implies |h(a)| = |g(a)|$ Lösungen besitzen.* *c*

Folgerung. *Es ist entscheidbar, ob unbalancierte Instanzen Lösungen besitzen, in denen jede Regel mindestens ein Mal verwendet wird.* *d*

Folgerung. $\mathcal{G}(1)$ *ist entscheidbar.* *e*

a

Balancieren mit Vorgabe

Balancieren mit Vorgabe wurde vom Autor bereits an anderer Stelle publiziert. (Rahn [35])

Wie gesehen, können unbalancierte Instanzen reduziert werden. Oftmals wird sogar eine Entscheidung möglich. Allerdings werden beim Zählen und beim Balancieren in 19e und 23b nur Aussagen über die kommutativen Bilder von Lösungen gemacht. Es wäre nun schön, diese Information mit der über die Vorgabemenge aus 19a zu verknüpfen. Auf den ersten Blick scheint das nicht so leicht möglich zu sein, haben wir es doch mit einer regulären Menge auf der einen Seite und mit dem Nullraum eines linearen Gleichungssystems auf der anderen Seite zu tun.

Doch die Darstellung des Gleichungssystems in der Form (20a) erlaubt es, die Einschränkungen an das kommutative Bild der Gleichheitsmenge in Form einer kontextfreien Sprache zu kodieren.

Dazu beschreiben wir eine (ganzzahlige) Linearkombination der Gleichungen durch einen Vektor $v \in \mathbb{Z}^{|B|}$ und definieren die *Balance-Abbildung* $\varrho_v(I) : A^* \to \mathbb{Z}$ *für* I *und* v durch

b
$$\varrho_v(I)(u) = \langle v|\, \psi_B(h(u))\rangle - \langle v|\, \psi_B(g(u))\rangle.$$

Es handelt sich um einen Morphismus von A^* nach $(\mathbb{N}, +, 0)$ und es gilt mit (20a) $\varrho_v(I)(\mathrm{Eq}(I)) = \langle v|\, \psi_B(s_\downarrow f_\downarrow)\rangle - \langle v|\, \psi_B(s_\uparrow f_\uparrow)\rangle$ oder anders formuliert $\mathrm{Eq}(I) \subseteq \varrho_v^{-1}(I)(d)$ mit $d = \langle v|\, \psi_B(s_\downarrow f_\downarrow)\rangle - \langle v|\, \psi_B(s_\uparrow f_\uparrow)\rangle$, woraus nach dem Abschnitt 19a über die Vorgabemenge sofort folgt

c
$$\mathrm{Eq}(I) \subseteq \varrho_v^{-1}(I)(d) \cap \mathrm{R}(I).$$

Diese Inklusion hilft nur dann weiter, wenn sich die Menge $\varrho_v^{-1}(I)(d)$ „unkompliziert“ verhält, sprich, wenn sich der Schnitt dieser Menge mit einer regulären Menge berechnen lässt. Das ist zum Glück der Fall, es gilt das

d **Lemma.** *Die Menge* $\varrho_v^{-1}(I)(d)$ *ist kontextfrei.*

e *Beweis:* Da $\varrho_v(I)$ ein Morphismus von A^* nach $(\mathbb{N}, +, 0)$ ist, gilt $\varrho_v^{-1}(I)(d) = \{u \mid \sum_{a\in A} |u|_a\, \varrho_v(I)(a) = d\}$. Die Menge auf der rechten Seite kann durch einen Kellerautomaten erkannt werden. Dazu nutzt der Kellerautomat seinen Stapel als unären Zähler und addiert bei jedem gelesenen Zeichen a den Wert $\varrho_v(I)(a)$ zum Wert des Zählers. Nachdem der Automat die Eingabe komplett gelesen hat, akzeptiert er genau dann, wenn der Zähler den Wert d enthält. □

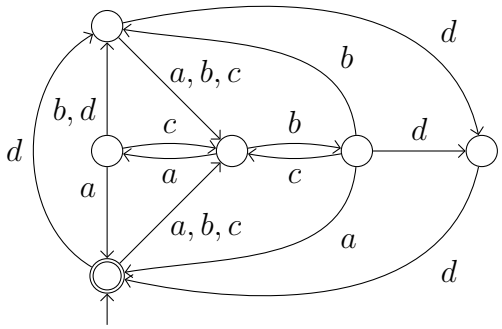
Da Leere und Endlichkeit für kontextfreie Sprachen entscheidbar sind, und da der Schnitt einer kontextfreien Sprache mit einer regulären Sprache wieder eine kontextfreie Sprache ist, haben wir eine weitere Möglichkeit die Gleichheitsmenge einzuschränken.

Das Lemma 24d gilt für beliebige Vektoren v, die natürlich nicht alle getestet werden können. In der Praxis sollten zumindest die Summen von Teilmengen der Einheitsvektoren untersucht werden.

Zum Abschluss dieses Abschnitt noch zwei Beispiele, die zeigen, dass durch die Kombination von Balancieren und Vorgeben tatsächlich mehr Instanzen entschieden oder reduziert werden können, als mit den einzelnen Techniken.

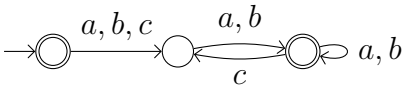
Beispiel: Für die Instanz $I_{25a} = \| \begin{smallmatrix} 0 \\ 00 \end{smallmatrix} \begin{smallmatrix} 0010 \\ 0 \end{smallmatrix} \begin{smallmatrix} 1 \\ 1011 \end{smallmatrix} \|$ ergibt sich die Vorgabemenge $\mathrm{R}(I_{25a}) = \{a, bcc, cacc\}^*$ und $\varrho^{-1}_{(1,1)}(I_{25a})(0)$ ist die Menge $\{u \mid |u|_a - 3|u|_b + 3|u|_c = 0\}$. Beide Mengen schränken die Gleichheitsmenge nicht so weit ein, dass eine Entscheidung oder eine Reduktion möglich wird. Betrachten wir jedoch den Schnitt und nehmen an, es gäbe ein $u \in \mathrm{R}(I_{25a}) \cap \varrho^{-1}_{(1,1)}(I_{25a})(0)$: Aus der Mitgliedschaft in $\mathrm{R}(I_{25a})$ folgt, dass u eine Iteration der Wörter a, bcc und $cacc$ ist. Seien x, y und z die Anzahlen der Vorkommen dieser Wörter in u, dann gilt $|u|_a = x + z$, $|u|_b = y$ und $|u|_c = 2y + 3z$. Da u ebenfalls in $\varrho^{-1}_{(1,1)}(I_{25a})(0)$ enthalten ist, muss gelten $|u|_a - 3|u|_b + 3|u|_c = x + z - 3y + 3(2y + 3z) = x + 3y + 10z = 0$. Daraus folgen $x = y = z = 0$ und $\mathrm{Eq}(I_{25a}) = \{\varepsilon\}$. ⌟ *a*

Beispiel: Bei der Instanz $I_{25b} = \| \begin{smallmatrix} 0 \\ 000 \end{smallmatrix} \begin{smallmatrix} 0 \\ 001 \end{smallmatrix} \begin{smallmatrix} 10 \\ 111 \end{smallmatrix} \begin{smallmatrix} 111 \\ 1 \end{smallmatrix} \|$ wird die Vorgabemenge durch den Automaten *b*



akzeptiert und es gilt $\varrho^{-1}_{(1,0)}(I_{25b})(0) = \{u \mid |u|_c = 2|u|_a + |u|_b\}$. Auch hier schränken die beiden Mengen die Gleichheitsmenge nicht entscheidend ein, ihr Schnitt enthält jedoch nur Wörter ohne das Symbol a. Das kann (ohne Lehrbuchkonstruktion des Kellerautomaten für den Schnitt) eingesehen werden, wenn beachtet wird, dass das Symbol d in $\varrho_{(1,0)}(I_{25b})$ das Gewicht 0 hat und

deshalb im Automaten für $\mathrm{R}(I_{25b})$ durch einen ε-Übergang ersetzt werden kann. Es ergibt sich dann der Automat



der die Projektion von $\mathrm{R}(I_{25b})$ auf $\{a, b, c\}^*$ akzeptiert. Den Symbolen a, b und c ordnet $\varrho_{(1,0)}(I_{25b})$ die Gewichte -2, -1 und 1 zu und jede Verwendung des Symbols a verhindert, dass die Summe der Gewichte in einem Endzustand gleich Null ist. Also kann die Regel a gelöscht werden, eine Information, mit deren Hilfe die komplette Instanz als unlösbar erkannt werden kann. ⌟

a

Periodische Instanzen

In den letzten Abschnitten konstruieren wir Obermengen der Gleichheitsmenge einer Instanz. In Gleichung (24c) ist diese Obermenge ein Schnitt einer kontextfreien mit einer regulären Sprache. Es liegt nahe, nach Mengen von Instanzen zu fragen, für die in (24c) sogar Gleichheit gilt. Eine solche Menge ist entscheidbar. Die Menge der periodischen Instanzen ist eine solche Menge. Es gilt

b **Lemma.** *Wenn $I \in \mathcal{D}$ eine periodische Instanz des GPCP ist, dann gilt* $\mathrm{Eq}(I) = \varrho_\iota^{-1}(I)(d) \cap \mathrm{R}(I)$ *wobei* $\iota = (1, \ldots, 1)$ *die Länge* $|B|$ *hat und* $d = \langle \iota | \psi_B(s_\downarrow f_\downarrow)\rangle - \langle \iota | \psi_B(s_\uparrow f_\uparrow)\rangle$ *ist.*

c *Beweis:* Sei I eine periodische Instanz und sei ohne Beschränkung der Allgemeinheit der Morphismus h periodisch. Dann gibt es ein $t \in B^*$, so dass $h(A^*) \subseteq t^*$ ist. Mit (24c) bleibt nur $\varrho_\iota^{-1}(I)(d) \cap \mathrm{R}(I) \subseteq \mathrm{Eq}(I)$ zu zeigen. Sei $u \in \varrho_\iota^{-1}(I)(d) \cap \mathrm{R}(I)$. Aus $u \in \mathrm{R}(I)$ folgt mit (19b) und (19c) $s_\downarrow g(u) f_\downarrow \in s_\uparrow h(A^*) f_\uparrow$, also gibt es ein $i \in \mathbb{N}$, so dass $s_\downarrow g(u) f_\downarrow = s_\uparrow t^i f_\uparrow$ gilt. Aus $u \in \varrho_\iota^{-1}(I)(d)$ und mit (24b) folgt $\varrho_\iota(I)(u) = \langle \iota | \psi_B(h(u))\rangle - \langle \iota | \psi_B(g(u))\rangle = \langle \iota | \psi_B(s_\downarrow f_\downarrow)\rangle - \langle \iota | \psi_B(s_\uparrow f_\uparrow)\rangle$ und damit schließlich $|s_\uparrow h(u) f_\uparrow| = |s_\downarrow g(u) f_\downarrow| = |s_\uparrow t^i f_\uparrow|$, da $\langle \iota | \psi_B(w)\rangle = |w|$. Aus $|h(u)| = |t^i|$ folgt aber $h(u) = t^i$, da es in t^* nur jeweils ein Wort mit einer bestimmten Länge gibt. □

d **Folgerung.** *$\mathcal{D}$ ist entscheidbar.*

e **Folgerung.** *Die Menge aller Instanzen mit $|B| = 1$ ist entscheidbar.*

f **Folgerung.** *$\mathcal{G}(2) \cap \mathcal{E}^{\geq 1}$ ist entscheidbar.*

g **Beispiel:** Für die Instanz $I_{15c} = \left|\begin{smallmatrix} 101 \\ \varepsilon \end{smallmatrix}\right| \left|\begin{smallmatrix} 0 & 01 \\ 10 & \varepsilon \end{smallmatrix}\right| \left|\begin{smallmatrix} \varepsilon \\ 1010 \end{smallmatrix}\right|$ gelten $\mathrm{R}(I_{15c}) = b^*a$, $\varrho_\iota^{-1}(I_{15c})(1) = \{u \mid 2|u|_b - |u|_a = 1\}$ und $\mathrm{Eq}(I_{15c}) = \{ba\}$. ⌟

h **Beispiel:** Für die Instanz $I_{26h} = \left\|\begin{smallmatrix} \varepsilon & 0 \\ 0 & \varepsilon \end{smallmatrix}\right\|$ gilt $\mathrm{R}(I_{26h}) = \{a, b\}^*$ und $\mathrm{Eq}(I_{26h}) = \varrho_\iota^{-1}(I_{26h})(0) = \{u \mid |u|_a = |u|_b\}$. ⌟

Suchen

a

Die bisherigen Abschnitte stellen Methoden dar, die mit Information über die kommutativen Bilder und über die möglichen h- oder die möglichen g-Bilder von Lösungen arbeiten. Keine Berücksichtigung findet die Tatsache, dass die Menge aller lösbaren Instanzen semientscheidbar ist: Ein Algorithmus, der für alle Elemente einer Aufzählung von A^* prüft, ob sie Lösung einer Instanz I sind oder nicht, wird eine existierende Lösung schließlich finden. Mit anderen Worten kann eine Turingmaschine die Menge aller lösbaren Instanzen akzeptieren, hält aber bei unlösbaren Instanzen eventuell nicht an.

In diesem Abschnitt wollen wir genauer untersuchen, was es heißt, nach einer Lösung zu suchen. Ausgangspunkt ist dabei der eben skizzierte Algorithmus, der eventuell existierende Lösungen garantiert findet, aber bei Instanzen ohne Lösung überhaupt kein Ergebnis produziert. Wie üblich bei solchen Problemen können wir mit diesem Algorithmus das *beschränkte GPCP* lösen. Dort wird nicht mehr nach der Existenz einer beliebigen Lösung gefragt, sondern nach der Existenz einer Lösung bis zu einer maximal erlaubten Länge. Dieses Problem ist NP-vollständig. (Garey, Johnson [6])

Nachdem wir in 28a klären, dass Suchen nichts anderes ist als das Traversieren eines bestimmten Graphen, untersuchen wir in den folgenden Abschnitten jeweils Techniken, diesen Graphen zu vereinfachen.

In 32a gehen wir der Frage nach, ob die Beschriftung eines Pfades im Suchgraphen Rückschlüsse auf dessen Produktivität erlaubt. Mit Hilfe der Erkenntnisse aus 18c gelingt es, Pfade als unproduktiv zu erkennen. Dabei können Pfade entweder als absolut unproduktiv erkannt werden oder als unproduktiv bezüglich einer vorgegebenen maximalen Lösungslänge.

In 34a werden nicht mehr ganze Pfade, sondern nur noch einzelne Knoten betrachtet und es werden Mengen von Überhängen konstruiert, die nicht auf dem Weg zu einer Lösung liegen. Die Methode in 36a hat sich als sehr effektiv erwiesen. Dort werden Überhänge konstruiert, die die Anwendung bestimmter Regeln erzwingen, aber die bei Anwendung der erzwungenen Regeln nicht zu einer Lösung führen.

Eine ähnliche Idee führt in 39b ebenfalls zu Mengen von Überhängen, die nicht auf dem Weg zu einer Lösung liegen können. Diesmal handelt es sich um Mengen von Überhängen, die durch Regelanwendung nicht mehr verlassen werden können. Enthält eine solche Menge keine Lösung, dann muss keines ihrer Elemente bei der Traversierung des Suchgraphen berücksichtigt werden.

In 42b beschreiben wir, wie die Position von Überhängen beschränkt werden kann, ein iterativer Prozess, der ausnutzt, dass im Suchgraphen einer Instanz ein Pfad zu einer Lösung stets einem transponierten Pfad im Suchgraphen der gespiegelten Instanz entspricht.

a

Suchen

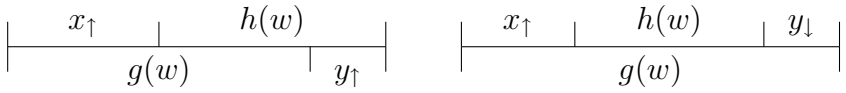
Es ist nicht effizient, *alle* Wörter aus $A^{\leq n}$ zu untersuchen, denn wenn für ein Wort w gilt, dass $s_\uparrow h(w)$ und $s_\downarrow g(w)$ nicht vergleichbar sind, dann kann w kein Präfix einer Lösung sein. Stattdessen sollten nur solche Wörter betrachtet werden, die Präfix einer Lösung sein können. Formaler sei die *Suchrelation* $\rightarrowtail_I : B^{*\updownarrow} \times A^* \times B^{*\updownarrow}$ *von* I gegeben durch

b

$$x \overset{w}{\rightarrowtail}_I y \iff x_\uparrow h(w) y_\downarrow = x_\downarrow g(w) y_\uparrow.$$

Werfen wir einen genaueren Blick auf (28b): Die Paare x und y stehen mit w gewichtet in Relation. Beide Paare sind in $B^{*\updownarrow}$ enthalten, also ist mindestens eine ihrer Komponenten das leere Wort ε. Nehmen wir an, x hat die Gestalt $x = (x_\uparrow, \varepsilon)$, dann stellt

c



die Relation zu y in den beiden Fällen $y = (y_\uparrow, \varepsilon)$ und $y = (\varepsilon, y_\downarrow)$ dar. In beiden Fällen sind $x_\uparrow h(w)$ und $g(w)$ miteinander vergleichbar und y beschreibt den Überhang und dessen Position.

Direkt aus der Definition ergeben sich der Zusammenhang zur Gleichheitsmenge und zur Suchrelation der gespiegelten Instanz.

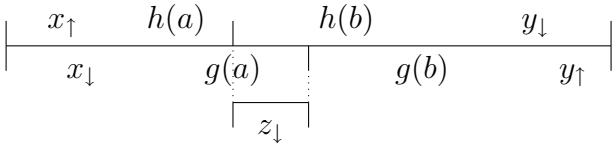
d **Folgerung.** $\mathrm{Eq}(I) = \{u \mid s \overset{u}{\rightarrowtail}_I \overline{f}\}$.

e **Folgerung.** (Spiegeln von $\rightarrowtail$) $x \overset{w}{\rightarrowtail}_I y \iff \overline{y}^{\mathrm{R}} \overset{w^{\mathrm{R}}}{\rightarrowtail}_{I^{\mathrm{R}}} \overline{x}^{\mathrm{R}}$.

Nützlich wird die Suchrelation durch das

f **Lemma.** (Zerlegen von $\rightarrowtail$) $x \overset{ab}{\rightarrowtail}_I y \iff \exists z : x \overset{a}{\rightarrowtail}_I z \overset{b}{\rightarrowtail}_I y$.

g *Beweis:* Nach Definition von $\rightarrowtail_I$ und weil h und g Morphismen sind, gilt $x \overset{ab}{\rightarrowtail}_I y \iff x_\uparrow h(a)h(b)y_\downarrow = x_\downarrow g(a)g(b)y_\uparrow$ also sind $x_\uparrow h(a)$ und $x_\downarrow g(a)$ vergleichbar. Nehmen wir zunächst an, $x_\uparrow h(a)$ ist ein Präfix von $x_\downarrow g(a)$: Dann gibt es ein $z_\downarrow$, so dass $x_\uparrow h(a) z_\downarrow = x_\downarrow g(a)$ und gleichzeitig $h(b)y_\downarrow = z_\downarrow g(b) y_\uparrow$:



Das ist der Fall, wenn für $z = (\varepsilon, z_\downarrow)$ sowohl $x \overset{a}{\rightarrowtail}_I z$ als auch $z \overset{b}{\rightarrowtail}_I y$ gilt. Der Beweis verläuft analog, wenn $x_\downarrow g(a)$ ein Präfix von $x_\uparrow h(a)$ ist. □

Als Konsequenz aus Lemma 28f können wir nun alternativ formulieren, dass das Suchen nach einer Lösung nichts anderes ist, als das Suchen nach einem Pfad durch $\rightarrowtail_I$, der in s beginnt und in $\overline{f}$ endet. Eine solche Suche ist bereits wesentlich effizienter als das naive Vorgehen, da nur noch Wörter betrachtet werden, deren Präfixe vergleichbare Bilder haben.

Wir bezeichnen die Hülle $\rightarrowtail_I^\star(\{s\})$ von $\rightarrowtail_I$ über $\{s\}$ als den *Suchgraphen von I*. Der Suchgraph kann als Automat mit unendlich vielen Zuständen und s als Start- und $\overline{f}$ als Endzustand betrachtet werden, der genau die Gleichheitsmenge akzeptiert. Daraus folgt *nicht*, dass $\rightarrowtail_I^\star(\{s\})$ genau dann endlich ist, wenn die Gleichheitsmenge regulär ist. Zwar erzwingt die Endlichkeit des Suchgraphen eine reguläre Gleichheitsmenge, die Umkehrung gilt aber nicht. Es kann trotz regulärer Gleichheitsmenge beliebig lange unproduktive Pfade geben, das sind Pfade, die nicht zu einem Endzustand führen.

Unendlich lange unproduktive Pfade gehören zu unendlichen Lösungen. Der *produktive* Teil des Suchgraphen ist genau dann endlich, wenn die Gleichheitsmenge regulär ist. Die Frage, ob ein Pfad produktiv ist oder nicht, ist aber genau die Frage nach der Lösbarkeit einer Instanz des GPCP und damit nicht entscheidbar. Das heißt, dass der Suchgraph im Allgemeinen selbst dann nicht explizit konstruiert werden kann, wenn bekannt ist, dass eine Instanz eine reguläre Gleichheitsmenge besitzt.

Betrachten wir nur den Teilgraphen von $\rightarrowtail_I^\star(\{s\})$, in dem sämtliche Knoten Überhänge enthalten, die höchstens die Länge k haben, dann erhalten wir einen endlichen Automaten, der die reguläre Sprache $\mathrm{Eq}^k(I)$ akzeptiert, die wir *k-beschränkte Gleichheitsmenge* nennen. Offensichtlich gilt $\mathrm{Eq}(I) = \bigcup_{k\geq 0} \mathrm{Eq}^k(I)$ und mit Hilfe der k-beschränkten Gleichheitsmengen erhalten wir die Folge

$$\mathrm{Eq}^0(I) \subseteq \mathrm{Eq}^1(I) \subseteq \cdots \subseteq \mathrm{Eq}(I) \qquad a$$

unterer Approximationen der Gleichheitsmenge. Es folgt, dass die Gleichheitsmenge genau dann regulär ist, wenn sie Teilmenge einer k-beschränkten Gleichheitsmenge für ein $k \in \mathbb{N}$ ist. (Harju, Karhumäki [13]) Darüber hinaus gilt das

Lemma. *Wenn $L \subseteq \mathrm{Eq}(I)$ eine reguläre Teilmenge der Gleichheitsmenge von I ist, dann gibt es ein $k \in \mathbb{N}$, so dass $L \subseteq \mathrm{Eq}^k(I)$ bereits eine Teilmenge der k-beschränkten Gleichheitsmenge von I ist.* b

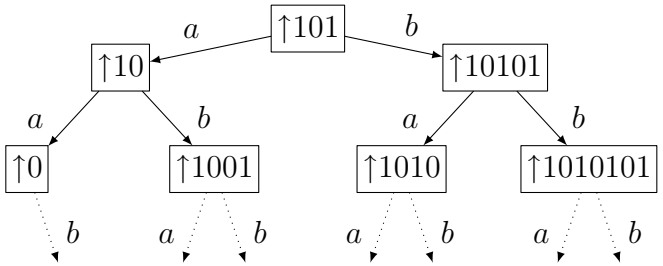
Beweis: Da die Sprache L regulär ist, wird sie von einem reduzierten endlichen Automaten A_L mit n Zuständen akzeptiert. Jede Schleife (q, w, q) in der erweiterten Überführungsrelation δ' von A_L ist mit einem Wort w beschriftet, für das $|h(w)| = |g(w)|$ ist, anderenfalls ergibt genügend häufiges Durchlaufen der Schleife einen Widerspruch zu $L \subseteq \mathrm{Eq}(I)$. Daraus folgt, dass alle Wörter c

in L eine maximale Differenz in Längen der h- und g-Bilder haben, die sich anhand der Beschriftungen der schleifenfreien Pfade in δ' effektiv ermitteln lässt. □

a **Folgerung.** *Für reguläre Mengen L ist entscheidbar, ob $L \subseteq \mathrm{Eq}(I)$ gilt.*

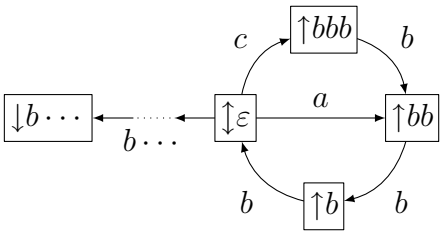
Die Entscheidung in 30a ist sogar effizient möglich: Berechne die Schranke m aus Beweis 29c von Lemma 29b und teste, ob $L \subseteq \mathrm{Eq}^m(I)$ ist.

b **Beispiel:** Für die Instanz $I_{15c} = |\,{}^{101}_{\varepsilon}\,|\,{}^{0}_{10}\,{}^{01}_{\varepsilon}\,|\,{}^{\varepsilon}_{1010}\,|$ hat der Suchgraph die Gestalt:



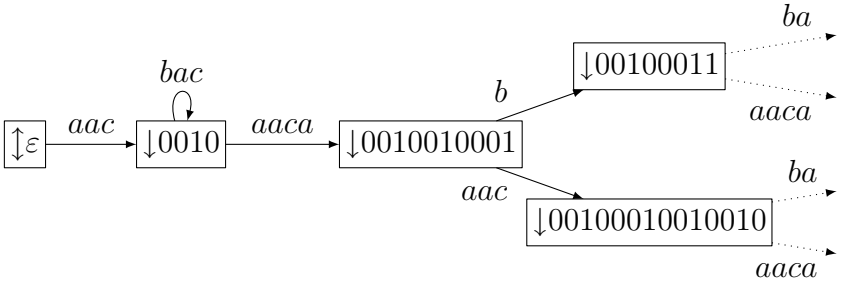
In solchen Darstellungen geben wir, wie bei Hasse Diagrammen von Halbordnungen, keine transitiven Kanten an. Dieser Graph ist unendlich groß, da Regel b stets angewendet werden kann. Der Pfad $s = \uparrow 101 \xrightarrow{b}_{I_{15c}} \uparrow 10101 \xrightarrow{a}_{I_{15c}} \uparrow 1010 = \overline{f}$ gehört zur Lösung ba. ⌟

c **Beispiel:** Für die Instanz $I_{15d} = \|\,{}^{abb}_{a}\,{}^{bb}_{bbb}\,{}^{cbbb}_{c}\,\|$ hat der Suchgraph die Gestalt:



Hier gibt es links den unendlichen Pfad b^ω, der im k-ten Schritt das Paar (ε, b^k) erzeugt und zur unendlichen Lösung b^ω gehört. ⌟

d **Beispiel:** Für die Instanz $I_{16b} = \|\,{}^{0}_{001}\,{}^{001}_{1}\,{}^{1}_{0}\,\|$ hat der Suchgraph die Gestalt:



Die liberalisierte Darstellung verschmilzt Pfade ohne Abzweig. In 16b wurde erwähnt, dass $\mu(I_{16b}) = 75$ gilt. Eine Möglichkeit des Beweises ist, neben der Angabe einer Lösung der Länge 75, das komplette Traversieren aller Pfade im Suchgraphen, die höchstens eine Länge von 74 haben. Es gibt 85 119 109 solcher Pfade, die zusammen 375 636 850 Knoten berühren. Das entspricht einer beschränkten Tiefensuche ohne Detektion von Schleifen. Soll mit dieser Methode zusätzlich die Anzahl der kürzesten Lösungen bestimmt werden, müssen die 109 918 385 Pfade mit einer Länge von höchstens 75 und ihren 485 555 235 Knoten traversiert werden.

Laut (14c) können wir auch den Suchgraphen von I^{R}_{16b} heranziehen. Dort gibt es nur 12 782 632 Pfade bis zur Länge 74 mit 61 962 784 Knoten und nur 16 105 042 Pfade bis zur Länge 75 mit 78 067 826 Knoten.

In beiden Fällen wurden auch Pfade mit Schleifen betrachtet. Alternativ müssten bereits betrachtete Knoten memoriert werden, zusammen mit der Tiefe auf der sie angetroffen wurden. Bei der Anzahl der Knoten ist klar, dass das nicht umfänglich und auf keinen Fall naiv erfolgen kann. In 110c werden wir Möglichkeiten zur partiellen Detektion von Schleifen diskutieren. ⌟

Beispiel: (Zhao [44]) Der Unterschied in der Größe der Suchgraphen für Instanz und gespiegelte Instanz kann extrem ausfallen. Für die Instanz $I_{31a} = \| \begin{smallmatrix}0\\01010\end{smallmatrix} \begin{smallmatrix}00000\\01\end{smallmatrix} \begin{smallmatrix}10100\\00\end{smallmatrix} \|$ gilt $\mu(I_{31a}) = 189$. Die einzige kürzeste Lösung w ist *acaacc aaaaccccaaaaaaaaccccccccaaaaaabbccccccccaaabbccccaaaaaaaccccccccaaaabbcc cccaaaaaaaaaccccccccccaaaaaaaabbccccccccccaaaaaaaabbccccccccaaaabbccccc cabbccaaacccaaaaaaccccccaabbcccaaaaaccccccbbc* und um das mit Hilfe des Suchgraphen von I_{31a} zu beweisen, müssen alle 38 244 752 Pfade bis zur Länge 189 mit insgesamt 268 869 440 Knoten betrachtet werden. Im Gegensatz dazu hat der Suchgraph von I^{R}_{31a} nur 2 Pfade bis zur Länge 189 mit 190 Knoten. Er beweist zusätzlich, dass $\mathrm{Eq}(I_{31a}) = w^*$ eine reguläre Sprache ist. ⌟ *a*

Beispiel: Ein extremes Beispiel ist die Instanz $I_{31b} = \| \begin{smallmatrix}0\\0010\end{smallmatrix} \begin{smallmatrix}000\\01\end{smallmatrix} \begin{smallmatrix}010\\0100\end{smallmatrix} \begin{smallmatrix}1000\\000\end{smallmatrix} \|$ bei der der Suchgraph von I^{R}_{31b} aus einem Pfad der Länge 1 und einem weiteren (nicht geschlossenen) Pfad der Länge 742 besteht und somit beweist, dass I_{31b} keine echte Lösung besitzt. Es ist erstaunlich, dass von den 4 Regeln 742 mal genau eine passt und dann keine mehr. Vom Suchgraphen von I_{31b} hingegen ist unbekannt, ob er endlich ist oder nicht. Zumindest wächst er zu Beginn ziemlich stark, es gibt dort zum Beispiel 927 339 463 Pfade bis zur Länge 70 mit 2 687 944 007 Knoten. ⌟ *b*

Beispiel: Instanzen mit wenigen Regeln und geringer Weite können nicht nur tiefe, sondern auch breite aber endliche Suchgraphen beschreiben. Der Suchgraph der Instanz $I^1_{31c} = \| \begin{smallmatrix}0\\0001\end{smallmatrix} \begin{smallmatrix}0\\0100\end{smallmatrix} \begin{smallmatrix}011\\1\end{smallmatrix} \begin{smallmatrix}101\\1101\end{smallmatrix} \|$ ist endlich und hat keinen *c*

Pfad der Länge 53. Es gibt aber 6 291 456 Pfade der Länge 52 und insgesamt 10 206 170 Pfade einer Länge von höchstens 52 mit 18 463 959 Knoten.

Der Suchgraph der Instanz $I^2_{31c} = \| \begin{smallmatrix} 0 \\ 1000 \end{smallmatrix} \begin{smallmatrix} 0 \\ 11 \end{smallmatrix} \begin{smallmatrix} 0001 \\ 00 \end{smallmatrix} \begin{smallmatrix} 1111 \\ 0111 \end{smallmatrix} \|$ hat keinen Pfad der Länge 77, aber 131 072 Pfade der Länge 76 und 21 030 648 Pfade einer Länge von höchstens 76 mit 44 660 201 Knoten. ⌟

a

Abschneiden

Wenden wir uns der Frage zu, ob Pfade im Suchgraphen einer Instanz mit Hilfe von Informationen über mögliche Lösungen als unproduktiv erkannt werden können. Wir unterscheiden dabei zwischen *genauen* und *beschränkten* Methoden. Eine genaue Methode kann unproduktive Pfade absolut erkennen, während eine beschränkte Methode lediglich Pfade erkennen kann, die in einer beschränkten Suche unproduktiv sind.

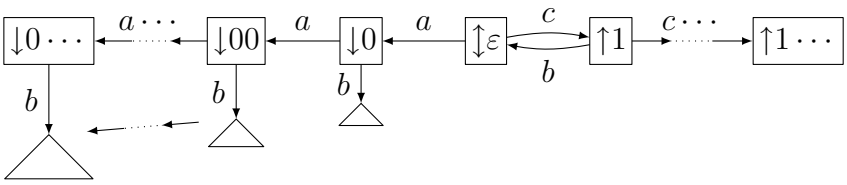
b

Genau

In 19a wird eine reguläre Obermenge der Gleichheitsmenge konstruiert. Wenn ein Pfad mit einem Wort beschriftet ist, das kein Präfix eines Wortes aus der Vorgabemenge ist, dann ist dieser Pfad unproduktiv und muss nicht weiter betrachtet werden. Das ist effektiv entscheidbar, da reguläre Sprachen gegenüber den Operationen Präfix und Komplement abgeschlossen sind. Wir können sogar effizient vorgehen, wenn wir einen deterministischen und vollständigen Automaten mit genau einem Startzustand für die Vorgabemenge $\mathrm{R}(I)$ heranziehen. Ein solcher Automat hat genau dann tote Zustände, wenn es solche verbotenen Präfixe gibt und kann parallel zum Suchgraphen arbeiten. Das heißt, dass jedem Knoten x im Suchgraphen genau ein Zustand q_x des Automaten zugeordnet wird und dass der einem Knoten y mit $x \overset{a}{\rightarrowtail}_I y$ zugeordnete Zustand q_y des Automaten nur von q_x und a abhängt. Handelt es sich bei q_y um einen toten Zustand, dann ist der Pfad, der über y führt nicht produktiv und y muss nicht weiter betrachtet werden.

c

Beispiel: Betrachten wir die Instanz $I_{32c} = \| \begin{smallmatrix} 0 \\ 00 \end{smallmatrix} \begin{smallmatrix} 0 \\ 10 \end{smallmatrix} \begin{smallmatrix} 11 \\ 1 \end{smallmatrix} \|$ mit der Vorgabemenge $\mathrm{R}(I_{32c}) = (aa)^* \cup (aa)^* c\{b,c\}(a \cup c\{b,c\})^*$ und dem Beginn des Suchgraphen



wobei sich am linken unendlichen Pfad jeweils endliche aber exponentiell wachsende Teilgraphen befinden. Diese Teilgraphen können alle abgeschnitten werden, da a^*b kein Präfix einer möglichen Lösung ist. ⌟

Beschränkt *a*

In 19e konstruieren wir ein Gleichungssystem, so dass das kommutative Bild jeder Lösung einer Instanz auch Lösung dieses Gleichungssystem ist. Mit Hilfe des ersten Schritts des Simplexalgorithmus kann entschieden werden, *ob* es Lösungen gibt, bei denen jede Regel mindestens ein Mal verwendet wird. Der Simplexalgorithmus erlaubt auch festzustellen, *wie lang* eine Lösung mindestens sein muss. Wird der Suchgraph traversiert, dann entspricht jeder Schritt der Änderung einer Nebenbedingung: Initial fordern wir für eine Regel a und eine Lösung u nur $|u|_a \geq 0$. Beim ersten Schritt mit a wird daraus die neue Nebenbedingung $|u|_a \geq 1$, beim k-ten Schritt mit a die Nebenbedingung $|u|_a \geq k$. Mit anderen Worten ist jedem Knoten des Suchgraphen ein kommutatives Bild eines Präfixes einer möglichen Lösung zugeordnet und mit dem Simplexalgorithmus kann berechnet werden, wie lang eine Lösung mindestens sein muss, die ein solches Präfix besitzt. Damit kann in jedem Knoten abgelesen werden, wie lang eine Lösung mindestens sein muss, deren Pfad durch diesen Knoten verläuft. Sollen nur Lösungen bis zu einer bestimmten Länge gesucht werden, kann diese Information benutzt werden, um Pfade nicht weiter zu betrachten.

Eine speziellere Überlegung betrachtet nur die Länge des Überhangs in einem Knoten: Pro Schritt kann diese Länge nur um einen konstanten Wert c vergrößert oder verkleinert werden. Befindet sich eine mit M beschränkte Suche auf Tiefe d, kann sich die Länge des Überhangs um maximal $c(M - d)$ ändern. Die zu erreichende Länge ist bekannt: Es muss die Länge des Überhangs in f erreicht werden. Also kann auch so eine minimale Lösungslänge für Lösungen auf einem bestimmten Pfad ermittelt werden. Diese Abschätzung verwendet nicht so viel Information wie die mit dem Simplexalgorithmus und ist deshalb potentiell schlechter, sie ist aber sehr leicht und extrem effizient implementierbar: Es wird lediglich die Länge des Überhangs eines Knotens benötigt, ein Wert, der mit konstantem Aufwand aktuell gehalten werden kann, gemessen sowohl bezüglich der Pfadlänge als auch bezüglich der Länge des Überhangs in einem Knoten. Darüber hinaus liegt dieser Wert mit hoher Wahrscheinlichkeit ohnehin vor.

Beispiel: Für die Instanz $I_{16b} = \| \begin{smallmatrix} 0 & 001 \\ 001 & 1 \end{smallmatrix} \begin{smallmatrix} 1 \\ 0 \end{smallmatrix} \|$ ist $\psi(\mathrm{Eq}(I_{16b})) \subseteq \mathrm{span}\{(1,1,1)\}$, das heißt in jeder Lösung kommt jede Regel gleich oft vor. Eine Lösung u hat also mindestens die Länge $m = 3 \cdot \max\{x_a, x_b, x_c\}$, wenn x_a, x_b und x_c die *b*

Anzahlen der Regeln sind, die angewendet wurden, um zu einem bestimmten Knoten x im Suchgraphen zu gelangen, der auf dem Pfad von u liegt. Wenn m größer als 75 ist, dann kann auf diesem Pfad keine Lösung der Länge höchstens 75 gefunden werden. In $\rightarrowtail_{I_{16b}}$ gibt es von $(\varepsilon, \varepsilon)$ ausgehend einen mit *aacaaca aacaacaaacaacaacaaacaacaaacaa* markierten Pfad der Länge 36. Da bereits 26-mal die Regel a benutzt wurde, kann dieser Pfad nicht zu einer Lösung der Länge höchstens 75 führen und muss nicht mehr betrachtet werden. Der dadurch abgeschnittene Teilgraph umfasst allein für diesen Knoten 86 000 Pfade mit 364 371 Knoten.

Insgesamt ergibt sich im Vergleich mit 30d für die Anzahlen der Pfade und Knoten bis zur Länge 75:

	Pfade		Knoten	
	30d	33b	30d	33b
I_{16b}	109 918 385	591 808	485 555 235	2 517 402
I^{R}_{16b}	16 105 042	1 388 773	78 067 826	6 557 540

Das Abschneiden hat die Suchgraphen von I_{16b} und der gespiegelten Instanz I^{R}_{16b} unterschiedlich stark verkleinert. Insbesondere hat sich das Größenverhältnis umgekehrt. ⌟

a

Ausschließen

In 32a benutzen wir die Information aus 18c um Pfade im Suchgraphen als unproduktiv zu erkennen. Dabei handelt es sich um Informationen über die *Beschriftung* eines Pfades. Jetzt wollen wir versuchen, einen Pfad anhand des *Überhangs* in einem Knoten als unproduktiv zu erkennen.

b

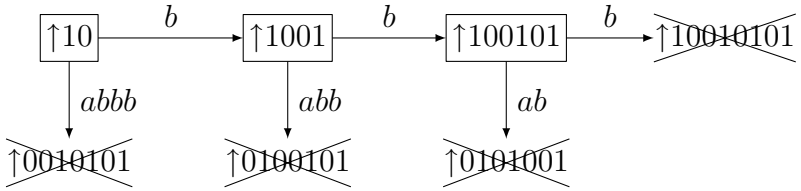
Falsches Bild

Die erste Idee ist sehr einfach: Eine Regel $(h(a), g(a))$ kann nur dann auf einen Überhang x angewendet werden, wenn $x_\uparrow$ und $g(a)$ oder $x_\downarrow$ und $h(a)$ vergleichbar sind. Daraus folgt unmittelbar, dass die ↑-Komponenten der Überhänge aus $B^{*\uparrow}$ auch in $L^\uparrow = g(A^*)\,\mathrm{pref}(g(A^*))$ enthalten sein müssen. Analog müssen die ↓-Komponenten der Überhänge aus $B^{*\downarrow}$ auch in $L^\downarrow = h(A^*)\,\mathrm{pref}(h(A^*))$ enthalten sein. Hier ist etwas Vorsicht geboten, denn es ist durchaus möglich, dass ein Überhang nicht in $L^\uparrow$ enthalten ist, aber dennoch auf dem Weg zu einer Lösung liegt oder gar selbst bereits Lösung ist. Um sicher zu gehen verwenden wir als Ausschlussmenge die Menge $\mathrm{Ex}^{\mathrm{B}}(I) = (\complement(L^\uparrow)B^{>|f_\downarrow|})^\uparrow \cup (\complement(L^\downarrow)B^{>|f_\uparrow|})^\downarrow$ für die gilt

c **Lemma.** $x \in \mathrm{Ex}^{\mathrm{B}}(I) \implies \{a \mid x \xrightarrow{a}_I \overline{f}\} = \emptyset$.

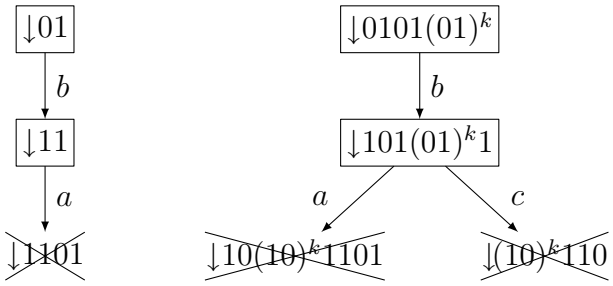
Beweis: Wir betrachten nur den Fall $x \in \complement(L^{\uparrow})B^{>|f_{\downarrow}|} \times \{\varepsilon\}$, das heißt $x_{\uparrow} = uv$ mit $u \notin L^{\uparrow}$ und $|v| > |f_{\downarrow}|$. Es gibt nach Konstruktion von $L^{\uparrow}$ keine Folge von Regelanwendungen die u komplett überdeckt. Also gilt für jedes a mit $x \overset{a}{\rightarrowtail}_I y$ stets $|y_{\uparrow}| > |f_{\downarrow}|$. □ *a*

Beispiel: In 30b wurde gezeigt, dass der Suchgraph für die Instanz $I_{15c} = |\begin{smallmatrix}101\\\varepsilon\end{smallmatrix}|\begin{smallmatrix}0\\10\end{smallmatrix}\begin{smallmatrix}01\\\varepsilon\end{smallmatrix}|\begin{smallmatrix}\varepsilon\\1010\end{smallmatrix}|$ unendlich groß ist. Insbesondere ist der Teilgraph unendlich groß, der am Knoten $(10, \varepsilon)$ beginnt. Die Ausschlussmenge für ↑-Komponenten ist die Menge $(10)^*\{0,11\}\{0,1\}^{>4}$ und mit ihrer Hilfe ergibt sich der Suchgraphausschnitt *b*



Hier sind die Knoten durchgestrichen, die in der Ausschlussmenge liegen und daher nicht weiter betrachtet werden müssen. ⌟

Beispiel: Die Instanz $I_{35c} = \|\begin{smallmatrix}\varepsilon\\01\end{smallmatrix}\begin{smallmatrix}0\\1\end{smallmatrix}\begin{smallmatrix}10\\0\end{smallmatrix}\|$ hat ebenfalls einen unendlich großen Suchgraphen, da Regel a stets angewendet werden kann. Genauer besteht der Suchgraph aus dem unendlichen Pfad a^{ω}, der zu den Überhängen $(\varepsilon, (01)^k)$ führt, an denen sich jeweils unendlich große Teilgraphen anschließen. Die Ausschlussmenge für ↓-Komponenten ist die Menge $\{0,1\}^*11\{0,1\}^{>0}$ und damit ergibt sich *c*



Also muss im Suchgraphen von I_{35c} nur der einzelne unendliche Pfad a^{ω} betrachtet werden. ⌟

In den beiden Beispielen wurden jeweils unendlich große Teilgraphen ausgeschlossen. Es lohnt sich aber auch bei nichtlöschenden Instanzen, Überhänge mit falschem Bild nicht weiter zu betrachten. Die ausgeschlossenen Teilgraphen sind dann zwar nicht mehr unendlich aber immer noch beliebig groß.

a **Falsche Fortsetzung**

Wir identifizieren Überhänge, die die Anwendung bestimmter Regeln erzwingen und die bei Anwendung der erzwungenen Regeln nicht auf dem Pfad zu einer Lösung liegen.

Die zweite Bedingung ist leicht zu erfüllen: Wenn $R \subseteq A$ eine Menge von *unbalancierten* Regeln ist, dann kann durch Anwendung dieser Regeln ein entsprechend balancierter Überhang nicht verkürzt werden. Sei ohne Beschränkung der Allgemeinheit R eine Menge von Regeln, bei denen sämtliche h-Bilder mindestens so lang sind wie die korrespondierenden g-Bilder, das heißt, es gilt $\forall r \in R : |h(r)| \geq |g(r)|$ und $x = (x_\uparrow, \varepsilon)$ sei ein $\uparrow$-Überhang. Dann gilt für alle y mit $x \xrightarrow{r}_I y$ für ein $r \in R$ stets $|y_\uparrow| \geq |x_\uparrow|$. Der Überhang wird nicht kürzer, wenn eine Regel aus R angewendet wird. Wenn nun außerdem $|x_\uparrow| > |f|$ ist, dann muss auf jedem Weg von x zu $\overline{f}$ mindestens ein Mal eine Regel aus $A \setminus R$ angewendet werden. Wir nennen Mengen unbalancierter Regeln $\uparrow$- *oder* $\downarrow$*-unbalanciert*, je nachdem, ob $\uparrow$- oder $\downarrow$-Überhänge bei Anwendung dieser Regeln nicht verkürzt werden. Die Mengen aller $\uparrow$- und aller $\downarrow$-unbalancierter Mengen von Regeln einer Instanz I bezeichnen wir mit $\mathrm{Ub}(I, \uparrow)$ und $\mathrm{Ub}(I, \downarrow)$.

Wir konstruieren auf zwei verschiedene Arten Überhänge, die die Anwendung bestimmter Regeln erzwingen. Dazu sei ohne Beschränkung der Allgemeinheit $R \subseteq A$ eine Menge von $\uparrow$-unbalancierten Regeln. Zunächst

b **Lemma.** *Die Menge* $\mathrm{Ex}^{\mathrm{F}}(I, R, \uparrow) = B^{>\mathrm{wd}(I)} \cap \complement(B^* g(A \setminus R) B^* \rhd h(R^*))$ *ist eine reguläre Ausschlussmenge für* $\uparrow$*-Überhänge.*

c *Beweis:* Es gilt

$$\begin{aligned}\complement(B^* g(A \setminus R) B^* \rhd h(R^*)) &= \complement(\{z \mid \exists y \in h(R^*) : zy \in B^* g(A \setminus R) B^*\}) \\ &= \{z \mid \forall y \in h(R^*) : zy \notin B^* g(A \setminus R) B^*\}.\end{aligned}$$

Sei $x = (x_\uparrow, \varepsilon)$ mit $x_\uparrow \in \mathrm{Ex}^{\mathrm{F}}(I, R, \uparrow)$, dann kann keine der Regeln aus $A \setminus R$ auf x angewendet werden, da einerseits $x_\uparrow$ so lang ist, dass ein g-Bild komplett (also nichtüberlappend) angewendet werden muss und andererseits $x_\uparrow$ nicht mit einem g-Bild einer Regel aus $A \setminus R$ beginnt. Wird jedoch eine der Regeln aus R auf x angewendet, dann entsteht wieder ein Überhang aus $\mathrm{Ex}^{\mathrm{F}}(I, R, \uparrow)$: Da R unbalanciert ist, wird der Überhang nicht kürzer und nach Definition ist $\mathrm{Ex}^{\mathrm{F}}(I, R, \uparrow)$ abgeschlossen gegenüber Konkatenation mit $h(R^*)$ und gegenüber Löschen von Präfixen. □

Wir definieren analog die Ausschlussmenge $\mathrm{Ex}^{\mathrm{F}}(I, R, \downarrow)$ für Mengen $\downarrow$-unbalancierter Regeln und $\downarrow$-Überhänge und schließlich die Ausschlussmenge

$$\mathrm{Ex}^{\mathrm{F}}(I) = \bigcup_{R\in\mathrm{Ub}(I,\uparrow)} \mathrm{Ex}^{\mathrm{F}}(I,R,\uparrow)^{\uparrow} \cup \bigcup_{R\in\mathrm{Ub}(I,\downarrow)} \mathrm{Ex}^{\mathrm{F}}(I,R,\downarrow)^{\downarrow}. \qquad a$$

Für die Menge $\mathrm{Ex}^{\mathrm{F}}(I)$ geben wir uns also eine Menge unbalancierter Regeln vor und konstruieren eine reguläre Menge, die die Anwendung von Regeln aus R erzwingt. Da R unbalanciert ist müssen Elemente der Menge $\mathrm{Ex}^{\mathrm{F}}(I)$ bei der Suche nach einer Lösung nicht weiter betrachtet werden.

Die zweite naheliegende Methode geht ebenfalls von einer Menge R unbalancierter Regeln aus und testet, ob Regeln aus $A \setminus R$ überhaupt irgendwann anwendbar sind, wenn vorher nur Regeln aus R angewendet wurden.

Dazu stellen wir zunächst eine Obermenge von Überhängen bereit, die entstehen, wenn nur Regeln aus R angewendet werden. Wenn x ein $\uparrow$-Überhang im Suchgraphen ist, der von s aus nur mit Regeln aus R erreichbar ist, das heißt es gilt $s \overset{a}{\rightarrowtail}_I x$ mit $a \in R^*$, dann ist $x_\uparrow$ in

$$\mathrm{Cf}(I,R,\uparrow) = s_\uparrow h(R^*) \lhd s_\downarrow g(R^*) = \{z \mid \exists y \in s_\downarrow g(R^*) : yz \in s_\uparrow h(R^*)\} \qquad b$$

enthalten. Für eine Regel $r \in A \setminus R$ stellt sich nun die Frage, ob r auf einen Überhang aus $\mathrm{Ex}^{\mathrm{A}}(I,R,\uparrow) = B^{>\mathrm{wd}(I)} \cap \mathrm{Cf}(I,R,\uparrow)$ anwendbar ist. Das ist dann der Fall, wenn es Wörter in $\mathrm{Ex}^{\mathrm{A}}(I,R,\uparrow)h(r)$ gibt, die mit $g(r)$ vergleichbar sind. Daraus folgt, dass r *nicht* anwendbar ist, wenn gilt

$$\mathrm{Ex}^{\mathrm{A}}(I,R,\uparrow)h(r) \lhd g(r) = g(r) \lhd \mathrm{Ex}^{\mathrm{A}}(I,R,\uparrow)h(r) = \emptyset. \qquad c$$

Wenn (37c) gilt, dann ist $\mathrm{Ex}^{\mathrm{A}}(I,R,\uparrow)$ eine Ausschlussmenge. Erneut sichert die Längenbedingung den Fall, dass bereits in $\mathrm{Cf}(I,R,\uparrow)$ die Lösung enthalten ist.

Zusammenfassend gilt für beide Techniken, dass alle beteiligten Mengen regulär sind und dass alle notwendigen Entscheidungen effektiv und effizient getroffen werden können. Zur Konstruktion der Mengen werden alle unbalancierten Teilmengen von Regeln aufgezählt. Für $\mathrm{Ex}^{\mathrm{F}}(I)$ werden dann Mengen von Überhängen gemäß (37a) direkt ausgerechnet. Die Mengen $\mathrm{Cf}(I,R)$ aus (37b) werden nur dann zur Ausschlussmenge, wenn die Bedingung (37c) erfüllt ist, wenn also keine der Regeln aus $A \setminus R$ anwendbar ist.

Beispiel: In 30c wird gezeigt, dass der Suchgraph der Instanz $I_{15d} = \| \begin{smallmatrix} abb & bb & cbbb \\ a & bbb & c \end{smallmatrix} \|$ aus einem endlichen Teil und dem unendlichen Pfad b^ω besteht, der Überhänge aus $b^{*\downarrow}$ erzeugt. Die Menge $\{b\}$, die nur die Regel b enthält, d

ist $\downarrow$-unbalanciert. Daraus folgt mit $B = \{a, b, c\}$

$$\begin{aligned}\mathrm{Ex}^{\mathrm{F}}(I_{15d}, \{b\}, \downarrow) &= B^{>4} \cap \complement(B^* h(\{a, c\}) B^* \rhd g(b^*)) \\ &= B^{>4} \cap \complement(B^* \{abb, cbbb\} B^* \rhd (bbb)^*) \\ &= B^{>4} \cap \complement(\{z \mid z(bbb)^* \cap B^* \{abb, cbbb\} B^* \neq \emptyset\}) \\ &= b^{>4}\end{aligned}$$

und

$$\begin{aligned}\mathrm{Ex}^{\mathrm{A}}(I_{15d}, \{b\}, \downarrow) &= B^{>\mathrm{wd}(I)} \cap g(b^*) \lhd h(b^*) \\ &= B^{>\mathrm{wd}(I)} \cap (bbb)^* \lhd (bb)^* \\ &= b^{>4}.\end{aligned}$$

Die Menge $\mathrm{Ex}^{\mathrm{A}}(I_{15d}, \{b\}, \downarrow)$ darf als Ausschlussmenge verwendet werden, da

$$\begin{aligned}&\mathrm{Ex}^{\mathrm{A}}(I_{15d}, \{b\}, \downarrow) g(a) \lhd h(a) = b^* bbbba \lhd abb = \{z \mid abbz \in b^* bbbba\} = \\ &h(a) \lhd \mathrm{Ex}^{\mathrm{A}}(I_{15d}, \{b\}, \downarrow) g(a) = abb \lhd b^* bbbba = \{z \mid b^* bbbbaz \ni abb\} = \emptyset\end{aligned}$$

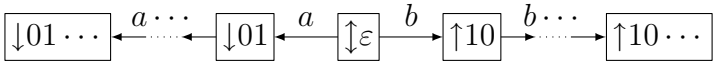
und

$$\begin{aligned}&\mathrm{Ex}^{\mathrm{A}}(I_{15d}, \{b\}, \downarrow) g(c) \lhd h(c) = b^* bbbbc \lhd cbbb = \{z \mid cbbbz \in b^* bbbbc\} = \\ &h(c) \lhd \mathrm{Ex}^{\mathrm{A}}(I_{15d}, \{b\}, \downarrow) g(c) = cbbb \lhd b^* bbbbc = \{z \mid b^* bbbbcz \ni cbbb\} = \emptyset\end{aligned}$$

gelten. Keine der Regeln a und c kann auf Überhänge aus $\mathrm{Ex}^{\mathrm{A}}(I_{15d}, \{b\}, \downarrow)^{\downarrow}$ angewendet werden.

Hier berechnen beide Methoden, dass der unendliche Pfad bei der Suche nach einer Lösung nicht betrachtet werden muss. ⌟

a **Beispiel:** Die Instanz $I_{38a} = \| \begin{smallmatrix} 01 \\ 0101 \end{smallmatrix} \begin{smallmatrix} 1010 \\ 10 \end{smallmatrix} \|$ erzeugt den Suchgraphen



mit den beiden unendlichen Pfaden a^ω und b^ω. Die Menge $R = \{b\}$ ist eine $\uparrow$-unbalancierte Menge von Regeln. Nun gilt mit $B = \{0, 1\}$

$$\begin{aligned}\mathrm{Ex}^{\mathrm{F}}(I_{38a}, \{b\}, \uparrow) &= B^{>4} \cap \complement(B^* g(a) B^* \rhd h(b^*)) \\ &= B^{>4} \cap \complement(B^* 01 B^* \rhd (10)^*) \\ &= B^{>4} \cap \complement(\{z \mid \exists y \in (10)^* : zy \in B^* 01 B^*\}) \\ &= \emptyset.\end{aligned}$$

Das „Problem“ ist hier, dass 1010 sowohl in $(10)^*$ als auch in B^*01B^* liegt, es kann also an jedes z einfach das Wort 1010 angehängt werden und dadurch wird immer ein Wort aus B^*01B^* erzeugt. Andererseits gilt

$$\begin{aligned}\mathrm{Ex}^{\mathrm{A}}(I_{38a}, \{b\}, \uparrow) &= B^{>4} \cap h(b^*) \lhd g(b^*) \\ &= B^{>4} \cap (1010)^* \lhd (10)^* \\ &= B^{>4} \cap \{z \mid \exists y \in (10)^* : yz \in (1010)^*\} \\ &= 101010(10)^*\end{aligned}$$

und die Menge $\mathrm{Ex}^{\mathrm{A}}(I_{38a}, \{b\}, \uparrow)$ darf als Ausschlussmenge verwendet werden, da gilt:

$$\begin{aligned}\mathrm{Ex}^{\mathrm{A}}(I_{38a}, \{b\}, \uparrow)h(a) \lhd g(a) &= 101010(10)^*01 \lhd 0101 \\ &= \{z \mid 0101z \in 101010(10)^*01\} = \\ g(a) \lhd \mathrm{Ex}^{\mathrm{A}}(I_{38a}, \{b\}, \uparrow)h(a) &= 0101 \lhd 101010(10)^*01 \\ &= \{z \mid 101010(10)^*01z \ni 0101\} = \emptyset.\end{aligned}$$

Daraus folgt, dass der Pfad b^ω nicht zu einer Lösung führt. Eine symmetrische Rechnung erlaubt diesen Schluss auch für den Pfad a^ω. ⌟

Beispiel: Manchmal schließt die Menge $\mathrm{Ex}^{\mathrm{F}}(I)$ mehr Überhänge von der weiteren Betrachtung aus als die Menge $\mathrm{Ex}^{\mathrm{A}}(I)$. In $\mathrm{Ex}^{\mathrm{A}}(I)$ sind nur Überhänge kodiert, die nur durch Anwendung von Regeln aus R entstehen, nicht jedoch Überhänge, die zunächst auch Regeln aus $A \setminus R$ verwenden, aber ab einem gewissen Punkt nur noch Regeln aus R zulassen. Ein Prototyp für ein solches Verhalten ist die Instanz $I_{39a} = |\,{}^{\varepsilon}_{1}\,|\,{}^{0}_{00}\,{}^{1}_{00}\,|\,{}^{\varepsilon}_{\varepsilon}\,|$ mit dem unendlichen Pfad $s \overset{b}{\rightarrowtail}_{I_{39a}} (\varepsilon, 00) \overset{a}{\rightarrowtail}_{I_{39a}} (\varepsilon, 000) \overset{a}{\rightarrowtail}_{I_{39a}} \cdots$, der initial die „falsche“ Regel b anwendet. Es gilt $\mathrm{Ex}^{\mathrm{A}}(I) = \emptyset$, da nur durch Anwendung der Regel b überhaupt keine Überhänge erzeugt werden können. Mit $B = \{0, 1\}$ gilt aber *a*

$$\begin{aligned}\mathrm{Ex}^{\mathrm{F}}(I_{39a}, \{a\}, \downarrow) &= B^{>2} \cap \complement(B^*h(b)B^* \rhd g(a^*)) \\ &= B^{>2} \cap \complement(B^*1B^* \rhd (00)^*) \\ &= B^{>2} \cap \complement(\{z \mid \exists y \in (00)^* : zy \in B^*1B^*\}) \\ &= B^{>2} \cap \complement(B^*1B^*) \\ &= 0000^*.\end{aligned}$$

Die Menge $\mathrm{Ex}^{\mathrm{F}}(I_{39a}, \{a\}, \downarrow)$ schließt den Pfad ba^ω also aus. ⌟

Einfangen *b*

Die in 34a gefundenen Überhänge erfordern die Anwendung bestimmter Regeln und führen dabei nicht zu einer Lösung. Anders betrachtet handelt

es sich um Überhänge, von denen aus der Überhang $\overline{f}$ nicht erreichbar ist. Allgemeiner identifizieren wir eine Menge M von Überhängen, für die die Hülle $\rightarrowtail_I^\star(M)$ nicht $\overline{f}$ enthält. Das nur bestimmte Regeln auf Elemente aus M angewendet werden können, ist bei dieser Überlegung ohne Bedeutung: Die Menge M kann als Ausschlussmenge verwendet werden.

Diese Idee lässt sich automatisiert umsetzen, allerdings gilt es zwei Probleme zu bewältigen: Erstens sind die Hüllen von $\rightarrowtail_I$ im Allgemeinen unendlich groß und zweitens ist unklar, woher die Menge M stammen soll.

Die Lösung für beide Probleme besteht darin, eine reguläre Menge M *zu raten*, genauer eine Beschreibung einer regulären Menge in Form zweier endlicher Automaten. Wir benötigen zwei Automaten, da $\uparrow$- und $\downarrow$-Überhänge zweckmäßigerweise getrennt dargestellt werden.

Ein einzelner Schritt mit $\rightarrowtail_I$ über M wird wie folgt berechnet: Zunächst berechnen wir die $\uparrow$- und die $\downarrow$-Überhänge, die von $\uparrow$- und von $\downarrow$-Überhängen erreicht werden, also *ohne* Änderung der Position. Dazu betrachten wir die Regeln von I und von $\overline{I}$ als Regeln eines Tag-Systems. Reguläre Sprachen sind gegenüber Schritten mit Tag-Systemen effektiv abgeschlossen, denn mit einer Regel (l, r) kann aus M genau die Menge $l \lhd Mr$ erreicht werden. Nun müssen noch die $\uparrow$- und die $\downarrow$-Überhänge berechnet werden, die von $\downarrow$- und von $\uparrow$-Überhängen erreicht werden, also *mit* Änderung der Position. Es gibt nur endlich viele solcher Übergänge. Wir zählen sie alle auf und nehmen $v = (\varepsilon, v_\downarrow)$ in die Menge der $\downarrow$-Überhänge auf, wenn $u \overset{a}{\rightarrowtail}_I v$ für ein $u \in M$ mit $u = (u_\uparrow, \varepsilon)$ und $a \in A$ gilt und analog für Übergänge von $\downarrow$ nach $\uparrow$.

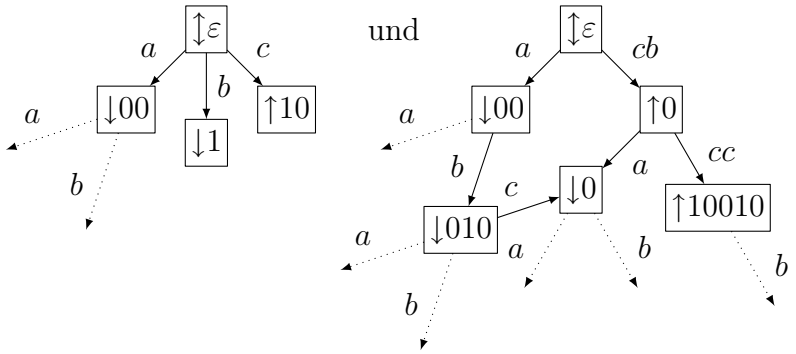
Nach Berechnung eines Schrittes mit $\rightarrowtail_I$ über M testen wir, ob $M \subseteq \rightarrowtail_I(M)$ ist. Ist das der Fall, dann ist M gegenüber $\rightarrowtail_I$ abgeschlossen und wir können M, falls $\overline{f}$ nicht enthalten ist, als Ausschlussmenge verwenden. Wenn $M \not\subseteq \rightarrowtail_I(M)$, dann fahren wir mit $M \cup \rightarrowtail_I(M)$ fort und nähern uns der Hülle $\rightarrowtail_I^\star(M)$ weiter an. Bei falscher Wahl von M kann sich eine Endlosschleife ergeben, das heißt, der gesamte Prozess darf nur eine gewisse maximale Anzahl von Schritten laufen. Alternativ kann auch eine beliebige andere Ressource begrenzt werden, zum Beispiel die Zeit oder die maximale Größe der Automaten.

Bleibt die Frage, wie M *konkret* geraten werden kann? Klar ist, dass M in irgendeiner Weise von I abhängen sollte und klar ist ebenfalls, dass wir nur über Heuristiken sprechen. Eine Heuristik, die in der Praxis recht erfolgreich ist, berechnet zunächst einen Anfangsteil des Suchgraphen, bestimmt alle darin vorkommenden Präfixe, Suffixe und Infixe einer bestimmten Minimallänge und rät dann, dass der gesamte Suchgraph nur Überhänge enthält, die solche Präfixe, Suffixe und Infixe haben.

Es sei erwähnt, dass natürlich *jede* unlösbare Instanz mit der Menge $\rightarrowtail_I^\star(\{s\})$ eingefangen werden kann. Allerdings kann $\rightarrowtail_I^\star(\{s\})$ kompliziert sein,

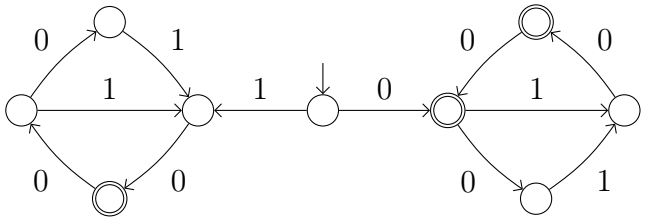
insbesondere nicht regulär. Die Regularität ist für das skizzierte Verfahren aber essenziell, da wir eine Sprachklasse benötigen, die gegenüber Schritten mit Tag-Systemen abgeschlossen ist und für die $\subseteq$ entscheidbar ist.

Beispiel: Die Suchgraphen der Instanzen $I_{41a} = \| \begin{smallmatrix} 0 \\ 000 \end{smallmatrix} \begin{smallmatrix} 0 \\ 01 \end{smallmatrix} \begin{smallmatrix} 010 \\ 0 \end{smallmatrix} \|$ und I_{41a}^{R} sind die Graphen

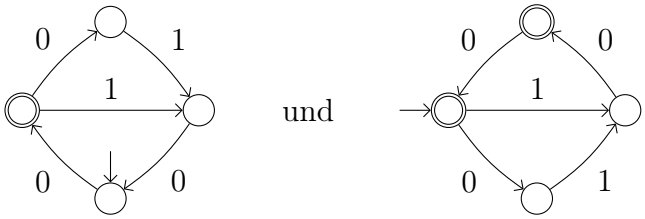


mit einem unendlichen ↓-Teilgraphen im Graphen von I_{41a} und einem unendlichen ↑-Pfad im Graphen von I_{41a}^{R}.

Betrachten wir die Menge $M^{\uparrow}$ mit $M = 10\{010, 0010\}^* \cup \{010, 0010\}^*0 \cup \{010, 0010\}^+$, die der Automat



akzeptiert. Offensichtlich können nur die Regeln b und c auf $M^{\uparrow}$ angewendet werden, was zu den beiden Automaten



führt, die beide jeweils eine Teilmenge von M akzeptieren. Daraus folgt, dass $M^{\uparrow}$ keine neuen ↑-Überhänge erzeugt und also als Ausschlussmenge verwendet werden kann. Die Graphen von I_{41a} und von I_{41a}^{R} enthalten nun beide nur noch ↓-Überhänge. Deshalb kann I_{41a} aus Symmetriegründen keine echte Lösung besitzen. ⌟

a **Beispiel:** In 16b wird erwähnt, dass die Instanz $J_{16b} = \| \begin{smallmatrix} 0 \\ 001 \end{smallmatrix} \begin{smallmatrix} 001 \\ 1 \end{smallmatrix} \begin{smallmatrix} 10 \\ 0 \end{smallmatrix} \|$ zu den schwierigsten in $\mathcal{P}(3,3)$ gehört. Mit Hilfe des Einfangens können wir sagen, dass J_{16b} unlösbar ist, weil mit $M^\uparrow = \{0,1\}^*\{100010101, 110000101\}\{0,1\}^*$ eine Ausschlussmenge für die gespiegelte Instanz $J_{16b}^\mathrm{R} = \| \begin{smallmatrix} 0 \\ 100 \end{smallmatrix} \begin{smallmatrix} 100 \\ 1 \end{smallmatrix} \begin{smallmatrix} 01 \\ 0 \end{smallmatrix} \|$ ist und weil jeder Pfad im Suchgraphen von J_{16b}^R nach spätestens 30 Schritten durch $M^\uparrow$ verläuft. Ein Beweis der Abgeschlossenheit von $M^\uparrow$ gegenüber Schritten mit $\rightarrowtail_{J_{16b}^\mathrm{R}}$ argumentiert über 23 Regelanwendungen mit einigen Zwischenreduktionen, das heißt, nach spätestens 23 bestimmten unvermeidbaren Schritten, die unter Umständen noch durch Zwischenschritte ergänzt sind, wiederholt sich ein Wort aus $M^\uparrow$.

Die Menge $M^\uparrow$ wurde *nicht* automatisch, sondern vom Autor durch intensive Beschäftigung mit dem Suchgraphen gefunden. In Kenntnis von $M^\uparrow$ wurde ein Automat mit 1528 Zuständen generiert, der gegenüber Schritten mit $\rightarrowtail_{J_{16b}^\mathrm{R}}$ abgeschlossen ist und einen automatisch überprüften Beweis der Unlösbarkeit von J_{16b} darstellt. ⌟

b

Maskieren

Beim Maskieren wird versucht, Überhänge allein anhand ihrer Position als nicht produktiv zu erkennen. Die Grundidee ist wie immer einfach: Wenn bei einer Instanz $\overline{f}$ zum Beispiel ein ↓-Überhang ist und diese Instanz keine Schritte erlaubt, die ↑- in ↓-Überhänge überführen, dann kann kein ↑-Überhang auf dem Pfad einer Lösung liegen. Wie bereits in 39b erwähnt, können alle endlich vielen möglichen Übergänge mit Änderung der Position effektiv aufgezählt werden. Mit anderen Worten: Es ist entscheidbar, ob Übergänge mit Änderung der Position möglich sind. Instanzen wie $\| \begin{smallmatrix} 0 \\ 00 \end{smallmatrix} \begin{smallmatrix} 001 \\ 10 \end{smallmatrix} \|$ können so schnell als unlösbar erkannt werden, doch wir wollen wesentlich mehr aus dieser Idee machen.

Folgerung 28e sagt, dass ein Schritt von einem ↑- zu einem ↓-Überhang im Suchgraphen von I, stets einem Schritt von einem ↓- zu einem ↑-Überhang im Suchgraphen von I^R entspricht. Zusammen mit Lemma 28f folgt sofort, dass es nicht ausreicht, wenn in I Schritte mit Änderung der Position möglich sind, sondern dass gleichzeitig in I^R Schritte mit entgegengesetzter Änderung der Position möglich sein müssen. Es gilt genauer, dass auf dem Weg zu einer Lösung $x \overset{a}{\rightarrowtail}_I y$ für ein $a \in A$ nur dann als Übergang mit Änderung der Position zu berücksichtigen ist, wenn einerseits $\overline{y}^\mathrm{R} \overset{a}{\rightarrowtail}_{I^\mathrm{R}} \overline{x}^\mathrm{R}$ ebenfalls ein möglicher Übergang mit Änderung der Position ist *und* wenn andererseits $\overline{y}^\mathrm{R}$ überhaupt von f^R aus durch Schritte mit $\rightarrowtail_{I^\mathrm{R}}$ erreichbar ist. Natürlich muss ebenfalls x von s durch Schritte mit $\rightarrowtail_I$ erreichbar sein.

Aus dieser Überlegung extrahieren wir zunächst einige Begriffe: Ein Überhang x heißt *erreichbar*, wenn $s \overset{w}{\rightarrowtail}_I x$ für ein $w \in A^*$ gilt und er heißt

überdeckbar, wenn $x \overset{w}{\rightarrowtail}_I \overline{f}$ gilt. Es folgt, dass x genau dann in I erreichbar ist, wenn $\overline{x}^{\mathrm{R}}$ in I^{R} überdeckbar ist. Im Allgemeinen ist nicht entscheidbar, ob ein Überhang erreichbar oder überdeckbar ist, da diese Fragen wieder dem Originalproblem entsprechen. Im beschränkten Fall können wir aber sehr wohl manchmal eine positive oder negative Antwort geben. Wir müssen also immer eine Schranke angeben, wenn wir einen dieser Begriffe verwenden.

Ein Überhang x heißt *Wechselüberhang*, wenn er erreichbar ist und es einen überdeckbaren Überhang y an entgegengesetzter Position gibt, mit $x \overset{a}{\rightarrowtail}_I y$ für ein $a \in A$. Der Wechselübergang x heißt darüber hinaus *echter Wechselübergang*, wenn von y aus nicht nur Wechselübergänge erreichbar sind. Ein Wechselübergang kann die Position zwar wechseln, könnte aber nach einer gewissen Anzahl von Schritten zu einem Rückwechsel gezwungen sein. Ein echter Wechselübergang gestattet das Verweilen auf der entgegengesetzten Position. Schließlich nennen wir Überhänge x *Start- oder Endüberhang*, wenn $s \overset{a}{\rightarrowtail}_I x$ oder wenn $x \overset{a}{\rightarrowtail}_I \overline{f}$ für ein $a \in A$ gilt. Start- und Endüberhänge sind die ersten und letzten Überhänge nach s und vor $\overline{f}$.

Mit Hilfe dieser Begriffe formulieren wir die Methode des Maskierens: Zunächst geben wir eine Schranke für die Begriffe erreichbar und überdeckbar vor. Nun berechnen die überdeckbaren Start-, die erreichbaren End- und echten Wechselüberhänge von I und I^{R}. In beiden Instanzen berechnen wir daraus die Einschränkung an die gültigen Positionen der Überhänge auf einem Lösungspfad und übernehmen für beide Instanzen die stärkere Einschränkung. Fanden tatsächlich Einschränkungen statt, beginnen wir erneut.

Diese Formulierung bedarf eines Kommentars: Eine Einschränkung an die gültigen Positionen ist eine Position, die als ungültig erkannt wurde. Es ist möglich, dass eine Position in I^{R} als ungültig aber die entgegengesetzte nicht als ungültig in I erkannt wird. Nach 28e dürfen wir sie dennoch als ungültig markieren. Schließlich muss der Vorgang nach tatsächlicher Einschränkung wiederholt werden, da mit dem neuen Wissen eventuell mehr Überhänge als nicht erreichbar oder als nicht überdeckbar erkannt werden können. Da nur endlich viele Einschränkungen möglich sind, wird das gesamte Verfahren in jedem Fall terminieren.

Eine weitere Verschränkung ist sinnvoll, denn die beim Maskieren gewonnenen Informationen haben Auswirkungen auf die Verfahren Ausschließen durch falsche Fortsetzung aus 36a und Einfangen aus 39b.

Beispiel: Die Instanz $I_{43a} = \| \begin{smallmatrix} 0 \\ 001 \end{smallmatrix} \begin{smallmatrix} 001 \\ 00 \end{smallmatrix} \begin{smallmatrix} 001 \\ 1 \end{smallmatrix} \|$ hat, zunächst ohne Berücksichtigung der Erreichbarkeit und der Überdeckbarkeit, die Startüberhänge $\downarrow 01$ und $\uparrow 1$, den Endübergang $\downarrow 00$ und den möglichen Wechsel von $\uparrow 0$ zu $\downarrow 1$. Daraus können keine Einschränkungen abgeleitet werden können. Nun ist aber $\downarrow 01$ nicht überdeckbar, da zwar $\downarrow 01 \overset{a}{\rightarrowtail}_{I_{43a}} \downarrow 1001$, aber $\downarrow 1001$ keine weiteren *a*

Übergänge zulässt. Der Überhang ↓1 ist ebenfalls nicht überdeckbar, da alle h-Bilder mit 0 beginnen.

Daraus folgt, dass eine Lösung einen Pfad von ↑1 nach ↓00 beschreibt, aber ohne dabei die Position zu ändern. Das ist nicht möglich, also hat I_{43a} keine gültige Position und keine echte Lösung. ⌟

a **Beispiel:** Bei der Instanz $I^{\mathrm{R}}_{44a} = \|\,{}^{0}_{000}\,{}^{00}_{110}\,{}^{110}_{1}\,\|$ muss ein Pfad von ↓00 zu ↑00 gefunden werden, unter Verwendung des Übergangs $\downarrow 1 \overset{c}{\rightarrowtail}_{I^{\mathrm{R}}_{44a}} \uparrow 0$. Alle beteiligten Überhänge sind erreichbar oder sind überdeckbar, doch ↓1 ist zwar Wechselüberhang, aber kein *echter* Wechselübergang, denn von ↑0 gibt es nur die Fortsetzung $\uparrow 0 \overset{a}{\rightarrowtail}_{I^{\mathrm{R}}_{44a}} \downarrow 0$. Es handelt sich nur scheinbar um einen Wechsel der Position. Daraus folgt, dass I_{44a} keine echte Lösung besitzt. ⌟

b **Beispiel:** Maskieren ist nicht nur nützlich, um Instanzen als unlösbar zu erkennen, sondern auch, um den Suchraum lösbarer Instanzen einzuschränken. Die Instanz $I_{44b} = \|\,{}^{0}_{0000}\,{}^{00}_{001}\,{}^{0111}_{000}\,{}^{1001}_{1}\,\|$ hat die einzige kürzeste Lösung *dbcdddd bdbdbdbbbbbbbacacbddbddbbdbbbdbadbabbcdbaadddbadbdbdbdbcdbbbbbbbcdddd bacaaddddbdbdbdbadddbbdbdbdbbbbbbbcdbdbdbadbbbbbaaacddddbbbbbcdbacacd dddbdbdbdbaacddddbadddbdddbdbdbdbbbbbbbcbdddbdbdbdbcdbdbdbbbdbdbbbb bbbaaacdddbbdbdbbbbbbbcddddbbbbbabbbbaaaabdddbdbdbadbbbaaacddddbdbd bdbcacacaadbdbdbbbbbcdbaabdddbdbdbdbbbbbbbcdddbddbddbbbbbbbaacddddbab bdbdbbbbbbbaaacddddbdbdbbbdbbbdbaaacddddbdbdbdbacdbbbcacaaddddbdbdb dbbbbbabbabbabdddbdbdbdbbbbbbbadddbacdddbddbbdbdbdbbbbbbbaaacaadbdb dbbbbbbbaaacdbdbdbdbadddddbdbdbbbdbadbbbbbaaaabddbbbbbbaaaabdddbbbbbb bcdbdbdbdbbbbbabbcdbaaaadbdbaaaabbdbdbaaacddddbbbbbbbaaaadddddbabbbb aabbbbabdddbdbdbdbaaaabbdbdbdbaaaaaadbdbdbbbbbbbaabbbbbbaadbbbbbaaa aaaabbaaaaaaa* der Länge 698. Der Suchgraph von I_{44b} besteht sowohl aus ↑- als auch aus ↓-Überhängen. Betrachten wir nur den ↑-Teil, dann finden wir dort 21 744 293 594 Pfade bis zu einer Länge von 698 mit 151 104 421 871 Knoten. Durch genaues Abschneiden wie in 32b schrumpft der ↑-Teil auf 123 014 139 Pfade der Länge höchstens 698 mit 745 563 443 Knoten. Arbeiten wir zusätzlich mit beschränktem Abschneiden wie in 33a, dann müssen nur noch 9 302 545 Pfade mit 51 565 603 Knoten betrachtet werden. Dabei wurden aus dem Nullraum $\{(0,4,1,1),(2,3,0,3)\}$ die Bedingungen $10x_c + 8x_d \leq 3M$, $4x_a + 6x_c \leq M$ und $7x_a + 6x_b \leq 4M$ abgeleitet, wenn x_i die Anzahl der Regel i in der Beschriftung eines Pfades ist und M die maximale Suchtiefe. Memorieren wir schließlich zusätzlich 4091 Überhänge bis zu einer Länge von 50, dann verbleiben 4 039 288 Pfade mit 22 195 350 Knoten. Mit aktueller Technik kann dieser Graph in weniger als 1 Sekunde traversiert werden.

Ohne Maskierung müsste auch der ↓-Teil betrachtet werden, der deutlich größer ist. Im gesamten Suchgraphen gibt es bereits 69 223 855 374 Pfade

der Länge höchstens 50 mit 156 982 160 464 Knoten. Bei gleichbleibendem Verzweigungsgrad entspricht das etwa 10^{151} Pfaden der Länge höchstens 698 mit etwa 10^{156} Knoten. Verwenden wir auch hier die Optimierungen wie oben, also genaues und beschränktes Abschneiden, und Memorieren von Überhängen bis zu einer Länge von 50 für ↑- und 25 für ↓-Überhänge, dann wird der Graph zwar kleiner, aber mit 6 602 748 294 Pfaden der Länge höchstens 125 und 16 399 160 676 Knoten bei 163 und 2722 memorierten ↑- und ↓-Überhängen erhalten wir eine geschätzte Anzahl von 10^{55} Pfaden der Länge höchstens 698 mit 10^{57} Knoten. Die Traversierung eines solchen Graphen liegt *weit* jenseits der Kapazitäten aktueller Technik. ⌟

Duales Suchen *a*

Der Suchgraph einer Instanz I enthält in den Knoten beliebig lange Überhänge und eine mit a beschriftete Transition verlängert sowohl die ↑-Komponente um $h(a)$ als auch die ↓-Komponente um $g(a)$. Wir wollen dual dazu eine Relation konstruieren, die in den Knoten Überhänge mit beschränkter Länge enthält und es dafür erlaubt, nur die ↑- oder nur die ↓-Komponente zu verlängern.

In 46a erlauben wir ohne Restriktionen die partielle Anwendung von Regeln und erhalten die allgemeine Suchrelation. Diese Relation hat noch die nützlichen Eigenschaften der Suchrelation aus 28a: Sie kann zerlegt werden und steht in engem Zusammenhang zur allgemeinen Suchrelation der gespiegelten Instanz. Der Wegfall jeglicher Restriktionen bleibt natürlich nicht ohne Folgen. Statt die Gleichheitsmenge zu beschreiben, beschreibt die allgemeine Suchrelation die Koinzidenzmenge, das ist die Menge von Paaren mit gleichem Bild, wenn die ↑-Komponente als Eingabe für h und die ↓-Komponente als Eingabe für g benutzt wird. Schlimmer ist die Größe der allgemeinen Suchrelation: Sie ist für nichtpathologische Instanzen stets unendlich groß und bringt keine wirklichen Vorteile bei der Entscheidung, ob eine Instanz echte Lösungen hat oder nicht.

Das Problem der Größe wird in 49a behoben, indem nur noch solche Transitionen zugelassen werden, die einen Überhang nicht verlängern. Der erste Ansatz in 49b geht dabei jedoch zu restriktiv vor und muss in 54a soweit gelockert werden, dass wir einerseits eine endliche Relation beibehalten und andererseits trotzdem die Koinzidenzmenge damit beschreiben können. Das gelingt und Satz 56a trägt die Früchte unserer Arbeit in Form eines Transducers, der die Koinzidenzmenge akzeptiert.

Dieses Resultat ist eines der Hauptresultate dieser Arbeit. Es erlaubt zunächst in einem Exkurs zu bedingten Koinzidenzmengen in 58a die Konstruktion der Kette (62b) oberer Approximationen der Gleichheitsmenge.

Diese Kette hat Juhani Karhumäki bereits 1980 beschrieben. (Karhumäki [21]) Er baut dabei neben eigenen auf Resultaten von Sheila Greibach, Oscar Ibarra und Chul Kim aus den Jahren 1975 und 1976 auf. (Greibach [7], Ibarra, Kim [17]) Wir geben ausgehend von Satz 56a neue einheitliche Beweise.

Neben dieser Approximationskette wird in 63b eine neue Darstellung der Theorie der Nachfolger von Instanzen gegeben. Der Begriff des Nachfolgers taucht bereits 1982 in der ersten Arbeit auf, in der die Entscheidbarkeit von $\mathcal{G}(2)$ gezeigt wurde, dort noch als „equality collector“ bezeichnet. (Ehrenfeucht, Karhumäki, Rozenberg [5]) Vesa Halava und einige Mitautoren haben dann in verschiedenen Arbeiten die eigentliche Theorie der Nachfolger gestaltet. (Halava et al. [8, 10, 11, 12]) Mit ihrer Hilfe wurden einfachere Beweise der Entscheidbarkeit von $\mathcal{G}(2)$ gefunden, die Menge $\mathcal{M}$ der markierten Instanzen und die Menge $\mathcal{U}$ der „unique block instances“ wurden als entscheidbar erkannt. Hier wird die Theorie der Nachfolger erstmals systematisch dargestellt und als Konsequenz aus Satz 56a ergibt sich die Entscheidbarkeit der oben genannten Mengen und darüber hinaus von Instanzen, für die bisher kein Entscheidungsalgorithmus existierte. Es wird erstmals klar, dass sich die bekannten Techniken alle unter dem Begriff des dualen Suchens zusammenfassen lassen.

a

Allgemeines Suchen

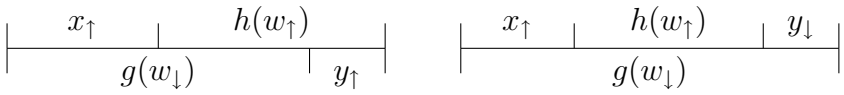
Die Grundidee beim Suchen nach einer Lösung wie in 28a besteht darin, einen Überhang aus $B^{*\updownarrow}$ zu verwalten und in einem Schritt mit der Regel a den Überhang an der $\uparrow$-Position um $h(a)$ und an der $\downarrow$-Position um $g(a)$ zu erweitern. Die Suchrelation in (28b) beschreibt dieses Vorgehen. Die dadurch entstehenden Suchgraphen enthalten in ihren Knoten Überhänge aus $B^{*\updownarrow}$, während die Beschriftungen mögliche Präfixe von Lösungen sind. In gewisser Weise beschreibt die Beschriftung eines Pfades aber nicht *ein* Wort, sondern *zwei*, nämlich eine Eingabe für den Morphismus h und eine Eingabe für den Morphismus g.

Was passiert, wenn wir nicht mehr fordern, dass die Eingaben für die beiden Morphismen stets gleich sind? Wir erlauben dann, dass einer der beiden Morphismen voraus eilt, aber welche Konsequenzen hat das? Probieren wir es aus und definieren die *allgemeine Suchrelation* $\hookrightarrow_I : B^{*\updownarrow} \times (A^* \times A^*) \times B^{*\updownarrow}$ *von* I durch

b

$$x \overset{w}{\hookrightarrow}_I y \iff x_\uparrow h(w_\uparrow) y_\downarrow = x_\downarrow g(w_\downarrow) y_\uparrow.$$

Anders als (28b) erlaubt (46b) nun unterschiedliche Eingaben für die beiden Morphismen. Die Darstellung 28c wird nun zu



bleibt also bis auf die Projektionen aus w heraus identisch. Die Folgerung 28e und das Lemma 28f gelten analog für $\hookmapsto$:

Folgerung. (Spiegeln von $\hookmapsto$) $x \overset{w}{\hookmapsto}_I y \iff \overline{y}^{\mathrm{R}} \overset{w^{\mathrm{R}}}{\hookmapsto}_{I^{\mathrm{R}}} \overline{x}^{\mathrm{R}}$. *a*

Hier wurde sogar *textuell* lediglich $\rightarrowtail$ durch $\hookmapsto$ ersetzt. Es ist aber zu beachten, dass die Operation Spiegeln in 28e auf ein *Wort* und in 47a auf ein *Wortpaar* angewendet wird.

Lemma. (Zerlegen von $\hookmapsto$) $x \overset{ab}{\hookmapsto}_I y \iff \exists z : x \overset{a}{\hookmapsto}_I z \overset{b}{\hookmapsto}_I y$. *b*

Auch hier wurde lediglich $\rightarrowtail$ durch $\hookmapsto$ ersetzt. Der Beweis verläuft komplett wie in 28g, wobei h nun die $\uparrow$- und g die $\downarrow$-Komponenten als Eingaben erhalten. Im Unterschied zu 28f ist nun aber ab eine Konkatenation von *Paaren* von Wörtern.

Das Analogon zu Folgerung 28d, wo der Zusammenhang zwischen der Suchrelation und der Gleichheitsmenge hergestellt wird, erfordert etwas Vorbereitung: Wenn wir unterschiedliche Eingaben für die Morphismen h und g erlauben, dann erzeugen wir eine Art verallgemeinerter Gleichheitsmenge. Diese Menge nennen wir *Koinzidenzmenge* $\mathbb{C}(I) \subseteq A^* \times A^*$ *von* I. Wir haben es mit einer Menge von Paaren zu tun und es gilt $\mathbb{C}(I) = \{u \mid s_\uparrow h(u_\uparrow) f_\uparrow = s_\downarrow g(u_\downarrow) f_\downarrow\}$. Wie üblich bezeichnen wir die echten Elemente der Koinzidenzmenge mit $\mathbb{C}^\circ(I) = \mathbb{C}(I) \setminus \{(\varepsilon, \varepsilon)\}$. Der Zusammenhang zwischen Koinzidenzmenge und $\hookmapsto$ lautet:

Folgerung. $\mathbb{C}(I) = \{u \mid s \overset{u}{\hookmapsto}_I \overline{f}\}$. *c*

und der zwischen Gleichheitsmenge und Koinzidenzmenge besteht durch

Folgerung. $\mathrm{id}(\mathrm{Eq}(I)) = \mathbb{C}(I) \cap \mathrm{id}(A^*)$. *d*

Es handelt sich bei $\hookmapsto$ tatsächlich um eine Verallgemeinerung von $\rightarrowtail$ in folgendem Sinn:

Folgerung. $x \overset{w}{\rightarrowtail}_I y \iff x \overset{(w,w)}{\hookmapsto}_I y$. *e*

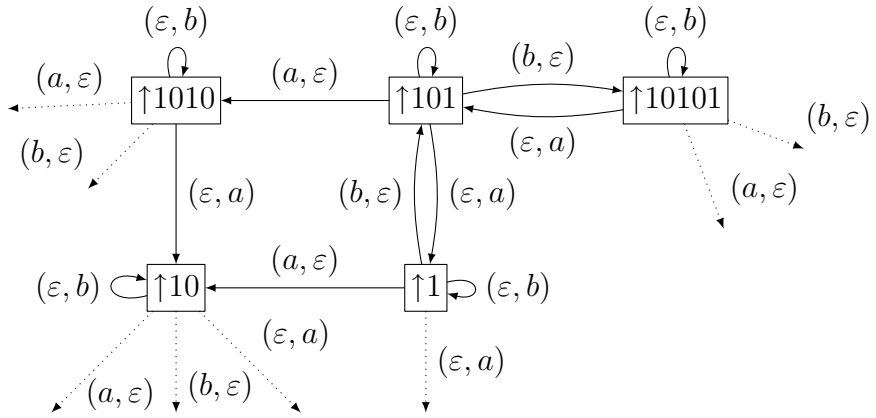
Das heißt, wenn es einen mit w beschrifteten Pfad durch $\rightarrowtail$ gibt, dann gibt es einen mit (w, w) beschrifteten Pfad durch $\hookmapsto$ und umgekehrt. Die Beziehung ist allerdings asymmetrisch, denn einerseits bestimmt zwar der Pfad durch $\hookmapsto$ den Pfad durch $\rightarrowtail$ eindeutig, aber andererseits kann es verschiedene mit

(w, w) beschriftete Pfade durch $\hookmapsto$ geben. Genauer folgt aus Lemma 47b sogar, dass *jede* Zerlegung von (w, w) einem Pfad durch $\hookmapsto$ entspricht. Es sei daran erinnert, dass $A^* \times A^*$ frei kommutativ aber nicht frei von $A^{\updownarrow}$ erzeugt wird und das Paar (w, w) auf $\binom{2|w|}{|w|}$ verschiedene Arten in Elemente aus $A^{\updownarrow}$ zerlegt werden kann.

Der durch $\hookmapsto$ erzeugte Graph ist in der Regel unendlich groß. Voraussetzung für einen endlichen Graphen ist, dass sämtliche h- oder sämtliche g-Bilder leer sind. Solche Instanzen sind periodisch und nach 26d entscheidbar.

Die Techniken aus 28a zur Einschränkung des Suchraums, also Abschneiden, Ausschließen, Einfangen und Maskieren wurden alle für die Suche nach Elementen der Gleichheitsmenge formuliert. Wenn wir nun von s beginnend $\hookmapsto$ auf der Suche nach $\overline{f}$ traversieren, dann suchen wir nach Elementen der Koinzidenzmenge. Das heißt, diese Methoden können nicht direkt übernommen werden. Wir könnten jetzt versuchen, Methoden zu finden, mit deren Hilfe das Traversieren von $\hookmapsto$ erleichtert wird, aber wir sind ja nicht wirklich an Elementen aus der Koinzidenzmenge interessiert. Deshalb werden wir das nicht tun, sondern im nächsten Abschnitt eine andere Richtung einschlagen, die es schließlich erlauben wird, gewisse Instanzen zu entscheiden.

a **Beispiel:** Für die Instanz $I_{15c} = |\,{}^{101}_{\varepsilon}\,|\,{}^{0}_{10}\,{}^{01}_{\varepsilon}\,|\,{}^{\varepsilon}_{1010}\,|$ erzeugt die allgemeine Suchrelation über $s = \uparrow 101$ den Graphen



Im Vergleich mit dem Suchgraphen von I_{15c} in 30b ist dieser Graph genauer: Der Pfad $\uparrow 101 \overset{a}{\rightarrowtail}_{I_{15c}} \uparrow 10$ aus 30b findet sich hier als der Pfad $\uparrow 101 \overset{(a,\varepsilon)}{\hookrightarrow}_{I_{15c}} \uparrow 1010 \overset{(\varepsilon,a)}{\hookrightarrow}_{I_{15c}} \uparrow 10$ und als der Pfad $\uparrow 101 \overset{(\varepsilon,a)}{\hookrightarrow}_{I_{15c}} \uparrow 1 \overset{(a,\varepsilon)}{\hookrightarrow}_{I_{15c}} \uparrow 10$. Statt eines einzigen mit der Lösung ba beschrifteten Pfades in 30b gibt es nun 6 solcher Pfade, von denen 3 in der obigen Abbildung zu sehen sind, zum Beispiel der Pfad $\uparrow 101 \overset{(b,\varepsilon)}{\hookrightarrow}_{I_{15c}} \uparrow 10101 \overset{(\varepsilon,b)}{\hookrightarrow}_{I_{15c}} \uparrow 10101 \overset{(\varepsilon,a)}{\hookrightarrow}_{I_{15c}} \uparrow 101 \overset{(a,\varepsilon)}{\hookrightarrow}_{I_{15c}} \uparrow 1010$. ⌟

Duales Suchen *a*

Beim Suchen mit $\rightarrowtail$ wie in 28a entstehen beliebig lange Überhänge aus $B^{*\updownarrow}$ in den Knoten des Suchgraphen. Die Beschriftungen der Pfade können wir als Paar ohne Überhang verstehen. Beim allgemeinen Suchen mit $\hookrightarrow$ wie in 46a entstehen beliebig lange Überhänge sowohl in den Knoten als auch in den Beschriftungen der Pfade. Wir wollen nun in Dualität zu 28a die Länge der Überhänge in den Knoten einschränken. Gelingt das, dann können wir einen Transducer mit endlicher Zustandsmenge konstruieren, der die Koinzidenzmenge akzeptiert.

Die allgemeine Suchrelation gestattet es den beiden Morphismen, beliebig weit voraus zu eilen. Dabei entstehen Überhänge, die dann vom jeweils anderen Morphismus wieder aufgeholt werden müssen. Wir können das vermeiden, indem wir festlegen, dass ein Morphismus nur dann zur Anwendung kommen darf, wenn dabei kein Überhang verlängert wird, der ohnehin vom anderen Morphismus überdeckt werden muss. Mit anderen Worten wollen wir die Situation vermeiden, dass ein Überhang der Form $\uparrow w$ mit $|w| > 0$ zu einem Überhang der Form $\uparrow wh(a)$ für ein $a \in A$ wird. Im Fall von herkömmlichen Instanzen, wenn also $f = (\varepsilon, \varepsilon)$ gilt, machen wir dabei keinen Fehler, da auf dem Weg zu $\overline{f}$ ja in jedem Fall die $\downarrow$-Komponente des Überhangs verlängert werden muss. Diese Verlängerung kann irgendwann auf dem Weg zu $\overline{f}$ erfolgen, also auch sofort. Im Fall allgemeiner Instanzen müssen wir uns zusätzlich Gedanken über den Fall machen, in dem der Übergang $\uparrow w \overset{a}{\hookrightarrow}_I \uparrow wh(a)$ notwendig ist, zum Beispiel weil $\overline{f} = \uparrow wh(a)$ ist.

Beschränkt *b*

Zunächst definieren wir die *beschränkte Suchrelation* $\hookrightarrow \subseteq \rightarrowtail$ *von* I durch

$$\hookrightarrow_I = \rightarrowtail_I \cap ((B^{*\uparrow} \times A^{\downarrow} \times B^{*\updownarrow}) \cup (B^{*\downarrow} \times A^{\uparrow} \times B^{*\updownarrow})) \qquad c$$

und verbieten in (49c) genau die oben beschriebenen Übergänge. Die beschränkte Suchrelation hat die gewünschte Eigenschaft, dass Überhänge in den Knoten nicht mehr beliebig lang sein können. Es gilt das

Lemma. $x \overset{a}{\hookrightarrow}_I y \implies |y| \leq \max\{|x|, |h(a_\uparrow)|, |g(a_\downarrow)|\}$. *d*

Beweis: Nach Definition gilt $\hookrightarrow \subseteq \rightarrowtail$ und damit $x_\uparrow h(a_\uparrow) y_\downarrow = x_\downarrow g(a_\downarrow) y_\uparrow$. Falls $x \in B^{*\uparrow}$ gilt, dann ist $a \in A^{\downarrow}$. Falls nun $y \in B^{*\uparrow}$, dann haben wir $x_\uparrow = g(a_\downarrow) y_\uparrow$ und daraus folgt $|y| = |y_\uparrow| \leq |x_\uparrow| = |x|$. Falls jedoch $y \in B^{*\downarrow}$, dann haben wir $x_\uparrow y_\downarrow = g(a_\downarrow)$ und daraus folgt $|y| = |y_\downarrow| \leq |g(a_\downarrow)|$. Der Fall $x \in B^{*\downarrow}$ folgt analog. □ *e*

Lemma 49d sagt aus, dass ein Schritt mit $\hookrightarrow$ einen Überhang nicht beliebig verlängern kann: Der neue Überhang ist höchstens so lang wie der alte Überhang falls die Position nicht geändert wurde und höchstens so lang wie das verwendete Regelwort falls die Position geändert wurde. Starten wir nun bei einem Überhang, dann geben uns seine Länge und die Weite von I eine Schranke für die Länge aller erreichbarer Überhänge.

a **Folgerung.** $y \in \mathrm{N}(\hookrightarrow_I^\star(s)) \implies |y| \leq \mathrm{wd}(I)$.

b **Folgerung.** $\hookrightarrow_I^\star(s)$ *ist endlich.*

Die Endlichkeit ist der erste Schritt zu einem Transducer, der die Koinzidenzmenge akzeptiert. Wenn $\overline{f}$ wie bei herkömmlichen Instanzen in $\hookrightarrow^\star(s)$ enthalten ist, dann sind wir an diesem Punkt sogar schon am Ziel. Im nächsten Abschnitt werden wir mit einem kleinen Kniff auch dann einen Transducer für die Koinzidenzmenge konstruieren, wenn $\overline{f}$ nicht von s mit $\hookrightarrow$ erreichbar ist.

Wir werden später die Eigenschaft von $\hookrightarrow_I^\star(\{(\varepsilon,\varepsilon)\})$ benötigen, dass die $\uparrow$-Knoten in dieser Hülle Suffixe des Morphismus h und die $\downarrow$-Knoten Suffixe des Morphismus g sind. Es gilt das

c **Lemma.** (Suffixlemma) $\mathrm{N}(\hookrightarrow_I^\star(\{(\varepsilon,\varepsilon)\})) \subseteq \mathrm{S}_h^\uparrow \cup \mathrm{S}_g^\downarrow$.

d *Beweis:* Sei $y \in \mathrm{N}(\hookrightarrow_I^\star(\{(\varepsilon,\varepsilon)\}))$. Induktion über die Länge von y: Wenn $y = (\varepsilon,\varepsilon)$, dann gilt natürlich $y \in \mathrm{S}_h^\uparrow \cup \mathrm{S}_g^\downarrow$. Sei nun $x \in \mathrm{N}(\hookrightarrow_I^\star(\{(\varepsilon,\varepsilon)\})) \cap (\mathrm{S}_h^\uparrow \cup \mathrm{S}_g^\downarrow)$ und $x \overset{a}{\hookrightarrow}_I y$ für ein $y \in A^{\updownarrow}$. Dann liegt einer der vier Fälle (i) $x_\uparrow = g(a_\downarrow)y_\uparrow$ mit $y_\uparrow \in \mathrm{S}_h$, (ii) $x_\uparrow y_\downarrow = g(a_\downarrow)$ mit $y_\downarrow \in \mathrm{S}_g$, (iii) $h(a_\uparrow) = x_\downarrow y_\uparrow$ mit $y_\uparrow \in \mathrm{S}_h$ oder (iv) $h(a_\uparrow)y_\downarrow = x_\downarrow$ mit $y_\downarrow \in \mathrm{S}_g$ vor. □

Eine weitere interessante Eigenschaft der beschränkten Suchrelation ist, dass die Koinzidenzen unendlicher Lösungen einer Instanz stets Beschriftungen von unendlichen Pfaden durch $\hookrightarrow_I^\star(s)$ entsprechen, unabhängig davon, ob $\overline{f}$ von s mit $\hookrightarrow$ erreichbar ist oder nicht. Dazu nennen wir einen unendlichen Pfad mit der Beschriftung $w_0w_1\cdots$ von s durch $\hookrightarrow_I$ einen *ω-Pfad*, wenn $(w_0w_1\cdots)_\uparrow$ mit $(w_0w_1\cdots)_\downarrow$ vergleichbar ist und wenigstens eine Komponente gleich ω ist, wenn also gilt $(w_0w_1\cdots)_\uparrow = \omega$ oder $(w_0w_1\cdots)_\downarrow = \omega$.

e **Lemma.** $\omega \in \mathrm{Eq}^\omega(I) \iff \exists\omega\text{-}Pfad$.

f *Beweis:* ($\Longleftarrow$) Sei $w_0w_1\cdots$ die Beschriftung eines ω-Pfades und sei ohne Beschränkung der Allgemeinheit $(w_0w_1\cdots)_\uparrow = \omega$. Nach Definition von $\hookrightarrow$ und dem Lemma 47b über die Zerlegung der allgemeinen Suchrelation sind

für jedes endliche Präfix p von $w_0w_1\cdots$ die Wörter $p_\uparrow$ und $p_\downarrow$ und die Wörter $s_\uparrow h(p_\uparrow)$ und $s_\downarrow g(p_\downarrow)$ vergleichbar.

Wenn $(w_0w_1\cdots)_\downarrow = u$ endlich ist, dann gibt es ein endliches Präfix $p_u = (v, u)$ von $w_0w_1\cdots$, so dass $h(v^{-1}\omega) = \varepsilon$ ist. Anderenfalls würde der Überhang in $(s_\uparrow h(p_\uparrow), s_\downarrow g(p_\downarrow))$ beliebig lang werden im Widerspruch zur Endlichkeit von $\hookrightarrow_I^\star(s)$. Also ist dann h löschend und ω ist eine unendliche Lösung von I.

Wenn $(w_0w_1\cdots)_\downarrow = \omega$ nicht endlich ist, dann sei v_i das längste gemeinsame Präfix von $(w_0w_1\cdots w_i)_\uparrow$ und $(w_0w_1\cdots w_i)_\downarrow$. Die Folge $(v_0, v_1, \dots)$ ist aufsteigend bezüglich $\preceq_\mathrm{p}$ und nach jeweils höchstens endlich vielen Schritten echt aufsteigend. Also ist $\lim_{i\to\infty} v_i = \omega$ eine unendliche Lösung von I.

($\Longrightarrow$) Sei ω eine unendliche Lösung von I. Wir konstruieren einen ω-Pfad: Wir beginnen beim Knoten s mit dem leeren Pfad $(\varepsilon, \varepsilon)$.

Erreichen wir mit dem Pfad p mit $p_\uparrow = p_\downarrow$ den Knoten x, dann bestimmt das erste Symbol von $p_\uparrow^{-1}\omega$ die von x ausgehende Transition bis auf die Position eindeutig. Mehrere Positionen sind nur für $x = (\varepsilon, \varepsilon)$ möglich, wir wählen dann eine beliebig aus.

Erreichen wir ohne Beschränkung der Allgemeinheit mit dem Pfad p mit $p_\uparrow \prec_\mathrm{p} p_\downarrow$ den Knoten x, dann wählen wir, falls $x \in B^{*\downarrow}$, die Transition (c, ε), wobei c das erste Symbol von $p_\uparrow^{-1}p_\downarrow$ ist. Falls $x \in B^{*\uparrow} \setminus \{(\varepsilon, \varepsilon)\}$ ist, dann wählen wir die Transition (ε, c), wobei c das erste Symbol von $p_\downarrow^{-1}\omega$ ist. Alle benötigten Transitionen existieren, anderenfalls wäre ω keine unendliche Lösung.

Im zweiten Fall kann es passieren, dass p nicht mehr in der $\uparrow$-Komponente verlängert wird. Es gibt dann eine Schleife um x, die nicht die Position wechselt. Das impliziert $g(p_\downarrow^{-1}\omega) = \varepsilon$. Jede Verlängerung von p entspricht dann einer Schleife um x, die jeweils das nächste Symbol von ω verbraucht. □

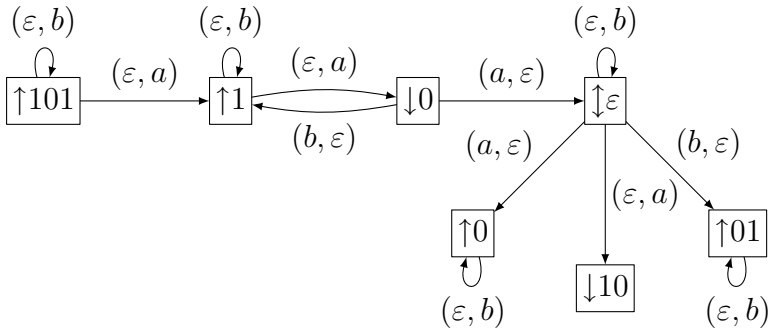
Dem Beweis entnehmen wir zusätzlich die

Folgerung. $I \in \mathcal{E}^0 \implies \mathrm{Eq}^\omega(I) = \{\omega \mid \exists\omega\text{-}Pfad\}$. *a*

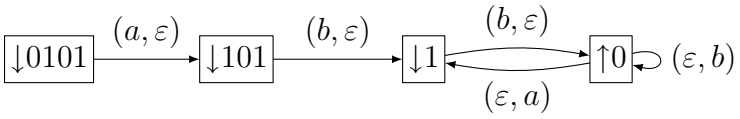
Es sei nochmals betont, dass die analoge Formulierung für die Gleichheitsmenge *nicht* gilt: Die Beispiele 51b und 53b zeigen, dass $\overline{f}$ nicht immer von s durch $\hookrightarrow$ erreichbar ist.

Unendliche Lösungen haben hingegen kein Ziel, es genügt, wenn stets eine Forsetzung möglich ist. Für herkömmliche Instanzen *ist* $\overline{f} = (\varepsilon, \varepsilon)$ natürlich stets von $(\varepsilon, \varepsilon)$ aus erreichbar. Das heißt, dass die beschränkte Suchrelation $\hookrightarrow$ zur Charakterisierung sowohl der Koinzidenzmenge als auch der unendlichen Koinzidenzmenge nichtlöschender herkömmlicher Instanzen ausreicht.

Beispiel: Für die Instanz $I_{15c} = \left|\begin{smallmatrix}101\\ \varepsilon\end{smallmatrix}\right| \left|\begin{smallmatrix}0 & 01\\ 10 & \varepsilon\end{smallmatrix}\right| \left|\begin{smallmatrix}\varepsilon\\ 1010\end{smallmatrix}\right|$ ist $\hookrightarrow_{I_{15c}}^\star(\uparrow 101)$ der Graph *b*



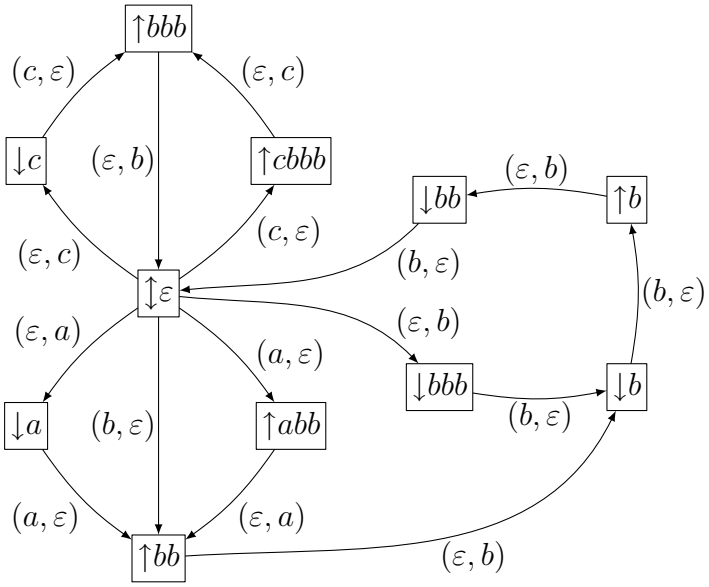
der $\overline{f} = \uparrow 1010$ *nicht* enthält. Auch der Graph $\hookrightarrow^{\star}_{I^{\mathrm{R}}_{15c}}(\downarrow 0101)$



der gespiegelten Instanz beschreibt *nicht* die Koinzidenzmenge von I^{R}_{15c}.

Der unendliche Pfad $\uparrow 101 \xhookrightarrow{(\varepsilon,a)}_{I_{15c}} \uparrow 1 \xhookrightarrow{(\varepsilon,a)}_{I_{15c}} \downarrow 0 \xhookrightarrow{(a,\varepsilon)}_{I_{15c}} \updownarrow\varepsilon \xhookrightarrow{(a,\varepsilon)}_{I_{15c}} \uparrow 0 \xhookrightarrow{(\varepsilon,b)^{\omega}}_{I_{15c}} \cdots$ in $\hookrightarrow^{\star}_{I_{15c}}(\uparrow 101)$ ist ein ω-Pfad mit der Beschriftung (aa, aab^{ω}). Tatsächlich ist aab^{ω} eine unendliche Lösung von I_{15c}, es gilt $\mathrm{Eq}^{\omega}(I_{15c}) = \{b^i auab^{\omega} \mid |u|_a = i\}$. Leserinnen und Leser mögen sich klar machen, dass für jede unendliche Lösung ein ω-Pfad in $\hookrightarrow^{\star}_{I_{15c}}(\uparrow 101)$ vorhanden ist. ⌟

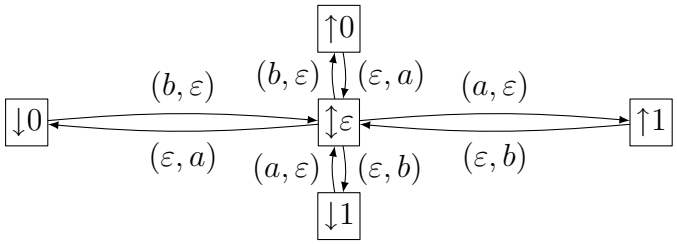
a **Beispiel:** Für die Instanz $I_{15d} = \| \begin{smallmatrix} abb & bb & cbbb \\ a & bbb & c \end{smallmatrix} \|$ ist $\hookrightarrow^{\star}_{I_{15d}}(\updownarrow\varepsilon)$ der Graph



in dem, da I_{15d} eine herkömmliche Instanz ist, die Pfade von $\updownarrow\varepsilon$ zu $\updownarrow\varepsilon$ genau mit Elementen aus der Koinzidenzmenge von I_{15d} beschriftet sind. Somit gilt $\mathbb{C}(I_{15d}) = \{(abb, abb), (bbb, bb), (c, cb)\}^*$.

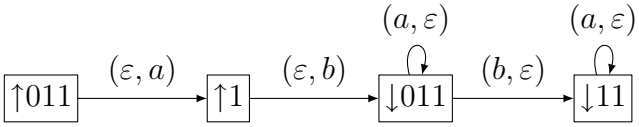
Die unendlichen Pfade $\updownarrow\varepsilon \xhookrightarrow{(\varepsilon,b)}_{I_{15d}} \downarrow bbb \xhookrightarrow{(b,\varepsilon)}_{I_{15d}} \downarrow b \xhookrightarrow{(b,\varepsilon)}_{I_{15d}} \uparrow b \xhookrightarrow{(\varepsilon,b)}_{I_{15d}} \downarrow bb \xhookrightarrow{(b,\varepsilon)}_{I_{15d}} \updownarrow\varepsilon \xhookrightarrow{\cdots}_{I_{15d}} \cdots$ und $\updownarrow\varepsilon \xhookrightarrow{(b,\varepsilon)}_{I_{15d}} \uparrow bb \xhookrightarrow{(\varepsilon,b)}_{I_{15d}} \downarrow b \xhookrightarrow{(b,\varepsilon)}_{I_{15d}} \uparrow b \xhookrightarrow{(\varepsilon,b)}_{I_{15d}} \downarrow bb \xhookrightarrow{(b,\varepsilon)}_{I_{15d}} \updownarrow\varepsilon \xhookrightarrow{\cdots}_{I_{15d}} \cdots$ gehören beide zur unendlichen Lösung b^ω. ⌟

Beispiel: Für die Instanz $I_{53a} = |\,{}^{0}_{\varepsilon}\,|\,{}^{1\ 0}_{0\ 1}\,|\,{}^{\varepsilon}_{0}\,|$ ist $\hookrightarrow^\star_{I_{53a}}(\uparrow 0)$ der Graph *a*

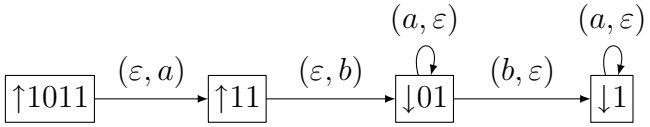


der zum Beispiel den Pfad $s = \uparrow 0 \xhookrightarrow{(\varepsilon,a)}_{I_{53a}} \updownarrow\varepsilon \xhookrightarrow{(\varepsilon,a)}_{I_{53a}} \downarrow 0 \xhookrightarrow{(b,\varepsilon)}_{I_{53a}} \updownarrow\varepsilon \xhookrightarrow{(b,\varepsilon)}_{I_{53a}} \uparrow 0 = \overline{f}$ enthält. Tatsächlich gilt $s_\uparrow h(bb) f_\uparrow = 000 = s_\downarrow g(aa) f_\downarrow$, das Paar (bb, aa) ist ein Element der Koinzidenzmenge von I_{53a}. ⌟

Beispiel: Für die Instanz $I_{53b} = |\,{}^{011}_{\varepsilon}\,|\,{}^{\varepsilon\ \ 0}_{01\ 1011}\,|\,{}^{1101}_{\varepsilon}\,|$ ist $\hookrightarrow^\star_{I_{53b}}(\uparrow 011)$ der Graph *b*



der $\overline{f} = \downarrow 1101$ ebenfalls nicht enthält, und auch hier beschreibt der Graph $\hookrightarrow^\star_{I^{\mathrm{R}}_{53b}}(\uparrow 1011)$



nicht die Koinzidenzmenge von I^{R}_{53b}.

Beide Graphen enthalten jeweils den einzigen ω-Pfad mit der Beschriftung (ab, aba^ω), also gilt $\mathrm{Eq}^\omega(I_{53b}) = \mathrm{Eq}^\omega(I^{\mathrm{R}}_{53b}) = aba^\omega$. ⌟

a **Beschränkt bidirektional**

Wir sind durch die Definition der beschränkten Suchrelation einen wesentlichen Schritt voran gekommen und haben eine Teilrelation der allgemeinen Suchrelation gefunden, die nur endlich viele Knoten enthält und für herkömmliche Instanzen die Koinzidenzmenge beschreibt. Im Allgemeinen ist $\hookrightarrow$ jedoch zu klein, das heisst, wir haben wie in 51b und 53b eventuell Pfade abgeschnitten, die essenziell auf dem Weg zu einer Lösung liegen.

Werfen wir einen zweiten Blick auf 51b und 53b, dann stellen wir fest, dass es in beiden Fällen gemeinsame Knoten im Graphen der Instanz und im gedrehten Graphen der gespiegelten Instanz gibt. Hier soll die Operation *Drehen* einer Transition $x \overset{w}{\hookrightarrow}_I y$ die Transition $\overline{y}^{\mathrm{R}} \overset{w^{\mathrm{R}}}{\hookrightarrow}_{I^{\mathrm{R}}} \overline{x}^{\mathrm{R}}$ zuordnen. Die Drehung des Graphen G bezeichnen wir mit $\circlearrowleft G$. Das heißt, in den beiden Beispielen beschreibt die *Vereinigung* von $\hookrightarrow_I^\star(s)$ und $\circlearrowleft \hookrightarrow_{I^{\mathrm{R}}}^\star(f^{\mathrm{R}})$ die Koinzidenzmenge. Wir definieren diese Menge als die *beschränkte bidirektionale Suchrelation* $\leftrightharpoons_I$ *von* I und werden zeigen, dass wir die Koinzidenzmenge *immer* mit $\leftrightharpoons_I$ beschreiben können.

Zunächst ergibt sich aus der Endlichkeit der beiden Hüllen die

b **Folgerung.** $\leftrightharpoons_I$ *ist endlich.*

und mit Lemma 47a über das Spiegeln der allgemeinen Suchrelation auch die

c **Folgerung.** $\leftrightharpoons_I \subseteq \hookrightarrow_I$.

Tatsächlich ist eine Transition entweder aus $\hookrightarrow_I^\star(s)$ und damit nach Definition der beschränkten Suchrelation $\hookrightarrow$ in der allgemeinen Suchrelation $\hookrightarrow$ enthalten, oder sie hat die Form $\circlearrowleft(x, a, y)$ für gewisse $x \overset{a}{\hookrightarrow}_{I^{\mathrm{R}}} y$ und ist nach Lemma 47a ebenfalls in der allgemeinen Suchrelation $\hookrightarrow$ enthalten.

Die Folgerungen 54b und 54c sichern, dass wir es bei der beschränkten bidirektionalen Suchrelation immer noch mit einem endlichen Teil der allgemeinen Suchrelation zu tun haben. Ein Pfad durch $\leftrightharpoons$ von s nach $\overline{f}$ beschreibt also ein Element der Koinzidenzmenge, es gilt die

d **Folgerung.** $\mathbb{C}(I) \supseteq \{u \mid s \overset{u}{\leftrightharpoons}_I \overline{f}\}$.

Wir wollen analog zu Folgerung 47c auch die Umkehrung zeigen. Dazu müssen wir zeigen, dass die in der beschränkten Suchrelation fehlenden Pfade in der beschränkten bidirektionalen Suchrelation wieder vorhanden sind. Genauer gilt das

e **Lemma.** (Vollständigkeit von $\leftrightharpoons$) *Wenn* $u \in \mathbb{C}^\circ(I)$ *ein echtes Element der Koinzidenzmenge von* I *ist, dann gibt es einen Pfad* $s \overset{u_0}{\leftrightharpoons}_I \cdots \overset{u_{k-1}}{\leftrightharpoons}_I \overline{f}$ *von* s *nach* $\overline{f}$ *durch* $\leftrightharpoons_I$*, der mit* $u = u_0 \cdots u_{k-1}$ *beschriftet ist.*

Beweis: Nach Voraussetzung ist $u \in \mathbb{C}^{\circ}(I)$, also gilt $s \xhookrightarrow{u}_I \overline{f}$ nach 47c. Sei p das maximale Präfix von u, für das $s \xhookrightarrow{p}_I x$ gilt, also das maximale Präfix, das zu einem Pfad gehört, der in s beginnt und die beschränkte Relation von I nicht verlässt. Wenn $p = u$, dann folgt $x = \overline{f}$ und wir haben den gesuchten Pfad bereits gefunden. Wenn $p \neq u$ und $u = pv$, dann $x \xhookrightarrow{v}_I \overline{f}$ nach Lemma 47b über die Zerlegung der allgemeinen Suchrelation. *a*

Da p maximal ist, gilt entweder $x_\uparrow = v_\uparrow = \varepsilon$ oder $x_\downarrow = v_\downarrow = \varepsilon$, denn nehmen wir zum Beispiel an, es wäre $x_\uparrow = \varepsilon$ und $v_\uparrow \neq \varepsilon$. Dann wäre $v = (a, \varepsilon)w$ für ein $a \in A$ und ein $w \in A^* \times A^*$. Nach Zerlegungslemma 47b gilt dann $x \xhookrightarrow{(a,\varepsilon)}_I y \xhookrightarrow{w}_I \overline{f}$ für ein $y \in B^{*\updownarrow}$ im Widerspruch zur Maximalität von p, denn es wäre auch $x \xhookrightarrow{(a,\varepsilon)}_I y$.

Sei also $x_\uparrow = v_\uparrow = \varepsilon$. Dann ist auch $\overline{f}_\uparrow = f_\downarrow = \varepsilon$, da ja $x \xhookrightarrow{v}_I \overline{f}$ gilt und v nichts zu der leeren $\uparrow$-Komponente von x hinzufügt. Durch wiederholte Anwendung des Zerlegungslemmas schließen wir nun auf die Existenz des Pfades $x = x_0 \xhookrightarrow{v_0}_I \cdots \xhookrightarrow{v_{m-1}}_I x_m = \overline{f}$, wobei $v = v_0 \cdots v_{m-1}$, sowie $v_{0\uparrow} = \cdots = v_{m-1\uparrow} = \varepsilon$ und $x_{0\uparrow} = \cdots = x_{m\uparrow} = \varepsilon$ gelten. Nach Lemma 47a über das Spiegeln der allgemeinen Suchrelation gibt es auch den Pfad $\overline{x_m}^{\mathrm{R}} \xhookrightarrow{v_{m-1}^{\mathrm{R}}}_{I^{\mathrm{R}}} \cdots \xhookrightarrow{v_0^{\mathrm{R}}}_{I^{\mathrm{R}}} \overline{x_0}^{\mathrm{R}}$, in dem sich sämtliche nichtleere Komponenten in den Überhängen $\overline{x_i}^{\mathrm{R}}$ entgegengesetzt zu den nichtleeren Komponenten in der Beschriftung v_j^{R} befinden. Deshalb können wir sogar auf die Existenz des Pfades $\overline{x_m}^{\mathrm{R}} \xhookrightarrow{v_{m-1}^{\mathrm{R}}}_{I^{\mathrm{R}}} \cdots \xhookrightarrow{v_0^{\mathrm{R}}}_{I^{\mathrm{R}}} \overline{x_0}^{\mathrm{R}}$ schließen und damit nach Definition der beschränkten bidirektionalen Suchrelation auch auf den Pfad $x = x_0 \overset{v_0}{\Longleftrightarrow}_I \cdots \overset{v_{m-1}}{\Longleftrightarrow}_I x_m = \overline{f}$.

Mit analogen Argumenten für den Fall $x_\downarrow = v_\downarrow = \varepsilon$ folgt die Existenz des gesuchten Pfades. □

Die Quintessenz von Beweis 55a lautet:

> „Bleibe so lange wie möglich in der beschränkten Suchrelation $\hookrightarrow_I$ von I und wenn $\hookrightarrow_I$ verlassen werden *muss*, dann von einem Überhang aus, der bereits in der gespiegelten beschränkten Relation $\hookrightarrow_{I^{\mathrm{R}}}$ der gespiegelten Instanz I^{R} liegt."

Es ist interessant, dass lediglich Aussagen über das Zerlegen und das Spiegeln der allgemeinen Suchrelation benutzt wurden, also Aussagen, die mit sehr einfachen Mitteln gezeigt werden können. Wesentlich haben wir benutzt, dass beim Drehen zwar die Positionen der Überhänge geändert werden, nicht jedoch die Position der Beschriftung einer Transition.

Lemma 54e zeigt, dass die beschränkte bidirektionale Suchrelation alle gewünschten Eigenschaften hat: Es handelt sich um eine *endliche* Relation

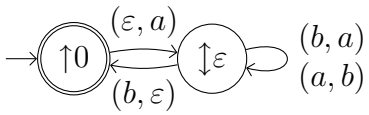
und die Beschriftungen der Pfade von s nach $\overline{f}$ sind *genau* die Elemente der Koinzidenzmenge. Wir können als Resultat festhalten:

a **Satz.** *Sei $I \in \mathcal{G}$ eine Instanz des verallgemeinerten Postschen Korrespondenzproblems. Die Koinzidenzmenge $\mathbb{C}(I)$ ist eine rationale Relation. Ein Transducer, der $\mathbb{C}(I)$ akzeptiert, kann effektiv konstruiert werden.*

b *Beweis:* Nach 54d und 54e akzeptiert der Transducer mit Überführungsrelation $\leftrightharpoons_I$, mit Startzustand s und mit Endzustand $\overline{f}$ die Koinzidenzmenge. □

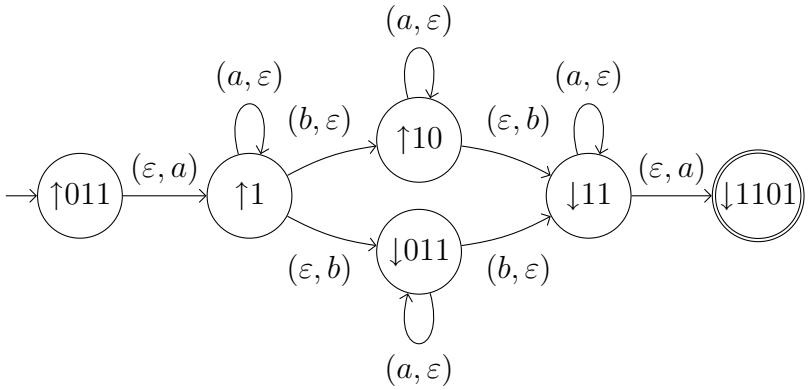
c **Folgerung.** *Es ist entscheidbar, ob $\mathbb{C}(I)$ leer ist. Es ist entscheidbar, ob $\mathbb{C}(I)$ endlich ist. $\mathbb{C}(I)$ kann effektiv aufgezählt werden.*

d **Beispiel:** Für die Instanz $I_{53a} = |\,{}^{0}_{\varepsilon}\,|\,{}^{1}_{0}\;{}^{0}_{1}\,|\,{}^{\varepsilon}_{0}\,|$ gilt $\hookrightarrow^{\star}_{I_{53a}}(\uparrow 0) = \circlearrowleft\hookrightarrow^{\star}_{I^{\mathrm{R}}_{53a}}(\downarrow 0)$, also akzeptiert der (liberalisiert dargestellte) Transducer



die Koinzidenzmenge $\mathbb{C}(I_{53a}) = (\varepsilon, a)\{(b, a), (a, b)\}^*(b, \varepsilon)$. ⌟

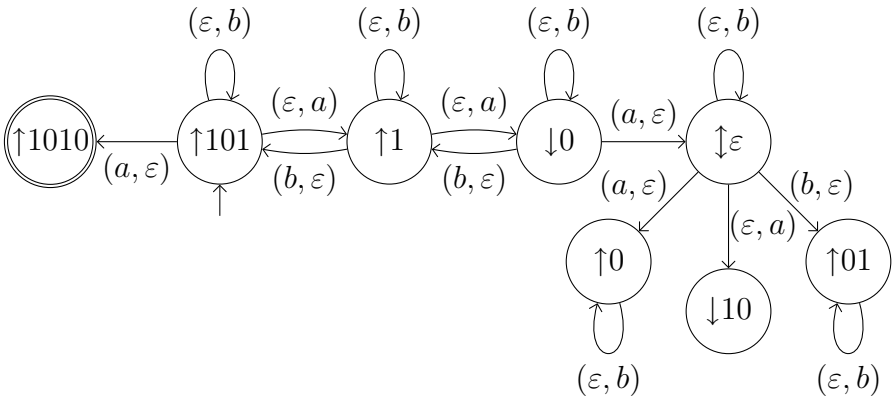
e **Beispiel:** Die Instanz $I_{53b} = |\,{}^{011}_{\varepsilon}\,|\,{}^{\varepsilon}_{01}\;{}^{0}_{1011}\,|\,{}^{1101}_{\varepsilon}\,|$ fällt ebenfalls in die Menge der Instanzen, für die die beschränkte Suchrelation zu klein ist, um die Koinzidenzmenge zu beschreiben. Der Transducer



akzeptiert $\mathbb{C}(I_{53b}) = (a, \varepsilon)^*(b, aba)(a, \varepsilon)^*$. Betrachten wir das Paar (b, aba) und die beiden Zerlegungen $(\varepsilon, a)(b, \varepsilon)(\varepsilon, b)(\varepsilon, a)$ und $(\varepsilon, a)(\varepsilon, b)(b, \varepsilon)(\varepsilon, a)$, die beide je einem akzeptierenden Pfad entsprechen. Die erste ist *nicht* die Zerlegung, die in Beweis 55a konstruiert wurde, denn hier wäre $p = (\varepsilon, a)$ nicht maximal und damit schlägt der Schluss fehl, dass sich alle weiteren

Schritte nur noch in einer Komponente abspielen. Konkret gilt zwar $(\uparrow 1)_\downarrow = \varepsilon$ aber nicht $(b, ba)_\downarrow = \varepsilon$. Die zweite Zerlegung hingegen verlässt $\hookrightarrow_{I_{53b}}$ so spät wie möglich, nämlich erst am Überhang $\downarrow 11$ und ist die in 55a konstruierte Zerlegung. Tatsächlich gilt $(\downarrow 11)_\uparrow = (\varepsilon, a)_\uparrow = (\downarrow 1101)_\uparrow = \varepsilon$. ⌟

Beispiel: In 51b wird gezeigt, dass die beschränkte Suchrelation für die *a*
Instanz $I_{15c} = |\,{}^{101}_{\varepsilon}\,|\,{}^{0}_{10}\,{}^{01}_{\varepsilon}\,|\,{}^{\varepsilon}_{1010}\,|$ nicht die Koinzidenzmenge beschreibt. Die beschränkte bidirektionale Suchrelation führt nun auf den Transducer



der die Koinzidenzmenge akzeptiert. Hier fehlen in der beschränkten Suchrelation von I_{15c} die beiden entscheidenden Transitionen $\uparrow 1 \overset{(b,\varepsilon)}{\Longleftrightarrow}_{I_{15c}} \uparrow 101$ und $\uparrow 101 \overset{(a,\varepsilon)}{\Longleftrightarrow}_{I_{15c}} \uparrow 1010$, die von der beschränkten Suchrelation der gespiegelten Instanz I^{R}_{15c} geliefert werden. Die Koinzidenzmenge ist die Menge $\mathbb{C}(I_{15c}) = \{(b^k a, v) \mid |v|_a = k, k \in \mathbb{N}\}$, die nach erneutem Betrachten der Regeln von I_{15c} auch intuitiv richtig ist.

Werfen wir einen Blick auf den unproduktiven Teil des Transducers, der über die Transition $\downarrow 0 \overset{(a,\varepsilon)}{\Longleftrightarrow}_{I_{15c}} \updownarrow\varepsilon$ erreichbar ist. Zwar trägt er nicht zu Elementen der Koinzidenzmenge bei, doch wenn es im unproduktiven Teil Schleifen gibt, dann wird dadurch analog zu einer unendlichen Lösung ein *unendliches Element* der Koinzidenzmenge beschrieben, denn für jede Beschriftung (u, v) eines Pfades im Transducer zu einem Überhang x gilt $s_\uparrow h(u) x_\downarrow = s_\downarrow g(v) x_\uparrow$. Das heißt aber, dass $s_\uparrow h(u)$ und $s_\downarrow g(v)$ vergleichbar sind. Als Beispiel sei im obigen Transducer der Pfad von $\uparrow 101$ nach $\uparrow 01$ genannt, der mit (ab, aab^ω) beschriftet ist. Nicht alle unendlichen Elemente der Koinzidenzmenge sind durch die beschränkte bidirektionale Suchrelation beschrieben, so gibt es keinen unendlichen Pfad, der mit $(a, aaab^\omega)$ beschriftet ist und *prinzipiell* kann es keinen mit (a^ω, ε) beschrifteten Pfad in einer endlichen Teilmenge von $\hookrightarrow_{I_{15c}}$ geben. ⌟

a

Bedingte Koinzidenz

Nach 47d ist die Gleichheitsmenge der Schnitt der Koinzidenzmenge mit der Identität. Anders formuliert liefert uns ein Element der Koinzidenzmenge eine Lösung, unter der Bedingung, dass es ebenfalls in der Identität enthalten ist. Nun ist einerseits unentscheidbar, ob es Lösungen gibt und andererseits entscheidbar, ob es echte Elemente in der Koinzidenzmenge gibt. Daher stellt sich die Frage, welche Bedingungen an die Koinzidenzmenge gestellt werden können, ohne die Entscheidbarkeit zu verlieren.

b

Gleiche Länge

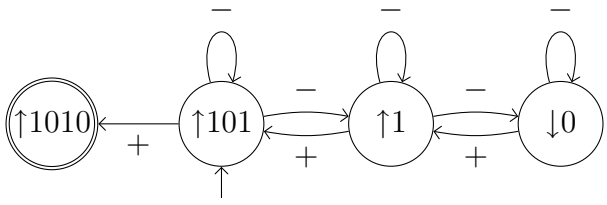
Da Elemente aus der Identität nicht nur gleich, sondern auch gleich lang sind, liegt es nahe zu fragen, ob es entscheidbar ist, ob die Menge $\mathbb{C}^{||}(I) = \mathbb{C}(I) \cap \{(u,v) \mid |u| = |v|\}$ leer ist oder nicht. Sheila Greibach hat diese Frage 1975 positiv beantwortet. (Greibach [7]) Wir wollen eine Begründung geben, die mit den hier verwendeten Begriffen arbeitet: Sei T der Transducer aus Satz 56a, der die Koinzidenzmenge akzeptiert. Wir konstruieren aus T einen endlichen Automaten, in dem wir die Beschriftungen der Transitionen $x \overset{a}{\Longrightarrow}_I y$ in $+$ oder in $-$ ändern, je nachdem, ob a in $A^{\uparrow}$ oder in $A^{\downarrow}$ enthalten ist. Der Automat akzeptiert eine reguläre Sprache L über $\{+,-\}$, deren Elemente die Positionen der Beschriftungen von akzeptierenden Pfaden durch T beschreiben. Daraus folgt, dass es genau dann Elemente (u,v) mit $|u| = |v|$ in $\mathbb{C}(I)$ gibt, wenn $L \cap E$ mit $E = \{w \mid |w|_+ = |w|_-\}$ nicht leer ist. Nun ist E kontextfrei und kontextfreie Sprachen sind abgeschlossen gegenüber Schnitten mit regulären Sprachen. Außerdem sind für kontextfreie Sprachen Leere und Endlichkeit entscheidbar. Schließlich haben wir T und damit $L \cap E$ effektiv gegeben und können also $L \cap E$ effektiv aufzählen. Jedes Element w aus $L \cap E$ entspricht Pfaden durch T, die mit Kenntnis von w rekonstruiert werden können.

c

Folgerung. *Es ist entscheidbar, ob $\mathbb{C}^{||}(I)$ leer ist. Es ist entscheidbar, ob $\mathbb{C}^{||}(I)$ endlich ist. $\mathbb{C}^{||}(I)$ kann effektiv aufgezählt werden.*

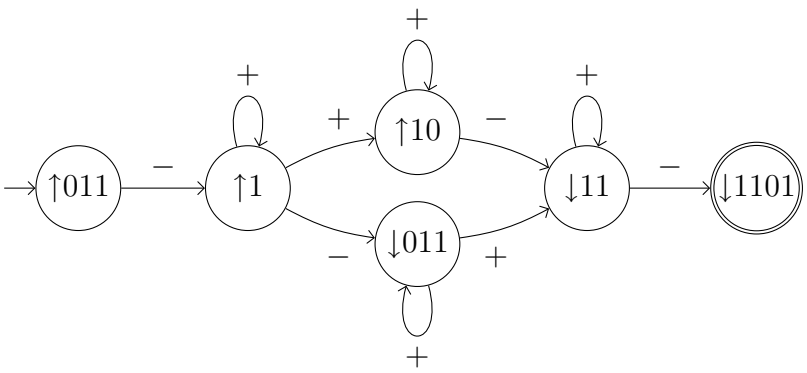
d

Beispiel: Der Transducer aus 57a für die Koinzidenzmenge $\mathbb{C}(I_{15c})$ der Instanz $I_{15c} = |\,{}^{101}_{\varepsilon}\,|\,{}^{0}_{10}\,{}^{01}_{\varepsilon}\,|\,{}^{\varepsilon}_{1010}\,|$ führt auf den (reduzierten) endlichen Automaten



für den $L \cap E$ offensichtlich unendlich groß ist. Aus $-+---+++$ können wir die beiden Pfade mit den Beschriftungen $(bbba, aaab) = (\varepsilon, a)(b, \varepsilon)(\varepsilon, a)(\varepsilon, a)(\varepsilon, b)(b, \varepsilon)(b, \varepsilon)(a, \varepsilon)$ und $(bbba, aaba) = (\varepsilon, a)(b, \varepsilon)(\varepsilon, a)(\varepsilon, b)(\varepsilon, a)(b, \varepsilon)(b, \varepsilon)(a, \varepsilon)$ rekonstruieren. Das Paar $(bbba, baaa) \in L \cap E$ kann aus $---++-++$ rekonstruiert werden. ⌟

Beispiel: Der Transducer aus 56e für die Koinzidenzmenge $\mathbb{C}(I_{53b})$ der Instanz $I_{53b} = |\,^{011}_{\varepsilon}\,|\,^{\varepsilon\ \ 0}_{01\ 1011}\,|\,^{1101}_{\varepsilon}\,|$ führt auf den endlichen Automaten *a*



für den $L \cap E$ die endliche Sprache $\{--+++-, -+-++-, -++-+-, -++--\}$ ist, aus der die Paare (aab, aba), (aba, aba) und (baa, aba) rekonstruiert werden können. ⌟

Gleiches kommutatives Bild *b*

Nach der Frage nach Elementen der Koinzidenzmenge mit gleicher Länge der beiden Komponenten schließt sich natürlich die Frage nach Elementen der Koinzidenzmenge an, deren Komponenten nicht nur gleiche Länge, sondern sogar das gleiche kommutative Bild haben. Oscar Ibarra und Chul Kim haben 1976 für herkömmliche Instanzen gezeigt, dass Leere und Endlichkeit der Menge $\mathbb{C}^{\psi}(I) = \mathbb{C}(I) \cap \{(u, v) \mid \psi_A(u) = \psi_A(v)\}$ entscheidbar sind. (Ibarra, Kim [17]) Sie konstruieren einen einfachen Mehrkopf-Kellerautomaten, der die Menge aller h-Bilder von Elementen aus $\mathbb{C}^{\psi}(I)$ akzeptiert. Einfache Mehrkopf-Kellerautomaten sind Kellerautomaten mit mehreren Köpfen, von denen aber nur einer in der Lage ist, Symbole aus A voneinander zu unterscheiden. Die anderen Köpfe können nur als Zähler verwendet werden. Für einfache Mehrkopf-Kellerautomaten sind Leere und Endlichkeit entscheidbar.

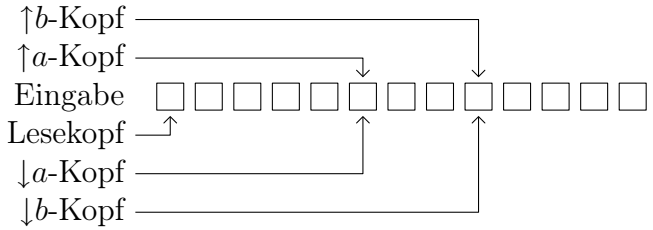
Angelehnt an die Konstruktion aus 58b wollen wir explizit die Konstruktion eines teilweise blinden endlichen Mehrkopfautomaten angeben. Dabei handelt es sich um einen endlichen Automaten mit einem normalen Lesekopf und k blinden Köpfen, die nicht zwischen verschiedenen Symbolen aus A

unterscheiden können. In einem Schritt wechselt ein solcher Automat nichtdeterministisch abhängig vom aktuellen Zustand und dem Symbol unter dem Lesekopf in einen Nachfolgezustand und darf jeden Kopf, also blinde Köpfe und den Lesekopf, um eine Position nach rechts versetzen. Wohlgemerkt *dürfen* die Köpfe versetzt werden, sie können ihre Position auch beibehalten. Initial befinden sich alle Köpfe über dem ersten Symbol der Eingabe und ein Wort wird akzeptiert, wenn der Automat einen Endzustand einnimmt und sich alle Köpfe unmittelbar hinter dem letzten Symbol der Eingabe befinden. Wird während der Berechnung ein Kopf weiter als ein Symbol nach dem Wortende nach rechts geschoben, gilt das Wort als nicht akzeptiert.

Oscar Ibarra und Bala Ravikumar zeigen, dass sich teilweise blinde endliche Mehrkopfautomaten und umkehrbeschränkten Mehrzählermaschinen gegenseitig simulieren können. Für umkehrbeschränkte Mehrzählermaschinen sind Leere und Endlichkeit entscheidbar. (Ibarra, Ravikumar [18], Ibarra [19])

Für die Konstruktion des teilweise blinden endlichen Mehrkopfautomaten gehen wir erneut vom Transducer T aus Satz 56a aus und konstruieren einen Automaten C mit $2|B|$ blinden Köpfen, der über $A^{\updownarrow}$ operiert. Wir konstruieren also einen Automaten, der nicht wie der von Oscar Ibarra und Chul Kim h-Bilder von Elementen aus $\mathbb{C}^{\psi}(I)$ akzeptiert, sondern *Zerlegungen* solcher Elemente.

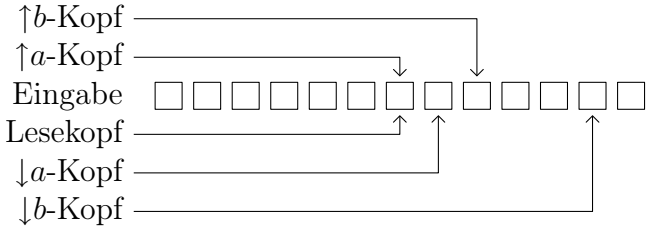
Die $2|B|$ blinden Köpfe dienen der Speicherung der Anzahl der Symbole aus B, die in der $\uparrow$- oder in der $\downarrow$-Komponente noch zu lesen sind. Initial wählt C nichtdeterministisch Anzahlen für jedes Symbol aus B und verschiebt die blinden Köpfe entsprechend weit nach rechts. Genauer wählt C nicht die Anzahl, sondern die Anzahl der virtuell bereits gelesenen Symbole. Würde der Automat C zum Beispiel eine Eingabe der Länge 13 erwarten, die 8-mal das Symbol a und 5-mal das Symbol b enthält, würde er nichtdeterministisch in die Situation



verzweigen. Der Lesekopf befindet sich immer noch über dem ersten Symbol der Eingabe, während die Köpfe für die Symbole a und b in den Positionen $\uparrow$ und $\downarrow$ bereits um $5 = 13 - 8$ und $8 = 13 - 5$ versetzt wurden.

Nun interpretiert C den Transducer T als endlichen Automaten über $A^{\updownarrow}$ und ändert seinen Zustand genau wie T. Zusätzlich versetzt er nach jedem

gelesenen Symbol den zu diesem Symbol gehörenden blinden Zähler um eine Stelle nach rechts. Hat der Automat C zum Beispiel zwei Mal das Symbol (ε, a), ein Mal das Symbol (a, ε) und drei Mal das Symbol (ε, b) gelesen, dann befindet er sich nun in der Situation



Der interne Zustand hat sich entsprechend der Tabelle von T geändert. Falls T für gewisse Eingaben keine Transitionen vorsieht, dann begibt sich C in einen nicht akzeptierenden Fehlerzustand, in dem nur noch die restliche Eingabe gelesen wird und der nicht mehr verlassen wird.

Eine Eingabe wird von C akzeptiert, wenn sie zu einem Endzustand von T führt *und* alle blinden Zähler direkt hinter dem letzten Symbol der Eingabe stehen. Es ist klar, dass C nur Zerlegungen von Paaren aus $\mathbb{C}^\psi(I)$ akzeptiert. Andererseits kann jedes Paar in $\mathbb{C}(I)$ passend für einen Pfad durch T zerlegt werden und es gibt eine zugehörige akzeptierende Rechnung von C.

Folgerung. *Es ist entscheidbar, ob $\mathbb{C}^\psi(I)$ leer ist. Es ist entscheidbar, ob $\mathbb{C}^\psi(I)$ endlich ist. $\mathbb{C}^\psi(I)$ kann effektiv aufgezählt werden.* *a*

Die Konstruktion von Oscar Ibarra und Chul Kim funktioniert im Prinzip genauso. (Ibarra, Kim [17]) Auch dort wird initial eine Anzahl von Symbolen geraten und dann verifiziert, dass die Eingabe ein h-Bild von dieser Anzahl von Symbolen ist. Hier haben wir aber den Transducer T zu Grunde gelegt, der automatisch sicherstellt, dass eine akzeptierte Eingabe nicht nur das richtige kommutative Bild hat, sondern gleichzeitig auch ein Element der Koinzidenzmenge ist. Ibarra und Kim steht T nicht zur Verfügung. Ibarra und Kim verifizieren beim Raten der Anzahlen für jedes neue Symbol, dass die entsprechenden Bilder vergleichbar sind, eine Arbeit, die hier in die Konstruktion von T verlagert wurde.

Der Vorteil unserer Konstruktion liegt also in der direkten Akzeptanz von Zerlegungen von Elementen aus $\mathbb{C}^\psi(I)$ und in der Trennung in die auch anderweitig nützliche Konstruktion von T und die spezielle Konstruktion von C zur Erkennung von $\mathbb{C}^\psi(I)$.

a **Gleiches verallgemeinertes kommutatives Bild**

In 58b fordern wir von Elementen (u, v) der Koinzidenzmenge, dass u und v gleiche Länge haben, in 59b, dass sie das gleiche kommutative Bild haben, also Permutationen sind. In beiden Fällen ergeben sich Mengen, für die Leere und Endlichkeit entscheidbar sind und die effektiv aufgezählt werden können. Wir wissen außerdem, dass $\mathrm{id}(\mathrm{Eq}(I)) = \mathbb{C}(I) \cap \mathrm{id}(A^*) \subseteq \mathbb{C}^{\psi}(I) \subseteq \mathbb{C}^{||}(I)$ ist. Es stellt sich die Frage, ob sich mit Hilfe weiterer Bedingungen weitere Mengen zwischen $\mathrm{id}(\mathrm{Eq}(I))$ und $\mathbb{C}^{\psi}(I)$ einfügen lassen, für die Leere und Endlichkeit entscheidbar sind. Auch diese Frage ist bereits positiv beantwortet: Juhani Karhumäki hat 1980 das *verallgemeinerte kommutative Bild* $\psi_k(w)$ eines Wortes w definiert, in dem nicht mehr das Auftreten einzelner Symbole in w gezählt wird, sondern das Auftreten der *Segmente* der Länge k. Zwei Wörter u und v sind k-äquivalent, $u \equiv_k v$, wenn sie das selbe verallgemeinerte kommutative Bild haben. (Karhumäki [21])

Es ist klar, dass das verallgemeinerte kommutative Bild mehr Information über w sammelt als das normale. Wenn wir mit $\psi_0(w)$ die Länge von w bezeichen und mit $\mathbb{C}^{\psi}_k(I)$ die Menge $\mathbb{C}(I) \cap \{(u, v) \mid u \equiv_k v\}$, dann gilt offensichtlich einerseits die Inklusionskette

b
$$\mathrm{id}(\mathrm{Eq}(I)) \subseteq \cdots \subseteq \mathbb{C}^{\psi}_1(I) \subseteq \mathbb{C}^{\psi}_0(I)$$

und andererseits $\mathrm{id}(\mathrm{Eq}(I)) = \bigcap_{k\geq 0} \mathbb{C}^{\psi}_k(I)$. Juhani Karhumäki zeigt, dass sich Fragen über Eigenschaften des verallgemeinerten kommutativen Bildes auf Fragen über Eigenschaften des normalen kommutativen Bildes bezüglich eines größeren Alphabets zurückführen lassen. Damit folgt aus der Entscheidbarkeit von Leere und Endlichkeit von $\mathbb{C}^{\psi}_1(I)$ auch die Entscheidbarkeit von Leere und Endlichkeit von $\mathbb{C}^{\psi}_k(I)$.

Wie in 59b können wir auf dem Transducer T aus Satz 56a aufbauen und einen teilweise blinden endlichen Mehrkopfautomaten konstruieren, der $\mathbb{C}^{\psi}_k(I)$ akzeptiert. Der Automat hat für jedes Wort bis zur Länge k zwei blinde Köpfe. Initial rät er die Anzahl der Segmente der Länge k und verschiebt die blinden Köpfe entsprechend. Der interne Zustand ist nun nicht mehr einfach der von T, sondern der Automat speichert zusätzlich die letzten k gelesenen Symbole. Hier sind Symbole wieder Elemente aus $A^{\updownarrow}$, der Automat versucht also eine Zerlegung eines Elements aus $\mathbb{C}^{\psi}_k(I)$ zu erkennen. Beim Lesen eines Symbols verschiebt er den zu den letzten k gelesenen Symbolen gehörenden blinden Kopf nach rechts und akzeptiert, falls der interne Zustand am Ende einem Endzustand von T entspricht und alle blinden Köpfe direkt hinter der Eingabe stehen. So werden nur Wörter akzeptiert, die Zerlegungen von Elementen aus $\mathbb{C}^{\psi}_k(I)$ entsprechen. Andererseits lässt sich jedes Element aus $\mathbb{C}^{\psi}_k(I)$ zu einer akzeptierenden Berechnung passend zerlegen.

Folgerung. *Es ist entscheidbar, ob $\mathbb{C}_k^\psi(I)$ leer ist. Es ist entscheidbar, ob $\mathbb{C}_k^\psi(I)$ endlich ist. $\mathbb{C}_k^\psi(I)$ kann effektiv aufgezählt werden.* a

Die Kette (62b) liefert also eine Folge oberer Approximationen der Gleichheitsmenge, für deren Elemente Leere und Endlichkeit jeweils entscheidbar sind. Es liegt der interessante Fall eines mit $k \in \mathbb{N}$ parametrisierten Problems vor, für das für jedes $k \in \mathbb{N}$ ein Entscheidungsalgorithmus existiert, jedoch kein einzelner Algorithmus, der das Problem für jedes k entscheidet. Zusammen mit der Kette unterer Approximationen in (29a) können wir uns sowohl von oben als auch von unten an die Gleichheitsmenge annähern.

Nachfolger b

Bisher werden mit Hilfe der allgemeinen Suchrelation $\hookmapsto$, der abgeleiteten beschränkten Suchrelation $\hookrightarrow$ und der vollständigen beschränkten bidirektionalen Suchrelation $\leftrightharpoons$ einerseits in 54a ein Transducer konstruiert, der die Koinzidenzmenge akzeptiert und andererseits in 59b eine Kette oberer Approximationen der Gleichheitsmenge.

Es folgt eine neue Darstellung der Theorie der Nachfolger: Ein Nachfolger einer Instanz I ist eine Instanz, deren Regeln die Bausteine der Elemente in der Koinzidenzmenge von I sind. In 64a werden diese Bausteine bereitgestellt. Für herkömmliche Instanzen handelt es sich schlicht um die Basis der Koinzidenzmenge. Für allgemeine Instanzen ist die Koinzidenzmenge nicht zwingend ein Monoid und wir benötigen für die Zerlegung einer Koinzidenz spezielle Start- und Endblöcke. In 72d zeigen wir, wie beliebige Koinzidenzen allgemeiner Instanzen in Bausteine zerlegt werden können, wobei die Bausteinmengen entweder rationale Relationen mit effektiv gegebenem Transducer sind, oder wenigstens effektiv aufgezählt werden können.

In 75b definieren wir den Begriff des Nachfolgers formal und zeigen, dass Lösbarkeit von Instanz und Nachfolger eng zusammenhängen. Darüber hinaus sind endliche Nachfolger von I einfacher als I. Es entsteht ein Graph von Nachfolgern, der genau dann lösbare transitive Nachfolger enthält, wenn I lösbar ist. Wir geben außerdem Beispiele für Instanzen, bei denen bisher unbekannt ist, ob es echte Lösungen gibt, die aber nun mit Hilfe des Nachfolgergraphen entschieden werden können.

Als konkrete Anwendung zeigt 79b wie markierte Instanzen entschieden werden können. Für markierte Instanzen ist der Nachfolgergraph stets endlich und kann effektiv konstruiert werden. Die Entscheidbarkeit binärer Instanzen folgt schließlich aus der Entscheidbarkeit markierter Instanzen.

a

Zerlegen von Koinzidenzen

Nach Satz 56a ist die Koinzidenzmenge eine rationale Relation und wir können effektiv einen Transducer T konstruieren, der $\mathbb{C}(I)$ akzeptiert. Betrachten wir T für den Fall herkömmlicher Instanzen, dann hat er den einzigen Start- und Endzustand $(\varepsilon, \varepsilon)$. Das bedeutet nichts anderes, als das $\mathbb{C}(I)$ für herkömmliche Instanzen ein Monoid ist. Es handelt sich um ein Monoid mit Länge und der Basis $\mathbb{C}^\circ(I) \setminus (\mathbb{C}^\circ(I)\,\mathbb{C}^\circ(I))$, deren Elemente Blöcke genannt werden. Daraus folgt, dass jedes Element der Koinzidenzmenge und damit auch jedes Element aus $\mathrm{id}(\mathrm{Eq}(I))$ in Blöcke zerlegt werden kann. Die Idee des Nachfolgers ist, nicht nach Lösungen von I zu suchen, sondern nach Lösungen einer Instanz, deren Regeln Blöcke von I sind.

Doch zunächst wollen wir Mengen von Blöcken definieren, die die Zerlegung der Koinzidenzen allgemeiner und nicht nur der herkömmlichen Instanzen erlauben.

b

Innere Blöcke

Wir nennen die Menge $\mathbb{I}(I) = \{(u, v) \mid h(u) = g(v)\}$, der Paare auf denen die Morphismen übereinstimmen, die Menge der *inneren Blöcke*. Für herkömmliche Instanzen ist $\mathbb{I}(I) = \mathbb{C}(I)$. Die Menge der *echten inneren Blöcke* ist $\mathbb{I}^\circ(I) = \mathbb{I}(I) \setminus \{(\varepsilon, \varepsilon)\}$. Bei $\mathbb{I}(I)$ handelt es sich um ein Monoid mit Länge, das von $\mathbb{B}(I) = \mathbb{I}^\circ(I) \setminus (\mathbb{I}^\circ(I)\,\mathbb{I}^\circ(I))$ erzeugt wird, es gilt $\mathbb{I}(I) = \mathbb{B}(I)^*$. Die Elemente von $\mathbb{B}(I)$ sind die eigentlichen *Blöcke*. Für herkömmliche Instanzen kann nun bereits jedes Element der Koinzidenzmenge in Blöcke zerlegt werden.

Der Transducer aus Satz 56a, der die Koinzidenzmenge akzeptiert, operiert über der Relation $\leftrightharpoons_I$, die aus den beiden Hüllen $\hookrightarrow_I^\star(s)$ und $\circlearrowleft\!\!\hookrightarrow_{I^{\mathrm{R}}}^\star(f^{\mathrm{R}})$ besteht. Wir konstruieren auf die selbe Art einen Transducer, der die Menge $\mathbb{I}(I)$ der inneren Blöcke akzeptiert. Einziger Unterschied ist, dass die Verwendung der Hülle $\hookrightarrow_I^\star((\varepsilon, \varepsilon))$ genügt, das heißt, wir berechnen die Koinzidenzmenge der herkömmlichen Instanz mit den selben Regeln wie I.

c

Folgerung. *Die Menge $\mathbb{I}(I)$ der inneren Blöcke ist eine rationale Relation. Ein Transducer, der $\mathbb{I}(I)$ akzeptiert, kann effektiv konstruiert werden.*

Tatsächlich kann der Beweis 55a direkt übernommen werden und da $s = f = (\varepsilon, \varepsilon)$ gilt, kann der Fall nicht eintreten, dass die beschränkte Suchrelation verlassen werden muss.

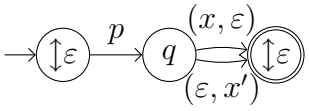
Wir können auch etwas über die Eindeutigkeit der Zerlegung sagen. Dazu nennen wir eine Instanz *ε-trivial (für a)*, wenn es ein $a \in A$ mit $h(a) = g(a) = \varepsilon$ gibt. Aus einer Lösung $u = u_0 \cdots u_{k-1} \in \mathrm{Eq}(I)$ einer ε-trivialen Instanz I kann stets die Menge $a^* u_0 a^* \cdots a^* u_{k-1} a^*$ von echten Lösungen von

I konstruiert werden. Wir können ε-triviale Instanzen von der Betrachtung ausschließen, ohne wirklich etwas zu verlieren. Doch zurück zur Eindeutigkeit der Zerlegung. Hier spielen ε-triviale Instanzen eine Sonderrolle, es gilt das

Lemma. *Die Menge $\mathbb{I}(I)$ der inneren Blöcke wird genau dann von der Menge $\mathbb{B}(I)$ der Blöcke frei erzeugt, wenn I nicht ε-trivial ist.* *a*

Beweis: Nehmen wir zunächst an, es gibt ein $a \in A$ mit $h(a) = g(a) = \varepsilon$. Dann sind sowohl (a, ε) als auch (ε, a) Blöcke. Außerdem ist $(a, a) \in \mathbb{I}(I)$ ein innerer Block, für den es die Zerlegungen $(a, a) = (a, \varepsilon)(\varepsilon, a)$ und $(a, a) = (\varepsilon, a)(a, \varepsilon)$ gibt. Also wird $\mathbb{I}(I)$ nicht frei von $\mathbb{B}(I)$ erzeugt. *b*

Wenn andererseits $\mathbb{I}(I)$ nicht frei von $\mathbb{B}(I)$ erzeugt wird, dann gibt es einen minimalen inneren Block $v \in \mathbb{I}(I)$, der sich auf zwei verschiedene Arten in Blöcke zerlegen lässt. Es gilt dann $v = bu = b'u'$ für voneinander verschiedene Blöcke $b \neq b'$. Sei p das längste gemeinsame Präfix von b und b', das heißt, es gilt ohne Beschränkung der Allgemeinheit $b = p(x, \varepsilon)$ und $b' = p(\varepsilon, x')$, wobei sowohl x als auch x' beide nicht leer sind, anderenfalls wäre b oder b' kein Block. Sprechen wir über Pfade im Transducer für $\mathbb{I}(I)$ aus 64c, dann haben wir also die Situation



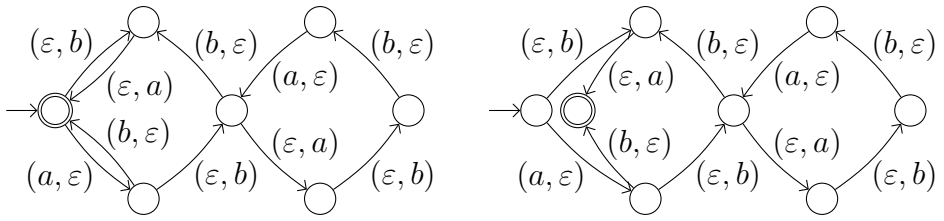
vorliegen. Nun muss $q = (\varepsilon, \varepsilon)$ sein, anderenfalls hätte $h(x)$ oder $g(x')$ negative Länge. Wenn aber $q = (\varepsilon, \varepsilon)$ ist, dann muss $p = (\varepsilon, \varepsilon)$ sein, anderenfalls wären b und b' keine Blöcke. Daraus folgt $h(x) = g(x') = \varepsilon$ und, da x und x' vergleichbar sind, dass es ein $a \in A$ mit $h(a) = g(a) = \varepsilon$ gibt. □

Wir können die Menge $\mathbb{B}(I)$ aller Blöcke ebenfalls effektiv aufzählen: Der Transducer, der $\mathbb{I}(I)$ akzeptiert, hat den Start- und Endzustand $(\varepsilon, \varepsilon)$. Spalten wir diesen Zustand auf, so dass eine Kopie der neue Startzustand ist und alle ausgehenden Transitionen behält und die andere Kopie der neue Endzustand mit allen eingehenden Transitionen, dann erhalten wir einen Transducer, der eine Obermenge von $\mathbb{B}(I)$ akzeptiert. Wir zählen diese Obermenge so auf, dass kurze Wörter zuerst genannt werden und übernehmen alle die Wörter in eine zweite Aufzählung, die sich nicht aus kürzeren bereits genannten Wörtern zusammensetzen lassen. Dann zählt die zweite Aufzählung alle Blöcke auf.

Folgerung. *Die Menge $\mathbb{B}(I)$ aller Blöcke kann effektiv aufgezählt werden.* *c*

Es stellt sich die Frage, ob entscheidbar ist, ob $\mathbb{B}(I)$ endlich ist. Klar ist, dass $\mathbb{B}(I)$ endlich ist, wenn der Transducer mit gespaltenem Start- und Endzustand

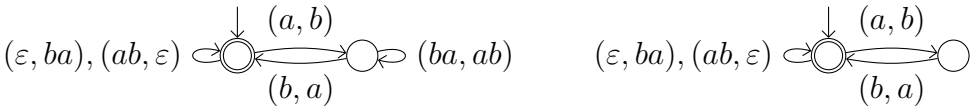
eine endliche Sprache akzeptiert. Es ist für Transducer im Allgemeinen möglich, dass es zwar eine endliche Basis gibt, das Aufspalten aber zu einem Transducer führt, der eine unendliche Sprache akzeptiert. Als Beispiel sei der Transducer



links genannt, der die Sprache $\{(ab, \varepsilon), (\varepsilon, ba)\}^*$ akzeptiert, für den das Aufspalten aber den Transducer rechts liefert, der eine unendlich große Sprache akzeptiert.

Für *sequentielle* Transducer, das sind Transducer, die Sprachen L mit $\forall u : |\{v \mid (u, v) \in L\}| \leq 1$ akzeptieren, gibt es zwar einen Minimierungsalgorithmus (Mohri [28]), aber hier haben wir es nicht mit sequentiellen Transducern zu tun. Die Menge $\mathbb{I}(I)$ der inneren Blöcke ist im Allgemeinen kein Graph einer rationalen Funktion (Berstel [1]), die Beispiele 68b und 68c zeigen, dass das bereits für nichtlöschende Instanzen eintreffen kann.

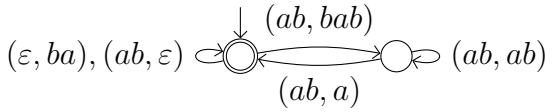
Darüber hinaus sind bis auf Leere und Endlichkeit im Prinzip alle interessanten Fragen für rationale Relationen unentscheidbar, insbesondere Enthaltensein und Gleichheit. (Berstel [1]) Ein Algorithmus, der als Eingabe den Transducer für $\mathbb{I}(I)$ erhält und einen Transducer für eine Obermenge von $\mathbb{B}(I)$ ausgeben soll, die genau dann endlich ist, wenn $\mathbb{B}(I)$ endlich ist, kann also nur schrittweise Transducer umformen und zum Beispiel nicht in einer Aufzählung aller Transducer nach einem passenden suchen. Der Transducer oben links kann zunächst zu den beiden Transducern



umgeformt werden. Die linke Darstellung ist die (genauer: eine) liberalisierte Version und die mit (ba, ab) beschriftete Schleife darf entfernt werden, da sie lediglich den mit (b, a) und (a, b) beschrifteten Pfad beschreibt, sie ist *zerlegbar*. Erneutes Liberalisieren und Entfernen der zerlegbaren Pfade führt auf die beiden Transducer

Nun erkennen wir am rechten Transducer die endliche Basis für die Sprache des Originaltransducers.

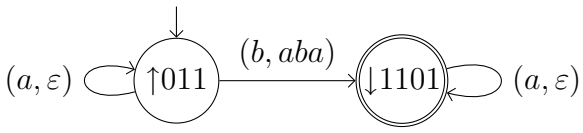
Dieses Vorgehen hat zunächst ein Effizienzproblem, denn es gibt nicht *die* liberalisierte Version eines Transducers. In der liberalisierten Version



ist zum Beispiel kein Pfad weiter zerlegbar und der Algorithmus stoppt. Wir können natürlich alle möglichen Liberalisierungen vornehmen und jeweils zerlegbare Pfade entfernen: Das Effizienzproblem.

Erschwerend kommen löschende Instanzen hinzu, denn die Regel $(h(a), \varepsilon)$ für ein $a \in A$ entspricht einer mit (ε, a) beschrifteten einfachen Schleife. Eine solche Schleife kann nicht weiter zerlegt werden und würde den Transducer aus Beispiel 56e vor weiteren Umformungen schützen. Allerdings können solche Schleifen manchmal entlang von Pfaden in Richtung Startzustand oder in Richtung Endzustand wandern. Genauer können wir die Situation

links in die Situation rechts überführen, falls der linke Zustand kein Endzustand ist. Natürlich können wir die Schleife auch wieder in die andere Richtung zurück schieben, die akzeptierte Sprache bleibt unverändert. Mit dieser Technik ist es möglich, den Transducer aus Beispiel 56e in die Darstellung

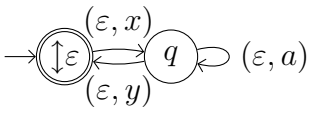


zu überführen. Wir können uns überlegen, dass mit Hilfe dieser Operationen, also Schieben der (ε, a)-Schleifen, Liberalisieren und Entfernen der zerlegbaren Schleifen, genau dann aus dem Transducer für $\mathbb{I}(I)$ ein Transducer mit dem einzigen Zustand $(\varepsilon, \varepsilon)$ konstruiert werden kann, wenn $\mathbb{B}(I)$ endlich ist.

Lemma. *Es ist entscheidbar, ob $\mathbb{B}(I)$ endlich ist. Endliche Blockmengen können effektiv ermittelt werden.* *a*

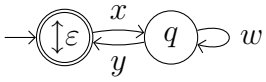
Beweis: Sei T der Transducer für $\mathbb{I}(I)$ aus 64c. Dann hat T den einzigen Start- und einzigen Endzustand $(\varepsilon, \varepsilon)$ und eventuell weitere Zustände. Wir konstruieren aus T in zwei Schritten einen Transducer, der genau dann den einzigen Zustand $(\varepsilon, \varepsilon)$ hat, wenn $\mathbb{B}(I)$ endlich ist. *b*

Wir verschieben zunächst alle einfachen Schleifen zum Zustand $(\varepsilon, \varepsilon)$. Wenn sich Schleifen nicht bis zum Zustand $(\varepsilon, \varepsilon)$ verschieben lassen, dann ist $\mathbb{B}(I)$ nicht endlich: Wir haben dann ohne Beschränkung der Allgemeinheit die Situation



mit $q \neq (\varepsilon, \varepsilon)$ vorliegen. Gäbe es eine endliche Basis, dann könnte $(\varepsilon, x)(\varepsilon, a)^*$ in Blöcke zerlegt werden. Daraus folgt, dass es für genügend großes k ein Präfix von (ε, xa^k) gibt, das ein innerer Block ist. Dann wäre aber $q = (\varepsilon, \varepsilon)$.

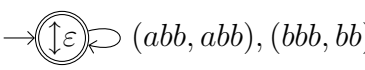
Im zweiten Schritt konstruieren wir *alle* Transducer, die alle Reihenfolgen von Anwendungen der Operationen Liberalisieren und Löschen von zerlegbaren Schleifen beschreiben. Das sind endlich viele, da entweder die Anzahl der Zustände oder die der Transitionen abnimmt. Angenommen $\mathbb{B}(I)$ ist endlich, aber keiner dieser Transducer besteht nur aus dem Zustand $(\varepsilon, \varepsilon)$. Dann haben wir die Situation



vorliegen. Da $\mathbb{B}(I)$ endlich ist, gibt es ein $k \in \mathbb{N}$, so dass $xw^k y = lr$ mit einem inneren Block l ist, der x als echtes Präfix hat. Daraus folgt, dass es ein Suffix u von l und ein Präfix v von r mit $uv = w$ gibt. Das heißt aber nichts anderes, als dass w zerlegbar ist, denn von q kommen wir mit u zu $(\varepsilon, \varepsilon)$ und von dort mit v zurück zu q. □

Es sei angemerkt, dass dieser Beweis *nicht* für *beliebige* Transducer gilt. Vielmehr haben wir essenziell benutzt, dass dem Transducer für $\mathbb{I}(I)$ die beschränkte Suchrelation $\hookrightarrow$ zu Grunde liegt. Deren Eigenschaften erlauben die Schlüsse auf $q = (\varepsilon, \varepsilon)$ einerseits und die Zerlegbarkeit von w andererseits.

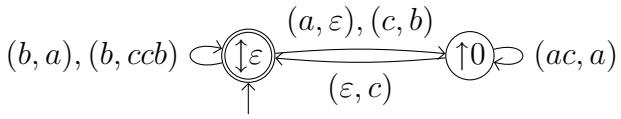
a **Beispiel:** In 52a wird gezeigt, dass die Koinzidenzmenge $\mathbb{C}(I_{15d})$ der Instanz $I_{15d} = \| \begin{smallmatrix} abb & bb & cbbb \\ a & bbb & c \end{smallmatrix} \|$ eine endliche Basis besitzt. Konstruieren wir den Transducer für $\mathbb{I}(I_{15d})$ explizit, dann erhalten wir mit



einen Transducer, der nach Aufspaltung auch die Berechnung von $\mathbb{B}(I_{15d}) = \{(abb, abb), (bbb, bb), (c, cb)\}$ erlaubt. ⌟

b **Beispiel:** Betrachten wir die Familie $I^k_{68b} = \| \begin{smallmatrix} 0 & 0^k1 & 1 \\ 0^k1 & 1 & 0 \end{smallmatrix} \|$, die die Instanz $I_{16b} = \| \begin{smallmatrix} 0 & 001 & 1 \\ 001 & 1 & 0 \end{smallmatrix} \| = I^2_{68b}$ aus 16b enthält. Alle diese Instanzen haben endliche Blockmengen, es gilt $\mathbb{B}(I^k_{68b}) = \{(a, c), (b, a), (c, b), (a^k c, a), (b, c^k b)\}$. ⌟

c **Beispiel:** Für die Instanz $J_{16b} = \| \begin{smallmatrix} 0 & 001 & 10 \\ 001 & 1 & 0 \end{smallmatrix} \|$ hat der Transducer für $\mathbb{I}(J_{16b})$ genau 10 Zustände und 19 Transitionen. Liberalisiert hat er die Gestalt



und damit ergeben sich mit $\mathbb{B}(J_{16b}) = \{(b,a),(b,ccb)\} \cup \{(a(ac)^i, a^ic) \mid i \in \mathbb{N}\} \cup \{(c(ac)^i, ba^ic) \mid i \in \mathbb{N}\}$ unendlich viele Blöcke. ⌟

Beispiel: Für die Instanz $I_{15c} = |\,{}^{101}_{\varepsilon}\,|\,{}^{0}_{10}\;{}^{01}_{\varepsilon}\,|\,{}^{\varepsilon}_{1010}\,|$ gilt $\mathbb{B}(I_{15c}) = \{(\varepsilon, b)\}$. ⌟ *a*

Beispiel: Für die Instanz $I_{53a} = |\,{}^{0}_{\varepsilon}\,|\,{}^{1}_{0}\;{}^{0}_{1}\,|\,{}^{\varepsilon}_{0}\,|$ gilt $\mathbb{B}(I_{53a}) = \{(a,b),(b,a)\}$. ⌟ *b*

Beispiel: Für die Instanz $I_{53b} = |\,{}^{011}_{\varepsilon}\,|\,{}^{\varepsilon}_{01}\;{}^{0}_{1011}\,|\,{}^{1101}_{\varepsilon}\,|$ gilt $\mathbb{B}(I_{53b}) = \{(a,\varepsilon)\}$. ⌟ *c*

Start- und Endblöcke *d*

Um die Koinzidenzen allgemeiner Instanzen zu zerlegen, benötigen wir zusätzlich die *Startblöcke* $\mathbb{S}(I) = \{(u,v) \mid s_\uparrow h(u) = s_\downarrow g(v)\} \setminus (\mathbb{S}(I)\,\mathbb{I}^\circ(I))$ und die *Endblöcke* $\mathbb{F}(I) = \{(u,v) \mid h(u)f_\uparrow = g(v)f_\downarrow\} \setminus (\mathbb{I}^\circ(I)\,\mathbb{F}(I))$. Diese Mengen beschreiben Wege von s zu $(\varepsilon,\varepsilon)$ und von $(\varepsilon,\varepsilon)$ zu $\overline{f}$, die keine Schleife von $(\varepsilon,\varepsilon)$ zu $(\varepsilon,\varepsilon)$ haben. Solche Schleifen entsprechen inneren Blöcken. Das Auftauchen der zu definierenden Mengen auf beiden Seiten der Gleichungen beschreibt, dass Startblöcke nicht aus Startblöcken und echten inneren Blöcken zusammengesetzt sein dürfen und analog, Endblöcke nicht aus echten inneren Blöcken und Endblöcken. Mit anderen Worten sind wir jeweils an den minimalen Mengen interessiert und sorgen dafür, dass ein echter innerer Block nicht gleichzeitig Startblock oder Endblock sein kann. Die geneigte Leserschaft möge überprüfen, dass $\mathbb{S}(I^{\mathrm{R}}) = \mathbb{F}(I)^{\mathrm{R}}$ gilt. Für herkömmliche Instanzen gibt es nur den Startblock $(\varepsilon,\varepsilon)$ und nur den Endblock $(\varepsilon,\varepsilon)$, allgemeiner gilt das

Lemma. $s = (\varepsilon,\varepsilon) \implies \mathbb{S}(I) = \{(\varepsilon,\varepsilon)\}$. $f = (\varepsilon,\varepsilon) \implies \mathbb{F}(I) = \{(\varepsilon,\varepsilon)\}$. *e*

Beweis: Wenn $s = (\varepsilon,\varepsilon)$ und $(u,v) \in \mathbb{S}(I)$ ein Startblock ist, dann gilt $h(u) = g(v)$, also ist $(u,v) \in \mathbb{I}(I)$ ebenfalls ein innerer Block. Da $(\varepsilon,\varepsilon)$ ein Startblock ist und ein Startblock nicht aus einem Startblock und einem echten inneren Block zusammengesetzt sein darf, folgt $\mathbb{S}(I) = \{(\varepsilon,\varepsilon)\}$. Der Fall $f = (\varepsilon,\varepsilon)$ folgt analog. □ *f*

Zum Aufzählen der Menge $\mathbb{S}(I)$ gehen wir wie folgt vor: Analog zur Konstruktion des Transducers für $\mathbb{I}(I)$ konstruieren wir ausgehend von den Hüllen $\hookrightarrow^\star_I(s)$ und $\circlearrowleft\!\!\hookrightarrow^\star_{I^{\mathrm{R}}}((\varepsilon,\varepsilon))$ einen Transducer, löschen alle Transitionen, die von $(\varepsilon,\varepsilon)$ ausgehen und adaptieren den Beweis 55a um zu zeigen, dass dieser Transducer eine *Obermenge* von $\mathbb{S}(I)$ akzeptiert. Es handelt sich jetzt nur

noch um eine Obermenge, da bei der Konstruktion in 55a nicht berücksichtigt wird, dass Startblöcke nicht aus Startblöcken und echten inneren Blöcken zusammengesetzt sein dürfen. Für die Aufzählung zählen wir die Sprache dieses Transducers auf und löschen alle die Elemente, die ein Suffix haben, das ein echter innerer Block ist. Wir können mit Hilfe des Transducers aus 64c entscheiden, ob das der Fall ist.

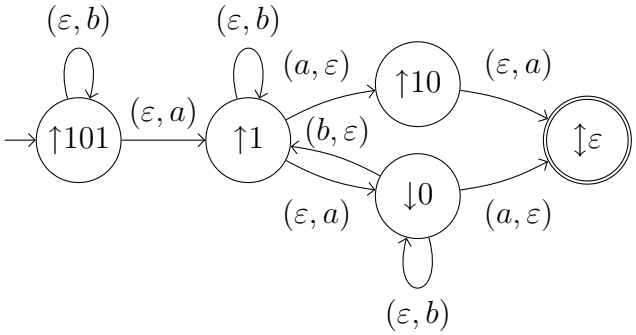
a **Folgerung.** *Die Mengen $\mathbb{S}(I)$ und $\mathbb{F}(I)$ können effektiv aufgezählt werden.*

Auch hier stellt sich die Frage, ob es entscheidbar ist, ob $\mathbb{S}(I)$ und $\mathbb{F}(I)$ endlich sind. Wir haben zwar einen Transducer für die Menge $\mathbb{X}(I) = \{(u, v) \mid s_\uparrow h(u) = s_\downarrow g(v)\}$, die eine Obermenge von $\mathbb{S}(I)$ ist und wir haben einen Transducer für die Menge $\mathbb{I}(I)$, doch es ist zunächst unklar, wie aus dem Transducer für $\mathbb{X}(I)$ alle die Pfade gelöscht werden können, die einem (echten) Suffix aus $\mathbb{I}^\circ(I)$ entsprechen. Wir wollen diese Frage hier nicht weiter vertiefen. Im weiteren Verlauf benötigen wir entweder die Start- und Endblöcke herkömmlicher oder markierter allgemeiner Instanzen. Für herkömmliche Instanzen wissen wir aus 69e, dass $\mathbb{S}(I) = \mathbb{F}(I) = \{(\varepsilon, \varepsilon)\}$ ist und für markierte allgemeine Instanzen werden wir gesondert darüber nachdenken, wie diese Mengen effektiv konstruiert werden können.

b **Beispiel:** Für die Instanz $I_{53a} = |\,{}^{0}_{\varepsilon}\,|\,{}^{1\ 0}_{0\ 1}\,|\,{}^{\varepsilon}_{0}\,|$ gelten $\mathbb{S}(I_{53a}) = \{(\varepsilon, a)\}$ und $\mathbb{F}(I_{53a}) = \{(b, \varepsilon)\}$. ⌟

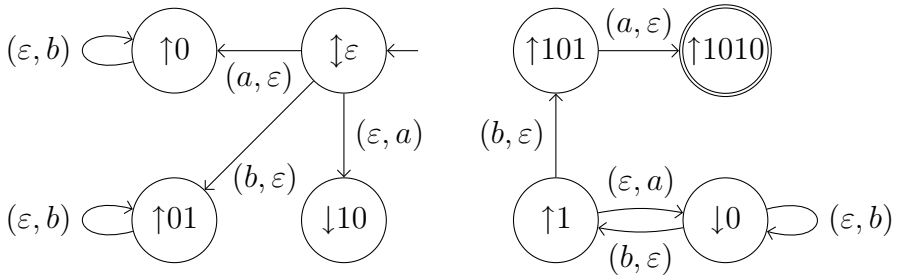
c **Beispiel:** Für die Instanz $I_{53b} = |\,{}^{011}_{\varepsilon}\,|\,{}^{\varepsilon\ \ 0}_{01\ 1011}\,|\,{}^{1101}_{\varepsilon}\,|$ gelten $\mathbb{S}(I_{53b}) = \emptyset$ und $\mathbb{F}(I_{53b}) = \emptyset$. ⌟

d **Beispiel:** Für die Instanz $I_{15c} = |\,{}^{101}_{\varepsilon}\,|\,{}^{0\ \ 01}_{10\ \ \varepsilon}\,|\,{}^{\varepsilon}_{1010}\,|$ ergibt sich für die Menge $\mathbb{S}(I_{15c})$ der Startblöcke der Transducer T



der die Sprache $L(T) = \{(b^k a, v) \mid |v|_a = k + 2, k \in \mathbb{N}\}$ akzeptiert. Laut 69a gilt $\mathbb{B}(I_{15c}) = \{(\varepsilon, b)\}$ und damit $\mathbb{I}^\circ(I_{15c}) = \{(\varepsilon, bb^i) \mid i \in \mathbb{N}\}$. Daraus folgt,

dass $L(T)$ eine echte Obermenge von $\mathbb{S}(I_{15c}) = \{(b^k a, va) \mid |v|_a = k+1, k \in \mathbb{N}\}$ ist: In $L(T)$ kann v die Form ub haben, in $\mathbb{S}(I_{15c})$ nicht. Konstruieren wir den Transducer für die Menge $\mathbb{F}(I_{15c})$ der Endblöcke, dann erhalten wir



Daraus folgt sofort $\mathbb{F}(I_{15c}) = \emptyset$. ⌟

Monolithen *a*

Mit Hilfe der inneren Blöcke, der Startblöcke und der Endblöcke lassen sich immer noch nicht alle Koinzidenzen einer allgemeinen Instanz zerlegen, denn es kann sein, dass der Pfad durch den Transducer aus Satz 56a nicht $(\varepsilon, \varepsilon)$ berührt. Wir benötigen zusätzlich die Menge der *Monolithen* $\mathbb{M}(I) = \mathbb{C}(I) \setminus (\mathbb{S}(I)\,\mathbb{I}(I)\,\mathbb{F}(I))$, die für herkömmliche Instanzen leer sein soll. Tatsächlich gilt

Lemma. $(\varepsilon, \varepsilon) \in \{s, f\} \implies \mathbb{M}(I) = \emptyset$. *b*

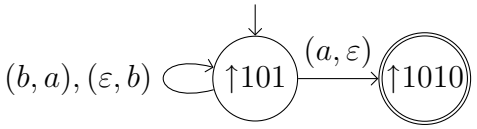
Beweis: Wenn $s = (\varepsilon, \varepsilon)$ und $x \in \mathbb{M}(I)$ ein Monolith ist, dann gilt $h(x_\uparrow) f_\uparrow = g(x_\downarrow) f_\downarrow$, also ist x in $\mathbb{I}(I)\,\mathbb{F}(I)$ enthalten. Widerspruch. Der Fall $f = (\varepsilon, \varepsilon)$ folgt analog. □ *c*

Der Transducer T aus 56a akzeptiert die Koinzidenzmenge und damit eine Obermenge der Monolithen. Die Idee liegt nahe, einfach den Zustand $(\varepsilon, \varepsilon)$ aus T zu löschen, um einen Transducer T' zu erhalten, der $\mathbb{M}(I)$ erkennt. Zwar akzeptiert T' immer noch eine Obermenge von $\mathbb{M}(I)$, aber es kann Pfade in T' geben, die mit einem Paar w beschriftet sind, für die es eine andere Zerlegung gibt, die einem Pfad in T entspricht, der durch $(\varepsilon, \varepsilon)$ führt. Eine Aufzählung von $\mathbb{M}(I)$ erhalten wir, wenn wir die von T' akzeptierte Sprache aufzählen und für jedes Element w prüfen, ob es einen Pfad in T gibt, der durch $(\varepsilon, \varepsilon)$ führt. Wir nehmen w genau dann in die Aufzählung von $\mathbb{M}(I)$ auf, wenn es keinen solchen Pfad gibt.

Folgerung. *Die Menge $\mathbb{M}(I)$ kann effektiv aufgezählt werden.* *d*

Wie bereits für die Start- und Endblöcke wollen wir uns nicht weiter mit der Frage beschäftigen, ob entscheidbar ist, ob $\mathbb{M}(I)$ endlich ist oder ob wir sogar endliche Monolithen effektiv konstruieren können. Auch hier wissen wir für herkömmliche Instanten bereits, dass $\mathbb{M}(I) = \emptyset$ ist. Unser anderer Anwendungsfall, der der markierten allgemeinen Instanzen, wird diese Fragen separat diskutieren.

a **Beispiel:** Für die Instanz $I_{15c} = |\,{}^{101}_{\varepsilon}\,|\,{}^{0}_{10}\;{}^{01}_{\varepsilon}\,|\,{}^{\varepsilon}_{1010}\,|$ ergibt sich durch Löschen des Zustandes $(\varepsilon, \varepsilon)$ aus dem Transducer aus 57a für die Koinzidenzmenge $\mathbb{C}(I_{15c})$ der liberalisierte Transducer



der $\mathbb{M}(I_{15c}) = \{(b^k a, v) \mid |v|_a = k, k \in \mathbb{N}\}$ akzeptiert. Im Transducer aus 57a gibt es keinen akzeptierenden Pfad, der durch $(\varepsilon, \varepsilon)$ führt. ⌟

b **Beispiel:** Für die Instanz $I_{53a} = |\,{}^{0}_{\varepsilon}\,|\,{}^{1}_{0}\;{}^{0}_{1}\,|\,{}^{\varepsilon}_{0}\,|$ gilt $\mathbb{M}(I_{53b}) = \{(\varepsilon, \varepsilon)\}$. ⌟

c **Beispiel:** Für die Instanz $I_{53b} = |\,{}^{011}_{\varepsilon}\,|\,{}^{\varepsilon}_{01}\;{}^{0}_{1011}\,|\,{}^{1101}_{\varepsilon}\,|$ erhalten wir $\mathbb{M}(I_{53b}) = (a, \varepsilon)^*(b, aba)(a, \varepsilon)^*$. ⌟

d **Zerlegen von Koinzidenzen**

Unser Ziel, die Koinzidenzen allgemeiner Instanzen in Blöcke zu zerlegen, haben wir erreicht, es gilt das

e **Lemma.** (Zerlegen von Koinzidenzen) $\mathbb{C}(I) = \mathbb{S}(I)\,\mathbb{B}(I)^*\,\mathbb{F}(I) \cup \mathbb{M}(I)$.

f *Beweis:* ($\supseteq$): Zunächst gilt $\mathbb{M}(I) \subseteq \mathbb{C}(I)$ nach Definition. Wenn nun $x \in \mathbb{S}(I)\,\mathbb{B}(I)^*\,\mathbb{F}(I) = \mathbb{S}(I)\,\mathbb{I}(I)\,\mathbb{F}(I)$ ist, dann kann x so in $x = lmr$ zerlegt werden, dass gilt $s_\uparrow h(x_\uparrow) f_\uparrow = s_\uparrow h(l_\uparrow) h(m_\uparrow) h(r_\uparrow) f_\uparrow = s_\downarrow g(l_\downarrow) g(m_\downarrow) g(r_\downarrow) f_\downarrow = s_\downarrow g(x_\downarrow) f_\downarrow$. Also gilt $x \in \mathbb{C}(I)$.

($\subseteq$): Sei $x \in \mathbb{C}(I)$ ein Element der Koinzidenzmenge. Nehmen wir zunächst an, dass es ein minimales Präfix $l \preceq_\mathrm{p} x$ von x gibt, so dass $s_\uparrow h(l_\uparrow) = s_\downarrow g(l_\downarrow)$ gilt. Dann gibt es auch ein minimales Suffix $r \preceq_\mathrm{s} x$ von x, so dass $h(r_\uparrow) f_\uparrow = g(r_\downarrow) f_\downarrow$ gilt. Darüber hinaus gilt $x = lmr$ für ein gewisses m mit $h(m_\uparrow) = g(m_\downarrow)$, anderenfalls wären l und r nicht minimal. Daraus folgt $x \in \mathbb{S}(I)\,\mathbb{I}(I)\,\mathbb{F}(I) = \mathbb{S}(I)\,\mathbb{B}(I)^*\,\mathbb{F}(I)$. Wenn es kein solches Präfix l gibt, dann ist $x \in \mathbb{M}(I)$ nach Definition ein Monolith. □

Lemma 72e wird für herkömmliche Instanzen zusammen mit den Lemmata 69e und 71b zur Aussage $\mathbb{C}(I) = \mathbb{B}(I)^*$, das heißt, jede Koinzidenz einer herkömmlichen Instanz lässt sich in Blöcke zerlegen. Schließen wir ε-triviale Instanzen aus, dann kann jede Koinzidenz sogar eindeutig in Blöcke zerlegt werden. Bei allgemeinen Instanzen ist entweder gar keine Zerlegung möglich oder es wird eine Folge von Blöcken durch genau einen Startblock und genau einen Endblock ergänzt.

Wir können bereits an diesem Punkt etwas über die Anzahl der Blöcke in einer Zerlegung einer Lösung aussagen, diese wird nämlich nicht größer, wenn wenigstens einer der Morphismen nicht löschend ist. Genauer gilt das

Lemma. *Seien $d = |s_\uparrow| - |s_\downarrow|$ und $c = |B|^{|d|}$. Wenn $(u,u) \in \mathrm{id}(\mathrm{Eq}(I)) \cap \mathbb{S}(I)\,\mathbb{B}(I)^k\,\mathbb{F}(I)$ eine Lösungskoinzidenz von I ist, die sich in einen Startblock, einen Endblock und k Blöcke zerlegen lässt, dann gelten $k \leq 2|u|$, $I \in \mathcal{E}^{\leq 1} \implies k \leq |u|$ und, wenn u minimal ist, $I \in \mathcal{E}^0 \implies |u| > c \implies k < |u|$.* *a*

Beweis: Nach Voraussetzung können wir (u,u) in $(u,u) = lb_0 \cdots b_{k-1}r$ zerlegen, wobei $l \in \mathbb{S}(I)$ ein Start-, $r \in \mathbb{F}(I)$ ein Endblock und $\{b_0, \ldots, b_{k-1}\} \subseteq \mathbb{B}(I)$ eine Menge von Blöcken sind. *b*

Nehmen wir zunächst $k > 2|u|$ an. Dann ist $|(u,u)| \geq |b_0 \cdots b_{k-1}| \geq k > 2|u| = |(u,u)|$, da jeder Block aus $\mathbb{B}(I)$ mindestens die Länge 1 hat. Widerspruch, also gilt $k \leq 2|u|$.

Sei nun $I \in \mathcal{E}^{\leq 1}$ und ohne Beschränkung der Allgemeinheit sei h nicht löschend. Dann ist die zweite Komponente von b_i für $i \in \{0, \ldots, k-1\}$ nicht leer, das heißt, es gilt $b_{i\downarrow} \neq \varepsilon$ und daraus folgt $|b_{i\downarrow}| \geq 1$. Da $(u,u)_\downarrow = u = (lb_0 \cdots b_{k-1}r)_\downarrow$ ist, folgt $|u| \geq |(b_0 \cdots b_{k-1})_\downarrow| \geq k$.

Nehmen wir schließlich an, dass $I \in \mathcal{E}^0$ nicht löschend ist und sei u mit $|u| > c$ eine minimale Lösung von I. Da h und g beide nicht löschend sind, sind beide Komponenten von b_i für $i \in \{0, \ldots, k-1\}$ nicht leer. Es gilt $b_{i\uparrow} \neq \varepsilon$ und $b_{i\downarrow} \neq \varepsilon$ und daraus folgt $|b_i| \geq 2$. Wir wissen bereits, dass $k \leq |u|$ gilt. Angenommen $k = |u|$, dann gilt $|(u,u)| \geq |b_0 \cdots b_{k-1}| \geq 2k = 2|u| = |(u,u)|$. Daraus folgt unmittelbar, dass sämtliche b_i die Länge 2 und dass l und r beide die Länge 0 haben. Es gilt also $|b_i| = 2$ für $i \in \{0, \ldots, k-1\}$ und $l = r = (\varepsilon, \varepsilon)$. Daraus folgt aber, dass die ersten und zweiten Komponenten sämtlicher b_i jeweils gleich sind, es gilt $\{b_0, \ldots, b_{k-1}\} \subseteq \mathrm{id}(A)$. Außerdem gilt $h(b_{i\uparrow}) = g(b_{i\downarrow})$ da b_i ein Block ist und daraus folgt insbesondere $|h(b_{i\uparrow})| = |g(b_{i\downarrow})|$ für $i \in \{0, \ldots, k-1\}$. Mit anderen Worten haben sämtliche Blöcke b_i die Form (a,a) für ein $a \in A$, wobei die h- und g-Bilder von a die selbe Länge haben.

Wir betrachten nun die Menge $\{x_0, \ldots, x_{k-1}\} \subseteq \mathrm{id}(A^*)$, die durch $x_i = b_0 \cdots b_i$ gegeben ist. Dann ist $x_{k-1} = (u,u)$ und es gilt $|s_\uparrow h(x_{i\uparrow})| - |s_\downarrow g(x_{i\downarrow})| = d$ für alle $i \in \{0, \ldots, k-1\}$: Sämtliche Überhänge in $(s_\uparrow h(x_{i\uparrow}), s_\downarrow g(x_{i\downarrow}))$ haben

die selbe Länge $|d|$. Da nach Voraussetzung $k = |u| > c = |B|^{|d|}$ ist, gibt es nach dem Schubfachprinzip zwei Indizes $\{n, m\} \subseteq \{0, \ldots, k-1\}$, so dass $n < m$ ist und der Überhang in $(s_\uparrow h(x_{n\uparrow}), s_\downarrow g(x_{n\downarrow}))$ der selbe ist wie der in $(s_\uparrow h(x_{m\uparrow}), s_\downarrow g(x_{m\downarrow}))$. Daraus folgt aber $b_0 \cdots b_n b_{m+1} \cdots b_{k-1} \in \mathrm{id}(\mathrm{Eq}(I))$ im Widerspruch zur Minimalität von u. □

Für herkömmliche Instanzen ist $d = 0$ und $c = 1$. Das Lemma 73a sagt dann für nichtlöschende Instanzen aus, dass eine minimale Lösung u entweder die Länge 1 hat, oder sich in eine Anzahl Blöcke zerlegen lässt, die echt kleiner als die Länge von u ist. Der Beweis könnte in diesem Fall bereits nach der Erkenntnis abgeschlossen werden, dass die b_i die Form (a, a) für ein $a \in A$ haben. Es wäre dann a direkt eine Lösung.

a **Beispiel:** Bei der Instanz $I_{15c} = |\,{}^{101}_{\varepsilon}\,|\,{}^{0}_{10}\;{}^{01}_{\varepsilon}\,|\,{}^{\varepsilon}_{1010}\,|$ ist $\mathbb{M}(I_{15c}) = \mathbb{C}(I_{15c})$. Das heißt, dass keine Koinzidenz in Blöcke zerlegbar ist. ⌟

b **Beispiel:** Für die Instanz $I_{53a} = |\,{}^{0}_{\varepsilon}\,|\,{}^{1}_{0}\;{}^{0}_{1}\,|\,{}^{\varepsilon}_{0}\,|$ kann für $i \in \mathbb{N}$ die Koinzidenz $(ab, ab)^{i+1}$ in den Startblock (ε, a), den Endblock (b, ε) und $2i + 1$ Blöcke $(a, b)((b, a)(a, b))^i$ zerlegt werden. ⌟

c **Beispiel:** Bei der Instanz $I_{53b} = |\,{}^{011}_{\varepsilon}\,|\,{}^{\varepsilon}_{01}\;{}^{0}_{1011}\,|\,{}^{1101}_{\varepsilon}\,|$ ist ebenfalls $\mathbb{M}(I_{53b}) = \mathbb{C}(I_{53b})$. Das heißt, dass keine Koinzidenz in Blöcke zerlegbar ist. ⌟

d **Beispiel:** Auch markierte herkömmliche Instanzen können viele Blöcke in der Zerlegung von Lösungskoinzidenzen benötigen: Betrachten wir die Familie $I_{74d}(k) = \|\,{}^{0}_{00}\;{}^{10^{2k}}_{1}\,\|$, dann gilt $\mathrm{Eq}(I_{74d}(k)) = (ba^{2k})^*$ und die Lösungskoinzidenz (ba^{2k}, ba^{2k}) lässt sich in $(b, ba^k)(aa, a)^k$ zerlegen. ⌟

e **Beispiel:** Bei der Instanz $I_{15d} = \|\,{}^{abb}_{a}\;{}^{bb}_{bbb}\;{}^{cbbb}_{c}\,\|$ lässt sich die Koinzidenz $(ccbbb, cbcbbb)$ in $(c, cb)(c, cb)(bbb, bb)$ zerlegen. Die zur minimalen Lösung abb gehörende Koinzidenz besteht aus dem einzigen Block (abb, abb). ⌟

f **Beispiel:** Die Instanz $I_{74f} = \|\,{}^{0}_{0000}\;{}^{000}_{0010}\;{}^{100}_{01}\;{}^{1010}_{010}\,\|$ hat die minimale Lösung *aa*
aaaaabbbbaabaabacacacacbaaabcbbbcbacbcbcbccaabaabaaacccacacccaaaaccccc
cdbbbcbbbcbbbbdcbcbdcbbbbbdddcacacacccacacccacacacccccccccacacacaccccb
cbcbdcbcbdcbcbcbddddcbcbcbcbddcccccccccccccccccccccccccccccccddddddddd
ddddddddcccccccccccccccccddddddddcccccccddddcccddccdcd der Länge 268, die sich in genau die 15 Blöcke *(aaaa,a)(aaabbb,aaa)(ba,a)(ab,a)(aabacacacac*
b,abbbba)(aaabcb,aba)(bbcb,aba)(acbcbcbccaabaa,cacacacbaa)(ba,a)(aacccacac
ccaaa,bcbbbcba)(acccccccdb,cbcbcbcca)(bbcb,aba)(bbcb,aba)(bbbdcbcbdcbb,aaccc
acacccaa)(bbbdddcacacacccacacccacacacccccccccacacacacccccbcbcbdcbcbdcbc
bcbddddcbcbcbcbddcccccccccccccccccccccccccccccccccddddddddddddddddcccccc

ccccccccccddddddddccccccccddddccccddccdcd,aaccccccdbbbcbbbcbbbbdcbcbdcb bbbbdddcacacacccacacccacacacccccccccacacacacccccbcbcbdcbcbdcbcbcbddddcb cbcbcbddcccccccccccccccccccccccccccccccccddddddddddddddddddccccccccccccc ccddddddddccccccccddddccccddccdcd) zerlegen lässt. Die Komponenten des letzten Blocks haben die Längen 173 und 203. ⌟

Beispiel: Eine extreme Instanz ist $I_{75a} = \| \begin{smallmatrix} 0 \\ 00001 \end{smallmatrix} \begin{smallmatrix} 0100 \\ 0 \end{smallmatrix} \begin{smallmatrix} 10000 \\ 001 \end{smallmatrix} \|$. Wenn u deren minimale Lösung *aaababaacbaababcbaacbbaababcabaacbbaacbabcababcbbaa cbacababcbccbbbcbaabaacbabcbb* der Länge 80 ist, dann ist die Koinzidenz (u, u) gleichzeitig einziger Block in ihrer Zerlegung. Wohlgemerkt *ist* das eine Zerlegung und kein Monolith, denn herkömmliche Instanzen haben keine Monolithen. ⌟ *a*

Nachfolger *b*

Kommen wir zum Begriff des Nachfolgers. Wir wissen, dass sich jede Koinzidenz in Bausteine aus den verschiedenen Blockmengen zerlegen lässt. Die Idee des Nachfolgers ist es, nicht mehr nach Lösungen einer Instanz I zu suchen, sondern nach Lösungen einer Instanz, deren Regeln genau diese Bausteine sind. Wir nennen die Instanz $I' = (A', A, s', (h', g'), f')$ genau dann einen *Nachfolger* der Instanz $I = (A, B, s, (h, g), f)$, wenn $\{(h'(a'), g'(a')) \mid a' \in A'\} \subseteq \mathbb{B}(I)$ gilt und es $\{x, y\} \subset \mathrm{id}(A^*)$ mit $xs' \in \mathbb{S}(I)$ und $f'y \in \mathbb{F}(I)$ gibt. Die Regeln von I' sind Blöcke von I und das Start- und das Endpaar von I' sind ein Suffix und ein Präfix eines Start- und eines Endpaars von I. Es ist zu beachten, dass Start- und Endpaare nicht aus $B^* \times B^*$, sondern aus $B^{*\updownarrow}$ sein müssen. Wir können also nicht einfach $s' \in \mathbb{S}(I)$ fordern, da $\mathbb{S}(I)$ im Allgemeinen Elemente enthält, deren beide Komponenten nicht leer sind. Die Menge aller Nachfolger von I bezeichnen wir mit $\mathrm{suc}(I)$. Da $\mathbb{B}(I)$, $\mathbb{S}(I)$ und $\mathbb{F}(I)$ alle effektiv aufgezählt werden können, gilt

Folgerung. *Die Menge* $\mathrm{suc}(I)$ *kann effektiv aufgezählt werden.* *c*

Für herkömmliche Instanzen ist $\mathbb{S}(I) = \mathbb{F}(I) = \{(\varepsilon, \varepsilon)\}$. Wenn dann außerdem $\mathbb{B}(I)$ endlich ist, interessieren wir uns vor allem für den *Hauptnachfolger von* I, das ist der Nachfolger, dessen Regelmenge genau die Blockmenge $\mathbb{B}(I)$ ist.

Wir zeigen, dass die Lösbarkeit einer Instanz I eng mit der Lösbarkeit von bestimmten Nachfolgern von I zusammenhängt. Zunächst impliziert die Lösbarkeit von I, dass entweder die Menge der Monolithen nicht leer ist, oder es einen lösbaren Nachfolger von I gibt.

Lemma. $\mathrm{Eq}(I) \neq \emptyset \implies \mathbb{M}(I) = \emptyset \implies \exists I' \in \mathrm{suc}(I) : \mathrm{Eq}(I') \neq \emptyset.$ *d*

a *Beweis:* Angenommen $u \in \mathrm{Eq}(I)$ ist eine Lösung von I, die keiner monolithischen Koinzidenz entspricht, also mit $(u,u) \notin \mathbb{M}(I)$. Nach Lemma 72e über die Zerlegung von Koinzidenzen und Folgerung 47d über den Zusammenhang zwischen Gleichheits- und Koinzidenzmenge gilt $(u,u) \in \mathrm{id}(A^*) \cap \mathbb{S}(I)\,\mathbb{B}(I)^*\,\mathbb{F}(I)$. Daraus folgt $(u,u) = xs'b_0 \cdots b_{k-1}f'y$ für gewisse Startblöcke $xs' \in \mathbb{S}(I)$, Endblöcke $f'y \in \mathbb{F}(I)$ mit $\{x,y\} \subset \mathrm{id}(A^*)$ und Blöcke $\{b_0,\ldots,b_{k-1}\} \subseteq \mathbb{B}(I)$. Sei $B = \{z_0,\ldots,z_{n-1}\} \subseteq \mathbb{B}(I)$ eine endliche Menge von Blöcken mit $\{b_0,\ldots,b_{k-1}\} \subseteq B$ und $\psi : \{0,\ldots,k-1\} \to \{0,\ldots,n-1\}$ die Abbildung, die durch $b_i = z_{\psi(i)}$ für $i \in \{0,\ldots,k-1\}$ gegeben ist. Wir haben in B alle Duplikate aus $\{b_0,\ldots,b_{k-1}\}$ gelöscht.

Sei nun $I' = (\{0,\ldots,n-1\}, A, s', (h',g'), f')$ die Instanz, deren Regeln die Elemente in B sind, also mit $(h'(i),g'(i)) = z_i$ für $i \in \{0,\ldots,n-1\}$. Dann ist $I' \in \mathrm{suc}(I)$ ein Nachfolger von I. Sei außerdem $u' \in \{0,\ldots,n-1\}^*$ das Wort $u' = \psi(0)\cdots\psi(k-1)$. Dann gilt

$$\begin{aligned}\begin{bmatrix} x_\uparrow s'_\uparrow h'(u') f'_\uparrow y_\uparrow \\ x_\downarrow s'_\downarrow g'(u') f'_\downarrow y_\downarrow \end{bmatrix} &= \begin{bmatrix} x_\uparrow s'_\uparrow h'(\psi(0))\cdots h'(\psi(k-1)) f'_\uparrow y_\uparrow \\ x_\downarrow s'_\downarrow g'(\psi(0))\cdots g'(\psi(k-1)) f'_\downarrow y_\downarrow \end{bmatrix} \\ &= xs' z_{\psi(0)} \cdots z_{\psi(k-1)} f'y \\ &= xs'b_0 \cdots b_{k-1}f'y \in \mathrm{id}(A^*).\end{aligned}$$

Also ist $u' \in \mathrm{Eq}(I')$ eine Lösung von I'. □

Andersherum folgt aus der Existenz eines lösbaren Nachfolgers von I die Lösbarkeit von I.

b **Lemma.** $\exists I' \in \mathrm{suc}(I) : \mathrm{Eq}(I') \neq \emptyset \implies \mathrm{Eq}(I) \neq \emptyset$.

c *Beweis:* Seien $I' = (A', A, s', (h',g'), f')$ ein Nachfolger von I und $u' = u'_0 \cdots u'_{k-1} \in \mathrm{Eq}(I')$ eine Lösung von I'. Dann gelten $xs' \in \mathbb{S}(I)$, $f'y \in \mathbb{F}(I)$ für gewisse $\{x,y\} \subset \mathrm{id}(A^*)$ und $\{(h'(a'),g'(a')) \mid a' \in A'\} \subseteq \mathbb{B}(I)$ nach Definition von $\mathrm{suc}(I)$. Daraus folgt

$$\begin{aligned}\begin{bmatrix} u \\ u \end{bmatrix} &= \begin{bmatrix} x_\uparrow s'_\uparrow h'(u') f'_\uparrow y_\uparrow \\ x_\downarrow s'_\downarrow g'(u') f'_\downarrow y_\downarrow \end{bmatrix} \\ &= \begin{bmatrix} x_\uparrow s'_\uparrow h'(u'_0)\cdots h'(u'_{k-1}) f'_\uparrow y_\uparrow \\ x_\downarrow s'_\downarrow g'(u'_0)\cdots g'(u'_{k-1}) f'_\downarrow y_\downarrow \end{bmatrix} \\ &= xs'b_0 \cdots b_{k-1}f'y \in \mathbb{S}(I)\,\mathbb{B}(I)^*\,\mathbb{F}(I)\end{aligned}$$

Also ist $u \in \mathrm{Eq}(I)$ eine Lösung von I. □

Betrachten wir nur herkömmliche Instanzen, für die nach 71b keine monolithischen Koinzidenzen existieren, dann sagen die Lemmata 75d und 76b zusammen aus, dass eine Instanz genau dann lösbar ist, wenn sie einen lösbaren Nachfolger hat. Sie geben außerdem eine explizite Konstruktion, wie sich Lösung von Instanz und Nachfolger aus der jeweils anderen ermitteln lassen. Aus Lemma 73a über die Anzahl der in einer Zerlegung einer Lösung verwendeten Blöcke folgt zudem, dass in einer Kette $(I, I', I'', \dots)$ von Nachfolgern einer lösbaren herkömmlichen Instanz I schließlich entweder alle Folgeglieder unlösbar sind oder es eine Instanz gibt, deren minimale Lösung die Länge 1 hat. Diese Beobachtung wird in 79b zu einem Entscheidungsalgorithmus für markierte Instanzen ausgebaut.

Doch zunächst einige Beispiele für Nachfolger und insbesondere ihre Anwendung in Unlösbarkeitsbeweisen. Geneigte Leserinnen und Leser mögen jeweils die Unlösbarkeit zeigen, ohne auf Nachfolger zu verweisen.

Beispiel: Bei der Instanz $I_{53a} = |\,{}^{0}_{\varepsilon}\,|\,{}^{1\ 0}_{0\ 1}\,|\,{}^{\varepsilon}_{0}\,|$ ist die Menge $\mathbb{B}(I_{53a})$ endlich und die Mengen $\mathbb{S}(I_{53a})$ und $\mathbb{F}(I_{53a})$ enthalten beide genau ein Element. Das heißt, wir können auch hier sinnvoll vom Hauptnachfolger $I'_{53a} = |\,{}^{\varepsilon}_{a}\,|\,{}^{b\ a}_{a\ b}\,|\,{}^{b}_{\varepsilon}\,|$ sprechen. Der Hauptnachfolger dieser Instanz ist die Instanz $I''_{53a} = |\,{}^{a}_{\varepsilon}\,|\,{}^{b\ a}_{a\ b}\,|\,{}^{\varepsilon}_{a}\,|$, die isomorph zu I_{53a} ist. Mit dem Lemma 73a über die Anzahl der in einer Zerlegung verwendeten Blöcke folgt daraus, dass I_{53a} genau dann eine nichtleere Lösung hat, wenn sie eine nichtleere Lösung der Länge höchstens $2 = |B|^{|d|}$ hat. Tatsächlich gelten $\mathrm{Eq}(I_{53a}) = (ab)^*$ und $\mathrm{Eq}(I'_{53a}) = a(ba)^*$. ⌟ *a*

Beispiel: Bei der Instanz $I_{15d} = \|\,{}^{abb\ \ bb\ \ cbbb}_{a\ \ bbb\ \ c}\,\|$ ergibt sich die Kette der Hauptnachfolger $I'_{15d} = \|\,{}^{abb\ bbb\ c}_{abb\ \ bb\ \ cb}\,\|$, $I''_{15d} = \|\,{}^{a\ \ bb\ \ cb}_{a\ bbb\ cb}\,\|$, $I'''_{15d} = \|\,{}^{a\ \ bbb\ c}_{a\ \ bb\ \ c}\,\|$, $I''''_{15d} = \|\,{}^{a\ \ bb\ \ c}_{a\ bbb\ c}\,\|$. Der Hauptnachfolger von I''''_{15d} ist wieder die Instanz I'''_{15d}. Aus der Lösung $c \in \mathrm{Eq}(I'''_{15d})$ rekonstruieren wir in I''_{15d} die Koinzidenz (cb, cb), die in I'_{15d} der Lösungskoinzidenz $(cbbb, cbbb)$ entspricht. Analog rekonstruieren wir aus $a \in \mathrm{Eq}(I'_{15d})$ die Lösung $abb \in \mathrm{Eq}(I_{15d})$.

Aus Lemma 73a über die Anzahl der in einer Zerlegung verwendeten Blöcke ergibt sich, dass herkömmliche Instanzen mit einem sich wiederholenden transitiven Nachfolger genau dann eine Lösung haben, wenn der sich wiederholende Nachfolger eine Lösung der Länge 1 hat. ⌟ *b*

Beispiel: Sei I_{77c} die Instanz $I_{77c} = \|\,{}^{0}_{010}\,{}^{1}_{00}\,{}^{1001}_{1}\,\|$. Diese Instanz hat keine echte Lösung, denn: Zunächst ist $I_{77c} = I^{\mathrm{R}}_{77c}$ und wir überlegen uns, dass eine Lösung ohne Beschränkung der Allgemeinheit mit c beginnt und mit a endet. Nach Anwendung von c kann aber a niemals angewendet werden, da die Anwendung von c und b nicht das Teilwort 010 erzeugen. Diese Beweisskizze müsste noch genauer ausgeführt werden und eine Implementation dieses *c*

Schemas ist nicht einfach. Insbesondere kann dieses Schema nicht *absolut* implementiert werden, denn die Erkenntnis, dass a nicht angewendet werden kann, erfordert die Berechnung der Hülle der Anwendungen von b und c und solche Hüllen können beliebig groß werden. Also benötigt ein Algorithmus eine künstliche Abbruchbedingung, die natürlich stets so gewählt wird, dass die Hülle noch nicht komplett berechnet wurde. Kurzum: I_{77c} hat keine echte Lösung und es ist schwierig dafür einen Beweis automatisch zu erzeugen.

Betrachten wir jedoch den Hauptnachfolger $I'_{77c} = \| \begin{smallmatrix} aca & c & b & aba & aa \\ aa & cbc & c & a & b \end{smallmatrix} \|$, dann ermitteln wir sofort, dass die Regeln (aba, a) und (aa, b) wegen falscher a-Balance gelöscht werden können: Sie erzeugen echt mehr Symbole a in den h-Bildern als in den g-Bildern und diese überzähligen Symbole können nicht wieder entfernt werden. Es bleibt die Instanz $\| \begin{smallmatrix} aca & c & b \\ aa & cbc & c \end{smallmatrix} \|$, die mit Hilfe der Methode des Maskierens aus 42b leicht entschieden werden kann: Es wäre notwendig einen Pfad von $\downarrow bc$ nach $\uparrow cb$ zu konstruieren, aber es gibt keine Möglichkeit, von $\downarrow$- zu $\uparrow$-Überhängen zu gelangen.

Daraus folgt die Unlösbarkeit von I_{77c} und alle Schritte der zweiten Beweisführung sind leicht implementierbar. ⌟

a **Beispiel:** Die Instanz $I_{78a} = \| \begin{smallmatrix} 0 & 001 & 0101 \\ 00101 & 1 & 0 \end{smallmatrix} \|$ hat den Hauptnachfolger $I'_{78a} = \| \begin{smallmatrix} a & b & c & bc & ac \\ c & ccb & cbcb & acb & a \end{smallmatrix} \|$ der ebenfalls automatisch als unlösbar erkannt werden kann: Lösche zunächst die Regel $(c, cbcb)$ wegen falscher c-Balance und anschließend die Regel (b, ccb), da cc nicht mehr überdeckt werden kann. Die verbleibende Instanz $\| \begin{smallmatrix} a & bc & ac \\ c & acb & a \end{smallmatrix} \|$ hat eine leere Endmenge, ist also unlösbar.

Auch ohne den Blick auf den Nachfolger kann die Unlösbarkeit von I_{78a} erkannt werden: Keine Lösung kann ab oder bb als Teilwort enthalten und da b keine Startregel ist, kann statt I_{78a} die Instanz $\| \begin{smallmatrix} 0 & 01010 & 0101 \\ 00101 & 01 & 0 \end{smallmatrix} \|$ betrachtet werden. Diese Instanz entsteht durch Ersetzen der Regel b durch bc, da in einer Lösung vor jedem b ein c stehen muss und ist äquivalent zur Instanz $\| \begin{smallmatrix} 0 & 2202 & 22 \\ 022 & 2 & 0 \end{smallmatrix} \|$. Die Unlösbarkeit der letzten Instanz lässt sich mit Hilfe der Technik des Einfangens aus 39b ermitteln. ⌟

b **Beispiel:** Die Instanz $I_{78b} = \| \begin{smallmatrix} 0 & 0101 & 1 \\ 1001 & 01 & 0110 \end{smallmatrix} \|$ hat den Hauptnachfolger $I'_{78b} = \| \begin{smallmatrix} ac & acca & b & bca & caac & cab & cabca \\ b & c & bb & bc & a & ab & ac \end{smallmatrix} \|$, der wegen falscher a-Balance unlösbar ist. ⌟

c **Beispiel:** Sei I_{78c} die Instanz $I_{78c} = \| \begin{smallmatrix} 0 & 0001 & 1 \\ 001 & 1 & 0 \end{smallmatrix} \|$ mit dem Hauptnachfolger $I'_{78c} = \| \begin{smallmatrix} a & aac & b & b & c \\ c & a & ca & cccb & b \end{smallmatrix} \|$. Mit Hilfe von Ausschlussmengen wie in 34a erkennt ein Programm, dass der relevante Teil des Suchgraphen von I'^{R}_{78c} endlich ist und keinen Lösungspfad enthält. ⌟

d **Beispiel:** Sei I_{78d} die Instanz $I_{78d} = \| \begin{smallmatrix} 0 & 0010 & 0110 \\ 00 & 011 & 1 \end{smallmatrix} \|$. Zwar ist leicht zu ermitteln, dass keine Lösung mit aaa beginnt, trotzdem ist es schwer, die Unlösbarkeit

von I_{78d} zu erkennen: Im Suchgraphen gibt es einen Pfad, der mit *accaccabac caacabaccaaccaacabaccaaccaccaaccaacabaccaaccaccabaccaaccaccaaccaacabac caaccaccabaccaacabaccaaccaccabaccaaccaccaaccaacabaccaaccaccabaccaacaba ccaaccaacabaccaaccaccabaccaacabaccaaccaccabaccaaacc*$\cdots$ beschriftet ist und dem weder leicht anzusehen ist, ob er unendlich ist, noch, ob es in ihm ein einfaches Muster gibt.

Mit $\mathbb{B}(I_{78d}) = \{(b, aca), (c, ba), (a, a)\}(c, cca)^*(a, \varepsilon)$ ergibt sich für I_{78d} eine unendlich große Blockmenge. Es gibt also nicht *den* Nachfolger von I_{78d}. Trotzdem erlaubt diese Menge den Schluss auf die Unlösbarkeit von I_{78d} durch ein Balanceargument: Der erste Block in einer Zerlegung einer Lösungskoinzidenz muss der Block $(a, a)(c, cca)(c, cca)(a, \varepsilon) = (acca, acca)(\varepsilon, cca)$ sein. Die Anzahl der nun überschüssigen c in der $\downarrow$-Komponente kann nur durch den Block (ca, ba) verringert werden, wobei sich die Anzahl der b erhöht. Die Anzahl der b in der $\downarrow$-Komponente kann nur durch den Block $(bc^ia, acac^{2i}a)$ verringert werden, der jedoch wieder die Anzahl der c erhöht. Es ergibt sich eine Schleife. Somit ist kein Nachfolger von I_{78d} lösbar, also auch nicht I_{78d}. ⌟

Beispiel: Sei I_{79a} die Instanz $I_{79a} = \| \begin{smallmatrix} 0 \\ 001 \end{smallmatrix} \begin{smallmatrix} 1 \\ 01 \end{smallmatrix} \begin{smallmatrix} 1010 \\ 1 \end{smallmatrix} \|$ mit der unendlichen Blockmenge $\mathbb{B}(I_{79a}) = \{(aa, a), (a, b), (\varepsilon, c)\}\{(ca, ba), (c, bb)\}^*(b, \varepsilon)$. Wegen der notwendigen Balance des Symbols a kann in einer Zerlegung einer Lösungskoinzidenz nur die Teilmenge $(\varepsilon, c)\{(ca, ba), (c, bb)\}^*(b, \varepsilon)$ verwendet werden, die aber keinen gültigen letzten Block enthält. Also ist I_{79a} unlösbar. ⌟ *a*

Markierte Instanzen *b*

Kommen wir zur Anwendung der Theorie der Nachfolger. Ziel ist es, den Satz

Satz. (Halava [12]) $\mathcal{M}$ *ist entscheidbar.* *c*

zu beweisen. Die Menge der markierten Instanzen ist nicht zuletzt deshalb interessant, weil sich die Entscheidbarkeit der Menge $\mathcal{G}(2)$ der binären Instanzen aus der Entscheidbarkeit von $\mathcal{M}$ ableiten lässt.

Zur Erinnerung: Eine Instanz heißt genau dann markiert, wenn beide Morphismen markiert sind, was genau dann der Fall ist wenn sie nicht löschend sind und wenn die Bilder unterschiedlicher Symbole mit unterschiedlichen Symbolen beginnen. Mit anderen Worten legen wir durch die Angabe eines Wortes $h(w) \in B^+$ das Wort w eindeutig fest: Das erste Symbol von $h(w)$ legt das erste Symbol a von w und das Wort $h(a)^{-1}h(w)$ eindeutig fest.

Die für uns wichtigste Eigenschaft von $\mathcal{M}$ ist der Abschluss gegen Nachfolgerbildung. Es gilt

Lemma. $I \in \mathcal{M} \implies \mathrm{suc}(I) \subset \mathcal{M}$. *d*

a *Beweis:* Angenommen es gäbe einen Nachfolger I' von I, der nicht markiert ist. Sei ohne Beschränkung der Allgemeinheit h' nicht markiert. Da g nicht löschend ist, hat kein Block aus $\mathbb{B}(I)$ die Form (ε, u), also ist h' nicht löschend, sondern es gibt zwei verschiedene Blöcke $\{(a,\varepsilon)x, (a,\varepsilon)y\} \subseteq \mathbb{B}(I)$, die in ihrer $\uparrow$-Komponente mit dem selben Symbol beginnen. Nun legt $h(a)$ das erste Symbol b von $x_\downarrow$ und $y_\downarrow$ *und* den Überhang in $(h(a), g(b))$ eindeutig fest. Dieser Überhang kann nicht gleich $(\varepsilon,\varepsilon)$ sein, anderenfalls wären entweder $x = y = (\varepsilon,\varepsilon)$ und die Blöcke nicht verschieden, oder es würde sich gar nicht um Blöcke handeln. Beachte, dass Blöcke minimal sind. Wenn der Überhang aber nicht $(\varepsilon,\varepsilon)$ ist, dann legt er entweder das nächste Symbol in der $\uparrow$- oder das nächste Symbol in der $\downarrow$-Komponente der beiden Blöcke eindeutig fest und es ergibt sich schließlich, dass sie nicht verschieden sein können. □

Der Beweis stellt fest, dass jedes Symbol nur in höchstens einem Block als erstes Symbol in der $\uparrow$-Komponente und aus Symmetriegründen auch nur in höchstens einem Block als erstes Symbol in der $\downarrow$-Komponente verwendet wird. Da die ersten Symbole der einen stets auch das erste Symbol der jeweils anderen Komponente eindeutig festlegen, kann ein Nachfolger einer markierten Instanz höchstens so viele Regeln wie die Originalinstanz haben.

b **Folgerung.** $I \in \mathcal{M} \implies I' \in \mathrm{suc}(I) \implies |I'| \leq |I|$.

c **Herkömmliche**

An dieser Stelle haben wir bereits einen interessanten Punkt erreicht: Aus Lemma 73a ergibt sich, dass die Länge der kürzesten Lösung einer markierten herkömmlichen Instanz entweder gleich 1 ist oder bei Nachfolgerbildung echt kleiner wird. Aus Lemma 79d und Folgerung 80b folgt, dass für markierte herkömmliche Instanzen nur der Hauptnachfolger von Interesse ist. Bilden wir also die Kette der Hauptnachfolger einer lösbaren markierten herkömmlichen Instanz, dann erhalten wir nach endlich vielen Schritten eine Instanz, deren kürzeste Lösung die Länge 1 hat. Genau diese Idee liegt schließlich dem Entscheidungsalgorithmus für markierte Instanzen zu Grunde. Allerdings gilt es noch einige Fragen zu klären, zum Beispiel, ob wir bei unlösbaren Instanzen ebenfalls ein Maß finden, für das Nachfolgerketten absteigen.

Ein solches Maß ist wohlbekannt, es ist die *Suffixkomplexität* σ_I *von* I, die gegeben ist durch die Summe der Anzahlen der Suffixe der Morphismen h und g, also $\sigma_I = |\,\mathrm{S}_h\,| + |\,\mathrm{S}_g\,|$. (Halava, Hirvensalo, de Wolf [11]) Aus dem Suffixlemma 50c folgt, dass die Zustände des Transducer aus 64c für die Menge $\mathbb{I}(I)$ der inneren Blöcke Suffixe von h und von g sind. Da bei der Ermittlung der Blockmenge $\mathbb{B}(I)$ keine neuen Zustände erzeugt werden, folgt, dass zur Erzeugung eines Suffixes von h' ein Zustand aus $\mathrm{S}_g^\downarrow$ überdeckt werden

muss. Da h markiert ist, kann die Anzahl der Suffixe von h' höchstens so groß wie die Anzahl der Suffixe von g sein und analog die Anzahl der Suffixe von g' höchstens so groß wie die Anzahl der Suffixe von h.

Folgerung. $I \in \mathcal{M} \implies I' \in \mathrm{suc}(I) \implies \sigma_{I'} \leq \sigma_I$. *a*

Es gibt nur endlich viele Instanzen mit beschränkter Suffixkomplexität, denn wenn $\sigma_I \leq m$ für ein $m \in \mathbb{N}$ ist, dann kann kein Bild von h oder g länger als m sein. Damit können wir einen Teil von Satz 79c beweisen, es gilt

Lemma. (Halava [12]) $\mathcal{M} \cap \mathcal{P}$ *ist entscheidbar.* *b*

Beweis: Wir konstruieren die Kette der Hauptnachfolger von I. Die Suffix- *c*
komplexität steigt nicht, dass heißt, diese Kette wird schließlich periodisch. Da die Länge der kürzesten Lösung, falls eine Lösung existiert, entweder gleich 1 ist oder von Nachfolger zu Nachfolger echt kleiner wird, hat die Originalinstanz genau dann eine Lösung, wenn die erste sich wiederholende Instanz eine Lösung der Länge 1 hat. □

Dieser Beweis der Entscheidbarkeit markierter herkömmlicher Instanzen geht über die bisher bekannten Beweise hinaus, da hier erstmals klar wird, wo die Markiertheit wirklich benutzt wird. Sie wird *nicht* benutzt, um den Begriff des Blocks zu definieren und folgerichtig auch nicht, um einige wesentliche Eigenschaften von Blöcken und allgemeiner Koinzidenzen zu zeigen. Sie wird ebenfalls *nicht* benutzt, um zu zeigen, dass die Länge einer Lösung der sich wiederholenden Instanz 1 sein muss. Es handelt sich dabei um eine Eigenschaft, die für beliebige lösbare herkömmliche Instanzen zutrifft. Die Markiertheit wird lediglich benutzt, um zu zeigen, dass der Graph der Nachfolger endlich ist, im speziellen Fall herkömmlicher Instanzen sogar aus nur einem einzigen periodischen Pfad besteht. Zwar fließt die Markiertheit in den Beweis ein, dass die Suffixkomplexität nicht steigen kann, doch auch hier profitieren wir von einer Eigenschaft der beschränkten Suchrelation, die nicht nur für markierte Instanzen zutrifft.

Unmittelbar aus der Konstruktion können wir auch die Struktur der Gleichheitsmengen markierter herkömmlicher Instanzen ermitteln: Betrachten wir alle Instanzen in der Kette der Nachfolger von I, dann gibt es endlich viele Instanzen mit Lösungen der Länge 1, aus denen sich genau alle minimalen Lösungen von I rekonstruieren lassen.

Folgerung. (Halava [12]) *Wenn* $I \in \mathcal{M} \cap \mathcal{P}$ *eine markierte herkömmliche Instanz ist, dann gilt* $\mathrm{Eq}(I) = \{u_0, \ldots, u_{|I|-1}\}^*$*, wobei* $\{u_0, \ldots, u_{|I|-1}\}$ *die Menge der minimalen Lösungen von* I *ist und effektiv ermittelt werden kann.* *d*

Dass die Anzahl der minimalen Lösungen nicht größer als $|I|$ sein kann, folgt aus der Eindeutigkeit der Fortsetzung nach Festlegung des ersten Symbols einer minimalen Lösung. Mit anderen Worten haben wir im Suchgraphen nur am Knoten $(\varepsilon, \varepsilon)$ eine Wahlmöglichkeit.

a **Beispiel:** Beispiel 77b zeigt einen Ablauf des Entscheidungsalgorithmus für markierte herkömmliche Instanzen. ⌟

b **Beispiel:** Die Instanz $I_{82b} = |\,{}^{\varepsilon}_{10}\,|\,{}^{0}_{010}\,{}^{101}_{1}\,|\,{}^{10}_{\varepsilon}\,|$ mit $\mathrm{Eq}(I_{82b}) = (b(ba)^*a)^*$ zeigt, dass Folgerung 81d nicht für markierte allgemeine Instanzen gilt. ⌟

c **Allgemeine**

Nach diesem Erfolg wollen wir uns markierten allgemeinen Instanzen zuwenden. Zwar steigt auch für markierte allgemeine Instanzen die Suffixkomplexität nicht an, die Anzahl der Regeln nimmt nicht zu und die Nachfolger sind markiert, aber neben der Möglichkeit von Monolithen, die bei der Nachfolgerbildung nicht berücksichtigt werden, haben wir es auch mit nichtleeren Start- und Endblöcken zu tun. Beginnen wir mit dem einfachen

d **Lemma.** *$I \in \mathcal{M} \implies |\mathbb{S}(I)| \leq 1$ und $\mathbb{S}(I)$ kann effektiv ermittelt werden. Wenn $\mathbb{S}(I) = \{(x, y)\}$ ist, dann gilt $||x| - |y|| \leq \mathrm{wd}(I)$.*

e *Beweis:* Zunächst folgt die Behauptung für $s = (\varepsilon, \varepsilon)$ aus Lemma 69e. Wenn $s \neq (\varepsilon, \varepsilon)$ ist, dann ist wie schon im Beweis von Lemma 79d die Fortsetzung in der $\uparrow$- oder der $\downarrow$-Komponente eindeutig bestimmt und wir können keine zwei Startblöcke finden, die verschieden sind.

In 69d wird beschrieben, wie $\mathbb{S}(I)$ aufgezählt werden kann. Dazu wird ein Transducer konstruiert, der eine Obermenge von $\mathbb{S}(I)$ akzeptiert und aus dessen Sprache werden die Elemente gelöscht, die einen echten inneren Block als Suffix haben. Dieser Transducer akzeptiert für markierte Instanzen nur eine endliche Sprache, anderenfalls gäbe es einen Zustand mit zwei verschiedenen Fortsetzungen.

Bleibt zu zeigen, dass die Längendifferenz $||x| - |y||$ des möglichen einzigen Startblocks (x, y) nicht größer als $\mathrm{wd}(I)$ sein kann. Wir zeigen sogar, dass $||x| - |y||$ nicht größer als $M = \max\{d, \mathrm{wd}(h), \mathrm{wd}(g)\}$ mit $d = ||s_\uparrow| - |s_\downarrow||$ sein kann. Hier sind $\mathrm{wd}(h) = \max\{h(a) \mid a \in A\}$ und $\mathrm{wd}(g) = \max\{g(a) \mid a \in A\}$ die Längen der längsten h- und der längsten g-Bilder. Der Grund ist einfach: Überdecken wir s mit den Bildern von (x, y), dann muss der Überhang nach spätestens $d \leq M$ Schritten die Position ändern. Der neue Überhang hat höchstens die Länge M, also kann die Differenz $||x| - |y||$ nicht größer als M werden. □

Etwas komplizierter ist der Umgang mit den Monolithen. Wir haben

Lemma. *$I \in \mathcal{M} \implies \mathbb{M}(I) = \mathrm{P}_{\mathbb{M}}(I)\,\mathrm{t}_{\mathbb{M}}(I)^*$, wobei $\mathrm{t}_{\mathbb{M}}(I) \in A^{*\updownarrow}$ und $\mathrm{P}_{\mathbb{M}}(I) \subset A^{*\updownarrow}$ mit $|\,\mathrm{P}_{\mathbb{M}}(I)| \leq 1$ effektiv ermittelt werden können.* *a*

Beweis: Zunächst gilt mit Lemma 71b für $s = (\varepsilon, \varepsilon)$ oder $f = (\varepsilon, \varepsilon)$ die Behauptung mit $\mathrm{P}_{\mathbb{M}}(I) = \emptyset$ und $\mathrm{t}_{\mathbb{M}}(I) = (\varepsilon, \varepsilon)$. Wenn $s \neq (\varepsilon, \varepsilon)$ und $f \neq (\varepsilon, \varepsilon)$ sind, dann betrachten wir zunächst alle schleifenfreien Pfade im Transducer aus 71a, die in s beginnen und in $\overline{f}$ enden. Gäbe es zwei verschiedene Beschriftungen, dann könnten wir erneut nicht die Stelle finden, an denen sie sich unterscheiden, da jede Fortsetzung eindeutig festgelegt ist. Also nehmen wir in $\mathrm{P}_{\mathbb{M}}(I)$ höchstens ein Element auf. Analog gibt es nun entweder keine Schleife von $\overline{f}$ zu $\overline{f}$ und wir setzen $\mathrm{t}_{\mathbb{M}}(I) = (\varepsilon, \varepsilon)$ oder es gibt genau eine und wir setzen $\mathrm{t}_{\mathbb{M}}(I)$ auf deren Beschriftung. *b*

Es bleibt zu prüfen, ob wir nicht doch zerlegbare Elemente erzeugt haben: Da $\mathbb{S}(I)$ und $\mathbb{B}(I)$ beide endlich sind, ist $\mathrm{P}_{\mathbb{M}}(I)$ nach endlich vielen Schritten überdeckt und in der Schleife um $\overline{f}$ können wir nur endlich viele Zwischenzustände erreichen. □

Damit stellen die Monolithen keine Hürde mehr dar:

Lemma. *Wenn $I \in \mathcal{M}$ eine markierte Instanz ist, dann ist entscheidbar, ob $\mathrm{id}(A^*) \cap \mathbb{M}(I)$ leer ist oder nicht. Außerdem können die Elemente in $\mathrm{id}(A^*) \cap \mathbb{M}(I)$ effektiv aufgezählt werden.* *c*

Beweis: Nach Lemma 83a können wir $\mathbb{M}(I) = pt^*$ für gewisse effektiv gegebene $\{p, t\} \subset A^{*\updownarrow}$ annehmen. Wenn $t \in \mathrm{id}(A^*)$, dann ist $pt^* \subset \mathrm{id}(A^*)$, genau dann, wenn $p \in \mathrm{id}(A^*)$ ist. Wenn $t \notin \mathrm{id}(A^*)$, dann ist $pt^k \in \mathrm{id}(A^*)$ für höchstens ein leicht zu berechnendes k. □ *d*

Es verbleibt die Behandlung der Endblöcke. Hier zeigen wir zunächst in Analogie zu 83a das

Lemma. *$I \in \mathcal{M} \implies \mathbb{F}(I) = \mathrm{P}_{\mathbb{F}}(I)\,\mathrm{t}_{\mathbb{F}}(I)^*$, wobei $\mathrm{t}_{\mathbb{F}}(I) \in A^{*\updownarrow}$ und $\mathrm{P}_{\mathbb{F}}(I) \subset A^{*\updownarrow}$ mit $|\,\mathrm{P}_{\mathbb{F}}(I)| \leq |A| - |\,\mathbb{B}(I)|$ effektiv ermittelt werden können. Darüber hinaus gilt $\{a, b\} \in A^{\updownarrow} \implies ax \in \mathbb{B}(I) \implies by \in \mathrm{P}_{\mathbb{F}}(I) \implies a \neq b$.* *e*

Beweis: Im Prinzip führen wir den Beweis von Lemma 83a noch einmal. Wir gehen diesmal allerdings von $(\varepsilon, \varepsilon)$ aus, so dass die erste Transition nicht festgelegt ist. Nach der ersten Transition folgt wie oben, dass es keine verschieden beschrifteten Pfad nach $\overline{f}$ geben kann und dass nur dort eine einzige Schleife möglich ist. Ebenfalls wie oben sichern wir, dass wir keine zerlegbaren Beschriftungen übersehen. Es bleibt $|\,\mathrm{P}_{\mathbb{F}}(I)| \leq |A| - |\,\mathbb{B}(I)|$ zu *f*

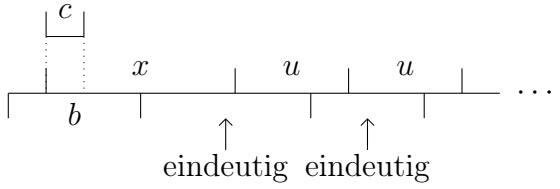
zeigen: Wir wissen, dass es in $\mathbb{B}(I)$ keine zwei verschiedenen Elemente mit dem selben Anfang aus $A^{\updownarrow}$ gibt. Also muss es zwei verschiedene Elemente in $\mathrm{P}_{\mathbb{F}}(I)$ mit dem selben Anfang geben oder einen gemeinsamen Anfang eines Elements aus $\mathbb{B}(I)$ und eines Elements aus $\mathrm{P}_{\mathbb{F}}(I)$. Das ist aber beides unmöglich, denn wir argumentieren erneut, dass wir für unterschiedliche Elemente mit gleichem Anfang nicht die Stelle finden können, an der sie sich unterscheiden. □

Auch für die Menge $\mathbb{F}(I)$ der Endblöcke markierter Instanzen kennen wir nun die genaue Struktur. Für Monolithen konnten wir damit zeigen, dass es entscheidbar ist, ob es monolithische Lösungen gibt. Die Endblöcke bereiten hier mehr Kopfzerbrechen, denn es kann unendlich viele davon geben und es ist nicht unmittelbar ersichtlich welche wichtig sind, denn es könnte sehr wohl unendlich viele geben, die alle suffixvergleichbar sind. Es ist aber doch möglich eine endliche Menge relevanter Endblöcke effektiv zu ermitteln.

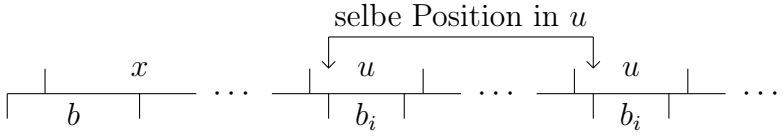
a **Lemma.** $I \in \mathcal{M} \implies \exists F \subseteq \mathbb{F}(I) : (\mathrm{id}(A^*) \cap \mathbb{S}(I)\,\mathbb{B}(I)^*\,\mathbb{F}(I) \neq \emptyset \implies \mathrm{id}(A^*) \cap \mathbb{S}(I)\,\mathbb{B}(I)^* F \neq \emptyset)$, *wobei* $|F| < \infty$ *effektiv ermittelt werden kann.*

b *Beweis:* Nach Lemma 83e können wir $\mathbb{F}(I) = \mathrm{P}_{\mathbb{F}}(I)\,\mathrm{t}_{\mathbb{F}}(I)^*$ für gewisse effektiv gegebene $\mathrm{t}_{\mathbb{F}}(I) \in A^{*\updownarrow}$ und $\mathrm{P}_{\mathbb{F}}(I) \subset A^{*\updownarrow}$ mit $|\,\mathrm{P}_{\mathbb{F}}(I)| < \infty$ annehmen. Sei $\mathrm{t}_{\mathbb{F}}(I) = (u, v)$. Wenn $|u| = |v|$ ist, dann erfüllt $F = \mathrm{P}_{\mathbb{F}}(I)$ die Behauptung. Sei deshalb ohne Beschränkung der Allgemeinheit $|u| > |v|$. Wir nehmen zunächst in F die endliche Menge $\{(xu^k, yv^k) \mid (x, y) \in \mathrm{P}_{\mathbb{F}}(I), |xu^k| \leq |yv^k|\}$ auf. Das sind alle Endblöcke, deren $\uparrow$-Komponente nicht länger als die $\downarrow$-Komponente ist. Für jedes der endlich vielen $(x, y) \in \mathrm{P}_{\mathbb{F}}(I)$ führen wir nun folgende Konstruktion aus: Wir bezeichnen mit $B = (\mathbb{S}(I) \cup \mathbb{B}(I))_{\downarrow}$ die Menge der $\downarrow$-Komponenten von Startblöcken und Blöcken von I und mit c das erste Symbol von xu. Die Menge B ist endlich, da $\mathbb{S}(I)$ und $\mathbb{B}(I)$ beide endlich sind und c existiert, anderenfalls wäre $\mathbb{F}(I) = \{(\varepsilon, \varepsilon)\}$ und die Behauptung würde für $F = \mathbb{F}(I)$ gelten.

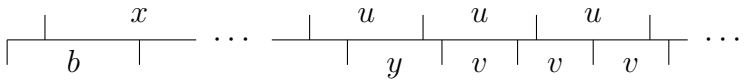
Wir suchen nach Positionen, an denen der Endblock (xu^k, yv^k) starten kann, wobei xu^k echt länger als yu^k ist, der Exponent k aber noch unbekannt. Da xu^k ebenfall mit c beginnt, muss es ein $b \in B$ geben, das das Symbol c irgendwo enthält. Wir versuchen nun den Exponenten k zu bestimmen. Dazu legen wir die $\uparrow$-Komponente auf xu^* fest, so dass c überdeckt wird und versuchen die $\downarrow$-Komponente um Wörter aus $B \cup \{y\}$ zu verlängern.



Nun ist jede Fortsetzung in der ↓-Komponente eindeutig festgelegt, da nach Lemma 83e jedes Wort in $B \cup \{y\}$ mit einem anderen Symbol beginnt. Wenn irgendwann keine Verlängerung mehr möglich ist, dann überprüfen wir einfach, ob wir einen Endblock gefunden haben und wenn ja, dann nehmen wir ihn in F auf. Da B endlich ist, wird anderenfalls nach höchstens $|u|$ Verlängerungen entweder y angewendet, oder das selbe Element b_i aus B wird zum zweiten Mal verwendet, um die selbe Position in u zu überdecken.



In diesem Fall haben wir eine Schleife gefunden, die kein neues Element zu F beiträgt. Wenn wir jedoch y anwenden können, dann versuchen wir durch Verlängern mit v einen kompletten Endblock zu erzeugen.



Da $|v| < |u|$, wird nach endlich vielen Verlängerungen die Anzahl der v die Anzahl der u übertreffen. Bei gleicher Anzahl testen wir, ob es sich um einen Endblock handelt und falls ja, nehmen wir ihn in F auf. Falls bereits vor Erreichen der gleichen Anzahl keine Verlängerung mit v möglich ist, dann haben wir keinen neuen Endblock gefunden. Der Beweis schließt mit der Bemerkung, dass wir insgesamt nur endlich viele Schritte benötigen, die alle effektiv ausführbar sind. □

Die Tatsache, dass nur endlich viele Endblöcke relevant sind, wird auch von Vesa Halava gezeigt, allerdings mit wesentlich mehr Aufwand: Er zeigt zunächst, dass $\mathbb{F}(I) = \mathrm{P}_{\mathbb{F}}(I)\,\mathrm{t}_{\mathbb{F}}(I)^* w$ ist, also eine schwächere Version von Lemma 83e. (Halava [12]) Vesa hat mir gesagt, dass er immer gefühlt hat, dass $w = (\varepsilon, \varepsilon)$ sein muss, aber nie einen Beweis dafür gefunden hat. Durch die komplexere Struktur von $\mathbb{F}(I)$ ist es für ihn auch schwieriger zu zeigen, dass nur endlich viele Endblöcke relevant sind. Aber kommen wir zum

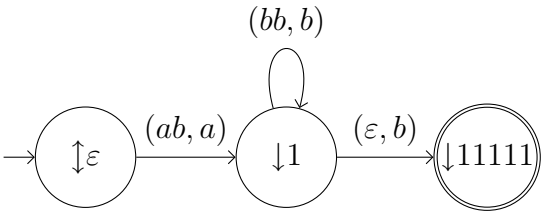
Beweis: (von Satz 79c) Aus den obigen Lemmata folgt zunächst, dass der relevante Teil des Nachfolgergraphen markierter Instanzen nur markierte Instanzen enthält und endlich ist, wenn wir nur die Morphismen betrachten. Für jede Instanz im Nachfolgergraphen können wir entscheiden, ob sie monolithische Lösungen hat. Falls es solche Instanzen gibt, dann ist die Originalinstanz lösbar. Falls nicht, dann betrachten wir nur die Pfade, die *a*

schließlich periodisch sind, da Instanzen ohne Nachfolger und ohne monolithische Lösungen unlösbar sind. In einem schließlich periodischen Pfad gibt es eine Instanz I_l, die transitiver Nachfolger von sich selbst ist. Wir stellen mit Lemma 82d fest, dass die Längendifferenzen aller Startblöcke aller Instanzen im Nachfolgergraphen durch eine effektiv gegebene Konstante c nach oben beschränkt sind. Zusammen mit Lemma 73a folgt, dass die Originalinstanz genau dann eine Lösung hat, wenn I_l eine Lösung hat, die nicht länger als $|B|^c$ ist. □

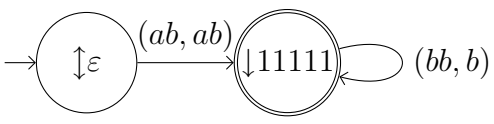
Der Aufwand für markierte allgemeine Instanzen ist doch erheblich höher als der für markierte herkömmliche. Dennoch stellt auch dieser Beweis einen deutlichen Fortschritt im Vergleich zu den bisher bekannten Beweisen dar, da die Markiertheit nur an wenigen, klar zu identifizierenden Stellen verwendet wird. Viele Fragen über Koinzidenzen haben wir auf allgemeinerer Ebene beantwortet und der Rückgriff auf die beschränkte Suchrelation erlaubt eine einheitliche Sprache für die restlichen technischen Details. Die Struktur der Endblöcke wird genauer als bisher beschrieben und dem roten Faden kann viel leichter als dem in der soweit kohärentesten Darstellung von Vesa Halava gefolgt werden. Ziehen wir noch in Betracht, dass wir auf der allgemeineren Ebene tatsächlich echt mehr als nur markierte Instanzen entscheiden können, dann können wir die neue Darstellung der Theorie der Nachfolger guten Gewissens als Erfolg bezeichnen.

a **Beispiel:** Beispiel 77a zeigt einen Ablauf des Entscheidungsalgorithmus für markierte allgemeine Instanzen. ⌟

b **Beispiel:** Die Instanz $I_{86b} = |\,{}^{\varepsilon}_{\varepsilon}\,|\,{}^{0}_{0111}\;{}^{11}_{1111}\,|\,{}^{11111}_{\varepsilon}\,|$ scheint Lemma 83e zu widersprechen, denn die explizite Konstruktion liefert zunächst den Transducer



der die Schleife *nicht* um $\overline{f} = \downarrow 11111$ hat. Dieser Transducer akzeptiert aber die selbe Sprache wie der Transducer



der $\mathbb{F}(I_{86b}) = (ab, ab)(bb, b)^*$ akzeptiert. Tatsächlich gelten $h(abb^{2i}) = 01^{4i+2}$ und $g(abb^i) = 01^{4i+7}$. Es gilt $\mathbb{B}(I_{86b}) = \{(bb, b)\}$ und, da $s = (\varepsilon, \varepsilon)$, gelten $\mathbb{S}(I_{86b}) = \{(\varepsilon, \varepsilon)\}$ und $\mathbb{M}(I_{86b}) = \emptyset$. Konstruieren wir die endliche Menge F relevanter Endblöcke wie in Lemma 84a, dann nehmen wir zunächst (ab, ab) in F auf und keine weiteren Endblöcke, da sich $B = \{b\}$ und $c = a$ ergeben. Die einzige Lösungskoinzidenz (ab, ab) lässt sich in $(\varepsilon, \varepsilon)(\varepsilon, \varepsilon)(ab, ab)$ zerlegen, wobei sowohl $(\varepsilon, \varepsilon) \in \mathbb{S}(I_{86b})$ ist, als auch $(\varepsilon, \varepsilon) \in \mathbb{B}(I_{86b})^*$. ⌟

Beispiel: Betrachten wir die Instanz $I_{87a} = |\begin{smallmatrix}010\\ \varepsilon\end{smallmatrix}|\begin{smallmatrix}111\\ 01\end{smallmatrix}\ \begin{smallmatrix}01\\ 11111011\end{smallmatrix}|\begin{smallmatrix}1\\ \varepsilon\end{smallmatrix}|$, dann erhalten wir $\mathbb{S}(I_{87a}) = \mathbb{F}(I_{87a}) = \emptyset$, $\mathbb{B}(I_{87a}) = \{(b, a)\}$ und $\mathbb{C}(I_{87a}) = \mathbb{M}(I_{87a}) = (\varepsilon, aa)(aab, b)^*$. Mit Lemma 83c ermitteln wir $\mathrm{Eq}(I_{87a}) = \{aab\}$. ⌟ *a*

Beispiel: Gehen wir von I_{87a} zu $I_{87b} = |\begin{smallmatrix}\varepsilon\\ \varepsilon\end{smallmatrix}|\begin{smallmatrix}\#010\ 111\\ \#\ \ 01\end{smallmatrix}\ \begin{smallmatrix}01\\ 11111011\end{smallmatrix}|\begin{smallmatrix}1\\ \varepsilon\end{smallmatrix}|$ über, dann überlegen wir uns zunächst, dass $\mathrm{Eq}(I_{87b}) = \{abbc\} = a\varphi(\mathrm{Eq}(I_{87a}))$ mit dem Morphismus $\varphi : \{a \mapsto b, b \mapsto c\}$ gelten muss. Nun gelten aber $\mathbb{S}(I_{87b}) = \{(\varepsilon, \varepsilon)\}$, $\mathbb{M}(I_{87b}) = \emptyset$ und $\mathbb{B}(I_{87b}) = \{(c, b)\}$. Die Menge der relevanten Endblöcke ermitteln wir mit Lemma 84a zu $F = \{(a, abb), (abbc, abbc)\}$. Das heißt, aus der monolithischen Lösung abb von I_{87a} ist die nichtmonolithische Lösung $abbc$ von I_{87b} geworden, die genau einem Endblock entspricht. ⌟ *b*

Markierte Nachfolger *c*

Markierte Instanzen sind wesentlich deshalb entscheidbar, weil die Nachfolgerbildung wieder auf markierte Instanzen führt. Es liegt also nahe, nach Mengen von Instanzen zu fragen, deren Nachfolger markiert sind. Wie Vesa Halava definieren wir die Menge $\mathcal{U}$ der „unique block instances“, die alle Instanzen umfasst, deren Blockmenge markiert ist. (Halava et al. [10, UC1]) Wir haben genau dann $I \in \mathcal{U}$, wenn gilt $a \in A^{\updownarrow} \implies \{ax, ay\} \subseteq \mathbb{B}(I) \implies x = y$, wenn also verschiedene Blöcke mit verschiedenen Symbolen beginnen. Aus Lemma 67a folgt, dass es entscheidbar ist, ob eine Instanz in $\mathcal{U}$ liegt.

Folgerung. $I \in \mathcal{U} \implies \mathrm{suc}(I) \subset \mathcal{M}$. *d*

Tatsächlich gehen wir ja stets davon aus, dass jede Regel höchstens ein Mal verwendet wird. Ein nichtmarkierter Nachfolger widerspricht dann direkt der Definition von $\mathcal{U}$. Aus dem Beweis von Lemma 79d entnehmen wir die

Folgerung. $\mathcal{M} \subset \mathcal{U}$. *e*

Die Inklusion ist echt, wie die Instanz $I_{87e} = \|\begin{smallmatrix}0\\ 01\end{smallmatrix}\ \begin{smallmatrix}102\\ 02\end{smallmatrix}\|$ mit $\mathbb{B}(I_{87e}) = \{(ab, ab)\}$ zeigt. Wir haben eine echte Obermenge der markierten Instanzen, die immer noch entscheidbar ist.

a **Lemma.** *Sei $\mathcal{I} \subseteq \mathcal{U}$ eine Instanzenmenge von Instanzen $I \in \mathcal{I}$ mit eindeutigen Blöcken, für die die Mengen $\mathbb{S}(I)$ und $\mathbb{F}(I)$ jeweils beide endlich sind und für die jeweils entscheidbar ist, ob $\mathrm{id}(A^*) \cap \mathbb{M}(I) = \emptyset$ gilt. Dann ist $\mathcal{I}$ entscheidbar.*

b *Beweis:* Nach Voraussetzung ist entscheidbar, ob I monolithische Lösungen hat. Falls das nicht der Fall ist, dann ist I nach den Lemmata 75d und 76b genau dann lösbar, wenn es einen lösbaren Nachfolger von I gibt. Wenn $\mathbb{S}(I)$ und $\mathbb{F}(I)$ beide endlich sind, dann hat I nur endlich viele Nachfolger, die nach Folgerung 87d markiert sind. Es ist also nach Satz 79c entscheidbar, ob einer der Nachfolger lösbar ist. □

c **Folgerung.** *$\mathcal{U} \cap \mathcal{P}$ ist entscheidbar.*

d **Beispiel:** Die Instanz $I_{88d} = \| \begin{smallmatrix} 0 & 010 & 11 \\ 01 & 0 & 110 \end{smallmatrix} \|$ ist aus $\mathcal{U}$. Die Kette der Hauptnachfolger ist $I'_{88d} = \| \begin{smallmatrix} a & b & ca \\ b & ab & c \end{smallmatrix} \|$, $I''_{88d} = \| \begin{smallmatrix} ab & b & cb \\ b & a & cb \end{smallmatrix} \|$, $I'''_{88d} = \| \begin{smallmatrix} a & b & c \\ ba & a & c \end{smallmatrix} \|$, $I''''_{88d} = \| \begin{smallmatrix} a & ba & c \\ b & a & c \end{smallmatrix} \|$, $I'''''_{88d} = \| \begin{smallmatrix} a & b & c \\ b & ab & c \end{smallmatrix} \|$, $I''''''_{88d} = \| \begin{smallmatrix} ab & b & c \\ b & a & c \end{smallmatrix} \|$ und $I'''''''_{88d} = I'''_{88d}$. Aus $\mathrm{Eq}(I'''_{88d}) = c^*$ rekonstruieren wir $\mathrm{Eq}(I''_{88d}) = c^*$, $\mathrm{Eq}(I'_{88d}) = (cb)^*$ und $\mathrm{Eq}(I_{88d}) = (cab)^*$. ⌟

e **Beispiel:** Die Instanz $I_{88e} = \| \begin{smallmatrix} 0 & 01 & 110 \\ 000 & 010 & 11 \end{smallmatrix} \|$ ist aus $\mathcal{U}$. Die Kette der Hauptnachfolger ist $I'_{88e} = \| \begin{smallmatrix} aaa & ba & caa \\ a & b & ca \end{smallmatrix} \|$, $I''_{88e} = \| \begin{smallmatrix} a & b & c \\ aaa & ba & ca \end{smallmatrix} \|$, $I'''_{88e} = \| \begin{smallmatrix} aaa & ba & ca \\ a & b & c \end{smallmatrix} \|$ und $I''''_{88e} = I''_{88e}$. Da I''_{88e} keine Lösung der Länge 1 hat, sind I_{88e} und alle Nachfolger unlösbar. Es gilt $\mathrm{Eq}^\omega(I_{88e}) = \{a, b, c\}a^\omega$, das ist die Menge der Fixpunkte des Morphismus $\{a \mapsto aaa, b \mapsto ba, c \mapsto ca\}$. ⌟

f **Beispiel:** Die Instanz $I_{88f} = \| \begin{smallmatrix} 00 & 0110 & 1001 & 11 \\ 0011 & 10 & 01 & 1100 \end{smallmatrix} \|$ ist aus $\mathcal{U}$. Die Kette der Hauptnachfolger ist $I'_{88f} = \| \begin{smallmatrix} ad & b & c & da \\ a & cb & bc & d \end{smallmatrix} \|$, $I''_{88f} = \| \begin{smallmatrix} a & bc & cb & d \\ ad & c & b & da \end{smallmatrix} \|$ und $I'''_{88f} = I'_{88f}$. Da I'_{88f} keine Lösung der Länge 1 hat, sind I_{88f} und alle Nachfolger unlösbar. Es gilt $\mathrm{Eq}^\omega(I_{88f}) = \{\pi\tau^\omega(0), \pi\tau^\omega(1)\}$, wobei τ der Morphismus aus Beispiel 11a und π der Morphismus $\{0 \mapsto a, 1 \mapsto d\}$ ist. ⌟

g **Beispiel:** (Halava et al. [10]) Die Instanz $I_{88g} = \| \begin{smallmatrix} ab & abc & cccc & b \\ a & babc & cc & bbc \end{smallmatrix} \|$ ist aus $\mathcal{U}$. Die Kette der Hauptnachfolger ist $I'_{88g} = \| \begin{smallmatrix} ab & c & db \\ ab & cc & b \end{smallmatrix} \|$, $I''_{88g} = \| \begin{smallmatrix} a & bb \\ a & b \end{smallmatrix} \|$, $I'''_{88g} = \| \begin{smallmatrix} a & b \\ a & bb \end{smallmatrix} \|$ und $I''''_{88g} = I''_{88g}$. Wir rekonstruieren $\mathrm{Eq}(I_{88g}) = (ab)^*$. Die unendliche Lösung $c^\omega \in \mathrm{Eq}^\omega(I_{88g})$ lässt sich aus dem Fixpunkt des Morphismus $\{c \mapsto cc\}$ rekonstruieren, allerdings ist die unendliche Lösung $adc^\omega \in \mathrm{Eq}^\omega(I_{88g})$ nicht mehr aus der Nachfolgerkette rekonstruierbar. ⌟

h **Beispiel:** (Halava et al. [10]) Die Instanz $I_{88h} = \| \begin{smallmatrix} a & ba & cabbb & ab \\ ac & bac & abbb & c \end{smallmatrix} \|$ ist aus $\mathcal{U}$. Darüber hinaus ist h nicht injektiv, da $h(ab) = aba = h(da)$ gilt. Die Kette der Hauptnachfolger ist $I'_{88h} = \| \begin{smallmatrix} ac & bc & c \\ ac & bc & dc \end{smallmatrix} \|$ und $I''_{88h} = \| \begin{smallmatrix} a & b \\ a & b \end{smallmatrix} \|$ und $I'''_{88h} = I''_{88h}$. Es gilt $\mathrm{Eq}(I_{88h}) = \{ac, bc\}^*$. ⌟

Unendliche Lösungen

a

Es ist wohl bekannt, dass für markierte herkömmliche Instanzen entscheidbar ist, ob es unendliche Lösungen gibt. Vesa Halava führt einen entsprechenden Beweis. (Halava [12]) Dazu teilt er die unendlichen Lösungen in drei Klassen ein: Die Klasse der unendlichen Lösungen, die aus endlichen Lösungen zusammengesetzt sind, die Klasse der unendlichen Lösungen, die nicht aus endlichen Lösungen zusammen gesetzt sind, deren Koinzidenz sich aber in Blöcke zerlegen lässt und die Klasse der unendlichen Lösungen, die nicht aus endlichen Lösungen zusammengesetzt sind und deren Koinzidenz sich nicht in Blöcke zerlegen lässt. Anschließend zeigt er, dass für alle drei Typen Entscheidungsalgorithmen existieren. Dazu untersucht er, wie sich die unendlichen Gleichheitsmengen bei Nachfolgerbildung verhalten. Je nach Typ der unendlichen Lösung bleibt sie entweder erhalten, entspricht dem Fixpunkt einer Wortfolge oder führt zu verschwindenden Symbolen, die den dritten Typ reflektieren.

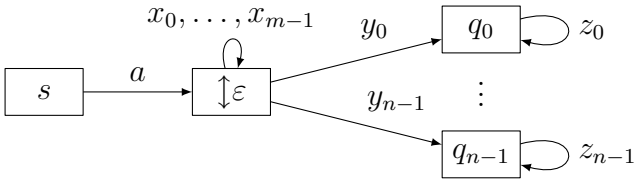
Wir werden zeigen, wie mit Folgerung 51a für markierte allgemeine Instanzen nicht nur entschieden werden kann, ob es unendliche Lösungen gibt, sondern sogar, wie die unendliche Gleichheitsmenge effektiv konstruiert werden kann.

Satz. *$\omega\,\mathcal{M}$ ist entscheidbar.*

b

Beweis: Sei $I \in \mathcal{M}$ eine markierte Instanz. Nach Folgerung 51a genügt es, alle unendlichen Pfade der Form (ω, ω) durch $\hookrightarrow_I$ effektiv zu konstruieren. Da I markiert ist, hat $\hookrightarrow_I^\star(s)$ die Struktur

c



wobei die Elemente der Menge $\{x_0, \ldots, x_{m-1}, y_0, \ldots, y_{n-1}\}$ paarweise verschieden beginnen. Zur Konstruktion eines ω-Pfades starten wir mit dem leeren Pfad in s. Nur am Knoten $\updownarrow\varepsilon$ ist eine Wahl der Fortsetzung möglich und nur dann, wenn der bisherige Pfad mit einem Element aus $\mathrm{id}(A^*)$ beschriftet ist. Erreichen wir zum *zweiten* Mal diese Situation, dann hat I unendliche Lösungen: Wir können den Pfad immer weiter verlängern, wenn wir immer wieder so wie beim ersten Mal wählen. Es ist entscheidbar, ob es solche Pfade gibt, denn das ist genau dann der Fall, wenn die markierte Instanz mit $\mathrm{id}(A^*)^{-1}a$

als Startpaar, $(\varepsilon, \varepsilon)$ als Endpaar und den Regeln $\{x_0, \ldots, x_{m-1}\}$ und die markierte herkömmliche Instanz mit den Regeln $\{x_0, \ldots, x_{m-1}\}$ jeweils Lösungen haben.

Interessant sind also nur Pfade, die höchstens ein Mal eine Wahl für die Fortsetzung erlauben. Pfade, die eine Wahl erlauben, setzen sich aus einer Lösung der markierten Instanz $(\mathrm{id}(A^*)^{-1}a, \{x_0, \ldots, x_{m-1}\}, (\varepsilon, \varepsilon))$ und einem von maximal m Pfaden zusammen, die in $(\varepsilon, \varepsilon)$ beginnen und keine Wahl der Fortsetzung erlauben. Insgesamt können wir uns auf Pfade beschränken, die in $(\varepsilon, \varepsilon)$ beginnen, bereits eine initiale Beschriftung haben und keine Wahl der Fortsetzung erlauben.

Wird irgendwann einer der y_i-Pfade gewählt, dann erreichen wir den Zustand q_i mit der Schleife $z_i = (\alpha, \beta)$ über einen Pfad, der ohne Beschränkung der Allgemeinheit mit (u, uv) beschriftet ist. Dieser Pfad kann genau dann zu einem ω-Pfad erweitert werden, wenn $\alpha^{|\beta|}v = v\beta^{|\alpha|}$ gilt. Es genügt, zu entscheiden, ob ein Pfad p, der keine Wahl der Fortsetzung erlaubt und nicht aus $\mathrm{id}(A^*)$ ist, unendlich oft durch $\updownarrow\varepsilon$ verlaufen kann. Das ist genau dann der Fall, wenn der Nachfolger $(\mathrm{id}(A^*)^{-1}p, \{x_0, \ldots, x_{m-1}\}, (\varepsilon, \varepsilon))$ von I unendliche Lösungen hat, die nicht aus endlichen Lösungen zusammen gesetzt sind. Wir starten die Prozedur rekursiv für diesen Nachfolger und brechen die Rekursion ab, wenn eine Instanz zum zweiten Mal auftaucht. Die Originalinstanz hat genau dann Lösungen, wenn die sich wiederholende Instanz eine mit (u, uv) oder (uv, u) beschriftete Schleife um $\updownarrow\varepsilon$ hat. □

Die verschiedenen Typen von Vesa Halava finden sich auch in diesem Beweis wieder: Pfade mit zwei oder mehr Wahlmöglichkeiten sind aus Lösungen zusammengesetzt, Pfade mit höchstens einer Wahlmöglichkeit verlaufen entweder unendlich oft durch $\updownarrow\varepsilon$ und sind dann in Blöcke zerlegbar, oder sie verlaufen nur endlich oft durch $\updownarrow\varepsilon$ und sorgen dafür, dass die Anzahl der Blöcke echt kleiner als die Anzahl der Symbole in A ist. Insbesondere wird hier klar, dass der dritte Halava-Typ einem *unproduktiven* Pfad durch $\hookrightarrow_I^\star(s)$ entspricht, in dem Sinne, dass über diesen Pfad der Knoten $\updownarrow\varepsilon$ nicht erreicht werden kann.

a

Binäre Instanzen

Als Anwendung der Entscheidbarkeit markierter Instanzen zeigen wir die Entscheidbarkeit binärer Instanzen. Betrachten wir den gesamten Bogen, von der Zerlegung der Koinzidenzmenge über den Begriff des Nachfolgers und die Entscheidbarkeit markierter Instanzen, dann stellen wir fest, dass der Beweis der Entscheidbarkeit binärer Instanzen erheblichen Aufwand verursacht. Die Entscheidbarkeit von Instanzen mit nur einer Regel folgt aus

einem einfachen Abzählargument, der Sprung zur Entscheidbarkeit binärer Instanzen ist gewaltig. Dazu im Gegensatz steht die Beobachtung, dass es nicht gelingt, schwierige binäre Instanzen zu konstruieren. Dem Autor ist zum Beispiel keine binäre herkömmliche Instanz bekannt, deren minimale Lösung länger als $\max\{1, ||h(a)| - |g(a)|| + ||h(b)| - |g(b)||\}$ ist. Vieles deutet darauf hin, dass es einen kurzen und elementaren Beweis der Entscheidbarkeit binärer Instanzen gibt.

Herkömmliche *a*

Wir zeigen, dass die Menge $\mathcal{P}(2) \setminus \mathcal{D}$ der nichtperiodischen binären herkömmlichen Instanzen entscheidbar ist, in dem wir aus einer Instanz $I \in \mathcal{P}(2) \setminus \mathcal{D}$ eine binäre markierte Instanz $\dot{I} \in \mathcal{M} \cap \mathcal{G}(2) \setminus \mathcal{D}$ mit der selben Gleichheitsmenge wie I konstruieren. Zunächst stellen wir fest, dass gemeinsame Präfixe von h und von g nach rechts rotiert werden dürfen. Für einen Morphismus $m : A^* \to B^*$ bezeichnen wir mit $\mathrm{P}(m) = \bigcap_{a \in A} \mathrm{pref}(m(a))$ die Menge aller gemeinsamen Präfixe von A-Bildern von m. Jedes Bild von m über A hat ein Präfix aus $\mathrm{P}(m)$.

Lemma. (Erstes Präfixlemma) *Sei $I \in \mathcal{P}$ eine herkömmliche Instanz mit dem gemeinsamen Präfix $p \in \mathrm{P}(h) \cap \mathrm{P}(g)$ aller A-Bilder von h und von g und $\dot{I}$ die herkömmliche Instanz mit $\dot{h}(u) = p^{-1}h(u)p$ und $\dot{g}(u) = p^{-1}g(u)p$. Dann gilt $\mathrm{Eq}(I) = \mathrm{Eq}(\dot{I})$.* *b*

Beweis: Zunächst sind $p^{-1}h(u)p$ und $p^{-1}g(u)p$ wohl definiert, da p Präfix jedes A-Bildes von h und von g ist. Nun folgt aus $h(u) = g(u) \iff h(u)p = g(u)p \iff p\dot{h}(u) = p\dot{g}(u) \iff \dot{h}(u) = \dot{g}(u)$ die Behauptung. □ *c*

Es gilt entweder $\mathrm{P}(h) \cap \mathrm{P}(g) = \{\varepsilon\}$ oder es gibt ein Symbol, das Präfix sämtlicher Bilder von h und von g ist. Dieses Symbol darf bei herkömmlichen Instanzen laut erstem Präfixlemma nach rechts rotiert werden, ohne die Gleichheitsmenge zu verändern. Für nichtperiodische Instanzen kann diese Operation nur endlich oft ausgeführt werden. Anschließend hat mindestens einer der Morphismen h und g zwei verschiedene Bilder, die mit verschiedenen Symbolen beginnen. Für binäre Instanzen heißt das aber nichts anderes, als dass dann einer der Morphismen markiert ist.

Folgerung. *$\mathcal{P}(2) \setminus \mathcal{D}$ ist entscheidbar $\iff$ $\mathcal{M}^1 \cap \mathcal{P}(2) \setminus \mathcal{D}$ ist entscheidbar.* *d*

Im ersten Präfixlemma haben wir gemeinsame Präfixe beider Morphismen rotiert. Betrachten wir nur noch gemeinsame Präfixe eines einzelnen Morphismus, dann können wir immer noch rotieren, wenn wir gleichzeitig zu einer allgemeinen Instanz übergehen.

a **Lemma.** (Zweites Präfixlemma) *Sei $I \in \mathcal{P}$ eine herkömmliche Instanz mit dem gemeinsamen Präfix $p \in \mathrm{P}(h)$ aller A-Bilder von h und $\dot{I}$ die Instanz mit $\dot{h}(u) = p^{-1}h(u)p$, $\dot{g} = g$, $\dot{s} = (p, \varepsilon)$ und $\dot{f} = (\varepsilon, p)$. Dann gilt $\mathrm{Eq}(I) = \mathrm{Eq}(\dot{I})$.*

b *Beweis:* Erneut ist $p^{-1}h(u)p$ wohl definiert, da p Präfix sämtlicher A-Bilder von h ist. Nun folgt aus $h(u) = g(u) \iff h(u)p = g(u)p \iff p\dot{h}(u) = g(u)p$ die Behauptung. □

Im Beweis haben wir lediglich benutzt, dass $p^{-1}h(u)p$ wohl definiert ist. Für nichtperiodische Instanzen gibt es ein maximales z_h, für das $z_h^{-1}h(u)z_h$ wohl definiert ist. Wenden wir auf eine herkömmliche Instanz zunächst das erste und anschließend das zweite Präfixlemma auf z_h an, dann erhalten wir eine allgemeine Instanz, deren beide Morphismen jeweils verschiedene Bilder haben, die mit verschiedenen Symbolen beginnen.

c **Folgerung.** *$\mathcal{P}(2) \setminus \mathcal{D}$ ist entscheidbar $\iff$ $\mathcal{M} \cap \mathcal{G}(2) \setminus \mathcal{D}$ ist entscheidbar.*

Zusammen mit der Entscheidbarkeit periodischer Instanzen aus Folgerung 26d, der Tatsache, dass entscheidbar ist, ob eine Instanz periodisch ist und der Entscheidbarkeit markierter Instanzen aus Satz 79c können wir festhalten:

d **Folgerung.** *$\mathcal{P}(2)$ ist entscheidbar.*

Dieses Resultat wurde 1982 erstmals gezeigt. (Ehrenfeucht, Karhumäki, Rozenberg [5]) Der erste Beweis ist etwa 20 Seiten lang und führt einige aufwändige Fallunterscheidungen durch. Später hat Vesa Halava einfachere Beweise gegeben, die vor allem deshalb einfacher sind, weil er die Theorie der Nachfolger für markierte Instanzen entwickelt und benutzt. (Halava et al. [8, 11, 12]) Wir haben nun nicht nur in 63b eine neue, einfachere und allgemeinere Darstellung der Theorie der Nachfolger und in 79b einen einfacheren Beweis der Entscheidbarkeit markierter Instanzen, auch der hier präsentierte Schritt von binären herkömmlichen zu binären markierten allgemeinen Instanzen ist sowohl allgemeiner formuliert als auch leichter verständlich.

e **Beispiel:** Die Instanz $I_{92e}(n,m) = \| \begin{smallmatrix} 0 \\ 0^n0^m1 \end{smallmatrix} \begin{smallmatrix} 0^n10^m \\ 0 \end{smallmatrix} \|$ hat nach erstem Präfixlemma die selbe Gleichheitsmenge wie die Instanz $\dot{I}_{92e}(n,m) = \| \begin{smallmatrix} 0 \\ 0^m10^n \end{smallmatrix} \begin{smallmatrix} 10^m0^n \\ 0 \end{smallmatrix} \|$, die nach zweitem Präfixlemma die selbe Gleichheitsmenge $(a^mb^m)^*$ wie die markierte Instanz $\ddot{I}_{92e}(n,m) = | \begin{smallmatrix} \varepsilon \\ 0^m \end{smallmatrix} | \begin{smallmatrix} 0 \\ 10^n0^m \end{smallmatrix} \begin{smallmatrix} 10^m0^n \\ 0 \end{smallmatrix} | \begin{smallmatrix} 0^m \\ \varepsilon \end{smallmatrix} |$ hat. ⌟

f **Beispiel:** Der Status der Instanz $I_{92f} = \| \begin{smallmatrix} 00 \\ 0000 \end{smallmatrix} \begin{smallmatrix} 000 \\ 0101 \end{smallmatrix} \begin{smallmatrix} 0101 \\ 10 \end{smallmatrix} \|$ ist nicht leicht zu ermitteln. Nach zweitem Präfixlemma ist sie aber äquivalent zur Instanz $\dot{I}_{92f} = | \begin{smallmatrix} 0 \\ \varepsilon \end{smallmatrix} | \begin{smallmatrix} 00 \\ 000 \end{smallmatrix} \begin{smallmatrix} 000 \\ 0101 \end{smallmatrix} \begin{smallmatrix} 1010 \\ 10 \end{smallmatrix} | \begin{smallmatrix} \varepsilon \\ 0 \end{smallmatrix} |$ die viel leichter ist: Für eine Lösung müsste ein Pfad von $\downarrow 0$ nach $\uparrow 000$ gefunden werden, aber im Gegensatz zu I_{92f} ist in $\dot{I}_{92f}$ kein Übergang mit Änderung der Position von $\downarrow$ nach $\uparrow$ möglich. ⌟

Allgemeine *a*

Das erste und das zweite Präfixlemma setzen beide herkömmliche Instanzen voraus. Wir würden gern die selbe Konstruktion auch für allgemeine Instanzen durchführen. Das heißt, wir würden zur Instanz $\dot{I}$ mit $\dot{s} = s \oplus_\mathrm{p} (p, \varepsilon)$ und $\dot{f} = (p^{-1}, \varepsilon) \oplus_\mathrm{s} f$ übergehen. Dabei löschen die Operation $\oplus_\mathrm{p}$ und $\oplus_\mathrm{s}$ vorhandene gemeinsame Präfixe oder vorhandene gemeinsame Suffixe, das heißt, $x \oplus_\mathrm{p} y \in \mathrm{id}(B^*)^{-1}xy$ und $x \oplus_\mathrm{s} y \in xy\,\mathrm{id}(B^*)^{-1}$. Beachte, dass $\oplus_\mathrm{p}$ und $\oplus_\mathrm{s}$ nur partielle Operationen sind. Zwar können wir uns überlegen, dass I nur dann eine Lösung haben kann, wenn $s_\uparrow p$ und $s_\downarrow$ vergleichbar sind, aber $p^{-1}f_\uparrow$ könnte nicht definiert sein. Die Lösung liegt im Einfügen eines Bildes $h(a)$ für ein $a \in A$. Das Wort $p^{-1}h(a)f_\uparrow$ ist definiert, wenn p gemeinsames Präfix jedes A-Bildes von h ist.

Lemma. (Allgemeines Präfixlemma) *Sei I eine Instanz mit dem gemeinsamen Präfix $p \in \mathrm{P}(h)$ aller A-Bilder von h und $\dot{I}(a)$ für ein $a \in A$ die Instanz mit $\dot{h}(u) = p^{-1}h(u)p$, $\dot{g} = g$, $\dot{s} = s \oplus_\mathrm{p} (p, \varepsilon)$ und $\dot{f} = (p^{-1}h(a), g(a)) \oplus_\mathrm{s} f$. Dann gilt $\mathrm{Eq}(I) \cap A^*a = \mathrm{Eq}(\dot{I}(a))a$.* *b*

Beweis: Da p gemeinsames Präfix sämtlicher A-Bilder von h ist, sind sowohl $\dot{h}$ als auch $p^{-1}h(a)$ wohl definiert. Wenn $\dot{s}$ nicht definiert ist, dann sind $s_\uparrow p$ und $s_\downarrow$ nicht präfixvergleichbar und sowohl $\mathrm{Eq}(I)$ als auch $\mathrm{Eq}(\dot{I}(a))$ sind beide leer. Wenn $\dot{f}$ nicht definiert ist, dann sind $p^{-1}h(a)f_\uparrow$ und $g(a)f_\downarrow$ nicht suffixvergleichbar und auch nicht $h(a)f_\uparrow$ und $g(a)f_\downarrow$. Also sind dann sowohl $\mathrm{Eq}(\dot{I}(a))$ als auch $\mathrm{Eq}(I) \cap A^*a$ beide leer. Wenn sowohl $\dot{s}$ als auch $\dot{f}$ definiert sind, dann gilt $ua \in \mathrm{Eq}(I) \iff s_\uparrow h(u)h(a)f_\uparrow = s_\downarrow g(u)g(a)f_\downarrow \iff s_\uparrow p\dot{h}(u)p^{-1}h(a)f_\uparrow = s_\downarrow g(u)g(a)f_\downarrow \iff u \in \mathrm{Eq}(\dot{I}(a))$. □ *c*

Das allgemeine Präfixlemma erlaubt ebenfalls die Rotation gemeinsamer Präfixe, es entstehen nun aber bis zu $|I|$ neue Instanzen. Außerdem haben die neuen Instanzen nicht die selbe Gleichheitsmenge wie die ursprüngliche, sondern es gilt nur noch $\mathrm{Eq}(\dot{I}(a))a = \mathrm{Eq}(I) \cap A^*a$, das heißt, I hat genau Lösungen, die auf a enden, wenn $\dot{I}(a)$ überhaupt Lösungen hat. Wenden wir das allgemeine Präfixlemma k Mal an, dann können wir nur noch sagen, dass I genau dann Lösungen mit einer Länge von mindestens k hat, wenn die k-fach rotierte Instanz überhaupt Lösungen hat. Mit anderen Worten können bei dieser Konstruktion nur kurze Lösungen bekannter Maximallänge verloren gehen.

Folgerung. $\mathcal{G}(2)$ *ist entscheidbar.* *d*

Tatsächlich entscheiden wir periodische Instanzen nach Folgerung 26d und für nichtperiodische kann das allgemeine Präfixlemma nur endlich oft angewendet werden, bis wir zu einer markierten Instanz kommen, die nach Satz 79c entschieden wird. Bei jeder Anwendung des allgemeinen Präfixlemmas entstehen nur endlich viele neue Instanzen, also auch nur endlich viele zu untersuchende markierte Instanzen. Schließlich prüfen wir noch, ob eventuell kurze Lösungen übersehen wurden.

Diese Konstruktion findet sich auch bei Vesa Halava, ist aber dort komplizierter ausgeführt, da nicht die Rotation eines gemeinsamen Präfixes betrachtet wird, sondern es wird direkt zur schließlich resultierenden markierten Instanz übergegangen. Es muss dann statt über $(p^{-1}h(a), g(a)) \oplus_\mathrm{s} f$ für ein einzelnes Symbol $a \in A$ sofort über $(z_h^{-1}h(x), g(x)) \oplus_\mathrm{s} f$ für ein zunächst unbekanntes Wort $x \in A^*$ und ein maximales z_h gesprochen werden. (Halava [12]) Mit anderen Worten behandeln wir hier einen einzelnen Schritt wo Vesa ganze Pfade betrachtet. Die benötigten Eigenschaften lassen sich aber am einzelnen Schritt leichter zeigen.

Darüber hinaus gilt das allgemeine Präfixlemma nicht nur für binäre Instanzen, sondern wir können *jede* Instanz in eine Form bringen, in der beide Morphismen verschiedene Bilder haben, die mit verschiedenen Symbolen beginnen.

a **Beispiel:** Sei I_{94a} die Instanz $I_{94a} = |\begin{smallmatrix}0\\\varepsilon\end{smallmatrix}|\begin{smallmatrix}0&010\\00&01\end{smallmatrix}|\begin{smallmatrix}0\\\varepsilon\end{smallmatrix}|$. Wenden wir das allgemeine Präfixlemma auf das gemeinsame Präfix $0 \in \mathrm{P}(h)$ an, dann erhalten wir die Instanz $\dot{I}_{94a}(a) = |\begin{smallmatrix}00\\\varepsilon\end{smallmatrix}|\begin{smallmatrix}0&100\\00&01\end{smallmatrix}|\begin{smallmatrix}\varepsilon\\0\end{smallmatrix}|$, da $(0^{-1}h(a), g(a)) \oplus_\mathrm{s} f = (0^{-1}0, 00) \oplus_\mathrm{s} (0, \varepsilon) = (\varepsilon, 0)$ ist. Die Komponenten von $(0^{-1}h(b), g(b))f = (0^{-1}010, 01)(0, \varepsilon) = (100, 01)$ sind nicht suffixvergleichbar, also ist $\dot{I}_{94a}(b)$ nicht definiert.

Wenden wir auf $\dot{I}_{94a}(a)$ das allgemeine Präfixlemma für das gemeinsame Präfix $0 \in \mathrm{P}(\dot{g})$ an, erhalten wir die Instanz $\ddot{I}_{94a}(a)(a) = |\begin{smallmatrix}0\\\varepsilon\end{smallmatrix}|\begin{smallmatrix}0&100\\00&10\end{smallmatrix}|\begin{smallmatrix}\varepsilon\\0\end{smallmatrix}|$, da $(\dot{h}(a), 0^{-1}\dot{g}(a)) \oplus_\mathrm{s} \dot{f} = (0, 0^{-1}00) \oplus_\mathrm{s} (\varepsilon, 0) = (\varepsilon, 0)$ ist. Erneut ist die Rotation mit b nicht möglich.

Die Gleichheitsmenge von $\ddot{I}_{94a}(a)(a)$ ist die Menge $\mathrm{Eq}(\ddot{I}_{94a}(a)(a)) = (ab)^*$. Wenn $u = (ab)^i$ ist, dann gelten $\dot{s}_\uparrow \dot{h}(ua)\dot{f}_\uparrow = 00(0100)^i0 = (0001)^i000 = \dot{s}_\downarrow \dot{g}(ua)\dot{f}_\downarrow$ und $s_\uparrow h(uaa)f_\uparrow = 0(0010)^i000 = (0001)^i0000 = s_\downarrow g(uaa)f_\downarrow$. ⌟

b **Beispiel:** Wir können das allgemeine Präfixlemma auch simultan für gemeinsame Präfixe $p_h \in \mathrm{P}(h)$ und $p_g \in \mathrm{P}(g)$ formulieren. Wir gehen dann zu $\dot{s} = s \oplus_\mathrm{p} (p_h, p_g)$ und $\dot{f} = (p_h^{-1}h(a), p_g^{-1}g(a)) \oplus_\mathrm{s} f$ über. Führen wir die beiden Schritte aus Beispiel 94a simultan aus, dann erhalten wir die Instanz $\dot{J}_{94a}(a) = |\begin{smallmatrix}0\\\varepsilon\end{smallmatrix}|\begin{smallmatrix}0&100\\00&10\end{smallmatrix}|\begin{smallmatrix}\varepsilon\\\varepsilon\end{smallmatrix}|$, da $(0^{-1}h(a), 0^{-1}g(a)) \oplus_\mathrm{s} f = (0^{-1}0, 0^{-1}00) \oplus_\mathrm{s} (0, \varepsilon) = (\varepsilon, \varepsilon)$ ist. Die Gleichheitsmenge ist nun $\mathrm{Eq}(\dot{J}_{94a}(a)) = a(ba)^*$ und für $u = a(ba)^i$ gilt $s_\uparrow h(ua)f_\uparrow = 00(0100)^i00 = s_\downarrow g(ua)f_\downarrow$. ⌟

Beispiel: Das simultane allgemeine Präfixlemma kann auch auf herkömmliche Instanzen angewendet werden. Für die Instanz $I_{95a} = \| \begin{smallmatrix} 0 \\ 001 \end{smallmatrix} \begin{smallmatrix} 01 \\ 011 \end{smallmatrix} \|$, erhalten wir mit $\dot{I}_{95a}(a) = | \begin{smallmatrix} \varepsilon \\ \varepsilon \end{smallmatrix} | \begin{smallmatrix} 0 \\ 010 \end{smallmatrix} \begin{smallmatrix} 10 \\ 110 \end{smallmatrix} | \begin{smallmatrix} \varepsilon \\ 01 \end{smallmatrix} |$ und $\dot{I}_{95a}(b) = | \begin{smallmatrix} \varepsilon \\ \varepsilon \end{smallmatrix} | \begin{smallmatrix} 0 \\ 010 \end{smallmatrix} \begin{smallmatrix} 10 \\ 110 \end{smallmatrix} | \begin{smallmatrix} \varepsilon \\ 1 \end{smallmatrix} |$ zwei unlösbare Instanzen: Es ist für $c \in \{a, b\}$ jeweils $\mathbb{M}(\dot{I}_{95a}(c)) = \mathbb{F}(\dot{I}_{95a}(c)) = \emptyset$. ⌟ *a*

Gleichheitsmengen *b*

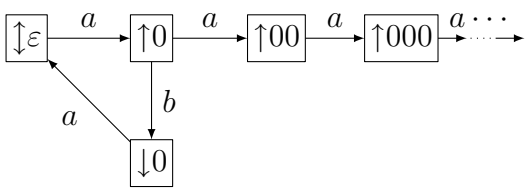
Die Struktur der Gleichheitsmengen binärer herkömmlicher Instanzen wurde 1983 im Zuge von Untersuchungen über Testmengen partiell geklärt. (Ehrenfeucht, Karhumäki, Rozenberg [4], Harju, Karhumäki [13]) Für nichtperiodische Instanzen gilt demnach $\mathrm{Eq}(I) = \{\alpha, \beta\}^*$ oder $\mathrm{Eq}(I) = (\alpha\gamma^*\beta)^*$ für Wörter $\{\alpha, \beta, \gamma\} \subset A^*$, wobei stets vermutet wurde, dass der zweite Fall nicht eintreten kann. Štěpán Holub hat 2003 einen Beweis dieser Vermutung veröffentlicht. (Holub [14]) Allerdings enthält diese Arbeit Fehler, die in einer neuen Version behoben worden sein sollen.

Die partielle Charakterisierung von Ehrenfeucht, Karhumäki und Rozenberg beruht wesentlich auf dem Konzept des kritischen Überhangs, das auch von Holub verwendet wird. Es sagt aus, dass es im Suchgraphen $\rightarrowtail_I^\star((\varepsilon, \varepsilon))$ einer nichtperiodischen Instanz I nur einen einzigen Überhang geben kann, der auf dem Weg zu zwei verschiedenen Lösungen liegt. Wir wollen diese Erkenntnis mit unseren Begriffen formulieren. Dazu sei I eine nichtperiodische binäre herkömmliche Instanz. Nach erstem Präfixlemma können wir ohne Beschränkung der Allgemeinheit annehmen, dass h markiert ist und nach zweitem Präfixlemma hat I die selbe Gleichheitsmenge wie die markierte Instanz $\dot{I}$ mit $\dot{g}(u) = z_g^{-1} g(u) z_g$, die das Startpaar (ε, z_g) und das Endpaar (z_g, ε) hat. Hier ist z_g das längste Wort, für das $\dot{g}$ wohl definiert ist. Simples Einsetzen der Definition der Suchrelation ergibt nun $(\varepsilon, \varepsilon) \overset{a}{\rightarrowtail}_I (z_g, \varepsilon)x \iff h(a_\uparrow)x_\downarrow = g(a_\downarrow) z_g x_\uparrow \iff h(a_\uparrow)x_\downarrow = z_g \dot{g}(a_\downarrow) x_\uparrow \iff (\varepsilon, z_g) \overset{a}{\rightarrowtail}_{\dot{I}} x$. Das heißt, die Suchgraphen von I und $\dot{I}$ sind isomorph. Da $\dot{I}$ markiert ist und also höchstens am Überhang $(\varepsilon, \varepsilon)$ eine Wahl der Fortsetzung möglich ist, folgt unmittelbar, dass in I höchstens am Überhang (z_g, ε) eine Wahl der Fortsetzung möglich ist.

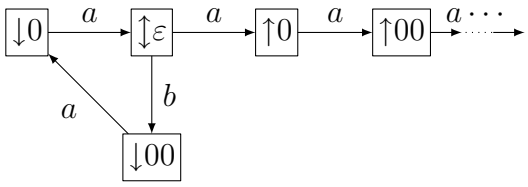
Nun ergibt sich die partielle Charakterisierung, da es im Suchgraphen von I entweder zwei Schleifen α und β um $(\varepsilon, \varepsilon)$ gibt, oder eine Schleife γ um (z_g, ε) mit dem Weg α von $(\varepsilon, \varepsilon)$ nach (z_g, ε) und dem Weg β zurück von (z_g, ε) nach $(\varepsilon, \varepsilon)$. Die volle Charakterisierung, die den zweiten Fall ausschließt, ist nicht so leicht zu haben, denn wie Beispiel 96b zeigt, kann der zweite Fall für binäre markierte allgemeine Instanzen eintreten. Solche Instanzen können nicht durch Anwendung der Präfixlemmas aus herkömmlichen Instanzen erzeugt werden. Im Prinzip zeigt Holub genau das, wir wollen die dazu notwendigen vielen technischen Details hier aber nicht wiedergeben. Der

Autor vermutet, dass es einen einfachen Beweis der Unmöglichkeit des zweiten Falls gibt, wahrscheinlich sogar eine einfache Argumentation, die sowohl die Entscheidbarkeit binärer Instanzen zeigt als auch die Struktur binärer Gleichheitsmengen klärt.

a **Beispiel:** Die Instanz $I_{96a} = \| \begin{smallmatrix} 00 & & 1 \\ 0 & & 010 \end{smallmatrix} \|$ hat den Suchgraphen

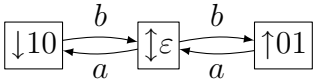


in dem nur am Knoten $\uparrow 0 = (z_g, \varepsilon)$ eine Wahl der Fortsetzung möglich ist. Die Instanz $\dot{I}_{96a} = | \begin{smallmatrix} \varepsilon \\ 0 \end{smallmatrix} | \begin{smallmatrix} 00 & & 1 \\ 0 & & 100 \end{smallmatrix} | \begin{smallmatrix} 0 \\ \varepsilon \end{smallmatrix} |$ hat den Suchgraphen



in dem höchstens am Knoten $\updownarrow\varepsilon$ eine Wahl der Fortsetzung möglich sein kann und der isomorph zum Suchgraphen von I_{96a} ist. ⌟

b **Beispiel:** Die Instanz $I_{96b} = \| \begin{smallmatrix} 0 & & 101 \\ 010 & & 1 \end{smallmatrix} \|$ hat die Gleichheitsmenge $\mathrm{Eq}(I_{96b}) = \{ab, ba\}^*$ und den Suchgraphen



der auch Suchgraph der Instanz $I_{82b} = | \begin{smallmatrix} \varepsilon \\ 10 \end{smallmatrix} | \begin{smallmatrix} 0 & & 101 \\ 010 & & 1 \end{smallmatrix} | \begin{smallmatrix} 10 \\ \varepsilon \end{smallmatrix} |$ ist. Die Instanz I_{82b} hat die Gleichheitsmenge $\mathrm{Eq}(I_{82b}) = (b(ba)^*a)^*$, der Fall einer unendlich erzeugten Gleichheitsmenge ist also sehr wohl möglich. Allerdings kann die Instanz I_{82b} nicht durch Anwendung der Präfixlemmas aus einer herkömmlichen Instanz erzeugt werden. ⌟

Praxis

Es gibt unzählige Einlassungen zum Zusammenhang zwischen Theorie und Praxis. Viele der in dieser Arbeit präsentierten Beispiele sind durch Programme aus teilweise sehr großen Kandidatenmengen ausgewählt worden. Sie erfüllen oft eine ganze Palette von Kriterien: Sie verdeutlichen eine bestimmte Eigenschaft sehr gut, sind klein, die notwendigen Rechnungen sind nicht ausufernd, bestimmte Eigenschaften fehlen, ... Die Programme wurden implementiert und natürlich getestet. Dazu wurden unter Verwendung von Bleistift und Papier Instanzen konstruiert, die Randfälle darstellen. Die experimentellen Daten haben dann zu neuen Vermutungen geführt oder zumindest geholfen, bekannte Techniken genauer zu verstehen und schärfer zu formulieren. Und so weiter und so fort.

Richard Lorentz hat im Jahr 2000 auf der Konferenz „Computer and Games“ ein Programm vorgestellt, das herkömmliche Instanzen erzeugen und lösen kann. (Lorentz [22]) Dieses Programm kann Instanzen mit leerer Start- oder mit leerer Endmenge wie in 19a und mit falscher Längen- oder mit falscher Elementbalance ähnlich wie in 23b erkennen. Es sucht anschließend im Suchgraphen aus 28a nach Lösungen. Als Suchstrategie kommt dabei iterative Tiefensuche zum Einsatz, also wird bis zu einer gewissen Tiefe m gesucht, die bei Misserfolg schrittweise bis zu einer Maximaltiefe M erhöht wird. Dabei werden pro Suche bereits besuchte Knoten memoriert und mehrfaches Traversieren von Teilgraphen wird verhindert.

Das Programm von Richard Lorentz ist zwar im Vergleich mit den vom Autor entwickelten Boliden nicht besonders effizient, doch es zeigt bereits die Grobstruktur, die alle späteren Entwicklungen beibehalten haben: Erkenne unlösbare Instanzen möglichst früh und suche schrittweise nach immer längeren Lösungen. Bereits mit diesem einfachen Programm kam Richard Lorentz zu der richtigen Vermutung, dass die Instanz I_{16b} die Instanz mit der längsten kürzesten Lösung (der Länge 75) in $\mathcal{P}(3,3) \cap \mathcal{E}^0$ ist.

Ling Zhao, ein Schüler von Richard Lorentz, hat dessen schwache Vermutungen über die schwierigsten Instanzen der Klassen $\mathcal{P}(3,4)$ und $\mathcal{P}(4,3)$ von 119 auf 252 und von 204 auf 302 verschärft. (Zhao [44]) Er liegt damit

zwar deutlich unterhalb der vom Autor gefundenen 452 und 595, aber in der Rangliste der schwierigsten bekannten Instanzen liegt die von Ling Zhao gefundene Instanz aus $\mathcal{P}(3,4)$ mit der kürzesten Lösung der Länge 252 auf dem zweitem Platz. Das Programm von Ling Zhao ist an mehreren Stellen gegenüber dem von Richard Lorentz verbessert: Es erlaubt das schnelle systematische Durchsuchen ganzer Klassen, wobei nur jeweils ein Vertreter einer Isomorphieklasse betrachtet wird, es sucht in der gespiegelten Instanz, falls der Suchgraph, wie in Beispiel 31a, dort kleiner ist und es verwendet eine einfache Variante des Maskierens aus 42b um unlösbare Instanzen zu erkennen. Ein wesentlicher Beitrag zur Beschleunigung der Suche ist das in 33a beschriebene Abschneiden von Pfaden, die aus Abzählgründen innerhalb der Suchschranke nicht mehr zu einer Lösung führen können.

Dank dieser Verbesserungen ist Ling Zhao in der Lage, auch in den Klassen $\mathcal{P}(3,5)$ und $\mathcal{P}(4,4)$ nach schwierigen Instanzen zu suchen. Er geht dabei zwar nicht mehr systematisch vor, sondern wählt zufällig Instanzen aus, zumindest in der Klasse $\mathcal{P}(3,5)$ findet er aber eine Instanz mit einer kürzesten Lösung der Länge 240, die in der Rangliste in dieser Klasse heute immer noch auf Platz 5 liegt. Die schwierigste bekannte Instanz in dieser Klasse hat eine kürzeste Lösung der Länge 528. In der Klasse $\mathcal{P}(4,4)$ ist die von Ling Zhao gefundene Instanz mit einer kürzesten Lösung der Länge 256 aus heutiger Sicht nicht wirklich schwierig. Es sind Dutzende schwierigere Instanzen bekannt, sogar drei verschiedene mit einer kürzesten Lösung der Länge 880. Der Autor ist heute mit Hilfe von in allen Bereichen nochmals stark verbesserten Programmen in der Lage, die Klassen $\mathcal{P}(3,5)$ und $\mathcal{P}(4,4)$ systematisch zu bearbeiten.

Ebenfalls randomisiert geht Heiko Stamer vor, der eine Sammlung von Rekordinstanzen unterhält. (Stamer [40]) Er hat gegenüber dem Programm von Ling Zhao zwar keine wesentlichen Verbesserungen implementiert, hat aber lange und intensiv gesucht und so auch einige schwierige Instanzen gefunden. Zusammen mit Johannes Waldmann und Mario Schmidt hat Heiko Stamer die Wichtigkeit des Ermittelns empirischer Daten betont: Sie können helfen, Vermutungen über die maximale Länge kürzester Lösungen aufzustellen. Sie vermuten, dass in $\mathcal{P}(3)$ die Länge der kürzesten Lösung in $O(\mathrm{wd}^2)$ liegt. (Waldmann, Schmidt, Stamer [42]) Mario Schmidt zeigt 2006, dass es Familien in $\mathcal{P}(3)$ gibt, deren kürzeste Lösung kubisch bezüglich der Weite wächst. (Schmidt [39]) Stärkeres Wachstum in $\mathcal{P}(3)$ ist nicht bekannt.

Übrigens hat bereits Richard Lorentz vermutet, dass in $\mathcal{P}(2)$ nur lineares Wachstum möglich ist. Es spricht einiges für diese Vermutung, die wir in 90a sehr scharf formulieren.

Kürzlich haben Štěpán Holub und Vesa Halava gezeigt, dass $\mathcal{G}(2)$ in polynomieller Zeit entscheidbar ist, doch sie können selbst exponentielles

Wachstum nicht ausschließen. (Holub, Halava [15])

Wir stellen in diesem Kapitel dar, wie effiziente Programme entwickelt werden können, die sowohl viele unlösbare Instanzen automatisch erkennen, als auch in der Lage sind, sehr große Suchgraphen zu bearbeiten. Diese beiden Aufgaben sind von recht verschiedener Natur: Beim Erkennen unlösbarer Instanzen handelt es sich um nichts anderes als das automatische Erzeugen von Beweisen. Diese Beweise können sehr unterschiedlich sein, sie können zum Beispiel aus dem Nullraum eines Gleichungssystems bestehen, sie können aber auch aus einem oder mehreren endlichen Automaten bestehen, die Mengen von Überhängen beschreiben. Entsprechende Programme müssen also mit unterschiedlichen Objekten umgehen, die darüber hinaus auch interagieren, wenn zum Beispiel in endlichen Automaten kodierte Information über verbotene Überhänge zum Maskieren verwendet werden soll. In Abschnitt 101b werfen wir einen Blick auf entsprechende Werkzeuge und deren effiziente Nutzung.

Beim Bearbeiten sehr großer Suchgraphen stellt sich eigentlich nur eine einzige Frage: Wie kann der Durchsatz maximiert werden? Diese Frage eröffnet einen sehr großen Raum für mögliche Implementationen, wir stellen in 110c einige Teilantworten vor. Neben der sequentiellen spielt in 116b auch die parallele Bearbeitung sehr großer Suchgraphen eine Rolle. Wir beschreiben Schemata zur Verteilung der Last, die sich bei der Bearbeitung von Suchgraphen mit bis zu 10^{15} Knoten als äußerst effizient heraus gestellt haben.

Doch wir beginnen mit einer weiteren, eigenständigen Aufgabe:

Erzeugen, Nummerieren, Normieren *a*

Die Frage, wie viele Instanzen in der Klasse $\mathcal{P}(g, w)$ enthalten sind, ist leicht zu beantworten: Es sind genau $|B^{\leq w}|^{2g}$, wobei $|B^{\leq w}|$ die Anzahl der Wörter über dem Alphabet B ist, die höchstens die Länge w haben. Es gilt $|\mathcal{P}(g, w)| = (\sum_{i=0}^{w} |B|^i)^{2g}$, zum Beispiel enthält $\mathcal{P}(3, 4)$ für $|B| = 2$ bereits 887 503 681 Instanzen.

Die Frage, wie diese Instanzen aufgezählt werden können, ist schon etwas komplizierter: Für kleine Klassen genügt ein triviales Programm, das $\mathcal{P}(g, w)$ irgendwie aufzählt. Bei größeren Klassen könnte eine rekursive Implementation bereits Probleme mit zu großen Stapeln bekommen. Darüber hinaus sollen die 852 891 037 441 Instanzen in $\mathcal{P}(4, 4)$ bei $|B| = 2$ vielleicht nicht am Stück bearbeitet werden, sondern zum Beispiel verteilt auf mehreren Maschinen. Es bietet sich also an, eine Bijektion $\mathrm{c}(g, w)$ zwischen $\mathcal{P}(g, w)$ und $\{0, \ldots, |\mathcal{P}(g, w)| - 1\}$ zu implementieren. Dazu gehen wir von einer Ko-

dierung der Wörter in $B^{\leq w}$ aus und implementieren zunächst generisch eine Kodierung von Paaren aus $B^{\leq w} \times B^{\leq w}$ und dann von Listen solcher Paare.

Haben wir $\mathrm{c}(g, w)$ vorliegen, dann können wir beliebige Teilmengen von $\mathcal{P}(g, w)$ kostengünstig aufzählen. Insbesondere haben wir das gleichverteilte Erzeugen zufälliger Instanzen auf das gleichverteilte Erzeugen von ganzen Zahlen aus $\{0, \ldots, |\mathcal{P}(g, w)| - 1\}$ zurückgeführt.

Außerdem verbietet sich das externe Erzeugen von Instanzenmengen, also das Erzeugen und Zwischenspeichern zur weiteren Verarbeitung. Einerseits sind die Datenmengen viel zu groß und andererseits würde das ein zusätzliches Einlesen erfordern. Statt dessen sollen Instanzen nach Bedarf erzeugt werden. Die klassische Lösung für dieses Problem ist die Coroutine, die leider in den meisten Programmiersprachen nicht vorgesehen ist.

a **Implementation:** Bereits an diesem Punkt stellt sich heraus, dass die Sprache Haskell sehr gut geeignet ist, die abstrakten Aufgaben zu implementieren. Es handelt sich um eine stark getypte funktionale Sprache mit fauler Auswertung und ohne Seiteneffekte. (Haskell Report [20]) Referentielle Transparenz erleichtert das Schreiben von korrekten Programmen und die faule Auswertung ermöglicht Komposition von Programmteilen. Insbesondere bekommen wir Coroutinen frei Haus, sogar ohne explizite Erwähnung. Haskell wird aktiv von der Nutzerinnengemeinschaft weiter entwickelt, die sich sowohl im akademischen als auch im industriellen Umfeld bewegt. Dank optimierender Übersetzer hoher Qualität werden in Haskell geschriebene Programme schnell und zuverlässig auf einer Reihe von Plattformen ausgeführt. ⌟

Als letzte vorbereitende Maßnahme wollen wir Instanzen so normieren, dass alle Mitglieder einer aus (14c) abgeleiteten Isomorphieklasse gleich sind. Dazu erzeugen wir alle Instanzen, die durch Hintereinanderausführung der Operationen Spiegeln, Transponieren, Permutieren des Quellalphabets A und Permutieren des Zielalphabets B entstehen und definieren die lexikographisch kleinste dieser Instanzen als Normalform. Eine Isomorphieklasse hat bis zu $4g!|B|!$ Mitglieder, die Menge $\mathcal{P}(4, 4)$ enthält bei $|B| = 2$ also mindestens 4 442 140 821 paarweise nichtisomorphe Instanzen.

b **Zahlen:** Einige Anzahlen von Elementen in $\mathcal{P}(g, w)$ bei $|B| = 2$:

g \ w	1	2	3	4	5	6
1	9	49	225	961	3 969	16 129
2	81	2 401	50 625	923 521	15 752 961	
3	729	117 649	11 390 625	887 503 681		
4	6 561	5 764 801	2 562 890 625			
5	59 049	282 475 249				
6	531 441					

Die zugehörigen Anzahlen von Isomorphieklassen in $\mathcal{P}(g, w)$ bei $|B| = 2$:

g \ w	1	2	3	4	5	6
1	4	14	47	163	589	2 213
2	17	231	3 748	60 376	999 842	
3	52	3 168	253 976	18 673 528		
4	147	37 653	13 991 481			
5	360	380 958				
6	819					

Für wachsendes w nähert sich die Anzahl der Isomorphieklassen recht schnell der minimal möglichen Anzahl. Das heißt, die Isomorphieklassen schöpfen im Schnitt ihre maximal mögliche Größe besser aus, was nichts anderes bedeutet, als das für eine größere Instanz die Chance geringer ist, dass sie invariant gegenüber den Transformationen innerhalb der Isomorphieklasse ist.

Diese Zahlen sind *nicht* direkt mit denen von Ling Zhao vergleichbar. Er lässt erstens keine löschenden Morphismen zu und zweitens gibt er die Zahlen für $\mathcal{P}(g, w+1) \setminus \mathcal{P}(g, w)$ an. (Zhao [44]) ⌟

Beispiel: Die Normalform der Instanz $\| \begin{smallmatrix} 11 \\ 110 \end{smallmatrix} \begin{smallmatrix} 0 \\ 1011 \end{smallmatrix} \begin{smallmatrix} 101 \\ 1 \end{smallmatrix} \begin{smallmatrix} 1010 \\ 10 \end{smallmatrix} \|$ ist die Instanz *a*
$I_{101a} = \| \begin{smallmatrix} 0 \\ 010 \end{smallmatrix} \begin{smallmatrix} 001 \\ 00 \end{smallmatrix} \begin{smallmatrix} 01 \\ 0101 \end{smallmatrix} \begin{smallmatrix} 0100 \\ 1 \end{smallmatrix} \|$, deren Isomorphieklasse 192 Instanzen enthält. ⌟

Filtern *b*

Unser Interesse besteht zunächst darin, solche Instanzen *schnell* zu erkennen, die offensichtlich keine echten Lösungen haben oder die offensichtlich entscheidbar sind. In den Abschnitten 18c und 27a werden einige passende Methoden vorgestellt. Wir unterteilen diese Methoden in *genaue* und *beschränkte*. Eine genaue Methode ist eine Methode, die ohne weitere Parameter auskommt. Eine solche Methode kann genau angewendet werden: Entweder eine Instanz wird als unlösbar erkannt, oder sie kann mit dieser Methode nicht als unlösbar erkannt werden. Beschränkte Methoden hängen hingegen von weiteren Parametern ab und eine nicht als unlösbar erkannte Instanz könnte bei anderer Wahl dieser Parameter eventuell doch als unlösbar erkannt werden. Solche Methoden arbeiten mit beschränkter Genauigkeit.

Wir vereinbaren, dass wir Instanzen nicht nur dann von der weiteren Bearbeitung ausschließen, wenn wir sie als unlösbar erkannt haben, sondern auch bereits dann, wenn wir zeigen können, dass gewisse Regeln in keiner echten Lösung verwendet werden können. Wir fragen also nicht mehr danach, ob die Gleichheitsmenge echte Lösungen enthält, sondern, ob die Gleichheitsmenge echte Lösungen enthält, in denen jede Regel mindestens ein Mal verwendet

wird. Dadurch geht im Prinzip nichts verloren, denn falls wir diese Frage negativ beantworten können, dann ist die Frage, ob es echte Lösungen gibt, zu der Frage äquivalent, ob eine Instanz mit weniger Regeln echte Lösungen besitzt, in denen jede Regel mindestens ein Mal verwendet wird. Diese Delegation der Fragestellung kann nur endlich oft vorgenommen werden, da in jedem Schritt die Anzahl der Regeln kleiner wird.

Die Reihenfolge, in der die einzelnen Methoden dargestellt werden, entspricht der Reihenfolge, in der sie vom Autor aus Effizienzgründen angewendet werden. Es gibt dabei einen gewissen Spielraum, der nicht zuletzt von der Qualität der einzelnen Implementationen abhängt. Die hier präsentierte Reihenfolge ist als Vorschlag zu verstehen, der sich in der Praxis bewährt hat. Die angegebenen Tabellen sind kumulativ in dieser Reihenfolge, dass heißt, die Grundgesamtheit der $(n+1)$-ten Tabelle sind die Instanzen, die in der n-ten als noch nicht entschieden markiert sind.

Die Zahlen für die Klassen $\mathcal{P}(4,3)$, $\mathcal{P}(4,4)$ und $\mathcal{P}(3,5)$ sind hier nicht angegeben. Der Autor hat diese Klassen mit früheren Versionen der Programme oder mit anderen Parametern systematisch bearbeitet. Die dabei ermittelten Zahlen sind nicht direkt mit den hier angegebenen Zahlen vergleichbar. Da die systematische Bearbeitung dieser Klassen etwa 2 Jahre in Anspruch genommen hat, verzichten wir auf eine neue Berechnung, zumal die Leistungsfähigkeit der einzelnen Methoden auch mit den hier gezeigten Daten eingeschätzt werden kann. Unter der Adresse `ftp://ftp.ira.uka.de/pub/pcp` finden sich auch Listen mit Instanzen aus diesen Klassen.

a

Genau

Zwar haben wir nicht viele genaue Methoden an der Hand, es handelt sich dabei aber um sehr leistungsfähige Methoden, die allesamt effizient implementiert werden können.

Zuerst könnten die in 19a definierten Startmengen $\mathrm{St}(I)$ und Endmengen $\mathrm{Fl}(I)$ einer Instanz I leer sein. Es ist leicht zu überprüfen, ob das der Fall ist und bei wenigen Regeln ist die Chance recht hoch, dass keine Regel eine Startregel oder eine Endregel ist. Weiterhin könnte der Nullraum des Systems (20b) entsprechend Folgerung 20c Regeln von der Benutzung ausschließen. Dazu implementieren wir den ersten Schritt des Simplexalgorithmus und wenden ihn auf die Matrix der Symboldifferenzen der Regeln an. Schließlich erlaubt Folgerung 23d zusätzlich die Identifikation von Instanzen, für die aus Balancegründen ein Maximum für die Länge der kürzesten Lösung festgestellt werden kann. Zum Erkennen unbalancierter Instanzen müssen lediglich die Vorzeichen der Längendifferenzen der Regeln bestimmt werden.

Zahlen: Diese vier Methoden angewendet auf kleine Klassen herkömmlicher Instanzen. *a*

Ein Häkchen bedeutet, dass der entsprechende Test passiert wurde. Eine Instanz mit einem Häkchen in der Spalte St hat also mindestens eine Startregel. Angegeben sind Anzahlen von Instanzen in Normalform mit dem entsprechenden Häkchenmuster. In der Klasse $\mathcal{P}(3,2)$ werden zum Beispiel nur 300 Isomorphieklassen nicht durch eine dieser Methoden als unlösbar erkannt oder zumindest reduziert. Nur die Instanzen in der ersten Zeile, bei denen alle vier Häkchen gesetzt sind, müssen weiter betrachtet werden.

				g	2				3			4	
St	Fl	23d	20c	w	1	2	3	4	1	2	3	1	2
✓	✓	✓	✓		1	12	99	730	5	300	19 868	19	6 851
✓	✓	✓	·		1	22	359	3 910	6	711	53 161	26	11 067
✓	✓	·	✓		4	18	81	474	8	106	2 390	15	657
✓	✓	·	·		9	120	1 364	12 531	31	1 642	89 804	84	16 495
✓	·	✓	✓		0	1	30	506	0	20	5 361	0	336
✓	·	✓	·		0	3	123	2 736	0	42	12 222	0	296
✓	·	·	✓		0	0	0	0	0	0	0	0	0
✓	·	·	·		0	18	486	8 408	0	162	25 228	0	1 059
·	✓	✓	✓		0	0	7	227	0	0	1 023	0	0
·	✓	✓	·		0	0	26	901	0	0	2 424	0	0
·	✓	·	✓		0	0	0	0	0	0	0	0	0
·	✓	·	·		0	0	120	3 092	0	0	5 410	0	0
·	·	✓	✓		0	1	22	586	0	5	2 443	0	52
·	·	✓	·		0	1	80	3 108	0	10	5 033	0	57
·	·	·	✓		1	11	151	2 302	1	54	4 523	2	264
·	·	·	·		1	24	800	20 865	1	116	25 086	1	519

Die vorhandenen Asymmetrien sind der Verwendung von Normalformen geschuldet. Es gibt zum Beispiel in der Klasse $\mathcal{P}(3,2)$ genau 20 Instanzen, die keine Endregel haben, aber alle anderen Tests bestehen. Aber es gibt keine Instanzen, die keine Startregel haben, aber alle anderen Tests bestehen. Würden wir die gespiegelte Instanz der lexikographisch kleinsten Instanz als Normalform definieren, dann wären diese beiden Zahlen vertauscht.

Wir sehen, dass alle vier Methoden ihre Berechtigung haben. Zwei Kombinationen sind nicht möglich, denn die Regeln einer unbalancierte Instanz, deren Nullraum Lösungen erlaubt, in denen alle Regeln verwendet werden, sind alle gleich lang. Dann ist aber die Startmenge gleich der Endmenge. ⌟

a ## Beschränkt

b ### Maskieren

Der Prototyp eines beschränkten Filters ist die Maskenmethode aus 42b. Sie hängt von den Begriffen *erzeugbar* und *überdeckbar* ab, die nur relativ zu einer Schranke für die Anzahl der Knoten in einem Suchgraphen definiert sind. Dabei empfiehlt es sich, in der Suchrelation bereits die Ausschlussmengen aus 34a zu berücksichtigen und genaues Abschneiden aus 32b vorzunehmen. Wie in 42b bereits erwähnt, muss die Maskenmethode bis zum Fixpunkt iteriert werden, der garantiert existiert.

c **Zahlen:** Maskenmethode angewendet auf die Instanzen, die nach dem genauen Filtern noch weiter zu betrachten sind. In Suchgraphen zur Entscheidung, ob ein Überhang erzeugbar oder überdeckbar ist, wurden maximal 100 Knoten betrachtet.

Ein Häkchen bedeutet, dass die entsprechende Position im Suchgraphen erlaubt ist. Zum Beispiel muss in der Klasse $\mathcal{P}(3,3)$ bei 528 Instanzen nach der Maskenmethode nur noch die ↑-Position beachtet werden. Alle Instanzen mit wenigstens einem Häkchen müssen weiter betrachtet werden.

		g	2				3				4	
↑	↓	w	1	2	3	4	1	2	3	4	1	2
✓	✓		1	7	28	85	5	168	3 935	83 339	19	4 056
✓	·		0	0	4	24	0	14	528	10 480	0	460
·	✓		0	3	29	173	0	29	1 555	36 196	0	638
·	·		0	2	38	448	0	89	13 850	871 860	0	1 697

Mit größerer Weite steigt der Anteil der Instanzen, die mit Hilfe der Maskenmethode als unlösbar erkannt werden. Das gibt die Tatsache wieder, dass längere Wörter eine geringere Chance haben, Präfix eines Überhangs zu sein. ⌟

d ### Kleines Suchen

Beim Filtern geht es nicht nur darum, unlösbare Instanzen zu erkennen, sondern wir wollen auch Instanzen erkennen, die kurze Lösungen haben.

Nach der Maskenmethode ist der richtige Zeitpunkt gekommen, eine kleine Lösungssuche vorzunehmen: Zum einen haben wir bereits Teile des Suchgraphen erzeugt und können diese eventuell erneut verwenden. Andererseits sind die noch verbleibenden Filter recht teuer und sollten daher auf

möglichst wenige Instanzen angewendet werden. Was eine *kleine* Suche ist, hängt von den vorhandenen Ressourcen und von der Anzahl der Instanzen ab. Wir gehen aber von einer Implementation aus, die nicht optimiert ist, dafür aber alle bisher gewonnenen Informationen berücksichtigt, insbesondere über Ausschlussmengen und abgeschnittene Pfade.

Zahlen: Kleines Suchen mit der Schranke 1000, dass heißt, die Suche wird eingestellt, wenn der Suchgraph bei Breitensuche mit Schleifendetektion mehr als 1000 Knoten enthält. *a*

Ein Häkchen bedeutet, dass in einer Instanz, in der gespiegelten Instanz oder in deren Kombination eine Lösung gefunden wurde, oder, dass einer der Suchgraphen endlich ist und keinen Lösungspfad enthält. Eine kombinierte Lösung uv^{R} besteht aus einem Pfad $s \overset{u}{\rightarrowtail}_I x$ und einem Pfad $f^{\mathrm{R}} \overset{v}{\rightarrowtail}_{I^{\mathrm{R}}} \overline{x}^{\mathrm{R}}$ für einen Überhang x. Es müssen nur die Instanzen ohne Häkchen weiter betrachtet werden.

				g	2		3				4	
I	I^{R}	↻	∅	w	4	5	1	2	3	4	1	2
·	·	·	·		0	0	0	1	85	2 194	0	38
·	·	·	✓		0	0	0	0	0	0	0	0
·	·	✓	·		0	0	0	0	7	93	0	2
·	✓	✓	·		0	0	0	0	2	35	0	3
✓	·	✓	·		0	0	0	0	2	34	0	4
✓	✓	✓	·		282	1 171	5	210	5 922	127 659	19	5 107

Die letzte Instanz in $\mathcal{P}(3,2)$ ist die Instanz $I_{105a} = \| \begin{smallmatrix} \varepsilon & 0 & 10 \\ 01 & 1 & 0 \end{smallmatrix} \|$, die mit einiger Berechtigung als die schwierigste in dieser Klasse bezeichnet werden kann. Ein Mensch erkennt ihre Unlösbarkeit sehr leicht: Jede Lösung muss mit $a^i b$ beginnen, der entsprechende ↓-Überhang enthält aber das Teilwort 11.

Diese Zahlen unterstützen einerseits die Vermutung, dass es einen einfachen Beweis der Entscheidbarkeit von $\mathcal{P}(2)$ gibt: Es gelingt schlicht nicht, schwierige Instanzen mit nur 2 Regeln zu finden. Andererseits darf die Komplexität der Maskenmethode mit einkodierten Ausschlussmengen und abgeschnittenen Pfaden nicht unterschätzt werden.

Die Zahlen zeigen außerdem, dass unter zufällig gewählten kleinen Instanzen ein hoher Anteil sehr leicht zu entscheiden sein wird.

Bei diesem Versuch wurden keine Instanzen gefunden, die endliche Suchgraphen ohne Lösungspfad erzeugen, wie die Zeile zeigt, die nur Nullen enthält. Instanzen wie in Beispiel 31c sind also wahrscheinlich recht selten. ⌟

a **Simples Einfangen**

Im nächsten Schritt wenden wir eine Variante des Einfangens aus 39b an. Beim Einfangen geht es ja darum, Mengen von Überhängen zu finden, die abgeschlossen gegenüber der Suchrelation sind und keine Lösung enthalten. Während 39b eine teure Heuristik beschreibt, die solche Mengen rät, wollen wir zunächst zwei simple Varianten betrachten: Bei der ersten vermuten wir, dass für eine Teilmenge $R \subseteq A$ die Menge $h(\mathrm{R}(I) \cap R^+) \times g(\mathrm{R}(I) \cap R^+)$ abgeschlossen gegenüber $\rightarrowtail_I$ ist, wobei $\mathrm{R}(I)$ die Vorgabemenge aus 19a ist. In der zweiten Variante vermuten wir, dass $(h(R^*) \cap B^{\geq \mathrm{wd}(I)})^\uparrow$ oder $(g(R^*) \cap B^{\geq \mathrm{wd}(I)})^\downarrow$ abgeschlossen sind, wobei R eine Teilmenge von Regeln ist. Diese Vermutungen können natürlich nur relativ zu einer Schranke überprüft werden. Sie versuchen, ähnlich wie bei der Konstruktion der Ausschlussmengen in 34a, die Situation zu erfassen, in der die Anwendung bestimmter Regeln nur Überhänge erzeugt, auf die keine der restlichen Regeln mehr angewendet werden können. Auf diese Weise kann zum Beispiel erkannt werden, dass bei der Instanz I_{105a} die Menge $((01)^* \cup 1(01)^*1)^\downarrow$ abgeschlossen gegenüber der Suchrelation mit Berücksichtigung der Ausschlussmengen ist.

b **Zahlen:** Simples Einfangen mit den Schranken 20 und 300: Es wird maximal 20 Schritte lang getestet, ob Abschluss vorliegt. Wenn die endlichen Automaten zur Beschreibung der Überhangmengen mehr als 300 Zustände haben, dann wird ebenfalls abgebrochen.

Ein Häkchen bedeutet, dass eine kombinierte Lösung gefunden wurde, oder dass der kombinierte Suchgraph endlich ist und keine Lösung enthält. Es müssen nur die Instanzen ohne Häkchen weiter betrachtet werden.

		g	3			4
↻	∅	w	2	3	4	2
·	·		0	37	1 216	11
·	✓		1	48	978	27
✓	·		0	0	0	0

Bei wenigen Regeln treffen die naiven Vermutungen recht häufig zu. Die Klasse $\mathcal{P}(3, 2)$ ist nun ebenfalls komplett automatisch klassifiziert. ⌟

c **Balancieren mit Vorgabe**

Das Balancieren mit Vorgabe aus 24a ist ein recht teurer Filter. Bisher waren die beteiligten Objekte Graphen von Überhängen und endliche Automaten. Nun kommen kontextfreie Sprachen ins Spiel, die durch Grammatiken und

Kellerautomaten repräsentiert werden. Die Lehrbuchkonstruktionen zur Berechnung des Schnitts einer kontextfreien mit einer regulären Sprache gehen von einem Kellerautomaten aus. Die Entscheidung, ob eine kontextfreie Sprache leer oder endlich ist, wird in der Regel mit Hilfe einer kontextfreien Grammatik getroffen.

Wir haben beide Modelle implementiert und beginnen mit einer Darstellung des Kerns $\varrho_v^{-1}(I)(d)$ der Balance-Abbildung durch einen Kellerautomaten. Wir berechnen dann den Schnitt mit der Vorgabemenge $\mathrm{R}(I)$ und erhalten einen weiteren Kellerautomaten. Um zu entscheiden, ob die resultierende Sprache leer oder endlich ist, berechnen wir eine Grammatik in Chomsky-Normalform, die, bis auf das leere Wort, die selbe Sprache wie dieser Kellerautomat akzeptiert. Wir gehen noch etwas genauer vor und berechnen den Schnitt mit der Menge $\mathrm{R}(I) \cap \mathrm{St}(I)A^* \cap A^* \,\mathrm{Fl}(I)$, also mit der Vorgabemenge, ergänzt um die Information über mögliche erste und letzte Regeln.

Die Parameter dieser Methode sind die Vektoren $v \in \mathbb{Z}^{|B|}$ und wie in 24a erwähnt, sollten wenigstens die Summen von Teilmengen der Einheitsvektoren untersucht werden.

Zahlen: Balancieren mit Vorgabe für die Vektoren $(0,1)$, $(1,0)$ und $(1,1)$. *a*

Ein Häkchen bedeutet, dass erkannt wurde, dass nicht alle Regeln in Lösungen verwendet werden können. Es müssen nur die Instanzen ohne Häkchen weiter betrachtet werden.

	g	3		4
ϱ	w	3	4	2
·		28	1 044	7
✓		9	172	4

Es werden relativ wenige Instanzen als unlösbar erkannt, die Verbindung aus kombinatorischen und algebraischen Argumenten ist aber tatsächlich mächtiger als die Summe der beiden Einzelteile. ⌟

Einfangen *b*

Der teuerste implementierte Filter ist das Einfangen mit der in 39b beschriebenen Heuristik. Kurz gesagt konstruieren wir einen Teil des Suchgraphen und vermuten dann, dass die dabei gefundenen Überhänge bereits genügen, um die Struktur der Überhänge des gesamten Suchgraphen zu beschreiben. Genauer vermuten wir, dass der gesamte Suchgraph nur Überhänge enthält, die nur Präfixe, Suffix und Infixe enthalten, die die Überhänge des konstruierten Teils bereits enthalten.

Die hohen Kosten entstehen, da für verschiedene (Paare von) Automaten überprüft werden muss, ob sie eine Menge von Überhängen beschreiben, die abgeschlossen gegenüber der Suchrelation ist.

a **Zahlen:** Einfangen mit der Heuristik aus 39b. Wir konstruieren einen Anfangsteils des Suchgraphen mit 1000 Knoten. Anschließend raten wir für $k \in \{3,4,5,6\}$, dass bereits alle Präfixe und Suffixe von Überhängen im Suchgraphen der Länge k und alle Infixe aufgetreten sind. Zur Überprüfung nehmen wir maximal 20 Schritte mit der Suchrelation über diesen Automaten vor, die maximal 500 Zustände haben dürfen.

Ein Häkchen bedeutet, dass Unlösbarkeit erkannt wurde. Es müssen nur die Instanzen ohne Häkchen weiter betrachtet werden.

				g	3		4
3	4	5	6	w	3	4	2
·	·	·	·		18	675	6
✓					1	84	1
·	✓				8	104	0
·	·	✓			1	151	0
·	·	·	✓		0	30	0

Die Parameter sind so gewählt, dass eine Bearbeitung vieler Instanzen in recht kurzer Zeit möglich ist. Dennoch werden einige Instanzen als unlösbar erkannt, ein Zeichen dafür, dass es schwierig ist, mit wenigen Regeln komplizierte Graphen zu beschreiben. ⌟

b **Gruppieren**

Manchmal können Teilwörter von h- oder von g-Bildern einer Instanz nur komplett und nichtüberlappend erzeugt werden. Technisch ist dann die freie Hülle der Menge $h(A) \cup g(A)$ nicht $\{0,1\}$. (Choffrut, Karhumäki [2]) Betrachten wir zum

c **Beispiel:** Die Instanz $I_{108c} = \| \begin{smallmatrix} 0 \\ 0010 \end{smallmatrix} \begin{smallmatrix} 0010 \\ 01 \end{smallmatrix} \begin{smallmatrix} 01 \\ 0 \end{smallmatrix} \|$. Dann ist $X = h(A) \cup g(A)$ die Menge $\{0, 0010, 01\}$ und das kleinste freie Monoid, das X^* enthält ist das Monoid $\{0,01\}^*$. Die freie Hülle $\hat{X}$ von X ist also die Menge $\{0,01\}$, die beschreibt, dass das Teilwort 01 nur komplett und nichtüberlappend erzeugt werden kann. Daraus folgt, dass I_{108c} die selbe Lösungsmenge wie die Instanz $J_{108c} = \| \begin{smallmatrix} 0 \\ 020 \end{smallmatrix} \begin{smallmatrix} 020 \\ 2 \end{smallmatrix} \begin{smallmatrix} 2 \\ 0 \end{smallmatrix} \|$ hat. Diese Instanz ist unlösbar, da ein Weg von $\downarrow 20$ nach $\uparrow 02$ gefunden werden müsste. Der einzige Übergang mit Änderung der Position ist aber $\downarrow 0 \overset{b}{\rightarrowtail}_{J_{108c}} \uparrow 0$, doch der Überhang $\downarrow 0$ hat ungerade Länge und ist deshalb nicht von $\downarrow 20$ erreichbar. ⌟

Das Beispiel zeigt, dass das Ersetzen von Teilwörtern, die nichtüberlappend erzeugt werden können, zu einfacheren Instanzen führen kann. Es lohnt sich, in diesem Fall alle Tests und Lösungsversuche nochmals an der ungruppierten Instanz vorzunehmen.

Zahlen: Zweiter Test bei vorhandenen Gruppen. *a*

Ein Häkchen bedeutet, dass Unlösbarkeit in der Instanz mit ersetzter Gruppe erkannt wurde. Es müssen nur die Instanzen ohne Häkchen weiter betrachtet werden.

	g	3		4
$\hat{X}$	w	3	4	2
·		15	659	6
✓		3	16	0

Ling Zhao gibt die Zahlen nur für nichtlöschende Instanzen an. (Zhao [44]) Seine Programme können in den Klassen $\mathcal{P}(3,3)\cap\mathcal{E}^0$, $\mathcal{P}(3,4)\cap\mathcal{E}^0$ und $\mathcal{P}(4,3)\cap\mathcal{E}^0$ genau 33, 3170 und 13 923 Instanzen nicht automatisch klassifizieren. Hier bleiben dagegen nur 4, 297 und 1740 nichtlöschende Instanzen übrig. Darunter befinden sich viele, die mit veränderten Parametern automatisch als unlösbar erkannt werden können, oder deren Lösungen mit den Techniken aus dem nächsten Abschnitt gefunden werden können. ⌟

Die restlichen Instanzen aus $\mathcal{P}(3,3)$ lassen sich alle klassifizieren. Unter ihnen befinden sich die beiden schwersten nichtlöschenden Instanzen $I_{16b} = \| \begin{smallmatrix} 0 & 001 & 1 \\ 001 & 1 & 0 \end{smallmatrix} \|$ und $J_{16b} = \| \begin{smallmatrix} 0 & 001 & 10 \\ 001 & 1 & 0 \end{smallmatrix} \|$ aus Beispiel 16b und insgesamt 11 löschende Instanzen, von denen hier einige exemplarisch klassifiziert werden sollen.

Beispiel: Die Instanz $I_{109b} = \| \begin{smallmatrix} \varepsilon & 00 & 011 \\ 000 & 1 & 0 \end{smallmatrix} \|$ erlaubt in einer Lösung keine einzelnen b, hat also die selbe Gleichheitsmenge wie die Instanz $\| \begin{smallmatrix} \varepsilon & 0000 & 011 \\ 000 & 11 & 0 \end{smallmatrix} \|$. Nach Gruppieren untersuchen wir die Instanz $\| \begin{smallmatrix} \varepsilon & 0000 & 01 \\ 000 & 1 & 0 \end{smallmatrix} \|$, in deren Suchgraphen nur ↑-Überhänge relevant sind. Jeder Pfad verläuft durch ein Element der Menge $\uparrow(\{0, 00000\}1)^*0000$, das unvermeidbar wieder auf ein Element aus dieser Menge führt. Also hat I_{109b} keine echte Lösung. ⌟ *b*

Beispiel: Bei der Instanz $I_{109c} = \| \begin{smallmatrix} \varepsilon & 0 & 01 \\ 001 & 1 & 0 \end{smallmatrix} \|$ stellen wir zunächst fest, dass in einer Lösung vor jedem b ein c stehen muss. Da b nicht erstes Symbol in einer Lösung sein kann, können wir zur Instanz $\| \begin{smallmatrix} \varepsilon & 010 & 01 \\ 001 & 01 & 0 \end{smallmatrix} \|$ übergehen, die nach Entfernen der Gruppe 01 die selbe Gleichheitsmenge wie die Instanz $\| \begin{smallmatrix} \varepsilon & 10 & 1 \\ 01 & 1 & 0 \end{smallmatrix} \|$ hat, deren Normalform die unlösbare Instanz I_{105a} ist. Also hat I_{109c} keine echte Lösung. ⌟ *c*

a **Beispiel:** Im Suchgraphen der Instanz $I_{110a} = \| \begin{smallmatrix} \varepsilon \\ 001 \end{smallmatrix} \begin{smallmatrix} 00 \\ 10 \end{smallmatrix} \begin{smallmatrix} 01 \\ 0 \end{smallmatrix} \|$ verläuft jeder Pfad durch einen Überhang der Form $\uparrow 0^{4k+3}$ mit $k \in \mathbb{N}$ aus dem die beiden Überhänge $\uparrow 0^{4(2k)+3}$ und $\uparrow 0^{4(2k+1)+3}$ abgeleitet werden können. Also hat I_{110a} keine echte Lösung. ⌟

b **Beispiel:** Betrachten wir den Beginn des Suchgraphen der Instanz $I_{110b} = \| \begin{smallmatrix} \varepsilon \\ 001 \end{smallmatrix} \begin{smallmatrix} 10 \\ 1 \end{smallmatrix} \begin{smallmatrix} 100 \\ 0 \end{smallmatrix} \|$, dann stellen wir fest, dass I_{110b} genau dann eine Lösung hat, wenn die allgemeine Instanz $| \begin{smallmatrix} 010100 \\ \varepsilon \end{smallmatrix} | \begin{smallmatrix} \varepsilon \\ 001 \end{smallmatrix} \begin{smallmatrix} 10 \\ 1 \end{smallmatrix} \begin{smallmatrix} 100 \\ 0 \end{smallmatrix} | \begin{smallmatrix} \varepsilon \\ \varepsilon \end{smallmatrix} |$ eine Lösung hat. In dieser Instanz muss in einer Lösung vor jedem b ein c stehen, I_{110b} ist also genau dann lösbar, wenn die Instanz $| \begin{smallmatrix} 010100 \\ \varepsilon \end{smallmatrix} | \begin{smallmatrix} \varepsilon \\ 001 \end{smallmatrix} \begin{smallmatrix} 10010 \\ 01 \end{smallmatrix} \begin{smallmatrix} 100 \\ 0 \end{smallmatrix} | \begin{smallmatrix} \varepsilon \\ \varepsilon \end{smallmatrix} |$ lösbar ist. In dieser Instanz kann aus Abzählgründen das Symbol c in keiner Lösung vorkommen und der restliche Suchgraph ist endlich und enthält keine Lösung. Also hat I_{110b} keine echte Lösung. ⌟

c

Lösen

Im letzten Abschnitt haben wir verschiedene Methoden angewendet, um einfache Instanzen zu identifizieren. Die Instanzen, die bis hierher nicht klassifiziert wurden, sollen nun einer *großen* Suche unterzogen werden. Eine große Suche ist eine Suche, die mit hohem Knotendurchsatz sehr große Suchgraphen in angemessener Zeit bearbeiten kann. Wir unterscheiden dabei zwischen einfacher Lösungssuche und Suche nach einer Lösung minimaler Länge. Eine einfache Lösungssuche kann sofort beendet werden, wenn eine Lösung gefunden wird, eine Suche nach einer Lösung minimaler Länge muss nach dem Finden einer Lösung noch alle kürzeren Pfade untersuchen.

Darüber hinaus suchen wir auch semantisch modifiziert: Wir schneiden einfach alle Pfade im Suchgraphen ab, deren Überhänge eine gewisse Maximallänge überschreiten. Die Semantik wurde verändert, klar, wir schneiden eventuell Pfade ab, die auf dem Weg zu einer Lösung liegen. Es hat sich aber nicht nur herausgestellt, dass dieses Vorgehen *gewaltiges* Beschleunigungspotential beinhaltet, sondern auch, dass es oft Lösungen gibt, deren Pfade keine langen Überhänge berühren. Ohne Suche mit veränderter Semantik wären viele der Rekordinstanzen aus 119b nicht gefunden worden. Es handelt sich um eine der wichtigsten, wenn nicht *der* wichtigsten, Technik zur Beschleunigung der Suche. Eigentlich ist das nicht überraschend, denn auch bei einer beschränkten Tiefensuche spielen wir mit der Semantik: Wir versprechen, dass keine minimale Lösung länger als die gegebene Schranke ist und vermeiden dadurch unendliche Abstiege.

Haben wir in einer Suche mit veränderter Semantik eine Lösung gefunden, dann ist es anschließend leichter nach einer minimalen Lösung zu suchen, da wir eine Maximaltiefe konkret kennen.

Die Anzahl der verbliebenen Instanzen ist zwar einerseits klein im Vergleich zur Anzahl der aussortierten, sie ist aber immer noch zu hoch, um per Hand Suchprogramme für jede Instanz zu implementieren.

Die Suchprogramme von Ling Zhao und Heiko Stamer sind deshalb generisch: Sie implementieren einen Suchalgorithmus, der mit einer Regelmenge parametrisiert wird. Dieses Vorgehen erlaubt die Bearbeitung großer Mengen von Instanzen mit ordentlichen Durchsätzen. Allerdings wird so eine Stufe der Indirektion erzeugt, die den Durchsatz prinzipiell verringert. Unser Ansatz ist daher ein anderer, der in der Mitte zwischen einem generischen und einem individuellen Programm liegt: Wir erzeugen automatisch für jede Instanz ein Programm, das möglichst viel Wissen über die Instanz enthält und gleichzeitig ohne jede Indirektion arbeitet.

Die Idee, aufgabenspezifische, optimierte Programme zu erzeugen ist keineswegs neu. So wählt beispielsweise die Bibliothek FFTW zur Berechnung der diskreten Fourier Transformation (`http://www.fftw.org/`) aus einer Menge vorhandener Routinen zur Laufzeit eine optimale Kombination aus. Dadurch passt sich die Bibliothek automatisch an die Aufgabe und an die Maschine an. Dieses Vorgehen hat dann Vorteile, wenn der Anpassungsaufwand klein im Vergleich zum Aufwand für das Lösen der Aufgabe ist.

Unser Grundmodell sieht wie folgt aus: Ein Haskell-Programm versucht eine Instanz mit einer der Filtermethoden zu klassifizieren. Dabei sammelt es Informationen wie Ausschlussmengen, gültige Positionen oder Kandidatenmengen. Ist keine Klassifikation möglich, dann erzeugt das Haskell-Programm ein C-Programm, das ohne Indirektion die Suchrelation implementiert und außerdem die gesammelte Information zuschaltbar berücksichtigen kann. Nun steuert das Haskell-Programm das C-Programm. Es führt prinzipiell eine Tiefensuche mit schrittweiser Vergrößerung der maximalen Suchtiefe aus und passt gleichzeitig die Parameter des C-Programms an. Dazu berechnet das Haskell-Programm den Durchsatz des C-Programms für verschiedene Einstellungen und wählt die Einstellung, die den größten Durchsatz verspricht.

Dieses Vorgehen hat den Vorteil, dass wir alle komplizierten Vorgänge bequem in Haskell implementieren können. Da nur für solche Instanzen eine große Suche durchgeführt wird, die nicht einfach sind, können wir außerdem davon ausgehen, dass der zusätzliche Aufwand für Programmerzeugung, Übersetzung und Kommunikation klein im Vergleich zu der Zeit ist, in der das C-Programm konkret sucht.

Beispiel: Sei $\nu(I, k)$ die minimale Länge von Überhängen, die betrachtet werden müssen, um eine Lösung von I zu finden, die höchstens die Länge k hat. Für die Instanz $I_{111a} = \| \begin{smallmatrix} 0 \\ 00010 \end{smallmatrix} \begin{smallmatrix} 01 \\ 0 \end{smallmatrix} \begin{smallmatrix} 0100 \\ 1 \end{smallmatrix} \|$ gelten $\nu(I_{111a}, 204) = 81$, $\nu(I_{111a}, 648) = 68$ und $\nu(I_{111a}, 1290) = 63$. *a*

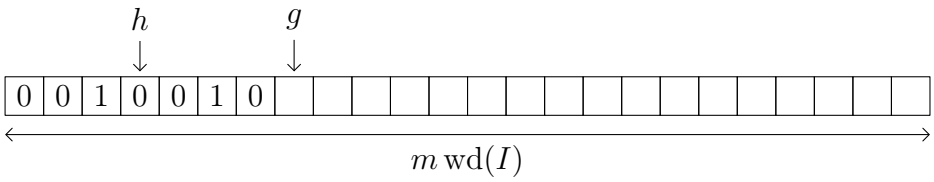
Werden bei einer Suche mit veränderter Semantik alle Überhänge abgeschnitten, die länger als 70 sind, dann wird zwar die minimale Lösung der Länge 204 nicht gefunden, sehr wohl aber eine Lösung der Länge 648. ⌟

a

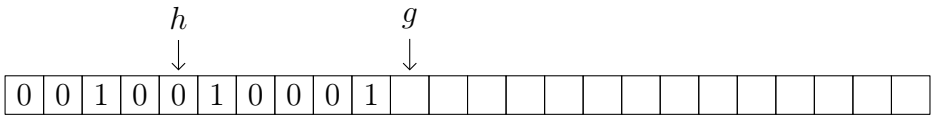
Sequentiell

Unser vornehmliches Ziel ist ein performantes C-Programm, das die Suchrelation ohne Indirektion kodiert.

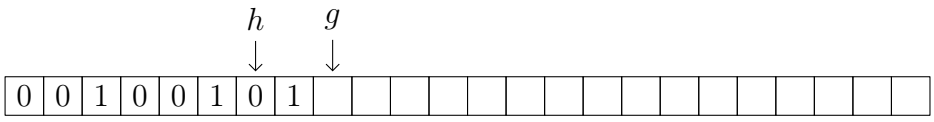
Dazu beobachten wir zunächst, dass die rekursive Traversierung des Suchgraphen neben dem Aufrufstapel nur konstant viel Speicher benötigt: In einer gegebenen Tiefenschranke m kann kein Überhang länger als $m \operatorname{wd}(I)$ werden. Ein Schritt ohne Änderung der Position mit $\rightarrowtail_I$ entspricht dem Anhängen rechts am aktuellen Überhang und dem Löschen links. Wir arbeiten also mit folgendem Grundmodell:



Wir stellen Platz der Größe $m \operatorname{wd}(I)$ bereit, in dem wir das aktuelle Bild von h *und* von g speichern. Da wir nach einer Lösung suchen, sind diese beiden Bilder vergleichbar. Es genügt also das längere Bild und die beiden Positionen zu speichern, an denen die Bilder fortgesetzt werden sollen. Die Abbildung zeigt die Situation, nachdem für die Instanz $I_{16b} = \| \begin{smallmatrix} 0 \\ 001 \end{smallmatrix} \begin{smallmatrix} 001 \\ 1 \end{smallmatrix} \begin{smallmatrix} 1 \\ 0 \end{smallmatrix} \|$ die Regeln aa c angewendet wurden. Tatsächlich gilt $h(aac) = 001$ und $g(aac) = 0010010$. Der aktuelle Überhang ist $\downarrow 0010$. In dieser Situation können sowohl Regel a als auch Regel b angewendet werden. Es ergeben sich dann die Situationen



und



mit den Überhängen $\downarrow 010001$ und $\downarrow 01$. An dieser Stelle ist es wichtig zu beobachten, dass nur die Positionen von h und g gespeichert werden müssen,

da nur am Ende angehängt wird. Ist der Pfad für die Anwendung von a abgearbeitet, dann müssen nur diese Positionen zurückgesetzt werden und der Pfad für die Anwendung von b kann mit dem *selben* Speicherbereich arbeiten wie der Pfad für die Anwendung von a. Das ist eine sehr gute Nachricht, denn so ist es möglich, ohne jedes Kopieren von Speicherbereichen zu arbeiten. Da auf dem Stapel lediglich 2 Positionen abgelegt werden müssen und weil der Programmtext nicht lang sein wird, können wir davon ausgehen, dass sich das Programm, der Stapel und alle Daten komplett im schnellsten Bereich der Speicherhierarchie befinden.

Dieses Modell hat einige Vorteile: Zunächst kann genaues Abschneiden wie in 32b sehr effizient implementiert werden. Es ist lediglich nötig, den Zustand des Automaten für $\mathrm{R}(I)$ mit auf dem Stapel zu speichern. Speichern wir zusätzlich die Anzahl der bisher angewendeten Regeln, dann kann auch beschränktes Abschneiden anhand der Bedingungen aus (20b) effizient implementiert werden.

Noch wichtiger ist, dass die Länge des Überhangs mit Hilfe der Positionen mit genau einer Subtraktion berechnet werden kann. Die Länge ist deshalb interessant, weil mit ihrer Hilfe das Vorwärtsabschneiden von Pfaden als Spezialfall des beschränkten Abschneidens wie in 33a implementiert werden kann. Dabei werden Pfade abgeschnitten, bei denen der Überhang zu lang ist, um noch innerhalb der Tiefenschranke überdeckt zu werden. Außerdem basiert das Suchen mit veränderter Semantik wie in 110c auf der Kenntnis der Länge des Überhangs. Nicht zuletzt wurde genau dann eine Lösung gefunden, wenn die Länge des Überhangs 0 ist.

Wenden wir uns der Frage zu, wie ein Schritt auf dieser Darstellung effizient ausgeführt werden kann. Die erste Idee ist einfach: Für jede Regel, verlängere zunächst das längere Bild und teste anschließend, ob sich das kürzere Bild verlängern lässt. Wenn sich das kürzere Bild nicht korrekt verlängern lässt, dann haben wir bei diesem Vorgehen unnötige Arbeit in die Verlängerung des längeren Bildes gesteckt. Da auch Schritte mit Änderung der Position möglich sind, wird der Test, ob sich das kürzere Bild verlängern lässt, im Allgemeinen nur ein Präfix testen, dessen Länge von der Länge des Überhangs abhängt. Allerdings sind nur bei bestimmten Längen Schritte mit Änderung der Position möglich.

Es bietet sich an, in Abhängigkeit von der Länge des Überhangs unterschiedliches Verhalten zu implementieren. Bei einer Länge, bei der keine Änderung der Position möglich ist, versuchen wir das *kürzere* Bild zu verlängern und nur bei Erfolg verlängern wir auch das längere. Bei einer Länge, bei der eine Änderung der Position möglich ist, testen wir für die in Frage kommenden Regeln nur ein bekanntes Präfix, für die restlichen Regeln verfahren wir normal.

Ein Beispiel anhand der Instanz $I_{16b} = \| \begin{smallmatrix} 0 \\ 001 \end{smallmatrix} \begin{smallmatrix} 001 \\ 1 \end{smallmatrix} \begin{smallmatrix} 1 \\ 0 \end{smallmatrix} \|$. Wenn ein $\uparrow$-Überhang vorliegt, dann sind folgende Übergänge möglich: $\uparrow 0 \overset{a}{\rightarrowtail}_{I_{16b}} \downarrow 1$, $\uparrow 001u \overset{a}{\rightarrowtail}_{I_{16b}} \uparrow u0$, $\uparrow 1u \overset{b}{\rightarrowtail}_{I_{16b}} \uparrow u001$ und $\uparrow 0u \overset{c}{\rightarrowtail}_{I_{16b}} \uparrow u1$. Das prototypische C-Programm hat die Gestalt:

```
1   void Top(int * h, int * g, uint depth)
2   {
3       if (depth < MAX_DEPTH) {
4           uint l = h - g;
5
6           if (l > 0) {
7               if (*(g + 0) == 0) {
8                   if (l > 2) {
9                       if (*(g + 1) == 0 && *(g + 2) == 1) {
10                          *(h + 0) = 0;
11                          Top(h + 1, g + 3, depth + 1);
12                      }
13                  } else if (l == 1) {
14                      *(g + 0) = 0;
15                      *(g + 1) = 0;
16                      *(g + 2) = 1;
17                      Bot(h + 1, g + 3, depth + 1);
18                  }
19
20                  *(h + 0) = 1;
21                  Top(h + 1, g + 1, depth + 1);
22              } else {
23                  *(h + 0) = 0;
24                  *(h + 1) = 0;
25                  *(h + 2) = 1;
26                  Top(h + 3, g + 1, depth + 1);
27              }
28          } else {
29              printf("Solution: depth = %d\n", depth);
30          }
31      }
32  }
```

Hier ist `MAX_DEPTH` die Tiefenschranke, deren Einhaltung in Zeile 3 sichergestellt wird. Die Funktionen `Top` und `Bot` sind die Funktionen zur Bearbeitung von $\uparrow$- und $\downarrow$-Überhängen. Die Parameter sind die Zeiger `h` und `g` auf die

Stellen, an denen die h- und g-Bilder verlängert werden sollen. In Zeile 6 wird geprüft, ob eine Lösung gefunden wurde, falls ja wird das in Zeile 29 ausgegeben. Wenn die in Zeile 4 berechnete Länge des Überhangs echt größer als 0 ist, dann verzweigen wir in Zeile 7 zunächst anhand des ersten Zeichens des Überhangs in die Zeilen 8–21 oder in die Zeilen 23–26. Der zweite Pfad wendet direkt die Regel b an. Dazu wird 001 an das h-Bild angehängt und das g-Bildes um ein Symbol weiter geschaltet. Im ersten Pfad wenden wir in den Zeilen 20–21 auf jeden Fall die Regel c an. Wenn nun die Länge des Überhangs mindestens 3 beträgt, dann wenden wir in den Zeilen 10–11 die Regel a an, falls sich das g-Bild entsprechend dem Test in Zeile 9 verlängern lässt. Es bleibt schließlich der Übergang mit Änderung der Position, der nur bei einem Überhang der Länge 1 (Zeile 13) stattfinden kann. Die Zeile 14 ist redundant und nur zur besseren Verständlichkeit des Beispiels angegeben.

Dieses Programm kodiert die Suchrelation $\rightarrowtail_{I_{16b}}$ für $\uparrow$-Konfigurationen vollständig und solche Programme lassen sich relativ leicht erzeugen. Der Autor hat einen Generator für eine Variante solcher Programme implementiert und später hat der Student Benjamin Vogel für den Autor einen Generator implementiert, der ein Programm erzeugen kann, das dem obigen sehr ähnlich ist.

Wir wollen hier nicht weiter auf Details eingehen, zum Beispiel ist es kein Zufall, dass sich die Blöcke 20–21 und 8–18 in dieser Reihenfolge befinden. Statt dessen wollen wir festhalten, dass dieser Programmtyp generiert werden kann und sich als sehr effizient heraus gestellt hat. Tatsächlich erreichen die so generierten Programme regelmäßig Durchsätze zwischen $50 \cdot 10^6$ und $100 \cdot 10^6$ Knoten pro Sekunde, das entspricht etwa 30 bis 40 Maschinenzyklen pro Knoten.

Für einige besonders schwere Instanzen hat der Autor Programme per Hand noch weiter optimiert, in einigen Fällen entstanden so Programme, die nur noch etwa 15 Maschinenzyklen pro Knoten benötigen. Solche Programme erlauben das Traversieren von Suchgraphen mit 10^{13} Knoten innerhalb eines Tages. Weitere Optimierungen auf Ebene des C-Programms sind schwierig. Für konkrete Maschinen ist vorstellbar, dass bei genauer Kenntnis der Arbeitsweise, zum Beispiel der Fließbandverwaltung oder der Sprungvorhersage, der Durchsatz nochmals um eine Größenordnung höher ausfallen könnte.

Die Techniken zum Beschleunigen der Suche, die mit der Beschriftung eines Pfades arbeiten, lassen sich in diesem Modell sehr effizient implementieren. Anders die Techniken, die mit den Knoteninhalten, also den Überhängen arbeiten. Die aktuelle Beschriftung wird nur am Ende verlängert. Das ermöglicht zum Beispiel den Automaten für die Vorgabemenge $\mathrm{R}(I)$ mitlaufen zu lassen, das heißt wir können dessen Zustand auf dem Stapel ablegen und mit konstantem Aufwand aktualisieren. Der aktuelle Überhang wird ebenfalls

am Ende verlängert, aber auch gleichzeitig am Anfang verkürzt. Wollen wir den Zustand ermitteln, den zum Beispiel ein endlicher Automat für eine der Ausschlussmengen nach Eingabe des aktuellen Überhangs erreicht, dann müssen wir Aufwand in der Größenordnung der Länge des Überhangs betreiben. In der Praxis zahlt es sich in der Regel nicht aus, solche Techniken zu implementieren: Die abgeschnittenen Teilgraphen müssen so groß sein, dass sich eine Verlangsamung des Programms um mehrere Größenordnungen lohnt. Ist das aber der Fall, dann kann meist bereits die Suchrelation anders implementiert werden, so dass diese Teilgraphen gar nicht erst entstehen.

Eine letzte Bemerkung zur Art der Traversierung: Wir nehmen eine beschränkte Tiefensuche vor, die bisher Schleifen nicht detektiert und also im Allgemeinen Teilgraphen mehrfach besucht. Zwar stellt sich in der Praxis einerseits heraus, dass Suchgraphen schwieriger Instanzen nicht viele Schleifen enthalten, aber andererseits kann eine einzige entdeckte Schleife, die weit oben im Suchgraphen auftritt, die Suche sofort um den Faktor 2 beschleunigen. Das macht die Richtung klar, in die wir bei Schleifendetektion gehen: Kurze Überhänge werden zusammen mit der Tiefe memoriert, auf der sie aufgetreten sind. Tauchen sie später weiter unten im Suchgraphen erneut auf, dann müssen sie nicht weiter betrachtet werden. Dieses Vorgehen nimmt Rücksicht auf die Gegebenheiten der Speicherhierarchie, tendiert dazu Schleifen zu detektieren, die weit oben im Suchgraphen liegen und verursacht für zu lange Überhänge nur konstante Kosten.

a **Implementation:** Verschiedene andere Schemata sind vorstellbar und bei deren Implementation und Evaluation hat sich die Bibliothek `Judy` (`http://judy.sourceforge.net/`) als äußerst effizientes Werkzeug zum Umgang mit dynamischen Feldern heraus gestellt. Diese Bibliothek schlägt regelmäßig selbst Implementationen, die mit aufgabenspezifischem Wissen arbeiten, sowohl im Speicherverbrauch als auch in der Geschwindigkeit. ⌟

In der Praxis hat sich bewährt, einmal mit eingeschalteter Detektion von Schleifen um Überhänge bis zu einer Länge von etwa 25 und einmal ohne Detektion von Schleifen zu suchen. Anhand der dabei ermittelten durchschnittlichen Verzweigungsgrade wählen wir dann die endgültige Einstellung.

b

Parallel

Die Natur des Postschen Korrespondenzproblems bringt es mit sich, dass wir es mit beliebig großen Suchgraphen zu tun haben. Nachdem wir im letzten Abschnitt sequentielle Programme bis nahe an die theoretischen Grenzen gebracht haben, wollen wir uns nun kurz der parallelen Suche widmen.

Wir implementieren kein eigenes paralleles Modell, sondern parallelisieren die bereits vorhandenen sequentiellen Programme. Insbesondere verwalten wir keine global sichtbare Struktur zur Memorierung bereits besuchter Überhänge.

Unsere Implementationen basieren auf dem Konzept nachrichtengekoppelter Parallelrechner. (Ungerer [41]) Mit MPI liegt eine Spezifikation einer Bibliothek zum Nachrichtenaustausch in industrieller Qualität vor, die auf vielen realen Plattformen zur Verfügung steht. (MPI [31]) Für unsere Versuche stand dem Autor der universitäre Parallelrechner HP XC6000 im Rechenzentrum der Universität Karlsruhe zur Verfügung. Diese Maschine ist ein Parallelrechner mit verteiltem Speicher, die für einzelne Berechnungen maximal 128 Intel Itanium2 Prozessoren mit einer Taktfrequenz von 1,5 GHz bereitstellt. Bei maximal 4 gleichzeitig abgearbeiteten Befehlen entspricht das maximal $7{,}68 \cdot 10^{11}$ Operationen pro Sekunde, das heißt zur Traversierung eines Graphen mit 10^{15} Knoten werden *mindestens* etwa 22 Minuten benötigt. Ziehen wir in Betracht, dass pro Knoten vermutlich $2\,\mathrm{wd}(I)$ Operationen für die Regelanwendung und vermutlich mindestens 8 Operationen für die Verwaltung des Stapels benötigt werden, dann erhalten für eine Instanz mit Weite 4 bereits etwa 6 Stunden minimale Bearbeitungszeit für einen Suchgraphen mit 10^{15} Knoten.

Zufälliges Anfragen *a*

Die erste Variante basiert auf der Arbeit von Peter Sanders. (Sanders [37], Sanders, Worsch [38]) Die Methode des asynchronen zufälligen Anfragens verteilt die Last bei baumförmigen Berechnungen beweisbar gut. Zusammen mit einer asymptotisch optimalen Terminierungserkennung wie sie Friedemann Mattern beschreibt, erhalten wir ein Verfahren mit sehr guter Effizienz, auch in heterogenen Umgebungen. (Mattern [26])

Allerdings kann das sequentielle Modell nicht direkt als Basis dienen: Beim zufälligen Anfragen teilen Prozesse ihre Arbeit auf und geben einen Teil an einen anfragenden Prozess ab. Die vorhandene Arbeit besteht aus dem Inhalt des Stapels, der nicht direkt, sondern nur als Aufrufstapel vorliegt. Wir müssen den Stapel explizit sichtbar machen und wollen trotzdem nur mit einem einzigen Speicherbereich für Überhänge arbeiten. Dazu ist es notwendig, auf dem Stapel die *alten* Positionen für die Fortsetzungen von h und g zusammen mit der nächsten anzuwendenden Regel abzulegen. Anderenfalls würde die zweite anwendbare Regel die Änderungen der ersten anwendbaren Regel überschreiben. Nach Einlesen des obersten Elementes des Stapels wenden wir die gespeicherte Regel an und erhalten neue Positionen. Für jede nun anwendbare Regel speichern wir diese Positionen zusammen mit der Regel. Zur Behandlung von Übergängen mit Änderung der Position können wir

entweder mit einem einzigen Stapel arbeiten und bei jedem Überhang die Position mit konstantem Aufwand bestimmen, oder wir arbeiten mit zwei getrennten Stapeln und je einer eigenen Funktion für jede der Positionen.

Das Modell mit explizitem Stapel hat keine prinzipiellen Nachteile gegenüber dem oben vorgestellten Modell mit implizitem Stapel und erreicht in der Praxis die selben Durchsätze.

Wenn der Stapel explizit vorliegt, dann kann mit den Techniken von Sanders und Mattern ein paralleles Programm mit sehr guter Effizienz einfach implementiert werden. Die Effizienz liegt für große Suchgraphen praktisch bei 1. Der Aufwand für das zufällige Anfragen und die Terminierungserkennung muss nur ein Mal betrieben werden. Wir benötigen zusätzlich einen Mechanismus für den kontrollierten Abbruch, der von Prozessen angestoßen werden kann, die eine Lösung gefunden haben. Schließlich müssen wir noch ein Intervall festlegen, in dem ein aktiver Prozess zur Kommunikation bereit ist. Dieses Intervall sollte nicht zu kurz sein, damit nicht zu oft Daten aus langsamen Ebenen der Speicherhierarchie benötigt werden.

a

Meister und Arbeiter

Die Methode des zufälligen Anfragens geht von einem sequentiellen Programm mit explizitem Stapel aus. Unsere Generatoren erzeugen aber Programme mit implizitem Stapel. Wir haben in unseren Versuchen deshalb meist ein anderes Parallelisierungsschema verwendet, das des Meisters und der Arbeiter. Dazu verwaltet ein Prozess, der Meister, die Arbeit und übergibt Teile davon an die anderen Prozesse, die Arbeiter, die den Meister über ihre Fortschritte unterrichten.

Bei der Suche nach einer Lösung verwaltet der Meister einen Teil des Suchgraphen. Zum Beispiel könnte der Meister den Suchgraphen bis zur Tiefe ηm mit $0 < \eta < m$ verwalten, wenn insgesamt bis zur Tiefe m gesucht werden soll. Jedes Blatt des Suchgraphen beim Meister entspricht einer Wurzel eines Suchgraphen für die Arbeiter, die bis zu einer Tiefe von $(1 - \eta)m$ suchen. Der Meister braucht höchstens ηm Schritte, um ein neues Blatt zu finden. Wir können also davon ausgehen, dass Arbeiter nicht lange auf neue Arbeit warten müssen. Andererseits stehen für den Meister bei genügend großer Anzahl von Arbeitern auch immer Abnehmer für neue Blätter bereit. Natürlich hängt diese Aussage von der Wahl des Parameters η ab. In der Praxis hat sich bewährt von $\eta = 1/2$ auszugehen.

Dieses Schema hat den Vorteil, dass die sequentiellen Programme mit implizitem Stapel als Basis dienen können. In der Praxis erreichen wir auch hier optimale Effizienzen. Es werden kleinere Nachrichten als beim zufälligen Anfragen versendet: Dort wird neben dem aktuellen h- oder dem aktuellen g-

Bild ein gesamter Stapel versendet, hier nur ein Element eines solchen Stapels. Im Gegenzug tendiert die Methode des zufälligen Anfragens dazu, weniger Nachrichten zu erzeugen.

Zusammengefasst sind beide Schemata sehr gut geeignet, große Suchgraphen zu traversieren. Bei einer Neuimplementation würde der Autor sich jedoch für die etwas komplexere Methode des zufälligen Anfragens entscheiden: Sie ist robust auch bei heterogenen Umgebungen und die Wahl der Länge eines Arbeitsintervalls fällt leichter als die Wahl des Anteils des Suchgraphen, den der Meister traversieren soll.

Zahlen: Zwei der größten traversierten Graphen hatten Knotenanzahlen von *a*
1 034 088 205 926 631 und 1 006 827 971 112 692. Beide wurden auf 128 Prozessoren des universitären Parallelrechners HP XC6000 in 99 284 und in 90 178 Sekunden durchsucht. Das entspricht einem Durchsatz von $1{,}04 \cdot 10^{10}$ und von $1{,}12 \cdot 10^{10}$ Knoten pro Sekunde, dass heißt, jeder der 128 Prozessoren hat mehr als einen Tag lang jede Sekunde mehr als 80 Millionen Knoten betrachtet, wobei dann pro Knoten nur noch etwa 18 Takte zur Verfügung stehen. Wohlgemerkt inklusive zusätzlichem Aufwand für die parallele Verarbeitung, die im wesentlichen aus dem Versenden und Empfangen von 469 636 908 und von 4 585 927 Nachrichten bestand, und inklusive zusätzlichem Aufwand zur Ermittlung der Knoten- und Nachrichtenanzahlen. ⌟

Rekordinstanzen *b*

Unsere empirischen Daten können nach verschiedenen Gesichtspunkten geordnet werden, die Länge der kürzesten Lösung ist nur ein Kriterium. Einige interessante Instanzen tauchen bereits an anderer Stelle in dieser Arbeit auf: Instanzen mit vielen verschiedenen kürzesten Lösungen (Beispiel 17c), mit und ohne lange Blöcke aufeinander folgender gleicher Symbole in einer kürzesten Lösung (Beispiele 17d und 18a), mit stark unterschiedlich großen Suchgraphen für die gespiegelte Instanz (Beispiel 31a), mit großen endlichen Suchgraphen (Beispiel 31c) oder mit Suchgraphen ohne Verzweigungen. (Beispiel 31b) Wir haben Lösungen gesehen, die sich in wenige Blöcke zerlegen lassen (Beispiele 74f und 75a) oder unendliche Lösungen mit nichttrivialer Struktur. (Beispiele 16a, 78d und 88f)

Der Autor hat eine umfangreiche Sammlung interessanter Instanzen angelegt, aus der hier ein kleiner Ausschnitt mit Instanzen mit langen kürzesten Lösungen dargestellt wird, ergänzt um die Schwierigkeit einer Instanz. Es handelt sich um Instanzen, deren Lösungen erstmals vom Autor gefunden wurde. Die Spalte $\mu_{\min}$ gibt die untere Schranke für die Länge von Lösungen an, die Spalte μ die Länge der kürzesten bekannten Lösung, die Spalte c

enthält die Anzahl kürzester Lösung, falls diese bekannt ist und die Spalte S schließlich gibt den abgerundeten Logarithmus zur Basis 10 der Anzahl der Knotenexpansionen an, die nötig waren, um die Daten dieser Zeile zu ermitteln. Dabei dienen optimierte Programme als Basis, die versuchen, möglichst wenige Knotenexpansionen vorzunehmen. Die Spalte S ist also ein Maßstab für die Schwierigkeit einer Instanz. Es ist natürlich vorstellbar, dass die Daten in einer Zeile mit weniger Knotenexpansionen zu ermitteln sind.

Klasse	Instanz	$\mu_{\min}$	μ	c	S
$(3,4)$	$\begin{array}{\|c\|}\hline 0\ \ 01\ 1001 \\ 001\ 0\ \ 1 \\ \hline\end{array}$	444	452		15
$(3,5)$	$\begin{array}{\|c\|}\hline 0\ \ 00100\ 1 \\ 0010\ \ 1\ \ 0 \\ \hline\end{array}$	528	539		14
	$\begin{array}{\|c\|}\hline 0\ \ 010\ 100 \\ 00010\ 01\ \ 0 \\ \hline\end{array}$	528	528	1	14
	$\begin{array}{\|c\|}\hline 0\ \ 00100\ \ 10 \\ 001\ \ 0\ \ 0100 \\ \hline\end{array}$	288	288	1	13
	$\begin{array}{\|c\|}\hline 0\ \ 110\ 11000 \\ 00110\ 0\ \ 1 \\ \hline\end{array}$	266	266		13
$(4,3)$	$\begin{array}{\|c\|}\hline 0\ \ 001\ 1\ \ 11 \\ 011\ \ 1\ \ 00\ 110 \\ \hline\end{array}$	595	595	1	15
	$\begin{array}{\|c\|}\hline 0\ \ 000\ 001\ 1 \\ 010\ 11\ \ 1\ \ 00 \\ \hline\end{array}$	546	546	1	9
$(4,4)$	$\begin{array}{\|c\|}\hline 0\ \ 0111\ 101\ 11 \\ 0111\ \ 11\ \ 001\ 01 \\ \hline\end{array}$	880	880	1	15
	$\begin{array}{\|c\|}\hline 0\ \ 0000\ 1000\ 1100 \\ 0010\ \ 1\ \ 00\ \ 0001 \\ \hline\end{array}$	880	880	1	10
	$\begin{array}{\|c\|}\hline 0\ \ 0101\ \ 011\ \ 11 \\ 111\ \ 01\ \ 0011\ 01 \\ \hline\end{array}$	880	880	1	4
	$\begin{array}{\|c\|}\hline 0\ \ 0\ \ 0011\ 101 \\ 0000\ 01\ \ 00\ \ 1 \\ \hline\end{array}$	874	874	1	15
	$\begin{array}{\|c\|}\hline 0\ \ 00\ \ 01\ 1111 \\ 000\ 010\ 1\ \ 10 \\ \hline\end{array}$	770	770	2	14

Diese Instanzen sind die absoluten Rekordinstanzen bezüglich minimaler Lösungslänge und/oder bezüglich Schwierigkeit, die vom Autor gefunden wurden. Die gesamte Liste, die unter der Addresse `ftp://ftp.ira.uka.de/pub/pcp` zur Verfügung steht, umfasst mehr als 500 Instanzen, von denen der überwiegende Teil erstmals gelöst wurde. Es sei betont, dass dafür nicht nur sehr viele Knotenexpansionen notwendig waren, sondern auch sehr gute Programme. Die äußerst leichte Instanz in der Klasse $(4,4)$ mit $\mu_{\min} = \mu = 880$ und $S = 4$ wurde bei früheren Versuchen nicht gefunden, weil die eingesetzten Programme nicht leistungsfähig genug waren: Sie waren zu langsam, um die gesamte Klasse zu untersuchen *und* sie waren zu langsam, um mit solch tiefen Suchen umzugehen. Insbesondere in der Klasse $(4,4)$ ist der Fortschritt gewaltig: Galten 2002 noch Instanzen mit $\mu_{\min} = 100$ als schwierig (Zhao [44]) und war die schwierigste bekannte Instanz eine Instanz mit $\mu_{\min} = 256$, so kennen wir nun allein 17 Instanzen mit $\mu_{\min} > 500$ und *hunderte* Instanzen mit $\mu_{\min} > 200$.

Schluss

Das Postsche Korrespondenzproblem ist ein Erreichbarkeitsproblem. Bereits 1946 hat Emil Post bei seinem Beweis der Unentscheidbarkeit des „correspondence decision problem“ eine Reduktion auf ein Erreichbarkeitsproblem verwendet. (Post [33]) Er benutzt, dass es in Tag-Systemen mit Regeln der Form (l, r), die es erlauben aus dem Wort lu das Wort ur abzuleiten, unentscheidbar ist, ob aus einer Prämisse A ein Satz B abgeleitet werden kann.

Die natürliche Form eines Erreichbarkeitsproblems gibt einen Knoten in einem Graphen an und fragt, ob es einen Weg in diesem Graphen zu einem bestimmten anderen Knoten gibt. Das macht das verallgemeinerte Postsche Korrespondezproblem zum eigentlichen Postschen Korrespondenzproblem und tatsächlich hat Post zunächst mit dem verallgemeinerten Problem gearbeitet. Er zeigt nicht nur, wie aus den Regeln eines Tag-Systems eine Instanz des verallgemeinerten Korrespondenzproblems konstruiert werden kann, sondern auch, wie aus einer nichtlöschenden Instanz des verallgemeinerten Problems eine Instanz des herkömmlichen Problems konstruiert werden kann. Doch damit nicht genug, er bringt auf den insgesamt 5 Seiten auch noch unter, wie sich für nichtlöschende Instanzen das Zielalphabet auf ein Alphabet mit nur zwei Symbolen reduzieren lässt und er erwähnt, dass konkrete Instanzen manchmal anhand einfacher kombinatorischer oder einfacher algebraischer Argumente als unlösbar erkannt werden können.

Zwar hat Post damit schon fast alles gesagt, die Einfachheit und Natürlichkeit der Formulierung des Postschen Korrespondenzproblems und der große Erfolg als Hilfsmittel in verschiedenen Unentscheidbarkeitsbeweisen lenkten immer wieder die Aufmerksamkeit auf dieses Problem.

Es wurde intensiv untersucht, wie viele Regeln nötig sind, damit das Problem unentscheidbar ist. Das triviale Resultat lautet: Eine Regel genügt nicht. Ende der 1970er Jahre gelang mit Hilfe von Konstruktionen von Yuri Matiyasevich der Beweis, dass 10 Regeln genügen, später verbesserte Matiyasevich seine Konstruktion und seit Mitte der 1990er Jahre wissen wir, dass 7 Regeln genügen, bei verallgemeinerten Instanzen bereits 5 Regeln. Es erscheint nur logisch, dass Matiyasevich sich eigentlich mit der Erreichbarkeit

in Semi-Thue-Systemen beschäftigte. Er konnte zeigen, dass es Semi-Thue-System mit drei Regeln gibt, für die die Erreichbarkeit unentscheidbar ist. Aus der Unentscheidbarkeit der Erreichbarkeit in Semi-Thue-Systemen mit k Regeln folgt die Unentscheidbarkeit des verallgemeinerten Korrespondenzproblems mit $k+2$ und des herkömmlichen Korrespondenzproblems mit $k+4$ Regeln.(Matiyasevich, Sénizergues [24, 25], Claus [3], Nicolas [32])

Andererseits war lange Zeit unklar, ob es einen Algorithmus gibt, der Instanzen mit zwei Regeln klassifizieren kann. Erst Anfang der 1980er Jahre wurde gezeigt, dass das tatsächlich der Fall ist. (Ehrenfeucht, Karhumäki, Rozenberg [5]) Dieser erste Beweis konstruiert bereits eine Form eines Nachfolgers einer Instanz und schließt eine umfangreiche Fallunterscheidung an. Wesentlich wird benutzt, dass sich die Lösbarkeit binärer Instanzen auf die Lösbarkeit binärer markierter allgemeiner Instanzen zurückführen lässt. Später hat vor allem Vesa Halava diese Ansätze ausgebaut und er konnte schließlich zeigen, dass alle markierten Instanzen entscheidbar sind und sogar, dass eine echte Obermenge der markierten Instanzen, die der Instanzen mit markiertem Nachfolger, ebenfalls entscheidbar ist. (Halava et al. [8, 10, 11, 12]) Die Entscheidbarkeit von $\mathcal{G}(2)$ folgt aus der Entscheidbarkeit markierter Instanzen.

Auch andere Klassen wurden untersucht, zum Beispiel hat Keijo Ruohonen Mitte der 1980er Jahre gezeigt, dass die Morphismen auch injektiv und sogar total synchronisiert sein dürfen, ohne zu einer entscheidbaren Klasse zu kommen. (Ruohonen [36]) Kürzlich haben Vesa Halava und Mitautoren gezeigt, dass im Allgemeinen Instanzen unentscheidbar sind, deren sämtlichen h- und g-Bilder höchstens die Länge 2 haben. (Halava et al. [9])

a

In dieser Arbeit geben wir eine einheitliche und umfassende Darstellung, welche Möglichkeiten es gibt, Instanzen zu klassifizieren. Wir gehen dabei stets von allgemeinen Instanzen aus und machen fallweise klar, welche Vereinfachung durch Beschränkung auf herkömmliche Instanzen erzielt werden können. Darüber hinaus lassen wir stets auch löschende Instanzen zu. Es stellt sich heraus, dass sich sämtliche Formulierungen in konsistenter Weise für allgemeine und für löschende Instanzen vornehmen lassen. Die einzige echte Sonderrolle spielen Instanzen, die eine Regel der Form $(\varepsilon, \varepsilon)$ enthalten. Für solche Instanzen ist die Zerlegung eines Elements der Koinzidenzmenge nicht mehr eindeutig, eine Tatsache, die keine weiteren Konsequenzen hat.

Einfache kombinatorische und einfache algebraische Methoden werden in 18c abgeleitet, angewendet, miteinander kombiniert und dabei an ihre Grenzen geführt. Diese Methoden sind in der Lage, sehr schnell viele Instanzen zu klassifizieren oder zumindest die Regelanzahl zu reduzieren. Als Höhepunkt

gestattet es in 26a eine Kombination aus kombinatorischen und algebraischen Methoden, die Gleichheitsmenge periodischer Instanzen effektiv zu konstruieren. Es handelt sich um die erste kompakte Darstellung dieser Methoden, frühere Arbeiten haben dieses Feld nicht systematisch oder nicht konsequent bearbeitet. Jeder Klassifizierungsversuch sollte zunächst die Möglichkeiten dieser Methoden ausschöpfen.

Kommen wir zurück zur Sicht auf das Postsche Korrespondenzproblem als ein Erreichbarkeitsproblem. In 27a klären wir die Frage, was es heißt, nach einer Lösung zu suchen und definieren den Suchgraphen einer Instanz. Das ist ein Graph, in dem es genau dann einen Pfad von s nach $\overline{f}$ gibt, wenn die Instanz Lösungen hat. Die Transitionen dieses Graphen spiegeln das natürliche Vorgehen beim Suchen nach einer Lösung wider: Wenn p ein Präfix einer Lösung ist, dann sind $h(p)$ und $g(p)$ vergleichbar und pa ist Präfix einer Lösung, wenn $h(pa)$ und $g(pa)$ vergleichbar sind. Merken wir uns nur den Überhang von $(h(p), g(p))$, dann definieren wir auf diese Art eine Relation über $B^{*\updownarrow} \times A^* \times B^{*\updownarrow}$, die wir als Suchgraphen auffassen. Wir untersuchen zunächst, wie die Struktur dieser Relation erkannt und in ihrer Komplexität verringert werden kann, immer mit dem Ziel, den Suchgraphen zu verkleinern. Dabei betrachten wir sowohl Methoden, die die Pfadbeschriftungen benutzen, als auch Methoden, die die Inhalte der Knoten verarbeiten. Auch hier gibt es eine Fülle an Möglichkeiten, im Gegensatz zu den einfachen Methoden aus 18c handelt es sich nun aber häufig um Methoden, die neben der Instanz noch weitere Eingaben erwarten, zum Beispiel eine Maximalzahl von Knoten in einem zu bearbeitenden Teilgraphen des Suchgraphen. Das heißt, dass je nach Wahl der Parameter mehr oder weniger Instanzen klassifiziert werden können.

Die Chance, dass eine zufällig gewählte Instanz nicht durch eine der bis hierher betrachteten Methoden entschieden werden kann, ist bereits sehr gering. Methoden wie das Einfangen aus 39b verwenden sehr leistungsfähige Heuristiken und sind in der Lage starke Vereinfachung in Suchgraphen zu erzielen. Die Tabelle 105a zeigt, dass es schon mit einem Teil der Methoden gelingt, alle Instanzen aus $\mathcal{P}(2)$ zu klassifizieren. Das heißt, es will nicht gelingen, komplizierte binäre Instanzen zu konstruieren. Ein alternativer Beweis der Entscheidbarkeit von $\mathcal{P}(2)$ ist aber nicht direkt in Sicht, da die beteiligten Methoden sich gegenseitig beeinflußen und so zu doch erheblicher Komplexität beitragen.

Jeder Instanz I ist eine interessante Menge zugeordnet, die Koinzidenzmenge $\mathbb{C}(I) = \{(u, v) \mid s_\uparrow h(u) f_\uparrow = s_\downarrow g(v) f_\downarrow\}$. Anders als bei der Gleichheitsmenge ist nun erlaubt, die Morphismen h und g auf verschiedene Eingaben anzuwenden. *a*

Als eines der Hauptresultate dieser Arbeit zeigen wir in Satz 56a, dass die Koinzidenzmenge eine rationale Relation ist und dass wir einen Transducer effektiv konstruieren können, der die Koinzidenzmenge akzeptiert. Auf dem Weg dorthin scheint es natürlich zu sein, die Suchrelation so zu verändern, dass auch dort unterschiedliche Eingaben für die beiden Morphismen möglich sind. Das machen wir in 46a, die dabei entstehende Relation ist zwar in der Lage, die Koinzidenzmenge zu beschreiben, sie ist ist aber im Allgemeinen unendlich groß und dadurch nicht geeignet, als Basis für die Überführungsfunktion eines endlichen Transducers zu dienen.

Also verkleinern wir diese Relation in 49a wieder. Wir fordern nun, dass ↑-Überhänge nur in der ↓-Position verlängert werden und umgekehrt. Als Ergebnis erhalten wir eine endliche Relation, die nun leider zu klein ist: Es kann Elemente in der Koinzidenzmenge geben, die keinem Pfad in der verkleinerten Relation entsprechen.

Die entscheidende Einsicht im Lemma 54e in 49b ist die, dass in der Vereinigung der endlichen Relation und der gedrehten endlichen Relation der gespiegelten Instanz diese fehlenden Pfade wieder vorhanden sind. Wir können nun diese Vereinigung direkt als Überführungsfunktion eines endlichen Transducers benutzen, der die Koinzidenzmenge akzeptiert. Wohlgemerkt für beliebige Instanzen, insbesondere für nichtmarkierte und für löschende.

Der Beweis dieser Aussage benutzt keine komplizierten Argumente. Es werden lediglich der enge Zusammenhang zwischen Instanz und gespiegelter Instanz und Zerlegungseigenschaften der beteiligten Relationen ausgenutzt, die alle quasi direkt aus den Definitionen folgen. Trotzdem können wir bereits hier die ersten Früchte ernten: Es ist bekannt, dass nicht nur entscheidbar ist, ob die Koinzidenzmenge leer ist, sondern auch, ob gewisse bedingte Koinzidenzmengen leer sind. Die Bedingungen an die Paare (u, v) sind dabei zum Beispiel, dass u und v gleiche Länge haben sollen, oder dass u und v das gleiche kommutative Bild haben sollen. (Greibach [7], Ibarra, Kim [17], Karhumäki [21]) Mit Hilfe von Satz 56a sind wir nicht nur in der Lage, diese Beweise mit einheitlicher Sprache in unserem Rahmen nachzuvollziehen, sondern wir können sogar Automaten effektiv konstruieren, die Zerlegungen von Elementen der bedingten Koinzidenzmengen akzeptieren und geeignet sind, die bedingten Koinzidenzmengen effektiv aufzuzählen. In den bisherigen Konstruktionen werden Automaten konstruiert, die Bilder von Elementen der Koinzidenzmenge akzeptieren.

a Um die Entscheidbarkeit von $\mathcal{G}(2)$ zu zeigen, wird der Umweg über die Entscheidbarkeit markierter Instanzen genommen, die ihrerseits gezeigt wird als Folgerung über Eigenschaften eines Graphen von Nachfolgerinstanzen. Der

Begriff des Nachfolgers wurde in früheren Arbeiten nur für markierte Instanzen definiert und es wurde nicht untersucht, welche allgemeineren Prinzipien hinter diesem Begriff stecken. Diese Lücke wird nun geschlossen. Wir definieren in 63b den Begriff des Nachfolgers einer Instanz erstmals umfassend für beliebige Instanzen. Es handelt sich dabei um Instanzen, deren Regeln Grundblöcke der Koinzidenzmenge der Originalinstanz sind.

Wir zeigen zunächst, dass und wie sich Elemente der Koinzidenzmenge in Blöcke zerlegen lassen. Dank des Transducers für die Koinzidenzmenge sind wir in der Lage, alle beteiligten Mengen von Grundblöcken effektiv aufzuzählen. Die Lösbarkeit von Instanz und Nachfolger hängen eng zusammen und darüber hinaus lassen sich Aussagen über die Anzahl der Blöcke in der Zerlegung einer Koinzidenz aus $\mathrm{id}(\mathrm{Eq}(I))$ machen.

Das alles zusammen führt auf die Entscheidbarkeit markierter Instanzen und von herkömmlichen Instanzen, die nur markierte Nachfolger haben. Die wesentliche Eigenheit markierter Instanzen ist, dass der Graph der Nachfolger nur markierte Instanzen enthält und endlich ist. Aus allgemein gültigen Eigenschaften von Nachfolgern folgt dann, dass eine markierte Instanz genau dann lösbar ist, wenn es in ihrem Nachfolgergraph eine Instanz gibt, die eine Lösung der Länge 1 hat. Die hier präsentierten Beweise gehen über die bisher bekannten in verschiedenen Punkten hinaus: Sie vermischen nicht die Definition des Begriffs Nachfolger mit dem Beweis der Entscheidbarkeit markierter Instanzen, sie verwenden die Markiertheit nur an wenigen klar zu identifizierenden Stellen und sie zeigen viele der verwendeten Eigenschaften auf der allgemeinen Ebene. Darüber hinaus erlaubt unsere Herangehensweise die automatische Klassifikation von Instanzen, die nicht markiert sind und die auch mit keiner bekannten Technik automatisch klassifiziert werden können.

Der endgültige Beweis der Entscheidbarkeit von $\mathcal{G}(2)$ konstruiert für jede binäre Instanz eine markierte binäre Instanz, die genau dann lösbar ist, wenn die Originalinstanz lösbar ist. Auch dieser letzte Beweisschritt wird hier allgemeiner formuliert und ist gleichzeitig leichter verständlich.

Insgesamt bleibt neben den einfachen kombinatorischen und einfachen algebraischen Techniken nur das Suchen in seiner allgemeinen Form. Tatsächlich basiert die Konstruktion des Transducers für die Koinzidenzmenge auf einer speziellen Form des Suchens. Beim Übergang zum Nachfolger fassen wir Beschriftungen von Schleifen in diesem Transducer zusammen und benutzen sie als neue Regeln. Das heißt, wir können den liberalisierten Transducer auch als Netz von *Instanzen* betrachten. Jeder Zustand entspricht einer Instanz mit den Regeln, die den Schleifen um diesen Zustand entsprechen. Neben der Anwendung dieser Regeln ist auch erlaubt, Regeln anzuwenden, die ausgehen- *a*

de Transitionen dieses Zustands sind. Dann wird zu der Instanz gewechselt, die am Ende der ausgehenden Transition steht. Falls der Transducer nur einen Zustand hat, also im Fall endlicher Blockmengen, dann entspricht diese Vorstellung der normalen Vorstellung einer einzelnen Instanz.

Wir haben also einen Transducer erzeugt, der die Koinzidenzmenge akzeptiert. Unser Ziel war, Vereinfachung zu erzielen, genauer, die Suche nach einer Lösung in Superschritten ablaufen zu lassen. Bekommen haben wir eine Verallgemeinerung des Ausgangsproblems: Wir haben ein Erreichbarkeitsproblem auf ein anderes Erreichbarkeitsproblem zurückgeführt.

a

Probleme

Die meisten Arbeiten über das Postsche Korrespondenzproblem stellen das

b **Problem:** *Welche der folgenden Mengen sind entscheidbar:* $\mathcal{P}(3)$*,* $\mathcal{P}(4)$*,* $\mathcal{P}(5)$*,* $\mathcal{P}(6)$*,* $\mathcal{G}(3)$ *und* $\mathcal{G}(4)$*?*

Auch hier haben wir keine neuen Antworten auf diese Frage gefunden. Die Möglichkeit des exponentiellen Wachstum der Länge der minimalen Lösung im Verhältnis zur Weite der Instanz in $\mathcal{G}(4)$ und $\mathcal{P}(6)$ deutet darauf hin, dass diese Klassen unentscheidbar sind, oder ein Entscheidbarkeitsbeweis extrem schwer zu finden ist.

In dieser Arbeit haben wir neben den markierten auch weitere Instanzen mit endlicher Blockmenge gesehen. Es ist allerdings keine Charakterisierung solcher Instanzen bekannt. Es stellt sich das

c **Problem:** *Welche Instanzen haben endliche Blockmengen? Welche Instanzen haben endliche Nachfolgergraphen?*

Aber auch Instanzen mit unendlichen Blockmengen können machmal durch Balanceargumente entschieden werden. Doch es bleibt das

d **Problem:** *Wie funktioniert Balancieren auf unendlichen Blockmengen?*

Schließlich gibt es die Diskrepanz zwischen der Tatsache, dass es keine schwierigen binären Instanzen zu geben scheint und dem doch recht komplexen Beweis der Entscheidbarkeit binärer Instanzen. Darüber hinaus geben Experimente Anlass zu der Vermutung, dass die Länge der minimalen Lösung binärer Instanzen nur linear bezüglich der Weite wachsen kann. Der Autor vermutet sogar eine konkrete Grenze und stellt das

e **Problem:** *Gibt es eine Instanz in* $\mathcal{P}(2)$*, deren kürzeste Lösung länger als* $\max\{1, ||h(a)| - |g(a)|| + ||h(b)| - |g(b)||\}$ *ist?*

Literatur

Angegeben sind auch die Nummern der Seiten, auf denen referenziert wird.

[1] J. Berstel. *Transductions and Context-Free Languages.* Teubner, 1979. *(11, 66)*

[2] C. Choffrut, J. Karhumäki. *Combinatorics of Words.* In: Handbook of Formal Languages, I:329–438, edited by G. Rozenberg and A. Salomaa, Springer: Berlin, 1997. *(9, 11, 108)*

[3] V. Claus. *Some Remarks on PCP(k) and Related Problems.* Bulletin EATCS 12:54–61, 1980. *(16, 122)*

[4] A. Ehrenfeucht, J. Karhumäki, G. Rozenberg. *On Binary Equality Sets and a Solution to the Test Set Conjecture in the Binary Case.* Journal of Algebra 85(1): 76–85, 1983. *(95)*

[5] A. Ehrenfeucht, J. Karhumäki, G. Rozenberg. *The (Generalized) Post Correspondence Problem with Lists Consisting of Two Words is Decidable.* TCS 21(2):119–144, 1982. *(2, 46, 92, 122)*

[6] M. R. Garey, D. S. Johnson. *Computers and Intractability.* W. H. Freeman: New York, 1979. *(27)*

[7] S. A. Greibach. *A Remark on Code Sets and Context-Free Languages.* IEEE Transactions on Computers C-24(7):741–742, 1975. *(46, 58, 124)*

[8] V. Halava, T. Harju, M. Hirvensalo. *Generalized Post Correspondence Problem for Marked Morphisms.* IJAC 10(6):757–772, 2000. *(2, 46, 92, 122)*

[9] V. Halava, T. Harju, M. Hirvensalo, J. Karhumäki. *Post Correspondence Problem for Short Words.* IPL: `http://dx.doi.org/10.1016/j.ipl.2008.04.013`, 2008, Im Druck. *(122)*

[10] V. Halava, T. Harju, J. Karhumäki, M. Latteux. *Extension of the Decidability of the Marked PCP to Instances with Unique Blocks.* TCS 380(3):355–362, 2007. *(3, 46, 87, 88, 122)*

[11] V. Halava, M. Hirvensalo, R. de Wolf. *Marked PCP is Decidable.* TCS 255:193–204, 2001. *(2, 46, 80, 92, 122)*

[12] V. Halava. *The Post Correspondence Problem for Marked Morphisms.* TUCS Dissertation (37), 2002. *(2, 15, 46, 79, 81, 85, 89, 92, 94, 122)*

[13] T. Harju, J. Karhumäki. *Morphisms.* In: Handbook of Formal Languages, I:439–510, edited by G. Rozenberg and A. Salomaa, Springer: Berlin, 1997. *(29, 95)*

[14] Š. Holub. *Binary Equality Sets are Generated by Two Words.* Journal of Algebra 259(1): 1–42, 2003. *(95)*

[15] Š. Holub, V. Halava. *Binary (Generalized) Post Correspondence Problem is in P.* TUCS Technical Report 785, 2006. *(99)*

[16] J. E. Hopcroft, J. D. Ullmann. *Einführung in die Automatentheorie, Formale Sprachen und Komplexitätstheorie.* Oldenbourg: München, Wien, 2000. *(14)*

[17] O. H. Ibarra, C. E. Kim. *A Useful Device for Showing the Solvability of Some Decision Problems.* JCSS 13(2):153–160, 1976. *(46, 59, 61, 124)*

[18] O. H. Ibarra, B. Ravikumar. *On Partially Blind Multihead Finite Automata.* TCS 356: 190–199, 2006. *(60)*

[19] O. H. Ibarra. *Reversal-Bounded Multicounter Machines and Their Decision Problems.* JACM 25(1):116–133, 1978. *(60)*

[20] S. P. Jones et al. *Haskell 98 Language and Libraries: The Revised Report.* JFP 13(1), 2003. *(100)*

[21] J. Karhumäki. *Generalized Parikh Mappings and Homomorphisms.* Information and Control 47(3):155–165, 1980. *(46, 62, 124)*

[22] R. L. Lorentz. *Creating Difficult Instances of the Post Correspondence Problem.* CG 2000, LNCS 2063:214–228, Springer: Berlin, Heidelberg, 2001. *(97)*

[23] M. Margenstern. *Frontier Between Decidablity and Undecidability: A Survey.* TCS 231(2): 217–251, 2000. *(2, 17)*

[24] Y. Matiyasevich, G. Sénizergues. *Decision Problems for Semi-Thue System With a Few Rules.* LNCS 96, IEEE Computer Society Press, Silver Spring: 523–531, 1996. *(2, 122)*

[25] Y. Matiyasevich, G. Sénizergues. *Decision Problems for Semi-Thue System With a Few Rules.* TCS 330(1): 145–169, 2005. *(2, 122)*

[26] F. Mattern. *Verteilte Basisalgorithmen.* Informatik Fachberichte (226), Springer, 1989. *(117)*

[27] M. L. Minsky. *Computation: Finite and Infinite Machines.* Englewood Cliffs: Prentice-Hall, 1967. *(1)*

[28] M. Mohri. *Minimization Algorithms for Sequential Transducers.* TCS 234: 177–201, 2000. *(66)*

[29] L. De Mol. *Tag-Systems and Collatz-like Functions.* TCS 390(1): 92–101, 2008. *(17)*

[30] L. De Mol. *Tracing Unsolvability: A Mathematical, Historical and Philosophical Analysis with a Special Focus on Tag-Systems.* PhD Thesis, Universität Gent, 2007. *(1)*

[31] *MPI-Standard 1.1*, `http://www.mpi-forum.org`. [15.12.2008] *(117)*

[32] F. Nicolas. *(Generalized) Post Correspondence Problem and Semi-Thue Systems.* `http://arxiv.org/abs/0802.0726`, 2008. *(16, 122)*

[33] E. L. Post. *A Variant of a Recursively Unsolvable Problem.* Bulletin AMS 52:264–268, 1946. *(2, 13, 121)*

[34] E. L. Post. *Formal Reductions of the General Combinatorial Decision Problem.* AJM 65(2):197–215, 1943. *(1)*

[35] M. Rahn. *More Decidable Instances of Post's Correspondence Problem: Beyond Counting.* IPL 106(3):115–119, 2008. *(24)*

[36] K. Ruohonen. *Reversible Machines and Post's Correspondence Problem for Biprefix Morphisms.* EIK 21(12):579–595, 1985. *(14, 122)*

[37] P. Sanders. *Lastverteilungsalgorithmen für parallele Tiefensuche.* Fortschrittsberichte VBI 10(463), 1997. *(117)*

[38] P. Sanders, T. Worsch. *Parallele Programmierung mit MPI: Ein Praktikum.* Logos: Berlin, 1997. *(117)*

[39] M. Schmidt. *Komplexität von Instanzenfamilien des Postschen Korrespondenzproblems.* Diplomarbeit, Universität Leipzig, 2006. *(3, 16, 98)*

[40] H. Stamer. *Projekt PCP@HOME.* `http://www.theory.informatik.uni-kassel.de/~stamer/pcp/` [15.12.2008] *(98)*

[41] T. Ungerer. *Parallelrechner und parallele Programmierung.* Spektrum, Akademie: Berlin, Heidelberg, 1997. *(117)*

[42] J. Waldmann, M. Schmidt, H. Stamer. *Busy Beaver PCPs.* Vortrag auf dem „5th International Workshop on Termination“, Utrecht, 2001. *(98)*

[43] S. Wolfram. *A New Kind of Science.* Wolfram Media, 2002. *(3)*

[44] L. Zhao. *Solving and Creating Difficult Instances of Post's Correspondence Problem.* Master Arbeit, Universität Alberta, 2002. *(31, 97, 101, 109, 120)*

Symbole

Angegeben ist die Nummer der Seite, auf der das Symbol definiert wird.

Monoid
- $\cdot^{-1}$ inverses Element 8
- e Einselement 8
- $\cdot^{\circ}$ $M^{\circ} = M \setminus \{e\}$ 8
- $\cdot^{i}$ i-te Potenz 8
- $\cdot^{\leq i}$ Potenzen bis i-te 8
- $\cdot^{>i}$ Potenzen ab $(i+1)$-ter 8
- $\cdot^{*}$ Hulle 8
- $\cdot^{+}$ positive Hulle 8

Morphismus
- $\cdot^{\omega}$ Fixpunkt 11
- P gemeinsame Prafixe 91
- S Suffixmenge 10
- $\cdot^{\mathrm{R}}$ gespiegelt 10
- τ $\{0 \mapsto 01, 1 \mapsto 10\}$ 11
- wd() Weite 82

Paar
- $\oplus_{\mathrm{p}}$ $x \oplus_{\mathrm{p}} y \in \mathrm{id}^{-1} xy$ 93
- $\oplus_{\mathrm{s}}$ $x \oplus_{\mathrm{s}} y \in xy\,\mathrm{id}^{-1}$ 93
- $\left[\cdot\right]$ $\left[\begin{smallmatrix} x \\ y \end{smallmatrix}\right] = (x, y)$ 7
- $\bar{\cdot}$ vertauscht 7
- $\cdot_{\uparrow}$ erste Komponente 7
- $\cdot_{\downarrow}$ zweite Komponente 7

Paarmenge
- $\cdot^{\uparrow}$ $X^{\uparrow} = X \times \{\varepsilon\}$ 8
- $\cdot^{\downarrow}$ $X^{\downarrow} = \{\varepsilon\} \times X$ 8
- $\cdot^{\updownarrow}$ $X^{\updownarrow} = X^{\uparrow} \cup X^{\downarrow}$ 8

Relation
- $\cdot^{\star}(\cdot)$ Hulle 8
- id Identitat 8
- N Knoten 8

Suchrelation
- $\circlearrowleft$ gedreht 54
- $\rightarrowtail$ normal 28
- $\hookrightarrow$ allgemein 46
- $\hookrightarrow$ beschrankt 49
- $\leftrightharpoons$ beschrankt bidirektional 54

Teilwort
- $\preceq_{\mathrm{p}}$ ist Prafix 9
- $\asymp_{\mathrm{p}}$ prafixvergleichbar 9
- $\preceq_{\mathrm{s}}$ ist Suffix 9
- $\asymp_{\mathrm{s}}$ suffixvergleichbar 9

Teilwortmenge
- inf Infixe 8
- pref Prafixe 8
- suff Suffixe 8

Vektor
- span aufgespannter Raum 7
- $\langle\cdot|\cdot\rangle$ Skalarprodukt 7

Wort
- $|\cdot|$ Lange 1
- ε leer 10
- ψ kommutatives Bild 10
  - ψ_k allgemeines 62
- $\cdot^{\mathrm{R}}$ gespiegelt 10

Wortmenge
- $\complement$ Komplement
- $\triangleleft$ linker Quotient 10
- $\triangleright$ rechter Quotient 10

Zahlenmenge
- $\mathbb{N}$ naturliche
- $\mathbb{Z}$ ganze

Beispiele

Beispielinstanzen, die an mehreren Stellen verwendet werden.

Stichwörter

Selbstständigkeitserklärung

Ich erkläre, dass ich diese Arbeit selbstständig angefertigt und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt sowie die wörtlich oder inhaltlich übernommenen Stellen als solche kenntlich gemacht und die Satzung der Universität Karlsruhe (TH) zur Sicherung guter wissenschaftlicher Praxis in der gültigen Fassung vom 15. Januar 2002 beachtet habe.

Mirko Rahn, Karlsruhe, den 15. Dezember 2008

Lebenslauf von Mirko Rahn

Geburtsdatum	20. Juli 1973
Nationalität	deutsch
Familienstand	verheiratet mit Juliane Landmann

Zuletzt

03/2005–	wissenschaftlicher Mitarbeiter am IAKS Vollmar an der Universität Karlsruhe (TH) Schwerpunkte: Parallelverarbeitung, insbesondere MPI, Postsches Korrespondenzproblem, Betreuung von Übungen und Praktika

Qualifikation

04/2002–12/2004	Diplomstudiengang an der Universität Karlsruhe (TH) Hauptfach: Informatik, Nebenfach: Geschichte der Philosophie und Logik
10/1999–03/2002	Diplomstudiengang an der Universität Leipzig Hauptfach: Informatik, Nebenfach: Logik
09/1988–06/1992	Gymnasium „Wilhelm Ostwald“ in Leipzig

Arbeit und Praktika

mehrfach	wissenschaftliche Hilfskraft an der Universität Karlsruhe (TH), Lehrstuhl Informatik für Naturwissenschaftler und Ingenieure (Prof. R. Vollmar) Schwerpunkte: Zellularautomaten, speziell der Automat „Regel 110“, Betreuung von Praktika
ein Jahr	wissenschaftliche Hilfskraft an der Universität Leipzig, Lehrstuhl für Theoretische Informatik (Prof. S. Gerber) Schwerpunkte: Betreuung und Entwicklung des Systems „autotool“ zur automatischen Korrektur von Übungsaufgaben
04/1994–09/1999	Angestellter beim „Projekt Verein e. V.“ (Conne Island) in Leipzig Schwerpunkte: Betreuung, Entwicklung und Ausbau der Infrastruktur zur Informationsverarbeitung, Herausgabe und Produktion von Monatszeitung, Programm und Werbematerial, Organisation und Durchführung von Veranstaltungen

Hobby

Radfahren im Wald

Havannah (Brettspiel für zwei Personen)