Komplexität von Suchproblemen und Beweissystemen

Anton Ehrmanntraut

31. Oktober 2023

Inhaltsverzeichnis

1 Einleitung

Viele andere Beweise der Komplexitätstheorie funktionieren auch in relativierten Umgebungen, also wenn alle beteiligten TM zu OTM ausgewechselt werden. Das Diagonalargument in einem typischen Beweis von $\mathrm{P} \subsetneq \mathrm{E}$ relativiert beispielsweise, sodass auch $\mathrm{P}^O \subsetneq \mathrm{E}^O$ für jedes beliebige Orakel O gilt. Wir sagen dann auch, dass diese Aussagen bzw. Beweise *relativieren*.

Die typischen Beweistechniken der Komplextätstheorie relativieren. Das macht eigens konstruierte Orakel zu Indizien, dass gewisse Aussagen schwer zu beweisen sind. Beispielsweise konstruieren Baker, Gill und Solovay (1975) ein Orakel A sodass $\mathrm{P}^A \neq \mathrm{NP}^A$. Mit diesem Fakt ist die Aussage „$\mathrm{P} = \mathrm{NP}$“ nicht mit relativierbaren Methoden beweisbar, da sonst ja auch $\mathrm{P}^A = \mathrm{NP}^A$ gelten würde.

2 Grundlagen

Dieses Kapitel legt die definitorischen Grundlagen für die folgenden Kapitel fest. In Abschnitt 2.1 legen wir mathematische Notationen für diese Arbeit fest. Abschnitt 2.2 spezifiziert das Maschinenmodell. Abschnitt 2.3 wiederholt einige Standarddefinitionen aus der Komplexitätstheorie. Abschnitt 2.4 setzt das hier verwendete Verständnis von Relativierungen fest. Abschließend geht Abschnitt 2.5 kurz auf Beweissysteme im Sinne von Cook und Reckhow (1979) ein.

2.1 Notation

Sei Σ das standardmäßige Alphabet mit $\Sigma = \{0, 1\}$. Elemente von Σ^* nenenn wir Wörter, sind also endliche Sequenzen von Zeichen aus Σ. Teilmengen von Σ^* nennen wir auch Sprachen. Wir bezeichnen die Länge eines Wortes $w \in \Sigma^*$ mit $|w|$. Das leere Wort bezeichnen wir mit ε. Das i-te Zeichen eines Wortes w für $0 \leq i < |w|$ identifizieren wir mit $w[i]$. Diese Notation erweitern wir auf Sequenzen von Indizes: für $0 \leq i_1, i_2, \ldots, i_k < |w|$ und $\alpha = (i_1, i_2, \ldots, i_k)$ sei $w[\alpha] = w[i_1]w[i_2]\cdots w[i_k]$. Insbesondere ist damit $w[0, 1, 2, \ldots, |w| - 1] = w$. Falls w ein (echter) Präfix von w ist dann schreiben wir $w \sqsubseteq v$ (bzw. $w \sqsubsetneq v$).

Die Menge aller natürlichen (nicht-negativen) Zahlen wird mit $\mathbb{N}$ bezeichnet. Die leere Menge notieren wir wie üblich als $\emptyset$. Die Kardinalität einer Menge A notieren wir wie üblich als $|A|$. Für eine Menge $A \subseteq \Sigma^*$ und $n \in \mathbb{N}$ definieren wir $A^{\leq n} = \{w \in A \mid |w| \leq n\}$. Analog definieren wir $A^{<n}, A^{=n}$, usw. Außerdem bezeichnet $\ell(A) = \sum_{w \in A} |w|$. Für solche Teilmengen A von Σ^* verstehen wir das Komplement $\overline{A}$ als $A\ \Sigma^*$.

Zweistellige bzw. binäre Relationen $R \subseteq A \times B$ können wir mit den üblichen Eigenschaften beschreiben: die Relation R ist

- *(links-)total* wenn jedes Element aus A mit mindestens einem Element aus B reliert,
- *rechtstotal* bzw. *surjektiv* wenn jedes Element aus B mit mindestens einem Element aus A reliert,
- *linkseindeutig* bzw. *injektiv* wenn jedes Element aus B mit höchstens einem Element aus A reliert,
- *(rechts-)eindeutig* bzw. *funktional* wenn jedes Element aus A mit höchstens einem Element aus B reliert,
- *bijektiv* wenn jedes Element aus A mit genau einem Element aus B reliert und umgekehrt, also genau dann wenn R funktional, surjektiv und injektiv ist.

Binäre Relationen nennen wir eine (partielle) *Funktion* wenn diese Relation funktional ist. Eine Funktion sei also im Folgenden im Allgemeinen nicht total. Sollte (Links-)Totalität explizit gefordert sein, sprechen wir von *totalen Funktionen*. Binäre Relationen über Wörtern aus Σ^*, welche nicht unbedingt Funktionen sind, verstehen wir manchmal auch aus historischen Gründen als (partielle) Multifunktionen, dem Begriff der „*partial multivalued function*“ nachempfunden.

Für eine binäre Relation $R \subseteq \Sigma^* \times \Sigma^*$ schreiben wir $\mathrm{Proj}(R)$ für die Menge $\{x \mid (x, y) \in R\}$. Für ein Wort $x \in \Sigma^*$ schreiben wir $\textit{set-}R(x) = \{y \mid (x, y) \in R\}$ für die Bildmenge von x auf R Manchmal werden wir binäre Relationen auch über die Spezifikation der jeweiligen Bildmengen definieren, also z.B. $\textit{set-}Q(n) = \{0, 1, \ldots, n\}$ schreiben um die Relation $Q = \{(a, b) \mid b \leq a\}$ zu definieren. Falls f eine Funktion bzw. funktional ist, meinen wir mit $f(x)$ wie üblich das *Bildelement* der Funktion f meinen, und nicht die *Bildmenge*.

Für eine Funktion f bezeichnen wir die Urbild- bzw. Bildmenge (domain und range) mit $\mathrm{dom}(f)$ und $\mathrm{ran}(f)$. (Beachte dass $\mathrm{Proj}(f) = \mathrm{dom}(f)$. Wir führen diese Unterscheidung nur wegen den Gewohnheiten dieser zwei Notationen ein.) Ist f eine injektive Funktion, dann bezeichnen wir mit f^{-1} dessen Umkehrfunktion. Beobachte dass f^{-1} funktional ist, ist ja f injektiv. Ist f sogar bijektiv, dann ist die Umkehrfunktion f^{-1} eine totale Funktion.

Eine Funktion $f\colon \Sigma^* \to \Sigma^*$ nennen wir *verlängernd* wenn $|f(x)| \geq |x|$ für alle $x \in \mathrm{dom}(f)$. Die Funktion f nennen wir *polynomiell längenbeschräkt* wenn ein Polynom p existiert sodass $|x| \leq p(|x|)$ für alle $x \in \mathrm{dom}(f)$. Die Funktion f nennen wir *ehrlich* wenn ein Polynom q existiert sodass $q(|f(x)|) \geq |x|$ für alle $x \in \mathrm{dom}(f)$.

Beachte dass Funktionen nur spezielle Relationen sind. Wenn also f eine Funktion ist, meinen wir mit „$f \in \mathrm{P}$“ dass *der Graph* von f in Polynomialzeit entschieden werden kann (i.e., gegeben Tupel (x, y), gilt $f(x) = y$?). Das ist eine schwächere Aussage als „$f \in \mathrm{FP}$“ die wie in üblicher Interpretation besagen soll, dass aus x das Bild $f(x)$ in Polynomialzeit berechnet werden kann.

Im Folgenden definieren wir noch den Begriff der *Verfeinerung*. Seien F, G zwei Multifunktionen. Wir nennen G eine *Verfeinerung* von F wenn $\mathrm{Proj}(F) = \mathrm{Proj}(G)$ und $\mathit{set}\text{-}G(x) \subseteq \mathit{set}\text{-}F(x)$ für alle $x \in \mathrm{Proj}(F)$ (bzw. äquivalent $\in \mathrm{Proj}(G)$). Ist F eine Multifunktion, und $\mathcal{G}$ eine Klasse von Multifunktioen, schreiben wir $F \in_{\mathrm{c}} \mathcal{G}$ wenn $\mathcal{G}$ eine Verfeinerung $G \in \mathcal{G}$ von F enthält. Für zwei Klassen $\mathcal{F}, \mathcal{G}$ von Multifunktionen schreiben wir $\mathcal{F} \subseteq_{\mathrm{c}} \mathcal{G}$ falls für jede Multifunktion $F \in \mathcal{F}$ auch $F \in_{\mathrm{c}} \mathcal{G}$ gilt.

Die endlichen Wörter Σ^* können über ihre quasi-lexikographischen Ordnung $\prec_{\mathrm{lex}}$ linear geordnet werden. Diese ist eindeutig definiert indem wir $0 \preceq_{\mathrm{lex}} 1$ fordern. Unter dieser Definition existiert ein Ordnungsisomorphismus zwischen $(\Sigma^*, \preceq_{\mathrm{lex}})$ und $(\mathbb{N}, <)$, welcher insbesondere eine Bijektion zwischen Σ^* und $\mathbb{N}$ induziert, der sowohl in Polynomialzeit berechenbar als auch invertiertbar ist. (Eine solcher Isomorphismus wird zum Beispiel durch eine dyadische Codierung realisiert.) Durch diese Identifikation können wir Wörter aus Σ^* als Zahlen aush $\mathbb{N}$ behandeln und umgekehrt. Es könenn also auch Notationen, Beziehungen und Operationen für Σ^* auf $\mathbb{N}$ übertragen werden und umgekehrt. Insbesondere können wir dann von einer Länge $|n|$ des Wortes sprechen, welches von $n \in \mathbb{N}$ repräsentiert wird. Insbesondere meint dieser Ausdruck nicht den Betrag von n. Ebenso bezeichnet die Ordnung $\leq$ sowohl die Kleiner-oder-gleich-Ordnung auf den natürlichen Zahlen als auch der quasi-lexikographischen Ordnung $\prec_{\mathrm{lex}}$ auf den endlichen Wörtern. Diese Übereinstimmung ist nach den Eigenschaften des Ordnungsisomorphismus auch kompatibel mit der Identifikation von Wörtern mit Zahlen. Beachte dass der Längenoperator $|\cdot|$ ordnungserhaltend ist: wenn $a \leq b$ (oder eben äquivalent $a \preceq_{\mathrm{lex}} b$) für zwei Wörter $a, b \in \Sigma^*$ dann ist auch $|a| \leq |b|$, bzw. ist das Wort a höchstens so lang wie das Wort b. Mit den Ausdrücken 0^n und 1^n meinen wir immer die zwei Wörter $000\cdots$ und $111\cdots$ aus Σ^n.

Wir definieren mit $\langle\cdots\rangle$ eine Paarungsfunktion von $\bigcup_{i\geq 0}(\Sigma^*)^i \to \Sigma^*$, welche injektiv und in Polynomialzeit sowohl berechenbar als auch invertierbar ist, und die im folgenden Sinne längeneffizient ist: $|\langle u_1, \ldots, u_n\rangle| = 2(|u_1| + \cdots + |u_n| + n)$. Eine solche Paarungsfunktion kann beispielsweise über $\langle u_1, \ldots, u_n\rangle \mapsto f(\#u_1\#\cdots\#u_n)$ realisiert werden, wobei f eine Codierung vom Alphabet $\{0, 1, \#\}$ auf Σ^* mittels $\{0 \mapsto 00, 1 \mapsto 11, \# \mapsto 01\}$ ist. Diese Paarungsfunktion werden wir häufig verwenden, um Tupel an Wörtern zu codieren, z.B. damit eine Turing-Maschine ein Tupel an Wörtern als Eingabe entgegen nehmen kann. Auf die konkrete Angabe dieser Paarungsfunktion wird aber im Folgenden meist verzichtet und sie wird nur implizit mitgedacht. So meinen wir mit dem Tupel (a, b) für $a, b \in \Sigma^*$ je nach Kontext entweder mathematisch präzise das Element aus dem Produkt $\Sigma^* \times \Sigma^*$, oder das Wort $\langle a, b\rangle \in \Sigma^*$. Ebenso verstehen wir je nach Kontext eine binäre Relation $R \subseteq \Sigma^* \times \Sigma^*$ auch als eine Sprache im Sinne einer Teilmenge von Σ^*, die bspw. von einer Turing-Maschine entschieden werden kann. Algorithmen und Turing-Maschinen verarbeiten nicht nur Wörter, sondern auch andere Objekte wie z.B. Graphen oder Turing-Maschinen. Daher werden wir die obige implizit mitgedachte Codierung auch auf andere Objekte ausweiten. Hierbei seien die jeweiligen Codierungen angemessen effizient, in dem Sinne dass die Codierungen kompakt sind und entsprechende Operationen auf den codierten Objekten in Polynomialzeit zulassen. Zum Beispiel lässt sich ein Graph mit Knotenmenge V und Kantenmenge E in polynomieller Länge abh. von $|V|$ und $|E|$ codieren, und auf der entsprechenden Codierung kann z.B. die Nachbarschaft eines ausgezeichneten Knotens ebenso in Polynomialzeit aufgezählt werden.

2.2 Maschinenmodell

Diese Arbeit baut auf dem Berechnungsmodell der Turing-Maschine (TM) auf. Wir betrachten hierbei sowohl die deterministische als auch die nichtdeterministische Variante. In dieser Arbeit haben TM sowohl ausgezeichnete Zustände zum Akzeptieren bzw. Ablehnen, ein Eingabeband, ein Arbeitsband, und ein Ausgabeband. Im Folgenden betrachten wir nur TM die immer terminieren. (Es ist einer TM im Allgemeinen nicht ansehbar, ob diese immer terminiert. Im Verlauf dieser Arbeit werden die TM aber so beschaffen sein, dass diese offensichtlich immer terminieren.)

Wir betrachten zunächst deterministische TM. Sei M eine deterministische TM, und x eine Eingabe. Dann induziert eine Berechnung $M(x)$ einen Rechenweg α, der in einem ausgezeichnetem Zustand q terminiert. Wir sagen dann auch, dass α *der Rechenweg von Berechnung* $M(x)$ ist. Wenn der terminierende Zustand q dieses Rechenwes α ein akzeptierender Zustand ist, dann sagen wir auch dass $M(x)$ *mit Ausgabe* y *akzeptiert* oder kurz $M(x)$ *akzeptiert* wobei y jenes Wort ist, welches auf dem Ausgabeband steht.

Eine solche deterministische TM M setzt nun gleichzeitig zwei unterschiedliche Funktionsweisen um. Einerseits die eines Akzeptors einer Menge, und andererseits die einer Funk-

tion:

- Die von M *entschiede Sprache* ist die Menge $L(M) = \{x \in \Sigma^* \mid M(x) \text{ akzeptiert}\}$.
- Die von M *berechnete Funktion* ist die Funktion $f_M \colon \Sigma^* \to \Sigma^*$ mit

$$f_M(x) = \begin{cases} y & \text{wenn } M(x) \text{ mit Ausgabe } y \text{ akzeptiert,} \\ \perp & \text{sonst.} \end{cases}$$

Wenn wir M im Kontext der zweiten Funktionsweise verstehen, dann sprechen wir auch von einem Turing-Transduktor. Wir kürzen dann auch „die von M berechnete Funktion“ durch „die Funktion M“ ab und verstehen den Turing-Transduktor M als genuine Funktion, und schreiben dann z.B. $M(x) = y$ anstelle $f_M(x) = y$.

Diese zwei Arten von Funktionsweisen einer TM erweitern wir nun auf nichtdeterministische TM. Sei N eine nichtdeterministische TM, und x eine Eingabe. Dann induziert analog eine Berechnung $N(x)$ nicht nur eine, sondern ggf. mehrere terminierende Rechenwege, die wir ebenso die Rechenwege von Berechnung $N(x)$ nennen. Terminiert ein solcher Rechenweg von $N(x)$ in einem akzeptierenden Zustand, nennen wir diesen Rechenweg auch einen *akzeptierenden Rechenweg*. Ähnlich wie im deterministischen Fall sagen wir dass $N(x)$ *auf Rechenweg α (mit Ausgabe y) akzeptiert* wenn α ein akzeptierender Rechenweg von $N(x)$ ist (und y auf dem Eingabeband steht). Beachte dass die Angabe eines Rechenwegs zwingend notwendig ist, da zu einer Berechnung $N(x)$ ja mehrere Rechenwege mit je unterschiedlichen Akzeptierverhalten und Ausgaben existieren. Im Sinne eines existentiellen Akzeptierverhaltens sagen wir dass $N(x)$ *akzeptiert* wenn *mindestens* ein akzeptierender Rechenweg α auf $N(x)$ existiert.

Analog ergeben sich nun wieder zwei Funktionsweisen, einerseits als Akzeptor, andererseits als Multifunktion:

- Die von N *entschiede Sprache* ist die Menge

$$L(N) = \{x \in \Sigma^* \mid N(x) \text{ akzeptiert } \} = \{x \in \Sigma^* \mid \text{ ex. akz. Rechenweg auf } N(x)\}.$$

- Die von N *berechnete Mulitfunktion* ist die Multifunktion $f_N \subseteq \Sigma^* \times \Sigma^*$ mit

$$f_N(x) = \{y \mid N(x) \text{ akz. auf einem Rechenweg mit Ausgabe } y\}$$

Die berechnete Mulitfunktion kann in anderen Worten auch so verstanden werden, dass x den Ausgaben von $N(x)$ zugeordnet wird, wobei jeder akzeptierender Rechenweg eine Ausgabe macht, nämlich das Wort was auf dem Ausgabeband steht. Wie im deterministischen Fall können wir von nichtdeterministischen Turing-Transduktoren sprechen, wenn wir die zweite Funktionsweise betonen wollen. Ebenso können wir wieder abkürzend von „der Multifunktion N“ sprechen.

In sowohl dem deterministischen und nichtdeterministischen Fall können wir Berechnungen eine *Laufzeit* zuordnen: für eine TM M sei

$$\text{time}_M(x) = \max\{\text{Anz. Rechenschritte in } \alpha \mid \alpha \text{ ist ein Rechenweg von } M(x)\}.$$

Ist $\text{time}_M(x)$ durch ein Polynom in Abhängigkeit von $|x|$ beschränkt, und M eine deterministische (bzw. nichtdeterministische) TM, sagen wir auch dass M eine *deterministische (bzw. nichtdeterministische) Polyomialzeit-Turing-Maschine* (PTM bzw. NPTM) ist.

2.3 Komplexitätsklassen

Auf Basis der Turing-Maschinen als Berechnungsmodell können die üblichen Komplexitätsklassen der Entscheidungsprobleme bzw. Sprachen $\mathrm{P}, \mathrm{NP}, \mathrm{coNP}$ usw. definiert werden:

$$\begin{aligned} \mathrm{P} &= \{L \subseteq \Sigma^* \mid \text{ex. PTM } M \text{ die } L \text{ entscheidet}\} \\ \mathrm{NP} &= \{L \subseteq \Sigma^* \mid \text{ex. NPTM } M \text{ die } L \text{ entscheidet}\} \\ \mathrm{coNP} &= \{L \subseteq \Sigma^* \mid \overline{L} \in \mathrm{NP}\} \end{aligned}$$

Die Einfach- und Doppelt-Exponentialzeitklassen definieren wir wie folgt:

$$\begin{aligned} \mathrm{E} &= \{L \subseteq \Sigma^* \mid \text{ex. TM } M \text{ die } L \text{ entscheidet, und ex. } c > 0 \text{ mit } \text{time}_M(x) \leq 2^{c|x|} \text{ für alle } x\} \\ \mathrm{EE} &= \{L \subseteq \Sigma^* \mid \text{ex. TM } M \text{ die } L \text{ entscheidet, und ex. } c > 0 \text{ mit } \text{time}_M(x) \leq 2^{2^{c|x|}} \text{ für alle } x\} \end{aligned}$$

Die nichtdeterministischen Varianten $\mathrm{NE}, \mathrm{NEE}$ und Komplementklassen $\mathrm{coNE}, \mathrm{coNEE}$ sind analog definiert.

Die Funktioneklassen FP, NPMV, NPSV ist analog definiert (Selman 1994):

$$\begin{aligned}\text{FP} &= \{f : \Sigma^* \to \Sigma^* \mid f \text{ ist eine Funktion und ex. PTM-Transduktor } M \text{ der } f \text{ berechnet}\}\\ \text{NPSV} &= \{f : \Sigma^* \to \Sigma^* \mid f \text{ ist eine Funktion und ex. NPTM-Transduktor } M \text{ der } f \text{ berechnet}\}\\ \text{NPMV} &= \{f \subseteq \Sigma^* \times \Sigma^* \mid f \text{ ist eine Multifunktion und ex. NPTM-Transduktor } M \text{ der } f \text{ berechnet}\}\end{aligned}$$

Wir defineren NPMV_t als die Teilmenge von NPMV der Multifunktionen, die linkstotal sind. Analog NPSV_t. Ist für eine injektive Funktion $f \in \text{FP}$ auch die Umkehrfunktion $f^{-1} \in \text{FP}$ sagen wir auch, dass f *p-invertierbar* ist. Beachte, dass die Ehrlichkeit von f eine notwendige Bedingung für die p-Invertierbarkeit von f ist.

Wie üblich können wir mittels Reduktionen die Sprachen der Komplexitätsklassen ordnen. Seien A, B zwei Sprachen:

- $A \leq^\text{p}_\text{m} B$ wenn eine Funktion $f \in \text{FP}$ existiert mit $x \in A \iff f(x) \in B$ (Many-one-Reduzierbarkeit).
- $A \leq^\text{p}_1 B$ wenn eine injektive Funktion $f \in \text{FP}$ existiert mit $x \in A \iff f(x) \in B$ (One-one-Reduzierbarkeit).
- $A \leq^\text{p}_{1,\text{i}} B$ wenn eine injektive und p-invertierbare Funktion $f \in \text{FP}$ existiert mit $x \in A \iff f(x) \in B$.

Für die Funktionenklassen hat sich folgender sehr starke Begriff von Many-one-Reduzierbarkeit herausgebildet (Köbler und Messner 2000; Beyersdorff, Köbler und Messner 2009; Pudlák 2017). Seien g, h zwei Multifunktionen:

- $g \leq^\text{p}_\text{m} h$ wenn eine Funktion $f \in \text{FP}$ existiert mit $\textit{set-}g(x) = \textit{set-}h(f(x))$.

Jede dieser Ordnungsrelationen ist eine Quasiordnung, i.e. reflexiv und transitiv. Beachte, dass (auf Mengen) $\leq^\text{p}_{1,\text{i}}$ feiner als $\leq^\text{p}_1$ ist, und diese feiner als $\leq^\text{p}_\text{m}$ ist. Beachte auch, dass P und NP auf $\leq^\text{p}_\text{m}$ nach unten abgeschlossen sind:

$$\begin{aligned}A \leq^\text{p}_\text{m} B \text{ und } B \in \text{NP} &\implies A \in \text{NP}\\ A \leq^\text{p}_\text{m} B \text{ und } B \in \text{P} &\implies A \in \text{P}\end{aligned}$$

Sei $\mathcal{C}$ eine Komplexitätsklasse und $\preceq$ eine der obigen Reduktionsordnungen. Wie üblich nennen wir nun eine Sprache A *$\preceq$-hart für $\mathcal{C}$* wenn A eine obere Schranke für $\mathcal{C}$ geordnet über $\preceq$ ist (d.h. $B \preceq A$ für alle $B \in \mathcal{C}$). Wir nennen A *$\preceq$-vollständig für $\mathcal{C}$* wenn $A \in \mathcal{C}$ ein größtes Element von $\mathcal{C}$ geordnet über $\preceq$ ist (d.h. $B \preceq A$ für alle $B \in \mathcal{C}$ und $A \in \mathcal{C}$). Auf Grundlage der Existenz universeller effizienter Turing-Maschinen können für die Klassen P und NP jeweils eine kanonische $\leq^\text{p}_\text{m}$-vollständige Menge angegeben werden. Für NP ist diese

Definition 2.1.

$\texttt{KAN} = \{(N, x, 1^n) \mid N \text{ ist eine NTM und akz. } x \text{ auf einem RW mit} \leq n \text{ vielen Schritten}\}.\triangleleft$

Lemma 2.2. *Die Menge* KAN *ist $\leq^\text{p}_{1,\text{i}}$-vollständig.*

Beweis. Sei $A \in \text{NP}$. Wir wollen zeigen dass $A \leq^\text{p}_{1,\text{i}} \texttt{KAN}$. Sei hierfür N eine NPTM welche A entscheidet. Es gibt also auch ein Polynom p welches die Laufzeit von N beschränkt. Definiere nun die Funktion $f(x) = (N, x, 1^{p(|x|)})$. Es gilt nun

$$\begin{aligned}&x \in A \iff N(x) \text{ akz. auf RW mit} \leq p(|x|) \text{ Schritten}\\ &\iff (N, x, 1^{p(|x|)}) \in \texttt{KAN} \iff f(x) \in \texttt{KAN}.\end{aligned}$$

Ferner ist leicht zu sehen, dass $f \in \text{FP}$, dass f injektiv und auch p-invertierbar ist. □

Die obigen Reduktionsordungen erzeugen je eine kanonische Äquivalenzordung („Duplikatrelation“):

- $A \equiv^\text{p}_\text{m} B \iff A \leq^\text{p}_\text{m} B \text{ und } B \leq^\text{p}_\text{m} A$.
- $A \equiv^\text{p}_1 B \iff A \leq^\text{p}_1 B \text{ und } B \leq^\text{p}_1 A$.
- $A \equiv^\text{p}_{1,\text{i}} B \iff A \leq^\text{p}_{1,i} B \text{ und } B \leq^\text{p}_{1,i} A$.

Wir definieren auch noch die *p-Isomorphie* als eine Verfeinerung von $\equiv^\text{p}_{1,\text{i}}$:

- $A \equiv^\text{p} B \iff$ es existiert eine bijektive und p-invertierbare Funktion $f \in \text{FP}$ mit $x \in A \leftrightarrow f(x) \in B$.

Gilt $A \equiv^\text{p} B$ dann sagen wir auch dass A und B *p-isomorph* sind. Im Folgenden werden noch die wichtigsten bekannten Aussagen bezüglich p-Isomorphie zusammengefasst:

Hartmanis und Berman (1976) zeigen dass aus $\equiv^\text{p}_{1,\text{i}}$-äquivalente Sprachen dann p-isomorph sind, wenn die jeweiligen Reduktionsfunktionen verlängernd sind.

Satz 2.3. *Gilt $A \leq^{\mathrm{p}}_{1,\mathrm{i}} B$ via f und $B \leq^{\mathrm{p}}_{1,\mathrm{i}} A$ via g, und f und g sind verlängernd, dann gilt $A \equiv^{\mathrm{p}} B$.*

Um die Voraussetzungen vom vorigen Satz 2.3 zu vereinfachen, führen sie den Begriff der *paddability* ein.

Definition 2.4. Eine Sprache $A \neq \emptyset$ heißt (Berman–Hartmanis-)*paddable* genau dann wenn eine injektive und p-invertierbare Funktion $g \in FP$ existiert sodass für alle $x, y \in \Sigma^*$ gilt:

$$x \in A \iff g(x, y) \in A.$$

Das heißt, g fügt einen beliebigen String y zur „Probleminstanz“ x hinzu, sodass die Mitgliedschaft zu A unverändert bleibt, und die beiden originalen Strings x und y wieder rekonstruiert werden können. ◁

Es gilt:

Satz 2.5. *(1) Ist A paddable so gibt es für jedes B mit $B \leq^{\mathrm{p}}_{\mathrm{m}} A$ eine injektive p-invertierbare verlängernde Funktion die $B \leq^{\mathrm{p}}_{1,\mathrm{i}}$ realisiert.*

(2) Sind A, B paddable, so folgt aus $A \equiv^{\mathrm{p}}_{\mathrm{m}}$ stets $A \equiv_{\mathrm{p}}$.

Alle bekannten $\leq^{\mathrm{p}}_{\mathrm{m}}$-vollständigen Mengen für NP sind zueinander p-isomorph. Berman und Hartmanis vermuteten, dass das für *alle* $\leq^{\mathrm{p}}_{\mathrm{m}}$-vollständigen Mengen gilt:

Vermutung 2.6 (IC)**.** *Alle $\leq^{\mathrm{p}}_{\mathrm{m}}$-vollständigen Mengen für* NP *sind p-isomorph. In anderen Worten: die $\leq^{\mathrm{p}}_{\mathrm{m}}$-Äquivalenzklasse der vollständigen Mengen ist gleich der $\equiv^{\mathrm{p}}$-Äquivalenzklasse von* SAT*.*

Mit obigem Begriff von Paddability lässt sich die $\equiv^{\mathrm{p}}$-Äquivalenzklasse von SAT folgendermaßen charakterisieren:

Satz 2.7. *Eine Menge $A \in$ NP ist genau dann p-isomorph zu* SAT *wenn A $\leq^{\mathrm{p}}_{\mathrm{m}}$-vollständig und und paddable ist.*

Als Konsequenz ergibt sich hieraus, dass die *bekannten* $\leq^{\mathrm{p}}_{\mathrm{m}}$-vollständigen Mengen alle paddable sind. (Das ist die eigentliche empirische Beobachtung von Berman und Hartmanis, auf welcher diese IC vermuteten.)

2.4 Orakel und Relativierungen

Wie in der Einleitung schon angesprochen, ist die Idee hinter Orakel-Berechnungen zu untersuchen, welche Probleme B effizient(er) durch einen Algorithmus gelöst werden können, wenn die Algorithmen eine (fiktive) Möglichkeit haben, ein (ggf. sehr schweres) Problem A ohne Rechenaufwand zu lösen. Der Zugriff auf A kann also wie ein „Nachschlagewerk“ eine „Blackbox-Funktion“ verstanden werden, die auf magische Weise A augenblicklich löst.

Diese Idee wird im Berechnungsmodell der Orakel-Turing-Maschine (OTM) formalisiert. Orakel-Turing-Maschinen sind eine Erweiterung der (deterministischen und nichtdeterministischen) Turing-Maschinen, die zum Eingabe-, Arbeits- und Ausgabeband auch noch ein separates Orakelband haben. Ferner existieren drei ausgezeichnete Zustände $q_?, q_{\mathrm{yes}}, y_{\mathrm{no}}$.

Gegeben ein Orakel $A \subseteq \Sigma^*$ können OTM nun Fragen der Form $x \stackrel{?}{\in} A$ an das Orakel stellen, indem sie ein Wort x auf das Frageband schreiben, und in den Zustand $q_?$ übergeht. Im unmittelbar nächsten deterministischen Schritt der Berechnung wird der Zustand q_{yes} eingenommen falls $x \in A$, sonst den Zustand q_{no}.

Aus dieser Beschreibung wird klar, dass eine Berechnung einer OTM sowohl von der Eingabe x abhängig ist, als auch vom Orakel A, *relativ* zu diesem $M(x)$ rechnet. Wir schreiben dann auch kurz M^A wenn wir die OTM M mit festem Orakel A meinen, und $M^A(x)$ die Berechnung der OTM M auf Eingabe x mit Orakel A. Entsprechend können wir auch die Laufzeit $\mathrm{time}^A_M(x)$ definieren, und von (deterministischen bzw. nichtdeterministischen) Polynomialzeit-Orakel-Turing-Maschinen (POTM, NPOTM) sprechen, wenn die Laufzeit auf allen Eigaben und allen Orkalen polynomiell durch die Eingabelänge beschränkt ist.

Wir können nun die relativierten Komplexitätsklassen $\mathrm{P}^O, \mathrm{NP}^O, \mathrm{FP}^O, \mathrm{NPMV}^O, \ldots$ relativ zu einem gegebenen Orakel O definieren, wobei in der jeweiligen Definition die TM mit OTM ersetzt werden, die Zugriff auf das Orakel O haben. Diese Relativierung überträgt sich auch auf unsere weiteren Definitionen, wie z.B. Reduktion und Vollständigkeit. Wir schreiben z.B. $A \leq^{\mathrm{p},O}_{\mathrm{m}} B$ wenn eine Funktion $f \in \mathrm{FP}^O$ existiert mit $x \in A \leftrightarrow f(x) \in B$. Die kanonische NP-vollständige Menge KAN kann ebenso zu $\mathtt{KAN}^O$ relativiert werden. Für *natürliche* Mengen wie SAT usw. werden wir dagegen keine relativierte Variante definieren. In diesem Sinne ist SAT im Allgemeinen *nicht* $\leq^{\mathrm{p},O}_{\mathrm{m}}$-vollständig, in dem Sinne dass ein Orakel O existiert und eine Menge $A \in \mathrm{NP}^O$ sodass $A \not\leq^{\mathrm{p},O}_{\mathrm{m}}$ SAT.

In allgemeineren Beweisen, die nicht konkrete natürliche Mengen betreffen, lassen sich üblicherweise alle beteiligten TM mit OTM auswechseln, ohne die Gültigkeit der Aussage zu verändern. Aussagen bzw. Beweise, die in solchen relativierten Umgebungen relativ zu jedem beliebigen Orakel O gelten, nennen wir *relativierende* Aussagen bzw. Beweise. Das Diagonalargument in einem typischen Beweis von $\mathrm{P} \subsetneq \mathrm{E}$ relativiert beispielsweise, sodass auch $\mathrm{P}^O \subsetneq \mathrm{E}^O$ für jedes beliebige Orakel O gilt. Ebenso relativiert die Aussage „$\mathtt{KAN} \in \mathrm{NP}$ ist $\leq^{\mathrm{p}}_{\mathrm{m}}$-vollständig“ (zu „$\mathtt{KAN}^O \in \mathrm{NP}^O$ ist $\leq^{\mathrm{p},O}_{\mathrm{m}}$-vollständig“).

Im Folgenden soll jede Aussage als relativierbar verstanden werden, es sei denn es wird auf die Nichtrelativierbarkeit hingewiesen, oder von konkreten natürlichen Mengen gesprochen, welche ohnehin nicht relativieren.

2.5 Beweissysteme

Beweissysteme wurden in der Einleitung schon kurz definiert. In diesem Abschnitt wird die präzise Definition von Cook und Reckhow (1979) wiedergegeben.

Definition 2.8. Eine Funktion $f \in \mathrm{FP}$ ist ein *Beweissystem* für L wenn $\mathrm{ran}(f) = L$. Ist $f(w) = x$ schreiben wir auch, dass w ein *f-Beweis* für x ist.

Existiert zudem ein Polynom q sodass für jedes $x \in L$ ein f-Beweis x der Länge $\leq q(|x|)$ existiert, sagen wir dass f *kurzen Beweise* hat. ◁

Hieraus stellt sich die erste Frage, welche Mengen Beweissysteme mit kurzen Beweisen haben.

Ein einfaches Beweissystem für die Menge $\mathtt{SAT} \in \mathrm{NP}$ wäre z.B. das *Standardbeweissystem* *std* für $\mathtt{SAT}$:

$$std(\varphi, w) = \begin{cases} \varphi & \text{wenn } w \text{ eine erfüllende Belegung für } \varphi \text{ ist,} \\ \bot & \text{sonst.} \end{cases}$$

Dieses Beweissystem hat kurze Beweise.

Cook und Reckhow machen dagegen die Beobachtung im Fall von der Menge $\mathtt{TAUT}$ der aussagenlogischen Tautologien die Beobachtung, dass $\mathtt{TAUT}$ genau dann kein Beweissystem mit kurzen Beweisen hat, wenn $\mathrm{NP} \neq \mathrm{coNP}$. Diese Einsicht motivierte das sogenannte *Cook–Reckhow-Programm*: man nähert sich der Frage $\mathrm{NP} \neq \mathrm{coNP}$ mittels Untersuchung immer stärerer Beweissysteme. Um nun die relative Stärke unterschiedlicher Beweissysteme zu vergleichen, führen Cook und Reckhow den Begriff der Simulation ein.

Definition 2.9. Seien f, g zwei Beweissysteme für L. Wir sagen dass f das Beweissystem g *simuliert* wenn eine (nicht notwendigerweise effiziente) polynomiell längenbeschränkte Funktion π existiert sodass

$$f(\pi(w)) = g(w).$$

Heißt, für jeden g-Beweis w für x existiert auch ein f-Beweis $\pi(w)$ für das gleiche x, und dieser f-Beweis $\pi(w)$ ist nur polynomiell länger als w.

Ist zusätzlich $\pi \in \mathrm{FP}$, dann ist sagen wir, dass f das Beweissystem g *p-simuliert*. Das ist äquivalent zur Aussage $g \leq^{\mathrm{p}}_{\mathrm{m}} f$. ◁

Wenn f das Beweissystem g p-simuliert, kürzen wir das entsprechend der Beobachtung in der Definition auch kurz mit $g \leq^{\mathrm{p}}_{\mathrm{m}} f$ ab.

Beobachte dass Beweissysteme mit kurzen Beweisen unter Simulation abgeschlossen sind: wenn Beweissystem f das Beweissystem g simuliert, und g kurze Beweise hat, dann hat auch f kurze Beweise. Auf ähnliche Beweise sind die Beweissysteme unter p-Simulation abgeschlossen, die einfach auffindbare Beweise haben: wenn Beweissystem f für L das Beweissystem g p-simuliert, und für jedes $x \in L$ in Polynomialzeit ein g-Beweis w gefunden werden kann, dann kann auch ein f-Beweis in Polynomialzeit gefunden werden.

Allgemein generiert die Relation der (p-)Simulation wieder eine Quasiordnung, die nach der Existenz von größten Elementen untersucht werden kann. Hieraus ergibt sich der Begriff der (p-)Optimalität.

Definition 2.10. Ein Beweissystem f für L ist *(p-)optimal* wenn es jedes Beweissyem g für L (p-)simulieren kann.

Die p-Optimalität von f ist äquivalent zur $\leq^{\mathrm{p}}_{\mathrm{m}}$-Vollständigkeit von f für die Teilmenge der Funktionen aus FP mit Bildmenge L. ◁

Es ist leicht zu sehen, dass jedes Beweissystem mit kurzen Beweisen auch optimal ist. Im Zusammenhang mit dem Cook–Reckhow-Programm weisen Krajíček und Pudlák (1989) darauf hin, dass die Existenz eines optimalen Beweissystems für $\mathtt{TAUT}$ wahrscheinlich schwächer ist, als die Existenz eines Beweissystems mit kurzen Beweisen, denn Ersteres folgt

schon aus NE = coNE. (Köbler, Messner und Torán, 2003, schwächen die Voraussetzung auf NEE = coNEE ab.)

Für die Mengen aus P bzw. NP existieren p-optimale bzw. optimale Beweissysteme:

Beobachtung 2.11. *(1) Ist $A \in \mathrm{P}$, dann existiert ein p-optimales Beweissystem für A mit kurzen Beweisen.*

(2) Ist $A \in \mathrm{NP}$, dann existiert ein optimales Beweissystem für A mit kurzen Beweisen.

Beweis. 1. Zu (1): Betrachte die Funktion

$$h(x) = \begin{cases} x & \text{wenn } x \in A \\ \bot & \text{sonst} \end{cases}.$$

Diese Funktion ist definitiv ein Beweissystem für A. Sie ist in FP, ist ja der Test „$x \in A$“ in Polynomialzeit möglich. Klar ist auch dass h kurze Beweise hat. Dieses Beweissystem ist p-optimal, denn wenn g ein weiteres Beweissystem für A ist, und wenn w ein g-Beweis für x ist, dann ist $h(g(w)) = h(x) = x$, also $g(w)$ ein h-Beweis für x, wie gewünscht.

2. Zu (2): Kann durch die bekannte Zertifikats-Definition von NP gezeigt werden (was im späteren Teil der Arbeit auch geschieht), zur Vollständigkeit lässt sich an dieser Stelle aber auch ein Beweis über die NPTM-Definition von oben angeben. Sei N eine NPTM, die A mit polynomieller Laufzeit p entscheidet. Definere nun

$$h(x, \alpha) = \begin{cases} x & N(x) \text{ akz. auf Rechenweg } \alpha \\ \bot & \text{sonst} \end{cases}$$

Nachdem $L(N) = A$ ist h definitiv ein Beweissystem für A. Es ist leicht zu sehen dass $h \in \mathrm{FP}$, ist der Test „α ist ein gültiger Rechenweg und akzeptiert“ in Polynomialzeit möglich. Ferner existiert für jedes $x \in A$ auch ein akzeptierender Rechenweg α auf $N(x)$ der Länge $\leq p(|x|)$, womit der Beweis (x, α) auch nur polynomiell länger als x ist. Da Beweissystem h hat also kurze Beweise, und ist damit auch optimal. □

Insbesondere mit dem letzten Punkt können die optimalen Beweissysteme der Mengen aus NP auch als Beweissysteme mit kurzen Beweisen charakterisiert werden:

Beobachtung 2.12. *Sei $A \in \mathrm{NP}$ und f ein Beweissystem für A. Das Beweissystem f ist optimal genau dann wenn jedes $x \in A$ einen f-Beweis der Länge $\leq q(|x|)$ hat, wobei q ein Polynom ist.*

Beweis. Richtung von rechts nach links klar, das gilt schon im Allgemeinen Fall. Für die andere Richtung sei f ein optimales Beweissystem für A. Dann muss f auch das Beweissystem h mit kurzen Beweisen aus voriger Beobachtung simulieren. Für jeden kurzen h-Beweis w für x existiert dann ein höchstens polynomiell längerer f-Beweis $\pi(w)$. □

3 Zur Konzeptionalisierung und Ordnung von Suchproblemen

TODO: Einleitung und Übersicht über das Kapitel. Was passiert hier?

3.1 Definition von Suchproblemen

Wir geben hier noch einmal die Definition von Suchproblemen wieder, welche schon in der Einleitung erarbeitet wurde. Als Suchprobleme verstehen wir das algorithmische Problem, gegeben eine Probleminstanz x, eine entsprechende positive Lösungsinstanz y zu berechnen, oder negativ abzulehnen. Hier noch einmal ein Beispiel aus der Einleitung: gegeben eine aussagenlogische Formel φ, berechne entweder eine Belegung y welche φ erfüllt, oder gebe „unerfüllbar“ aus. Die wesentliche Einschränkung, welche wir auch schon in der Einleitung festgelegt haben, ist die Einschränkung auf „NP-Suchprobleme“. Wir meinen damit, dass

- die Lösungen nur polynomiell länger als die Probleminstanzen sind, und
- effizient in Polynomialzeit verifiziert werden kann, ob zu einer gegebenen Probleminstanz x ein beliebiges Wort y tatsächlich eine (positive) Lösung im Sinne des Suchproblems darstellt oder nicht.

(Wir fordern im Übrigen nicht, dass negatives Ablehnen effizient verifiziert werden kann.) Um das Beispiel wieder aufzugreifen: Zum einen haben Formeln φ, welche überhaupt erfüllbar sind, eine erfüllende Belegung in Länge von φ. Zum anderen kann effizient geprüft werden, ob y tatsächlich eine erfüllbare Belegung von φ ist.

Diese Einschränkung wird durch die empirische Einsicht gestützt, dass viele natürliche Suchprobleme, für die momentan kein effizienter Algorithmus bekannt ist, genau in eine solche Einschränkung fallen. Also Suchprobleme, die „verifizierbar“ sind und „kurze Lösungen“ haben. Einige weitere Beispiele werden wir im Folgenden noch betrachten.

Zunächst werden wir die beiden obigen Punkte noch einmal in eine formale Definition gießen:

Definition 3.1 (NP-Relation, FNP)**.** Eine *NP-Relation* ist eine zweistellige Relation $R \subseteq \Sigma^* \times \Sigma^*$, sodass diese

(1) in Polynomialzeit entscheidbar ist, d.h. $R \in \mathrm{P}$, und

(2) p-balaciert ist, d.h. es existiert ein Polynom q, sodass

$$(x, y) \in R \implies |y| \leq q(|x|) \quad \text{für alle } x, y \in \Sigma^*. \tag{3.1}$$

Die Wörter der ersten Komponente nennen wir *Probleminstanzen* oder *Instanzen* oder *Probleme* von R, die Wörter der zweiten Komponente nennen wir die *Zertifikate* (oder manchmal *Lösungen*) von R. Wir sagen dann für $(x, y) \in R$, dass y *ein Zertifikat für* x ist. In diesem Sinne sagt (3.1) aus, dass Zertifikate y für x nicht superpolynomiell länger als x sein dürfen. Das Polynom q nennen wir auch die *Zertifikatsschranke* zu R.

Wir schreiben FNP für die Klasse aller NP-Relationen. ◁

Das oben diskutierte Suchproblem zu einer NP-Relation R kann jetzt wie folgt formal formuliert werden:

> Suchproblem zur Relation R:
>
> **Gegeben:** Instanz x.
>
> **Gesucht:** Zertifikat y mit $(x, y) \in R$ falls ein solches y überhaupt existiert, sonst „keine Lösung“ ausgeben.

Zur Erinnerung:

$$\mathrm{Proj}(R) = \{x \mid \exists y \in \Sigma^*, (x, y) \in R\} \in \mathrm{NP}.$$

Die Menge $\mathrm{Proj}(R)$ ist also die Menge der Probleminstanzen, für welche ein zugehöriges Zertifikat existiert; damit entspricht $\mathrm{Proj}(R)$ derjenigen Menge, die üblicherweise bei algorithmischen Entscheidungsproblemen betrachtet wird. Um die beiden Varianten noch einmal gegenüberzustellen: das entsprechende Entscheidungsproblem einer Relation R lautet

Entscheidungsproblem zur Relation R:

Gegeben: Instanz x.

Gesucht: Akzeptieren falls ein Zertifikat y mit $(x, y) \in R$ existiert, sonst ablehnen.

Das entspricht also dem Entscheiden der Sprache Proj(R). Damit wird auch klar, dass das entsprechende Entscheidungsproblem bzw. die Sprache Proj(R) nicht von der konkreten Relation R abhängig ist. Vielmehr: es existieren zur Sprache L ggf. unendlich viele NP-Relationen R mit Proj(R) = L. Für eine Sprache L sagen wir dann auch, dass R *eine NP-Relation für* L ist.

Die Zugehörigkeit des entsprechenden Suchproblems zu NP folgt hierbei unmittelbar aus der Definition von NP-Relationen. (Rate nichtdeterministisch ein Zertifikat und akzeptiere wenn dieses korrekt ist.) Im nächsten Abschnitt wird die Beziehung zwischen Suchproblemen bzw. NP-Relationen einerseits, und Entscheidungsproblemen bzw. Mengen aus NP andererseits, weiter behandelt. Festhalten können wir hier aber schon, dass das Suchproblem offenbar „schwieriger“ ist als das alleinige Entscheidungsproblem.

Im Folgenden werden eine Beispiele von natürlichen NP-Relationen angegeben. Um diese von den aonst üblicherweise verwendeten Labels für Mengen bzw. Suchprobleme abzugrenzen, sind im Verlauf dieser Arbeit *NP-Relationen* zu natürlichen Suchproblemen immer mit einem `r` am Anfang gekennzeichnet.

- `rPERFECTMATCHING` = $\{(G, M) \mid G$ ist ein Graph, M ein perfektes Matching auf $G\}$.
- `rSAT` = $\{(\varphi, w) \mid \varphi$ ist eine aussagenlogische Formel, w erfüllende Belegung für $\varphi\}$.
- `rVC` = $\{((G, k), C) \mid G$ ist ein Graph, C eine Knotenüberdeckung, und $|C| \leq k\}$.
- `rHAMCYCLE` = $\{(G, P) \mid G$ ist ein Graph, P ein Zyklus der jeden Knoten genau einmal berührt$\}$.
- `rANOTHERHAMCYCLE` = $\{((G, P), P') \mid G$ ist ein Graph, P, P' je ein Zyklus der jeden Knoten genau einmal berührt, $P \neq P'\}$.
- `rFACTORIZATION` = $\{(n, (p_1, p_2, \ldots, p_k)) \mid n \in \mathbb{N}, n > 1$, alle p_i Primzahlen ungleich 2 oder n, und $n = p_1 \cdots p_k\}$.
- `rFACTOR` = $\{(n, p) \mid n$ ist nicht prim, und p ist ein nichttrivialer Faktor von $n\}$.
- `rSMALLFACTOR` = $\{((n, a), p) \mid n$ ist nicht prim, und p ist ein nichttrivialer Faktor von n und $p \leq a\}$.
- `rGI` = $\{((G, H), \sigma) \mid G, H$ sind Graphen mit gleicher Knotenmenge, und σ ist ein Graphisomorphismus von G nach $H\}$.

Es ist leicht zu sehen, dass jede dieser relationen auch eine NP-Relation ist. Beachte dass die Menge der Primzahlen in Polynomialzeit etnscheidbar ist Agrawal, Kayal und Saxena (2004). Bei jeder der obigen natürlichen Relationen gilt, dass die Projektion auch der üblichen Sprache aus NP entspricht. Wir haben z.B.

$$\text{Proj}(\texttt{rVC}) = \{(G, k) \mid \text{ex. Knotenüberdeckung } C \text{ von Graph } G \text{ mit } |C| \leq k\}.$$

Die Definition Suchproblemen als NP-*Relationen* lässt es zu, Suchprobleme bzw. NP-Relationen als „partielle Multifunktionen“ zu verstehen. Selman (1994) definiert in seiner Taxonomie der Funktionsklassen die Klasse NPMV_g als die Klasse derjenigen Multifunktionen $f \in$ NPMV, für die (der Graph) f in P liegt. Es lässt sich leicht sehen, dass die hier definierte Klasse FNP identisch zu Selman definierten Klasse NPMV_g ist, solange man Multifuktionen mit binäre Relationen identifiziert.

Mit dieser Perspektivierung ist auch einfach zu definieren, was mit „Suchproblem lösen“ gemeint ist. Wir machen hierbei Gebrauch von Verfeinerungen (von Multifunktionen). Wir sagen, dass das Suchproblem zur NP-Relation R *in Polynomialzeit lösbar ist*, wenn $R \in_\text{c}$ FP. Diese Aussage bedeutet ja, dass eine Verfeinerung f von R existiert, und f ist dabei eine (partielle) Funktion. Für eine Eingabeinstanz x wird also entweder $f(x)$ einen Wert y ausgeben für den $y \in$ *set-*$R(x)$ gilt, bzw. in anderen Worten, ein y für das $(x, y) \in R$ und damit ist die Ausgabe y eine Lösung für x. Oder, falls $f(x)$ ablehnt, dann ist $x \notin \text{dom}(f) = \text{Proj}(R)$, heißt „$f(x)$ lehnt ab“ bedeutet dass x keine Lösung hat.

Wir können damit auch schon die obige intuitive Aussage beweisen, dass das Suchproblem „schwieriger“ ist als das entsprechende Entscheidungsproblem, in dem Sinne dass sich das Entscheidungsproblem auf das Suchproblem reduzieren lässt:

Beobachtung 3.2. *Sei R eine NP-Relation. Falls $R \in_\text{c}$ FP, dann gilt* Proj(R) $\in$ P.

Beweis. Sei $f \in$ FP die Verfeinerung von R nach Voraussetzung. Teste ob $f(x) \neq \perp$. Falls ja, dann ist $f(x) \in$ *set-*$R(x)$ und damit hat x eine Lösung; akzeptiere. Falls nicht, dann ist $x \notin \text{dom}(f) = \text{Proj}(x)$; lehne ab. □

Der aktuelle Stand zur Lösbarkeit der oben gennanten natürlichen Suchprobleme ist:

- $\mathtt{rPERFECTMATCHING} \in_c \mathrm{FP}$.
- $\mathrm{NP} = \mathrm{P} \iff \mathtt{rSAT} \in_c \mathrm{FP} \iff \mathtt{rVC} \in_c \mathrm{FP} \iff \mathtt{rHAMCYCLE} \in_c \mathrm{FP} \iff \mathtt{rANOTHERHAMCYCLE} \in_c \mathrm{FP}$.
- Unklar ob $\mathtt{rSMALLFACTOR}, \mathtt{rFACTOR}, \mathtt{rFACTORIZATION} \overset{?}{\in_c} \mathrm{FP}$. Wir haben aber $\mathrm{UP} \cap \mathrm{coUP} = \mathrm{P} \implies \mathtt{rSMALLFACTOR}, \mathtt{rFACTOR}, \mathtt{rFACTORIZATION} \in_c \mathrm{FP}$.
- Unklar ob $\mathtt{rGI} \overset{?}{\in_c} \mathrm{FP}$.

Die genannten Implikationen und Äquivalenzen werden in den nächsten Abschnitten beweisen.

Bevor nun im nächsten Abschnitt die Suchprobleme den Entscheidungsproblemen näher gegenübergestellt werden, schließen wir diesen Abscnhitt noch mit einer kurzen Diskussion zu *totalen* Suchchproblemen ab.

Die oben formulierte Definition von FNP ist genau diejenige, die von Megiddo und Papadimitriou (1991) als erstes in dieser Form und Bezeichnung definiert wurde. Ihre Motivation war, hierbei insbesondere die *totalen* Suchprobleme in den Blick zu nehmen. Also solche Suchprobleme, bei der jede Proleminstanz immer mindestens ein Zertifikat bzw. Lösung hat. Die Faktorisierung ist beispielsweise ein solches totales Suchproblem, da ja jede natürliche Zahl sich faktorisieren lässt.

Das sind – entsprechend dieser Definition von FNP bzw. Konzeptionalisierung von Suchproblemen – genau jene NP-Relationen welche (links-)total sind: für jedes $x \in \Sigma^*$ existiert ein $y \in \Sigma^*$ mit $(x, y) \in R$. Die Relationen $\mathtt{rFACTORIZATION}$ und $\mathtt{rFACTOR}$ wie oben definiert sind nicht total, nachdem die negativen Instanzen aber besonders „einfach“ sind, können für beide NP-Relationen effektiv äquivalente Relationen angegeben werden, die total sind:

- $\mathtt{rFACTORIZATION}' = \mathtt{rFACTORIZATION} \cup \{(n, \text{„}\textit{ungültig}\text{“}) \mid n \leq 1\}$.
- $\mathtt{rFACTOR}' = \mathtt{rFACTOR} \cup \{(n, \text{„}\textit{ungültig}\text{“}) \mid n \leq 1 \text{ oder } n \text{ ist prim}\}$.

Megiddo und Papadimitriou (1991) fassen diese totalen NP-Relationen zur Klasse TFNP zusammen:

Definition 3.3 (TFNP)**.** Die Klasse TFNP ist die Teilmenge von FNP derjenigen NP-Relationen R, welche linkstotal sind, heißt zu jedem $x \in \Sigma^*$ existiert ein $y \in \Sigma^*$ mit $(x, y) \in R$. ◁

Hierzu gehören die oben genannten Varianten $\mathtt{rFACTORIZATION}'$ und $\mathtt{rFACTOR}'$. Für Megiddo und Papadimitriou befinden sich in TFNP eine Vielzahl von interessanten und schwierigen Suchproblemen, bei denen die Frage der Lösbarkeit in Polynomialzeit noch offen ist. Das betrifft u.a. zahlentheoretische Probleme aus der Kryptographie wie Faktorisierung, diskreter Logarithmus. Beachte dass TFNP nicht identisch ist zur Klasse NPMV_t; es macht sich hier die gleiche Unterscheidung wie bei FNP vs. NPMV auf: Die Klasse TFNP ist eine Teilmenge von NPMV_t jener totalen Multifunktionen $f \in \mathrm{NPMV}_t$, für die der (der Graph) f in P liegt. Beachte dass TFNP sogar eine echte Teilmengen von NPMV_t ist, außer $\mathrm{P} = \mathrm{NP}$:

Beobachtung 3.4 (Vgl. Fenner u. a. 2003, Prop. 5)**.** *Wenn für alle* $f \in \mathrm{NPMV}_t$ *auch (der Graph)* $f \in \mathrm{P}$ *ist, dann gilt* $\mathrm{P} = \mathrm{NP}$.

> *Beweis.* Betrachte folgenden NPTM-Transduktor N auf Eingabe $\varphi \in \Sigma^*$: zunächst spaltet sich die Berechnung nichtdeterministisch auf. In der ersten Rechnung wird sofort 1 ausgegeben. In der zweiten Rechnung wird eine Belegung w für die aussagenlogische Formel φ geraten, und 2 ausgegeben wenn w die Formel φ erfüllt. Sei f die Multifunktion, welche von N berechnet wird. Damit gilt:
>
> $$\textit{set-}f(x) = \{1, 2\} \text{ falls } x \in \mathtt{SAT}, \text{ und } \{1\} \text{ sonst}$$
>
> und $f \in \mathrm{NPMV}_t$. Nach Annahme ist $f \in \mathrm{P}$. Nun kann aber SAT in Polynomialzeit entschieden werden, denn $\varphi \in \mathtt{SAT}$ genau dann wenn $(\varphi, 2) \in f$.
>
> Die Aussage relativiert, wenn anstelle SAT z.B. das kanonische vollständige Problem gewählt wird. □

Mit der Beschäftigung mit TFNP-Problemen kam es zu einer umfassenden Theoriebildung. So kam z.B. eine verfeinerte Betrachtung durch Unterklassen von TFNP hinzu. Jede dieser Unterklassen verinnerlicht hierbei jeweils das kombinatorische Prinzip, „warum“ ein Suchproblem total ist. Exemplarisch werden hier zwei Unterklassen skizziert:

- Die Unterklasse PLS („polynomial local search“) umfasst die Suchprobleme, welche in die Form eines Suchgraphen polynomiellen Grads gebracht werden können, worauf ein lokales Optimum gesucht ist. Das zugrunde liegende kominatorische Prinzip zur Totalität wäre „*endliche Suchgraphen haben immer ein lokales Optimum*“ oder allgemeiner „*Jeder endliche gerichtete azyklische Graph hat eine Senke*“.

Ein Beispiel hierfür wäre die Suche nach einem lokal kürzesten Rundreise in einem Graphen; „lokal kürzesten“ im Sinne dass das Austauschen von zwei Kanten mit zwei anderen zu keiner kürzereren Rundreise führt.

- Die Unterklasse PPP („polynomial pigeon principle“) umfasst Suchprobleme, welche aufgrund des kombinatorischen Schubfachprinzip total sind.

  Ein Beispiel hierfür ist das Gleiche-Summe-Suchproblem: gegeben n positive ganze Zahlen die sich zu $< 2^n - 1$ aufsummieren, finde zwei unterschiedliche nichtleere Teilmengen dieser Zahlen welche die gleiche Summe haben. (Existiert nach Schubfachprinzip: es existieren $2^n - 1$ viele nichtleere Teilmengen, jede davon mit Summe $< 2^n - 1$, die Summen können also nicht alle unterschiedlich sein.)

Auf die weitere Theorie der TFNP-Probleme wird in dieser Arbeit nicht weiter eingegangen. Wir werden aber in Abschnitt **??** noch Reduktionen auf NP-Relationen definieren; dieser Reduktionsbegriff ist der identische wie auf den TFNP-Problemen Megiddo und Papadimitriou (1991).

3.2 Suchprobleme vs. Entscheidungsprobleme

Tatsächlich lässt sich über die Projektionen von NP-Relationen genau die Klasse NP von Entscheidungsproblemen charakterisieren: zu jeder Sprache bzw. Entscheidungsproblem $L \in$ NP existiert (mind.) eine NP-Relation R mit $L = \text{Proj}(R)$, und zu jeder NP-Relation bzw. NP-Suchproblem R ist $\text{Proj}(R) \in$ NP. Das ist die übliche „Zerifikats-Charakterisierung“ von NP aus den Lehrbüchern.

Beobachtung 3.5. $\text{NP} = \{\text{Proj}(R) \mid R \textit{ ist eine NP-Relation}\}$.

Beweis. Für die Inklusion von links nach rechts starten wir mit einer Sprache L und einer NPTM N die L entscheidet, wobei die Laufzeit durch das Polynom p beschränkt ist. Definiere nun die Relation

$$R_N = \{(x, \alpha) \mid N(x) \text{ akz. mit RW } \alpha, \alpha \text{ hat } \leq p(|x|) \text{ viele Schritte}\}$$

Diese Relation ist eine NP-Relation. Der Test ist offenbar in Polynomialzeit möglich, und die Relation ist p-balanciert, ist $|\alpha| \in O(\#\text{ Schritte von } \alpha) \in O(p(|x|))$. Aus Definition geht hervor dass $L(N) = \text{Proj}(R_N)$.

Für die Inklusion von rechts nach links konstruieren wir uns zu einer gegebenen NP-Relation R mit Zertifikatsschranke q eine NPTM N_R die $\text{Proj}(R)$ entscheidet. Die NPTM N_R arbeitet auf Eingabe x wie folgt:

1 Rate nichtdeterministisch ein $y \in \Sigma^*$
2 **wenn** $(x, y) \in R$ **dann** akzeptiere
3 **sonst** ablehnen

Es ist wegen $R \in$ P klar, dass diese NTM in Polynomialzeit läuft. Wieder geht aus Definition hervor dass $L(N) = \text{Proj}(R_N)$. □

Damit ist im Übrigen die obige Definition von NP-Relationen auch nicht „neu“ sondern schon immer mitgedacht. Die eben formulierte Charakterisierung findet sich in allen üblichen Einführungswerken zur Komplexitätstheorie. Dagegen machen die unterschiedlichen Einführungswerke ihren Zugang manchmal stärker von der Perspektive der Suchprobleme abhängig, und manchmal stärker von der typischen Herangehensweise über Entscheidungsproblemen. Vgl. z.B. Goldreich (2008) wlecher in seinem Lehrbuch die P-vs.-NP-Frage zunächst als die äquivalente Frage der Beziehung zwischen den „*efficiently solvable search problems*“ und den „*search problems with efficiently checkable solutions*“ (letzteres sind genau die NP-Relationen) formuliert. Erst später wird mittels *Search-reduces-to-Decision*-Argumenten dafür argumentiert, NP-Entscheidungsproble als die zentralen Untersuchungsobjekte der Komplexitätstheorie anzusehen.

3.3 Levin-Reduzierbarkeit

Definition 3.6 (Levin-Reduzierbarkeit)**.** Seien Q, R zwei NP-Relationen. Wir sagen dass Q *sich auf R (Polynomialzeit-)Levin-reduzieren lässt*, bzw. $Q \leq_{\text{L}}^{\text{p}} R$ wenn zwei Funktionen $f, g \in$ FP existieren sodass

(1) $x \in \text{Proj}(Q) \iff f(x) \in \text{Proj}(R)$,

(2) $(f(x), y) \in R \implies (x, g(x, y)) \in Q$.

Punkt (1) sagt also nur aus, dass f eine Many-one-Polynomialzeit-Reduktion zwischen den entsprechenden Entscheidungsproblemen ist. Punkt (2) sagt nun aus, dass wenn y ein Zertifikat für die Instanz $f(x)$ aus R ist, dann lässt sich aus y wieder ein Zertifikat $g(x, y)$ für die originale Instanz x berechnen.

Die Funktion f nennen wir *Reduktionsfunktion*, die Funktion g nennen wir *Translationsfunktion*.

Wir schreiben $Q \leq^{\mathrm{p}}_{\mathrm{L},1} R$ falls f zusätzlich injektiv ist. Wir schreiben $Q \leq^{\mathrm{p}}_{\mathrm{L},1,\mathrm{inv}} R$ falls f zusätzlich injektiv und p-invertierbar ist. Klar ist:

$$Q \leq^{\mathrm{p}}_{\mathrm{L},1,\mathrm{inv}} R \implies Q \leq^{\mathrm{p}}_{\mathrm{L},1} R \implies Q \leq^{\mathrm{p}}_{\mathrm{L}} R \implies \mathrm{Proj}(Q) \leq^{\mathrm{p}}_{\mathrm{m}} \mathrm{Proj}(R).$$

Wir sagen dass $R \leq^{\mathrm{p}}_{\mathrm{L}}$-*vollständig* ist, wenn $Q \leq^{\mathrm{p}}_{\mathrm{L}} R$ für alle NP-Relationen Q gilt. Die $\leq^{\mathrm{p}}_{\mathrm{L},1}$- und $\leq^{\mathrm{p}}_{\mathrm{L},1,\mathrm{inv}}$-Vollständigkeit ist analog definiert. ◁

Satz 3.7. *Die kanonische NP-Relation*

$$\mathtt{rKAN} = \{((N, x, 1^n), \alpha) \mid \alpha \textit{ ist ein akz. Rechenweg auf } N(x) \textit{ und } |\alpha| \leq n\}$$

ist $\leq^{\mathrm{p}}_{\mathrm{L},1,\mathrm{inv}}$*-vollständig.*

Beweis. Sei R eine beliebige NP-Relation mit Zertifikatsschranke r, i.e. $(x, y) \in R \implies |y| \leq r(|x|)$. Sei M die PTM welche R entscheidet, mit Laufzeitschranke p. Sei N eine NPTM welche auf Eingabe x zunächst ein Zertifikat $y, |y| \leq r(|x|)$ rät, und dann testet ob $M(x, y)$ akzeptiert. Die Laufzeit von N ist beschränkt auf $p(|(x, y)|) \in O(p(r(|x|)))$ (hier nutzen wir die effiziente Listencodierung von **??** aus). Sei daher q ein Polynom, welches die Laufzeit von N beschränkt.

Definiere die Reduktionsfunktion $f(x) = (N, x, 1^{q(|x|)})$. Wir zeigen zunächst dass

$$x \in \mathrm{Proj}(R) \iff f(x) \in \mathrm{Proj}(\mathtt{rKAN}).$$

Wenn $x \in \mathrm{Proj}(R)$, dann existiert ein $y, |y| \leq r(|x|)$ sodass $(x, y) \in R$. Dann wird auch $N(x)$ akzeptieren, nämlich auf jenem Pfad welcher y rät. Es existiert also ein Rechenweg α mit $|\alpha| \leq q(|x|)$ sodass $N(x)$ auf α akzptiert. Dann gilt aber auch $(f(x), \alpha) = ((N, x, 1^{q(|x|)}), \alpha) \in \mathtt{rKAN}$. Die Rückrichtung $x \notin \mathrm{Proj}(R) \implies f(x) \notin \mathrm{Proj}(R)$ folgt analog. Es ist klar, dass f injektiv ist, dass f Polynomialzeit-berechenbar und -invertierbar ist.

Es lässt sich außerdem einfach eine Translationsfunktion $g \in \mathrm{FP}$ angeben, die für $g(f(x), \alpha) = y$ aus α das entsprechende geratene Zertifikat y aus α berechenen kann. □

3.4 Zur gemeinsamen Struktur von vollständigen Suchproblemen

Beobachtung 3.8 (Buhrman, Kadin und Thierauf 1998)**.** *Für jede Menge* $L \in \mathrm{P}$ *die p-isomorph zu* SAT *ist, existiert eine NP-Relation* R_L *sodass* $\mathrm{Proj}(R_L) = L$.

Beweis. Nach Voraussetzung haben wir eine bijektive p-invertierbare Funktion $h \in \mathrm{FP}$ mit $x \in L \iff h(x) \in \mathtt{SAT}$. Definiere nun

$$R_L = \{(x, w) \mid (h(x), w) \in \mathtt{rSAT}\}.$$

Es ist leicht zu sehen, dass R_L eine NP-Relation ist. Es ist auch leicht zu sehen dass $\mathrm{Proj}(R_L) = L$.

Wir zeigen nun, dass R_L auch $\leq^{\mathrm{p}}_{\mathrm{L}}$-vollständig ist. Sei hierfür Q eine beliebige NP-Relation. Nachdem rSAT ja $\leq^{\mathrm{p}}_{\mathrm{L}}$-vollständig ist, existieren Reduktions- und Translationsfunktionen f, g die $Q \leq^{\mathrm{p}}_{\mathrm{L}} \mathtt{rSAT}$ realisieren. Definiere nun

$$f'(x) = h^{-1}(f(x)).$$

Insbesondere ist $h^{-1}(\cdot)$ wohldefiniert, ist ja h^{-1} surjektiv. Damit gilt zum einen für f'

$$x \in \mathrm{Proj}(Q) \iff f(x) \in \mathtt{SAT} \iff h(\underbrace{h^{-1}(f(x))}_{f'(x)}) \in \mathtt{SAT} \iff f'(x) \in \mathrm{Proj}(R_L),$$

und zum anderen gilt

$$(f'(x), w) \in R_L \implies (h(h^{-1}(f(x))), w) \in \mathtt{rSAT} \implies (f(x), w) \in \mathtt{rSAT} \implies (x, g(x, w)) \in Q.$$

Damit erfüllen also f' und g die Voraussetzungen an eine Reduktions- bzw. Translationsfunktion und $Q \leq^{\mathrm{p}}_{\mathrm{L}} R_L$, wie gewünscht. □

Damit haben (im unrelativierten Fall) insbesondere alle *bekannten* $\leq^{\mathrm{p}}_{\mathrm{m}}$-vollständigen Mengen, i.e. zu SAT p-isomorphen Mengen, eine entsprechende NP-Relation, auch wenn hierbei die Zertifikate nicht „natürlich“ sind. Es ist leicht zu sehen, dass diese Aussage relativiert, wenn anstelle rSAT eine andere beliebige $\leq^{\mathrm{p}}_{\mathrm{L}}$-vollständige Relation R gewählt wird.

Definition 3.9 (Geizige Reduktionen)**.** Seien Q, R NP-Relationen. Wir sagen dass sich Q auf R (in Polynomialzeit) *geizig* (*„parsimonious“*) reduzieren lässt, bzw. $Q \leq^{\mathrm{p}}_{\mathrm{pars}} R$ wenn eine Funktion $f \in \mathrm{FP}$ existiert mit

$$x \in \mathrm{Proj}(Q) \iff f(x) \in \mathrm{Proj}(R) \quad \text{und} \quad |\textit{set-}Q(x)| = |\textit{set-}R(f(x))|.$$

In anderen Worte, f realisiert eine Many-one-Reduktion von Q auf R, und haben sowohl die originale Q-Instanz x als auch die reduzierte R-Instanz $f(x)$ die gleiche Anzahl an Lösungen. Definiere $Q \leq^{\mathrm{p}}_{\mathrm{pars}}$-Vollständigkeit entsprechend. ◁

Definition 3.10 (Strukturerhaltende Reduktion; Lynch und Lipton 1978)**.** Seien Q, R NP-Relationen. Wir sagen dass sich Q auf R (in Polynomialzeit) *strukturerhaltend reduzieren lässt*, bzw. $Q \leq^{\mathrm{p}}_{\mathrm{st}} R$, zwei Funktionen $f, g \in \mathrm{FP}$ existieren, und

(1) $(x, y) \in Q \implies (f(x), g(x, y)) \in R$ (Vorwärts-Translation von Zertifikaten),

(2) $(f(x), z) \in R \implies \exists y.\, (x, y) \in R \wedge g(x, y) = z$ (g ist quasi “surjektiv”),

(3) Falls $y_1, y_2 \in \textit{set-}Q(x)$ und $y_1 \neq y_2$, dann ist auch $g(x, y_1) \neq g(x, y_2)$ (g ist quasi “injektiv”). ◁

In einer ähnlichen Weise definieren Fischer, Hemaspaandra und Torenvliet 1995 *zertifikats-isomorphe* Reduktionen („witness-isomorphic reduction“) zwischen zwei NP-Relationen. Obwohl dies nicht explizit angegeben ist, kann ihre Definition so umformuliert werden, dass sie der Definition von strukturerhaltenden Reduktionen nach Lynch und Lipton entspricht, wobei zusätzlich die p-Invertierbarkeit der Funktionen f und g gefordert wird, welche die Reduktion realisieren.

Definition 3.11 (Zertifikats-Isomorphie; Fischer, Hemaspaandra und Torenvliet 1995)**.** Seien Q, R NP-Relationen. Wir sagen dass sich Q auf R (in Polynomialzeit) *zertifikats-isomorph reduzieren lässt*, bzw. $Q \leq^{\mathrm{p}}_{\mathrm{wi}} R$, wenn zwei Funktionen $f, g \in \mathrm{FP}$ existieren, die p-invertierbar (also auch injektiv) sind, und

TODO: das steht so direkt aber nicht bei FHT...

(1) $(x, y) \in Q \implies (f(x), g(x, y)) \in R$ (Vorwärts-Translation von Zertifikaten),

(2) $(f(x), z) \in R \implies \exists y.\, (x, y) \in R \wedge g(x, y) = z$ (g ist quasi “surjektiv”),

(3) Falls $y_1, y_2 \in \mathit{set}\text{-}Q(x)$ und $y_1 \neq y_2$, dann ist auch $g(x, y_1) \neq g(x, y_2)$ (g ist quasi “injektiv”). ◁

Definition 3.12 (Universelle Relationen; Agrawal und Biswas 1992)**.** Sei R eine NP-Relation mit Zertifikatsschranke q. Wir nennen R *streng* wenn für R gilt, dass $(x, y) \in R \implies |y| = q(|x|) > 0$. In anderen Worten, jedes Zertifikat y für x ist nicht ε und hat genau die Länge $q(|x|)$.

Eine Funktion $f \in \mathrm{FP}$ ist eine *projektive Levin-Reduktion* einer strengen Relation Q zu einer zweiten strengen Relation R wenn diese die folgenden Bedingungen erfüllen

(1) $f(x) = (z, \alpha)$ wobei $x, z \in \Sigma^*$ und $\alpha \in \mathbb{N}_{>0}^{q(|x|)}$ ist eine Sequenz von positiven paarweise verschiedenen Indizes der Länge $q(|x|)$.

(2) $\{y[\alpha] \mid y \in \mathit{set}\text{-}R(z)\} = \mathit{set}\text{-}Q(x)$

Wir nennen eine strenge Relation R *universell* wenn sie vollständig bezüglich projektiven Levin-Reduktionen ist. In anderen Worten, wenn für jede strenge Relation Q eine projektive Levin-Reduktion von Q auf R existiert. ◁

Lemma 3.13. *(1) Seien Q, R strenge NP-Relationen. Ist Q auf R reduzierbar über eine projektive Levin-Reduktion, dann ist $Q \leq^{\mathrm{p}}_{\mathrm{L}} R$.*

(2) Ist R universell, dann ist R auch $\leq^{\mathrm{p}}_{\mathrm{L}}$-vollständig. Damit gilt insbesondere auch $Q \leq^{\mathrm{p}}_{\mathrm{L}} R$ für NP-Relationen Q die nicht *streng sind.*

Definition 3.14 (Agrawal und Biswas 1992)**.** Sei R eine NP-Relation mit Zertifikatsschranke q, wobei zusätzlich für R gilt, dass $(x, y) \in R \implies |y| = q(|x|)$. In anderen Worten, jedes Zertifikat y für x hat genau die Länge $q(|x|)$. Wir definieren nun bezüglich einer solchen Relation R:

(1) Die Relation R hat einen *building block*, wenn es ein Element $\mathit{block} \in \mathrm{Proj}(R)$ gibt, sowie paarweise verschiedene $b_1, b_2, b_3 \in \mathbb{N}_{>0}$ sodass

$$\{y[b_1, b_2, b_3] \mid y \in \mathit{set}\text{-}R(\mathit{block})\} = \Sigma^3 - \{000\}$$

(2) Die Relation R ist *joinable* wenn es eine Funktion $\mathit{join} \in \mathrm{FP}$ gibt sodass

$$\mathit{join}(x_1, \ldots, x_n) = (z, \alpha) \text{ wobei } x_1, \ldots, x_n, z \in \Sigma^* \text{ und } \sum_{k=1}^{n} q(|x_k|) = |\alpha| \leq q(|z|),$$

wobei $\alpha \in \mathbb{N}_{>0}^*$ eine Sequenz von *paarweise verschiedenen* Indizes ist, und

$$\{y'[\alpha] \mid y' \in \mathit{set}\text{-}R(z)\} = \{y_1 \circ y_2 \circ \cdots \circ y_n \mid (\forall k \leq n).\, y_k \in \mathit{set}\text{-}R(x_k)\}.$$

(3) Die Relation R ist *coupable* wenn es eine Funktion $\mathit{cpl} \in \mathrm{FP}$ gibt sodass

$$\mathit{cpl}(x, (i_1, \ldots, i_n), (j_1, \ldots, j_n)) = (z, \alpha) \text{ wobei } x \in \Sigma^*, \; 1 \leq i_1, \ldots, i_n, j_1, \ldots, j_n \leq q(|x|) \text{ und } |\alpha| = q(|x|),$$

wobei wieder $\alpha \in \mathbb{N}_{>0}^*$ eine Sequenz von *paarweise verschiedenen* Indizes ist, und

$$\{y'[\alpha] \mid y' \in \mathit{set}\text{-}R(z)\} = \{y \mid y \in \mathit{set}\text{-}R(x) \text{ und } (\forall k \leq n)(y[i_k] \neq y[j_k])\}.$$

◁

Satz 3.15. *Sei R eine strenge NP-Relation. Folgende Aussagen sind äquivalent:*

(1) R ist eine universelle Relation.

(2) R hat einen building block*, ist* joinable *und ist* coupable*.*

Diese Äquivalenz gilt nur im unrelativierten Fall.

4 Suchprobleme und die Hypothese Q im Kontext des Pudlákschen Programms

- Will Q in den Pudlák-Baum einordnen: dafür ist es notwendig, diese ordentlich zu relativieren. Insb. will ich zeigen, dass einige bisherige Resultate natürlicherweise auf „Standardbeweissysteme“ vollständiger Mengen übertragen (nicht nur das Standardbeweissystem für SAT).

Definition 4.1 (Levin-Paddability)**.** Eine NP-Relation R ist *Levin-paddable* wenn Funktionen $pad \in \mathrm{FP}$ und $padsol \in \mathrm{FP}$ existieren, sowie ein Polynom r sodass

(1) $x \in \mathrm{Proj}(R) \iff pad(x, 1^n) \in \mathrm{Proj}(R)$,

(2) $(pad(x, 1^n), y) \in R \implies (x, padsol(x, 1^n, y)) \in R$,

(3) $r(|pad(x, 1^n)|) \geq n$. (Funktion pad ist ehrlich bzgl. der zweiten Komponente.) $\lhd$

Definition 4.2 (Standardbeweissystem)**.** Sei R eine NP-Relation. Wir definieren bezüglich R das *Standardbeweissystem* std_R für $\mathrm{Proj}(R)$ wie folgt:

$$std_R(w) = \begin{cases} x & \text{wenn } w = (x, y) \text{ und } (x, y) \in R, \\ \bot & \text{sonst.} \end{cases} \qquad \lhd$$

Beobachtung 4.3. *Für jede NP-Relation R ist das Standardbeweissystem std_R ehrlich.*

Beweis. Sei q die Zertifikatsschranke von R, und sei $w = (x, y)$ gegeben sodass $std_R(x, y) = x$. An dieser Stelle müssen wir auf die konkrete Codierung von Beweisen $w = (x, y)$ eingehen. Wie in **??** beschrieben, codieren wir Tupel in einer solchen Weise sodass

$$|w| = |(x, y)| = 2(|x| + |y| + 2) = 2|x| + 2|y| + 4.$$

Da $(x, y) \in R$ gilt für y auch $|y| \leq q(|x|)$. Damit also

$$|w| \leq 2|x| + 2q(|x|) + 4 \leq q'(|x|) = q'(|std_R(w)|).$$

für ein geeignetes Polynom q', wie gewünscht. $\square$

Folgendes Lemma ist eine Generalisierung von Messner (2001, Thm. 5.2)

Lemma 4.4. *Sei R eine NP-Relation die Levin-paddable ist. Folgende Aussagen sind äquivalent:*

(1) Das Standardbweissystem std_R bzgl. R ist p-optimal.

(2) Für alle NTM N (ohne Laufzeitbeschränkung) mit $L(N) = \mathrm{Proj}(R)$ lassen sich akzepierende Rechenwege von N in Zertifikate umrechnen: es existiert eine Funktion $h \in \mathrm{FP}$ sodass

$$N(x) \text{ akz. mit Rechenweg } \alpha \implies (x, h(x, \alpha)) \in R.$$

(3) Für alle NPTM N mit $L(N) = \mathrm{Proj}(R)$ lassen sich akzepierende Rechenwege von N in Zertifikate umrechnen: es existiert eine Funktion $h \in \mathrm{FP}$ sodass

$$N(x) \text{ akz. mit Rechenweg } \alpha \implies (x, h(x, \alpha)) \in R.$$

Beweis. (1)⇒(2): Für NTM f_N können wir das assoziierte Beweissystem

$$f_N(x, \alpha) = \begin{cases} x & N(x) \text{ akz. mit Rechenweg } \alpha \\ \bot & \text{sonst} \end{cases}$$

angeben. Es ist klar, dass f_N ein Beweissystem für $\mathrm{Proj}(R)$ ist. Aus der p-Optimalität von std_R gilt nun $std_R \leq^{\mathrm{p}} f_N$, bzw. p-simuliert das Standardbeweissystem das Beweissystem f_N. Damit existiert eine Funktion $h \in \mathrm{FP}$ sodass

$$std_R(h(x, \alpha)) = f_N(x, \alpha).$$

Diese Funktion h erfüllt genau die Eigenschaften von (2): Wir haben

$$\begin{aligned} &N(x) \text{ akz. mit Rechenweg } \alpha \implies h(x, \alpha) = x \\ &\implies std_R(h(x, \alpha)) = x \\ &\implies (x, h(x, \alpha)) \in R, \end{aligned}$$

wie gewünscht.

(2)⇒(3): Klar.

(3)⇒(1): Angenommen (3) gilt. Seien pad, $padsol$ die entsprechenden Funktionen, welche die Levin-Paddability von R realisieren. Das Polynom r sei so gewählt dass $r(|pad(x, 1^n)|) \geq n$ (vgl. 4.1(3)).

Wir wollen nun zeigen, dass std_R auch p-optimal ist. Sei hierfür f ein beliebiges Beweissystem für $\mathrm{Proj}(R)$. Wir zeigen nun, dass $std_R \leq^{\mathrm{p}} f$. Seien pad, $padsol$ die entsprechenden Padding-Funktionen von R. Definiere nun

$$f'(w) = \begin{cases} pad(x, 1^{|w|}) & \text{falls } w = 1z \text{ und } f(z) = x, \\ x & \text{falls } w = 0z \text{ und } std_R(z) = x, \\ \bot & \text{sonst.} \end{cases}$$

Es ist leicht zu sehen, dass f' ehrlich ist: es ist ehrlich für Eingaben $0z$, denn das Standardbeweissystem std_R ist ehrlich nach Beobachtung 4.3. Es ist ehrlich für Eingaben $w = 1z$, denn

$$|1z| = |w| \leq r(|\underbrace{pad(x, 1^{|w|})}_{f'(1z)}|) = r(|f'(|w|)|).$$

Sei im Folgenden dann das Polyom r' so gewählt, dass $|w| \leq r'(|f'(w)|)$ gilt.

Definiere nun die NPTM $N_{f'}$ welche auf Eingabe x erst nichtdeterministisch einen Beweis w, $|w| \leq r'(|x|)$ rät, und genau dann akzeptiert falls $f'(w) = x$. Es ist klar, dass $L(N_{f'}) = \mathrm{Proj}(R)$. Nach Voraussetzung (3) gibt es also nun eine Funktion $h \in \mathrm{FP}$ sodass

$$N_{f'}(x) \text{ akz. mit Rechenweg } \alpha \implies (x, h(x, \alpha)) \in R. \tag{4.1}$$

Jetzt können wir $std_R \leq^{\mathrm{p}} f$ zeigen: sei z ein f-Beweis für x, d.h. $f(z) = x$. Wir wissen, dass $f'(1z) = pad(x, 1^{|1z|}) = x'$. Daher können wir aus z einen Rechenweg α_z konstruieren, sodass $N_{f'}(x')$ akzeptiert, nämlich jener der den f'-Beweis $1z$ rät. Die Abbildung $z \mapsto \alpha_z$ lässt sich in Polynomialzeit leisten.

Nun gilt

$$\begin{aligned} & N_{f'}(x') \text{ akz. mit } \alpha_z \implies (x', \underbrace{h(x', \alpha_z)}_{y'}) \in R \text{ nach (4.1)} \\ & \implies (pad(x, 1^{|1z|}), y') \in R \text{ mit } y' = h(x', \alpha_z) \text{ und obiger Def. von } x' \\ & \implies (x, \underbrace{padsol(x, 1^{|1z|}, y')}_{y}) \in R \\ & \implies std(x, y) = x \text{ mit } y = padsol(x, 1^{|1z|}, y') \end{aligned}$$

und wir haben aus dem f-Beweis z für x einen std_R-Beweis (x, y) für x bestimmt. Es ist klar, dass die Übersetzung $z \mapsto (x, y)$ in Polynomialzeit möglich ist. □

Folgendes Lemma ist eine Generalisierung von Fenner u. a. (2003, Thm. 2)

Lemma 4.5. *Sei R eine $\leq^{\mathrm{p}}_{\mathrm{L}}$-vollständige NP-Relation, mit der zusätzlichen Eigenschaft dass für die jeweilige entsprechende Problem-Reduktionsfunktion $f\colon Q \to R$ für $Q \leq^{\mathrm{p}}_{\mathrm{L}} R$ immer gilt, dass f ehrlich ist. Folgende Aussagen sind äquivalent:*

(1) Für alle NPTM N mit $L(N) = \mathrm{Proj}(R)$ lassen sich akzepierende Rechenwege von N in Zertifikate umrechnen: es existiert eine Funktion $h \in \mathrm{FP}$ sodass

$$N(x) \text{ akz. mit Rechenweg } \alpha \implies (x, h(x, \alpha)) \in R.$$

(2) Für alle NPTM N mit $L(N) = \Sigma^*$ lassen sich aus Eingabe x Rechenwege von $N(x)$ effizient bestimmen: es existiert $r \in \mathrm{FP}$ sodass $N(x)$ auf Rechenweg $r(x)$ akzeptiert. (Das ist die Aussage Q*.)*

Beweis. (2)⇒(1): Sei R eine beliebige NP-Relation mit Zertifikatsschranke q, und sei N eine beliebige NPTM mit $L(N) = \mathrm{Proj}(R)$. Definiere nun die NPTM $N'(w)$ wie folgt:

```
1 wenn w nicht von der Form (x, α) dann akzeptiere
2 (x, α) ← w
3 wenn N(x) akzeptiert nicht auf Rechenweg α dann akzeptiere
4 sonst
  | (Ab hier gilt x ∈ Pr(R), also auch set-R(x) ≠ ∅)
5 | Rate nichtdeterministisch y ∈ Σ^{≤q(|x|)}
6 | Akzeptiere genau dann wenn (x, y) ∈ A.
```

Es ist nun leicht zu sehen dass $L(N') = \Sigma^*$. Nach Voraussetzung (2) existiert eine Funktion $r \in \mathrm{FP}$ sodass für alle x die Maschine $N(w)$ auf Rechenweg $r(w)$ akzeptiert. Nun gilt

$$\begin{aligned} & N(x) \text{ akz. mit Rechenweg } \alpha \\ & \implies N'(x, \alpha) \text{ akz. in Z. 6} \\ & \implies N'(x, \alpha) \text{ akz. mit Rechenweg } r(x, \alpha) \text{ in Z. 6,} \end{aligned}$$

und aus diesem Rechenweg $r(x, \alpha)$ kann effizient der geratene Zeuge $y \in set\text{-}R(x)$ aus Z. 5 ausgelesen werden. Da $r \in \mathrm{FP}$ existiert also auch ein $h \in \mathrm{FP}$ sodass $h(x, \alpha)$ genau diesen geratenen Zeugen y berechnet. Wir haben dann also

$$\implies (x, h(x, \alpha)) \in R,$$

wie gewünscht.

(1)⇒(2): Sei R eine $\leq^{\mathrm{p}}_{\mathrm{L}}$-vollständige NP-Relation unter ehrlichen Problem-Reduktionsfunktionen, und Zertifikatsschranke p. Sei nun N eine NPTM mit $L(N) = \Sigma^*$. Betrachte die entsprechende NP-Relation

$$R_N = \{(x, \alpha) \mid N(x) \text{ akz. mit Rechenweg } \alpha\}$$

Da R ja vollständig ist, gilt $R_N \leq^{\mathrm{p}}_{\mathrm{L}} R$ via $f, g \in \mathrm{FP}$ und (nach Voraussetzung) ist f ehrlich; es existiert ein Polynom q sodass $q(|f(x)|) \geq |x|$.

Definiere nun die folgende NPTM $N'(w)$:

1 Rate nichtdeterministisch $x \in \Sigma^{\leq q(|w|)}$
2 **wenn** $f(x) = w$ **dann** akzeptiere
(Ab hier kann man x wegwerfen)
3 Rate nichtdeterministisch $y \in \Sigma^{\leq p(|w|)}$
4 Akzeptiere genau dann wenn $(w, y) \in R$.

Wichtig: Ehrlichkeit ist hier notwendig, denn wir *müssen* das Urbild raten, und können das nicht als zweite Eingabe mitführen, damit $L(N') = \mathrm{Proj}(R)$. Auch alternative Wege über Beweissysteme laufen darauf hinaus, Levin-Paddability vorauszusetzen, woraus ohenhin wieder Ehrlichkeit folgt.

Wir zeigen nun, dass $L(N') = \mathrm{Proj}(R)$. Wir müssen hierfür nur die Fälle betrachten, wenn $N'(w)$ in Z. 2 akzeptiert. In diesem Fall gilt $f(x) = w$, und wir haben

$$x \in \Sigma^* \implies x \in \mathrm{Proj}(R_N) \implies f(x) \in \mathrm{Proj}(R) \implies w \in \mathrm{Proj}(R),$$

wie gewünscht.

Nach Voraussetzung (1) gilt nun also, dass eine Funktion $h \in \mathrm{FP}$ existiert sodass

$$N'(w) \text{ akz. mit Rechenweg } \alpha \implies (w, h(w, \alpha)) \in R.$$

Beobachte wie für $N'(f(x))$ immer ein trivialer akzeptierender Rechenweg α_x existiert: nämlich jener, welcher in Z. 1 das Urbild x rät. Beobachte dass die Umformung $x \mapsto \alpha_x$ in Polynomialzeit möglich ist.

Um nun (2) zu zeigen müssen wir aus $x \in \Sigma^*$ effizient einen akzeptierenden Rechenweg für N bestimmen. Wir haben

$$\begin{aligned}
&N'(f(x)) \text{ akz. mit Rechenweg } \alpha_x \implies (f(x), h(f(x), \alpha_x)) \in R \\
&\implies (x, \underbrace{g(h(f(x), \alpha_x))}_{r(x)}) \in R_N \quad \text{nach Translationsfunktion } g \\
&\implies N(x) \text{ akz. mit Rechenweg } r(x)
\end{aligned}$$

mit $r \in \mathrm{FP}$, $r(x) = g(h(f(x), \alpha_x))$, wie gewünscht. □

Lemma 4.6. *Die in Lemma 4.4 und 4.5 genannten Voraussetzungen an die NP-Relation R werden von allen solchen R erfüllt, die $\leq^{\mathrm{p}}_{\mathrm{L}}$-vollständig sind und Levin-paddable sind.*

Beweis. Es ist sofort klar, dass R die Voraussetzungen von Lemma 4.4 erfüllt. Es bleibt nur zu zeigen, dass für jede NP-Relation Q eine $\leq^{\mathrm{p}}_{\mathrm{L}}$-Reduktion angegeben werden kann, bei dem die Problem-Reduktionsfunktion ehrlich ist. Wir nutzen hierbei aus, dass R eine Levin-paddable Relation ist.

Nachdem R vollständig ist, gilt $Q \leq^{\mathrm{p}}_{\mathrm{L}} R$; sei $f, g \in \mathrm{FP}$ die Reduktions- bzw. Translationsfunktion welche diese Reduktion realisieren. Wir werden nun Funktionen $f', g' \in \mathrm{FP}$ angeben, welche die gleiche Reduktion realisieren, aber f' ehrlich, wie gewünscht.

Sei $pad, padsol$ die zu R zugehörigen Padding-Funktionen. Definiere

$$f'(x) = pad(f(x), 1^{|x|}).$$

Es gilt

$$x \in \mathrm{Proj}(Q) \iff f(x) \in \mathrm{Proj}(R) \iff pad(f(x), 1^{|x|}) = f'(x) \in \mathrm{Proj}(R),$$

wobei erste Implikation die Eigenschaft der Reduktionsfunktion f ist, und die zweite aus der Definition von Levin-Paddability folgt. Aus der Definition von Levin-Paddability folgt auch $r(|f'(x)|) \geq |x|$ für ein geeignetes Polynom r, und damit ist auch f' ehrlich.

Definiere

$$g'(x, z) = g(x, padsol(f(x), 1^{|x|}, z)).$$

Sei nun $(f'(x), z) \in R$. Die Funktion g' berechnet nun ein Zertifikat y für x: Wir haben $(pad(f(x), 1^{|x|}), z) \in R$, also gilt nach Levin-Paddability dass

$$(f(x), padsol(f(x), 1^{|x|}, z)) \in R,$$

und nach Definition der Translationsfunktion g gilt dann

$$(x, g(x, padsol((f(x), 1^{|x|}, z))) \in Q,$$

und das ist genau $(x, g'(x, z)) \in Q$, wie gewünscht. □

4.1 Welche Suchprobleme sind paddable?

Beobachtung 4.7. *Die kanonische Levin-vollständige NP-Relation* `rKAN` *ist Levin-paddable.*

Beobachtung 4.8. *(1) Gilt* `rKAN` $\leq^{\mathrm{p}}_{\mathrm{L}} R$*, und ist die zugehörige Reduktionsfuktion f ehrlich, dann ist R Levin-paddable*

(2) Jede $\leq^{\mathrm{p}}_{\mathrm{L,inv}}$-vollständige NP-Relation R ist auch Levin-paddable.

Korollar 4.9. *Jede $\leq^{\mathrm{p}}_{\mathrm{L,inv}}$-vollständige Relation R erfüllt die in Lemma 4.4 und 4.5 genannten Voraussetzungen an die NP-Relation R.*

Das sind im unrelativierten Fall u.a. `rSAT`*,* `rSETCOVER`*,* `rVERTEXCOVER`*,* `rCLIQUE`*,* `r3COLORABILITY`*.*

So das Textbook von Goldreich (2008).

Beweis zu Beobachtung 4.8. Aussage (2) folgt unmittelbar aus (1): Wir haben $\text{rKAN} \leq^{\text{p}}_{\text{L,inv}} R$ und damit ist die entsprechende Reduktionsfunktion f p-invertierbar, und damit ehrlich.

Für (1) nutzen wir die Levin-Paddability von rKAN aus: übersetze Instanz x von R nach rKAN, padde dort hoch, und überetze zu R-Instanz x' zurück. Ist dann y' ein Zertifikat für x', dann lässt sich dies auf ähnlichem Weg wieder zu einem Zertifikat für x zurückrechnen.

Seien f, g die Reduktions- bzw. Translationsfunktion, welche $\text{rKAN} \leq^{\text{p}}_{\text{L}} R$ bezeugen, und seinen analog f', g' jene Funktionen, welche $R \leq^{\text{p}}_{\text{L}} \text{rKAN}$ bezeugen. Erstere existieren nach Voraussetzung, zweitere existieren weil $\text{rKAN} \leq^{\text{p}}_{\text{L}}$-vollständig ist. Nach Voraussetzung ist f ehrlich. Und nach Beobachtung 4.7 existieren für rKAN Padding-Funktionen pad_{rKAN}, $padsol_{\text{rKAN}}$. Sei q ein entsprechendes Polynom mit $q(|pad_{\text{rKAN}}(x, 1^n)|) \geq n$, $q(|f(x)|) \geq |x|$.

Definiere nun

$$pad_R(x, 1^n) = f(pad_{\text{rKAN}}(f'(x), 1^n)).$$

Die Zugehörigkeit zu $\text{Proj}(R)$ bleibt erhalten:

$$\begin{aligned} x \in \text{Proj}(R) &\iff f'(x) \in \text{KAN} \iff pad_{\text{rKAN}}(f'(x), 1^n) \in \text{KAN} \\ &\iff f(pad_{\text{rKAN}}(f'(x), 1^n)) \in \text{Proj}(R) \iff pad_R(x, 1^n) \in \text{Proj}(R). \end{aligned}$$

Ferner gilt

$$\begin{aligned} &q(q(|pad_R(x, 1^n)|)) \\ &= q(q(|f(pad_{\text{rKAN}}(f'(x), 1^n)|)) \\ &\geq q(|pad_{\text{rKAN}}(f'(x), 1^n)|) \\ &\geq n. \end{aligned}$$

und damit ist pad_R wie gewünscht ehrlich bzgl. n (mit Polynom $q \circ q$).

Es verbleibt noch die Funktion $padsol_R$. Nehme hierfür an dass wir ein y' haben mit $(pad_R(x, 1^n), y') \in R$. Wir können über g, g' das Zertifikat y' zu Zertifikat y mit $(x, y) \in R$ zurück übersetzen: Sei $p = pad_{\text{rKAN}}(f'(x), 1^n)$, dann gilt

$$(f(p), y') \in R \implies (p, \underbrace{g(p, y')}_{z}) \in \text{rKAN}.$$

Definere $z = g(p, y')$. Nun haben wir

$$\begin{aligned} &(p, z) = (pad_{\text{rKAN}}(f'(x), 1^n), z) \in \text{rKAN} \\ &\quad \implies (f'(x), \underbrace{padsol_{\text{rKAN}}(f'(x), 1^n, z)}_{z'}) \in \text{rKAN} \end{aligned}$$

und mit $z' = padsol_{\text{rKAN}}(f'(x), 1^n, z)$ gilt

$$(f'(x), z') \in \text{rKAN} \implies (x, \underbrace{g'(x, z')}_{y}) \in R.$$

Es ist leicht zu sehen, dass sich eine Funktion $padsol_R \in \text{FP}$ angeben kann, die aus $x, 1^n$ dieses entsprechende y berechnen kann. □

Beobachtung 4.10. *Jede NP-Relation mit einem* building block *und die* joinable *ist, ist auch Levin-paddable.*

Korollar 4.11. *Im unrelativierten Fall erfüllt jede universelle Relation R die in Lemma 4.4 und 4.5 genannten Voraussetzungen an die NP-Relation R.*

Das sind u.a. rSAT, rHAM, rINDSET, rKNAPSACK, rMAXCUT.

Beweis zu Beobachtung 4.10. Sei R eine NP-Relation, mit zugehörigem Polynom q, welches die Zertifikatsgröße spezifiziert. Zur Erinnerung, dieses Polynom ist streng monoton steigend, und aus $(x, y) \in R$ folgt $|y| = q(|x|)$. Wir zeigen zunächst, wie wir für beliebige Instanz x und $n \in \mathbb{N}$ auf eine Instanz x' hochpadden, in dem Sinne dass $q(|x'|) \geq n$.

Nach Voraussetzung hat die Relation R einen *building block block*. Es lässt sich leicht aus der Definition eines *building block* ableiten, dass $|block| > 0$ und $block \in \text{Proj}(R)$. Damit gilt auch dass die Zertifikate y zu *block* die Länge $l = q(|block|) \geq |block| \geq 1$ haben.

Nach Voraussetzungen ist die Relation R auch *joinable*, das heißt wir haben eine Funktion $join \in \text{FP}$. Sei

$$(x', \delta) = join(x, \underbrace{block, block, ..., block}_{n \text{ mal}}).$$

Wir werden nun über die Länge $|\delta|$ auf die Länge von Zertifikaten zu x' schließen, und damit $|x'|$ beschränken. Nach Definition ?? gilt

$$|\delta| = q(|x|) + q(n) \cdot q(|block|) = q(|x|) + q(n) \cdot l \geq n.$$

Beob. dass unter Definition ?? alle Zertifikate y' für x' die feste Länge $q(|x'|)$ haben. Zur Erinnerung: wir haben

$$\begin{aligned} \{y'[\delta] \mid y' \in \Sigma^{q(|x'|)}, (x', y') \in R\} = \{yy_1y_2 \cdots y_n \mid y \in \Sigma^{q(|x|)}, y_1, y_2, ... \in \Sigma^l, \\ (x, y), (block, y_1), (block, y_2), ... \in R\} \end{aligned} \quad (4.2)$$

Die Sequenz δ besteht nach Definition aus paarweise verschiedenen Indizes, daher können wir argumentieren, dass auch alle Zertifikate y' (mit vorgegebener Länge $q(|x'|)$) mindestens die Länge $|\delta|$ haben. Damit gilt

$$q(|x'|) \geq |\delta| \geq n$$

wie gewünscht.

Es sieht nicht danach aus, dass wir über die Anzahl an Zertifikaten für x' die Länge abschätzen können: ist $x \notin \text{Proj}(R)$ dann hätte auch x' keine Zertifikate, und könnte sich erlauben entsprechend kurz zu sein.

Sei nun *pad* genau jene polynomialzeit-berechenbare Funktion, die aus x und 1^n die Instanz x' konstruiert:

$$pad(x, 1^n) = x' \quad \text{wobei } (x', \delta) = join(x, \underbrace{block, block, \dots, block}_{q(n) \text{ mal}}).$$

Dann gilt schon sofort, dass $q(|pad(x, 1^n)|) = q(|x'|) \geq n$ wie gewünscht.

Wir zeigen jetzt, dass die Zugehörigkeit zu $\text{Proj}(R)$ erhalten bleibt: Gilt $x \notin \text{Proj}(R)$, dann ist die rechte Menge in (4.2) leer, also auch die linke Menge und damit $x' = pad(x, 1^n) \notin \text{Proj}(R)$. Falls anders herum $x \in \text{Proj}(R)$, dann ist die rechte Menge nicht leer, existiert ja ein Zertifikat y für x und je ein weiters y_i für *block*. Also ist auch die linke Menge nicht leer, damit $pad(x, 1^n) \in \text{Proj}(R)$.

Die noch verbleibende Funktion *padsol* ist durch die bitweise Projektion durch δ leicht möglich:

$$padsol(x, 1^n, y') = y'[\delta[1 : q(|x|)]] \quad \text{wobei } (\cdot, \delta) = join(x, \underbrace{block, block, \dots, block}_{n \text{ mal}}).$$

Wir verifizieren: Sei $(pad(x, 1^n), y') \in R$, dann ist nach (4.2) $y'[\delta] = yy_1y_2\cdots$ wobei $y \in \Sigma^{q(|x|)}$, $(x, y) \in R$. Wir haben

$$padsol(x, 1^n, y') = y'[\delta[1 : q(|x|)]] = (yy_1y_2\cdots)[1 : q(|x|)] = y$$

und damit $(x, padsol(x, 1^n, y')) = (x, y) \in R$, wie gewünscht. □

4.2 Hypothese Q und die Vollständigkeit von Suchproblemen

Satz 4.12 (Äquivalente Formulierungen der Hypothese Q; Fenner u. a. 2003; Messner 2001)**.** *Folgende Aussagen sind äquivalent:*

(1) Hypothese Q*: Für jede NPTM N mit $L(N) = \Sigma^*$ existiert eine Funktion $g \in \text{FP}$ sodass für alle x das Bild $g(x)$ eine akzeptierende Berechnung von $N(x)$ ist.*

(2) $\text{NPMV}_t \subseteq_{\text{c}} \text{FP}$

(3) $\text{P} = \text{NP} \cap \text{coNP}$ *und* $\text{NPMV}_t \subseteq_{\text{c}} \text{NPSV}_t$

(4) Jede surjektive ehrliche Funktion $f \in \text{FP}$ ist p-invertierbar.

(5) Für jede Menge $L \in \text{P}$ und jede NPTM N mit $L(N) = L$ existiert eine Funktion $h \in \text{FP}$ mit

$$x \in L \implies N(x) \text{ akz. mit Rechenweg } h(x).$$

(6) Für jedes Paar von NP-Relationen A, B und jede Funktion $f \in \text{FP}$ gilt:

$$\text{Proj}(A) \leq^{\text{p}}_{\text{m}} \text{Proj}(B) \text{ via } f \iff A \leq^{\text{p}}_{\text{L}} B \text{ via Reduktionsfunktion } f.$$

(7) Für jedes Beweissystem h gilt: h ist optimal $\iff$ h ist p-optimal.

(8) Es existiert eine $\leq^{\text{p}}_{\text{L}}$-vollständige Levin-paddable NP-Relation R sodass für alle NPTM N mit $L(N) = \text{Proj}(R)$ gilt: es existiert eine Funktion $h \in \text{FP}$ mit

$$N(x) \text{ akz. mit Rechenweg } \alpha \implies (x, h(x, \alpha)) \in R.$$

(9) Es existiert eine $\leq^{\text{p}}_{\text{L}}$-vollständige Levin-paddable NP-Relation R für welche das Standardbeweissystem std_R p-optimal ist.

(10) Es existiert eine $\leq^{\text{p}}_{\text{L,1,inv}}$-vollständige NP-Relation R sodass für jede Menge $S \in \text{P}$ mit $S \subseteq \text{Proj}(R)$ gilt: es existiert eine Funktion $g \in \text{FP}$ sodass

$$x \in S \implies (x, g(x)) \in R.$$

Beweis. 1. (1) $\iff$ (2) $\iff$ (3) $\iff$ (4) $\iff$ (5): nach Fenner u. a. (2003, Thm. 2).

2. (1) $\iff$ (8) $\iff$ (9): nach Lemma 4.5 und 4.4.

3. (5) $\implies$ (10): Wir zeigen eine stärkere Variante von (10), welche sich über *alle* NP-Relationen R estreckt (und damit auch über $\leq^{\text{p}}_{\text{L,1,inv}}$-vollständige `rKAN`, wie von (10) gefordert). Sei R eine beliebige NP-Relation, wobei Polynom q die Zertifikatsgröße beschränkt. Sei nun $S \subseteq \text{Proj}(R)$ mit $S \in \text{P}$. Definiere die NPTM N, welche auf Eingabe x folgendes leistet: teste zuerst ob $x \in S$; falls nicht, lehne sofort ab. Rate dann ein $y \in \Sigma^{\leq q(|x|)}$ und akzeptiere genau dann wenn $(x, y) \in R$.

Klar ist, dass $L(N) = S$. Nach (5) existiert nun eine Funktion $h \in \text{FP}$, die für $x \in S$ einen akzeptierenden Rechenweg $h(x)$ von $N(x)$ ausgibt. Wir können sogar aus $h(x)$ das geratene Zertifikat y extrahieren. Es ist daher leicht eine Funktion $g \in \text{FP}$ anzugeben für die $(x, g(x)) \in R$ für alle $x \in S$.

4. (10) $\implies$ (5): Sei $L \in \text{P}$ und sei N eine NPTM mit $L(N) = L$, wobei das Polynom q die Laufzeit beschränkt. Wir wollen eine Funktion $h \in \text{FP}$ definieren sodass $h(x)$ ein akzeptierender Rechenweg von $N(x)$ für $x \in L$ ist. Definiere die NP-Relation

$$Q = \{(x, y) \mid N(x) \text{ akzeptiert mit Rechenweg } y \in \Sigma^{\leq q(|x|)}\}.$$

Nachdem (10) gilt, haben wir eine $\leq^{\mathrm{p}}_{\mathrm{L},1,\mathrm{inv}}$-vollständige NP-Relation R. Damit gilt $Q \leq^{\mathrm{p}}_{\mathrm{L}} R$ mittels Reduktions- bwz. Translationsfunktion $f, k \in FP$. Insbesondere existiert eine Inverse $f^{-1} \in \mathrm{FP}$ zu f.

Sei $S = f(L)$ die Bildmenge der Elemente aus L, also

$$S = \{f(x) \mid x \in L\}.$$

Es ist leicht zu sehen dass $S \subseteq \mathrm{Proj}(R)$. Außerdem ist $S \in \mathrm{P}$: teste $z \in S$ indem getestet wird ob $f^{-1}(z) = x \neq \bot$ und ob $x \in L$.

Damit sind die Voraussetzungen von (10) erfüllt, und es existiert eine Funktion $g \in \mathrm{FP}$ sodass $(z, g(z)) \in R$ für alle $z \in S$. Damit gilt

$$\begin{aligned} x \in L &\implies f(x) \in S \implies (f(N, x), g(f(x))) \in R \\ &\implies ((x), k(g(f(x)))) \in Q \\ &\implies N(x) \text{ akz. mit Rechenweg } \underbrace{k(g(f(x)))}_{h(x)}. \end{aligned}$$

Definiere nun die gesuchte Funktion $h \in \mathrm{FP}$ mit $h(x) = k(g(f(x)))$. Damit gilt für alle $x \in L$ dass $N(x)$ mit Rechenweg $h(x)$ akzeptiert, wie gewünscht.

5. (1) $\implies$ (6): Die Richtunng von rechts nach links ist klar. Für die andere Richtung sei $\mathrm{Proj}(A) \leq^{\mathrm{p}}_{\mathrm{m}} \mathrm{Proj}(B)$ mit A, B NP-Relationen. Sei q hierbei das Polynom was die Zertifikatslänge in A begrenzt. Wir wollen nun eine Levin-Reduktion von A auf B angeben. Sei $f \in \mathrm{FP}$ die Funktion, welche die Reduktion $\mathrm{Proj}(A) \leq^{\mathrm{p}}_{\mathrm{m}} \mathrm{Proj}(B)$ realisiert.

Definiere folgende NPTM N, die wie folgt auf Eingabe w arbeitet:

```
1 wenn w nicht von der Form (x, y') dann akzeptiere
2 (x, y') ← w
3 wenn (f(x), y') ∉ B dann akzeptiere
4 sonst
    (Ab hier gilt f(x) ∈ Pr(B) und x ∈ Pr(A))
5   Rate nichtdeterministisch y ∈ Σ^{q(|x|)}
6   Akzeptiere genau dann wenn (x, y) ∈ A.
```

Es ist leicht zu sehen dass $L(N) = \Sigma^*$. Nach (1) existiert nun also eine Funktion g sodass, für alle w, $g(w)$ eine akzeptierender Rechwenweg von $N(w)$ ist.

Damit lässt die Levin-Reduktion von A auf B angeben: wähle f als Reduktionsfunktion, und definiere g' als Translationsfunktion, welche aus dem akzeptierenden Rechenweg $g(x, y')$ das geratene Zertifikat y von Zeile 5 ausliest. Dann gilt

$$\begin{aligned} &(f(x), y') \in B \implies N(x, y') \text{ akz. auf einem Rechenweg in Z. 6, ratet } y \\ &\implies (x, y) = (x, g'(x, y')) \in A \end{aligned}$$

wie gewünscht. Wir haben $A \leq^{\mathrm{p}}_{\mathrm{L}}$ via f, g'.

6. (6) $\implies$ (8): Sei R eine beliebige $\leq^{\mathrm{p}}_{\mathrm{L}}$-vollständige und Levin-paddable NP-Relation (diese existiert immer) und sei N eine NPTM N mit $L(N) = \mathrm{Proj}(R)$, wobei das Polynom q die Laufzeit von N beschränkt. Definiere

$$Q = \{(x, y) \mid N(x) \text{ akzeptiert mit Rechenweg } y \in \Sigma^{\leq q(|x|)}\}.$$

Es ist klar, dass $\mathrm{Proj}(R) \leq^{\mathrm{p}}_{\mathrm{m}} \mathrm{Proj}(Q) = L(N) = \mathrm{Proj}(R)$ über die Identiätsfunktion. Nach (6) gilt nun auch $R \leq^{\mathrm{p}}_{\mathrm{L}} Q$. mit Identitätsfunktion als Reduktionsfunktion und $h \in \mathrm{FP}$ als Translationsfunktion. Damit gilt

$$N(x) \text{ akz. mit RW } \alpha \implies (x, \alpha) \in Q \implies (x, h(x, \alpha)) \in R$$

wie gewünscht.

7. (2) $\implies$ (7): Die Richtung von rechts nach links ist klar. Sei für die andere Richtung h ein optimales Beweissystem für eine Menge L. Wir wollen zeigen, dass h auch p-optimal ist. Sei dafür g ein weiteres Beweissystem für L. Nach Voraussetzung haben wir $g \leq h$, das heißt es existiert eine (nicht notwendigerweise effiziente) Funktion f sodass $g(w) = h(f(w))$, und gleichzeitig ist $|f(w)| \leq q(|w|)$ für ein geeignetes Polynom q.

Betrachte folgende Multifunktion f':

$$f'(w) \mapsto y \iff \exists y \in \Sigma^{\leq q(|w|)}, g(w) = h(y).$$

Es lässt sich leicht zeigen, dass $f' \in \mathrm{NPMV}$, über einen geeigneten NPTM-Transduktor. Es ist sogar $f' \in \mathrm{NPMV}_t$, denn für jedes w mindestens $f(w) \in \mathit{set}\text{-}f'(w)$.

Nach (2) gilt also $f' \in \mathrm{NPMV}_t \subseteq_{\mathrm{c}} \mathrm{FP}$, also existiert eine Funktion $f'' \in \mathrm{FP}$ welche eine Verfeinerung von f' ist. Diese Funktion übersetzt g-Beweise w für x effizient in h-Beweise für x: Sei $g(w) = x$, dann gilt

$$f''(w) = y \quad \text{mit } y \in \Sigma^{\leq q(|w|)}, x = g(w) = h(y)$$

also ist $h(f''(w)) = x$ bzw. $f''(w)$ ein h-Beweis für x, wie gewünscht.

8. (7) $\implies$ (9): klar, denn rKAN ist $\leq^{\mathrm{p}}_{\mathrm{L}}$-vollständig, ist Levin-paddable, und das Standardbeweissystem $\mathit{std}_{\mathsf{rKAN}}$ ist (wie jedes Standardbeweissystem einer NP-Relation) optimal. Zusammen mit (7) ist es also auch p-optimal. □

Satz 4.13 (Formulierung Hypothese Q durch NP-Relationen, ∀-Variante)**.** *Folgende Aussagen sind äquivalent:*

(1) Hypothese Q

(8′) Für jede $\leq^{\mathrm{p}}_{\mathrm{L}}$-vollständige Levin-paddable NP-Relation R, und für alle NPTM N mit $L(N) = \mathrm{Proj}(R)$ gilt: es existiert eine Funktion $h \in \mathrm{FP}$ mit

$$N(x) \text{ akz. mit RW } \alpha \implies (x, h(x, \alpha)) \in R.$$

(9′) Für jede $\leq^{\mathrm{p}}_{\mathrm{L}}$-vollständige Levin-paddable NP-Relationen R ist das Standardbeweissystem std_R p-optimal.

(10′) Für jede NP-Relation P und jede Menge $S \in \mathrm{P}$ mit $S \subseteq \mathrm{Proj}(P)$ gilt: es existiert eine Funktion $g \in \mathrm{FP}$ sodass

$$x \in S \implies (x, g(x)) \in P.$$

Beweis. 1. (8′) $\Longrightarrow$ (1), (9′) $\Longrightarrow$ (1), (10′) $\Longrightarrow$ (1): Die NP-Relation $\mathtt{rKAN}$ ist $\leq^{\mathrm{p}}_{\mathrm{L,1,inv}}$-vollständig (und damit auch Levin-paddable). Gilt also (8′), dann auch die existentielle Variante (8) von vorigem Satz 4.12, und damit (1). Beweise der anderen Implikationen sind analog.

2. (1) $\Longrightarrow$ (8′): Folgt aus Lemma 4.5.

3. (8′) $\Longrightarrow$ (9′): Sei R eine beliebige $\leq^{\mathrm{p}}_{\mathrm{L}}$-vollständige Levin-paddable NP-Relation. Nach Voraussetzung können wir für jede NPTM N, $L(N) = \mathrm{Proj}(R)$ eine Funktion $h \in FP$ angeben sodass

$$N(x) \text{ akz. mit RW } \alpha \implies (x, h(x, \alpha)) \in R.$$

Das ist genau die erste der der äquivalenten Aussagen in Lemma 4.4. Damit gilt auch bezüglich *diesem* gewählten R auch die zweite der äquivalenten Aussagen, nämlich dass std_R p-optimal ist.

4. (1) $\Longrightarrow$ (10′): Im Beweis von Satz 4.12 wurde (1) $\Longrightarrow$ (5) $\Longrightarrow$ (10) gezeigt. Beachte insbesondere dass im Beweis zur zweiten Implikation sogar eine stärkere Aussage bewiesn wurde, welche die Aussage (10) für alle NP-Relationen (und nicht nur vollständige) zeigt. Das entspricht genau der hier aufgestellten Aussage (10′), wie gewünscht.

□

4.3 Karp-Vollständigkeit vs. Levin-Vollständigkeit

Vermutung 4.14 (Karp-vs-Levin-Vermutung; KvL)**.** *Es existiert eine NP-Relation R sodass* $\mathrm{Proj}(R) \leq^{\mathrm{p}}_{\mathrm{m}}$*-vollständig für* NP *ist, aber R ist nicht $\leq^{\mathrm{p}}_{\mathrm{L}}$-vollständig (für alle NP-Relationen).*

Satz 4.15. $\mathsf{KvL} \implies \neg\mathsf{Q}$.

Beweis. Wir zeigen die Kontraposition, und starten mit der Voraussetzung Q. Wir wollen nun ¬KvL zeigen. Sei hierfür R eine beliebige NP-Relation sodass $\mathrm{Proj}(R) \leq^{\mathrm{p}}_{\mathrm{m}}$-vollständig ist. Damit gilt also schon für alle weiteren NP-Relationen A, dass $\mathrm{Proj}(A) \leq^{\mathrm{p}}_{\mathrm{m}} \mathrm{Proj}(R)$. Nach Satz 4.12 gilt also auch die Aussage 4.12(6), und damit $A \leq^{\mathrm{p}}_{\mathrm{L}} R$. Also ist R auch $\leq^{\mathrm{p}}_{\mathrm{L}}$-vollständig, wie gewünscht und wir haben ¬KvL gezeigt. □

Was sind natürlich notwendige Bedingungen für die Hypothese KvL? Diese Frage erscheint tatsächlich wesentlich schwieriger als gedacht. Insbesondere scheint es unklar, ob aus irgend einer von Pudláks Hypothesen die Aussage KvL folgt.

Besonders interessant erscheint aber die Beziehung zur Hypothese ¬Q, also genau die Umkehrung von Satz 4.15. Betrachten wir hier exemplarisch den Fall für Relationen, die SAT bezeugen.

Starten wir mit ¬Q, dann haben wir über Satz 4.12 auch die Negation von Aussage 4.12(8). Nachdem $\mathtt{rSAT}$ eine $\leq^{\mathrm{p}}_{\mathrm{L}}$-vollständige Levin-paddable NP-Relation ist, muss auch eine NPTM N existiere, für die $L(N) = \mathtt{SAT}$, aber für alle Funktionen $h \in \mathrm{FP}$ gilt

$$N(\varphi) \text{ akz. mit Rechenweg } w \implies (\varphi, h(\varphi, w)) \in \mathtt{rSAT}. \tag{4.3}$$

In anderen Worten: es existiert zwar eine NPTM N welche die ($\leq^{\mathrm{p}}_{\mathrm{m}}$-vollständige Menge) SAT entscheidet, aber aus den akzeptierenden Rechenwegen w von $N(x)$ auf $x \in \mathtt{SAT}$ kann nicht effizient eine akzeptierende Belegung für x abgeleitet werden.

Wir können N äquivalent als NP-Relation R_N repräsentieren, mit $(\varphi, w) \in R_N$ genau dann wenn $N(x)$. mit Rechenweg w akzeptiert. Damit kann Gleichung 4.3 so verstanden werden, dass $\mathtt{rSAT} \not\leq^{\mathrm{p}}_{\mathrm{L}} R_N$ *falls die Reduktionsfunktion f die Identitätsfunktion ist.* An dieser Stelle muss erneut hervorgehoben werden, dass im Allgemeinen $\mathtt{rSAT} \leq^{\mathrm{p}}_{\mathrm{L}} R_N$ mit Funktionen f, g gelten kann, notwendig hierfür ist aber dass $f \neq \mathrm{id}$.

Gleichzeitig wäre die Existenz einer solchen Reduktion überraschend. Angenommen $\mathtt{rSAT} \leq^{\mathrm{p}}_{\mathrm{L}} R_N$ wird von f, g ($f \neq \mathrm{id}$) realisiert, dann haben wir nach Definition

$$N(f(\varphi)) \text{ akz. mit Rechenweg } w \implies \varphi \text{ wird von Belegung } g(\varphi, w) \text{ erfüllt.}$$

Einerseits ist es also nicht möglich, aus w effizient eine akzeptierende Belegung für $f(\varphi) \neq \varphi$ zu bestimmen. Andererseits reicht der „Beweis“ w aber aus, um (zusammen mit der Information φ) effizient wieder eine erfüllende Belegung für φ zu berechnen.

Ich vermute, dass solche Funktionen f, g nicht jeweils für alle NPTM N mit $L(N) = \mathtt{SAT}$ existieren können. Tatsächlich können wir die eben formulierte Vermutung auch in der Theorie der Beweissystemen formulieren. Wir definieren zunächst eine abgeschwächte Variante der p-Simulation.

Definition 4.16. Seien h, h' Beweissysteme für L. Das Beweissystem h *p-simuliert effektiv* h' falls Funktionen $f, g \in \mathrm{FP}$ existieren sodass

(1) $x \in L \implies f(x) \in L$,

(2) $h'(w) = f(x) \implies h(g(x, w)) = x$.

Wir schreiben in diesem Fall auch $h \leq^{\mathrm{p}}_{\mathrm{eff}} h'$. $\lhd$

Nicht vergessen: es ist offenbar unklar, wie das Symbol $\leq$ bzgl. Simulation gebraucht wird. Dose/Glaßer würden in der Def. das Ordnungszeichen spiegeln, Krajíček/Pudlák dagegen genau so wie hier, Krajíček im Sammelband dagegen wieder gespiegelt.

In anderen Worten, falls $h \leq^{p}_{\mathrm{eff}} h'$, dann kann h zwar nicht *jeden* h'-Beweis w für $x \in L$ in einen h-Beweis für (das gleiche) x effizient umrechnen, es kann aber zumindest alle *relevanten* h'-Beweise effizient umrechnen, nämlich für jedes $x \in L$ die h'-Beweise für $f(x)$ in h-Beweise für x.

Klar ist: p-Simulation impliziert effektive p-Simulation impliziert Simulation unter Beweissystemen.

Vermutung 4.17 (KvL formuliert unter Beweissystemen)**.** *Das Standardbeweissystem sat für* SAT *kann nicht alle anderen optimalen Beweissysteme für* SAT *effektiv p-simulieren.*

In anderen Worten, es existiert optimales Beweissystem h sodass $std \not\leq^{\mathrm{p}}_{\mathrm{eff}} h$.

Dass die Formulierung der Vermutungen 4.14 und 4.17 äquivalent sind, zeigt folgende Beobachtung:

Beobachtung 4.18. *Folgende Aussagen sind äquivalent:*

(1) Jede NP-Relation R mit $\leq^{\mathrm{p}}_{\mathrm{m}}$-vollständigem $\mathrm{Proj}(R)$, ist auch $\leq^{\mathrm{p}}_{\mathrm{L}}$-vollständig.

(2) Für alle optimalen Beweissysteme h für SAT *gilt $std \leq^{\mathrm{p}}_{\mathrm{eff}} h$.*

Beweis. (1)⇒(2): Wir zeigen, dass *sat* jedes andere optimale Beweissystem h effektiv p-simulieren kann. Nachdem h optimal ist, hat es auch kurze Beweise: für jedes $\varphi \in \mathtt{SAT}$ einen h-Beweis w mit $|w| \leq q(|\varphi|)$. Definiere

$$R_h = \{(\varphi, w) \mid |w| \leq q(|\varphi|), h(w) = \varphi\}.$$

Diese Relation ist offenbar eine NP-Relation und $\mathrm{Proj}(R_h) = \mathtt{SAT}$ und damit ist $\mathrm{Proj}(R_h)$ auch $\leq^{\mathrm{p}}_{\mathrm{m}}$-vollständig.

Nach Voraussetzung ist also R_h auch $\leq^{\mathrm{p}}_{\mathrm{L}}$-vollständig. Insbesondere gilt also auch $\mathtt{rSAT} \leq^{\mathrm{p}}_{\mathrm{L}} R_h$. Damit existieren also Funktionen $f, g \in \mathrm{FP}$ sodass $x \in \mathtt{SAT} \leftrightarrow f(x) \in \mathtt{SAT}$ und

$$(f(\varphi), w) \in R_h \implies (\varphi, g(\varphi, w)) \in \mathtt{rSAT}.$$

Nach Definition gilt also

$$h(w) = f(\varphi) \implies sat(g(\varphi, w) = \varphi,$$

und damit ist $sat \leq^{\mathrm{p}}_{\mathrm{eff}} h$.

(2)⇒(1): Sei R eine NP-Relation wobei $\mathrm{Proj}(R) \leq^{\mathrm{p}}_{\mathrm{m}}$-vollständig ist. Wir zeigenen nun, dass R auch $\leq^{\mathrm{p}}_{\mathrm{L}}$-vollständig ist. Aus der $\leq^{\mathrm{p}}_{\mathrm{m}}$-Vollständigkeit folgt unmittelbar die Existenz einer Reduktionsfunktion f mit

$$\varphi \in \mathtt{SAT} \iff f(\varphi) \in \mathrm{Proj}(R).$$

Definiere

$$h(w) = \begin{cases} \varphi & \text{falls } w = (x, y, \varphi) \text{ und } f(\varphi) = x \text{ und } (x, y) \in R \\ \bot & \text{sonst.} \end{cases}$$

Wir zeigen, dass h ein Beweissystem für SAT ist. Es ist offenbar dass $h \in \mathrm{FP}$. Die Funktion h ist korrekt: wenn $h(x, y, \varphi) = \varphi$ dann ist $f(\varphi) = x \in \mathrm{Proj}(R)$ und nach Eigenschaft von f auch $\varphi \in \mathtt{SAT}$. Die Funktion h ist vollständig: Sei $\varphi \in \mathtt{SAT}$. Dann ist schon $f(\varphi) \in \mathrm{Proj}(R)$ und es gibt ein y mit $(f(\varphi), y) \in R$. Also ist $(f(\varphi), y, \varphi)$ ein h-Beweis für φ.

Außerdem ist klar, dass h ehrlich ist. Definiere

$$R_h = \{(\varphi, w) \mid h(w) = \varphi\},$$

diese Relation ist damit eine NP-Relation. Wir wollen nun zeigen dass $\mathtt{rSAT} \leq^{\mathrm{p}}_{\mathrm{L}} R_h \leq^{\mathrm{p}}_{\mathrm{L}} Q$, womit dann Q auch $\leq^{\mathrm{p}}_{\mathrm{L}}$-vollständig ist, wie gewünscht.

Wir starten mit der zweiten Reduktion. Es gilt $R_h \leq^{\mathrm{p}}_{\mathrm{L}} Q$, denn es gilt

$$(f(\varphi), y) \in Q \implies h(\underbrace{f(\varphi), y, \varphi}_{g(\varphi, y)}) = \varphi \implies (\varphi, g(\varphi, y)) \in R_h,$$

wobei $g(\varphi, y) = (f(\varphi), y, \varphi)$.

Für die erste Reduktion nutzen wir die Voraussetzung. Es gilt nach Voraussetzung $std \leq^{\mathrm{p}}_{\mathrm{eff}} h$. Damit existieren also Funktionen $f', g' \in \mathrm{FP}$ mit

$$\varphi \in \mathtt{SAT} \iff f'(\varphi) \in \mathtt{SAT}$$
$$h(w) = f'(\varphi) \implies sat(g'(\varphi, w)) = \varphi.$$

Jetzt ist aber auch klar, dass f', g' die Reduktion $\mathtt{rSAT} \leq^{\mathrm{p}}_{\mathrm{L}} R_h$ realisieren, denn nun gilt

$$(f'(\varphi), w) \in R_h \implies (\varphi, g'(\varphi, w)) \in \mathtt{rSAT}.$$

□

Mit der Definition der effektiven p-Simulation und der eben beweisenen äquivalenten Formulierung der KvL-Vermutung lässt sich nun zumindest die Hypothese SAT so verstärken, dass diese hinreichend für KvL ist.

Vermutung 4.19 ($\mathsf{SAT}^{\mathsf{eff}}$)**.** *Kein optimales Beweissystem für* SAT *kann alle anderen optimalen Beweissysteme für* SAT *effektiv p-simulieren. In anderen Worten, für jedes optimales Beweissystem h existiert ein optimales Beweissysteme h' sodass $h \not\leq^{\mathrm{p}}_{\mathrm{eff}} h'$.*

Satz 4.20. *(1)* $\mathsf{SAT}^{\mathsf{eff}} \implies \mathsf{SAT}$

(2) $\mathsf{SAT}^{\mathsf{eff}} \implies \mathsf{KvL}$

Beweis. 1. Zu (1): Klar aus Kontraposition. Wenn ein p-optimales Beweissystem h für SAT existiert, dann kann dieses (optimale) h auch alle anderen Beweissysteme p-simulieren, und damit insbesondere auch alle optimalen Beweissyste h' effektiv p-simulieren.

2. Zu (2): Wieder klar aus Kontraposition. Unter $\neg\mathsf{KvL}$ folgt mit der Formulierung aus Vermutung 4.17 dass das (optimale) Standardbeweissystem *sat* alle optimalen Beweissysteme effektiv p-simulieren kann. Dann existiert also auch *ein* optimales Beweissystem welches dies leistet. □

4.4 Bekannte Implikationen, Offene Orakel

Satz 4.21 (Dingel 2022)**.** *Folgende Aussagen sind äquivalent:*

(1) NP = coNP

(2) Es existiert eine NP*-harte Funktion $f \in$ TFNP: für jede Menge $L \in$ NP und jede Verfeinerung f' von f gilt $L \in \mathrm{P}^{f'}$.*

Der folgende Satz ist eine Generalisierung von Dingel (2022).

Satz 4.22. *Wenn* NP = coNP *dann existiert eine $\leq^{\mathrm{p}}_{\mathrm{m}}$-vollständige Multifunktion f für* NPMV_t.

Beweis. Nach Voraussetzung können wir in NP testen, ob ein Wort x im Urbild einer beliebigen NPMV-Multifuktion liegt. Es gilt KAN $\in$ NP und damit KAN $\in$ coNP. Insbesondere ist dann die Menge

$$U = \{(i, x, 1^n) \mid T_i \text{ akz. auf keinem Rechenweg der Länge} \leq n\} \in \mathrm{NP},$$

und wird von der NPTM N_u in Laufzeit $q(|(i, x, 1^n)|)$ entschieden.

Betrachte nun die Multifunktion f, die durch folgenden nichtdeterministischen Transduktor $T'(i, x, 1^n)$ berechnet wird:

1 **wenn** T kein Transduktor ist oder $n \neq |x|^i + i$ **dann**
2 | akzeptiere
3 Rate nichtdeterministisch einen Rechenweg α von T_i der Länge $\leq n$
4 Rate nichtdeterministisch einen Rechenweg β von N_u der Länge $\leq q(|i, x, 1^n|)$
5 **wenn** Falls $T_i(x)$ mit α akzeptiert **dann**
6 | $y \leftarrow$ Ausgabe von $T_i(x)$ auf α
7 | Gebe y aus
8 **sonst wenn** Falls $N_u(i, x, 1^n)$ mit β akzeptiert **dann**
9 | Gebe ε aus
10 **sonst**
11 | Lehne ab

Es ist leicht zu sehen dass T' in Polynomialzeit arbeitet. Wir betrachten nun Eingaben $(i, x, 1^n), n = |x|^i + i$. Es gilt nun:

- Entweder ist $\mathit{set}\text{-}T_i(x) \neq \emptyset$, dann existiert für jedes $y \in \mathit{set}\text{-}T_i(x)$ ein akzeptierender Rechenweg α der Länge $\leq n$ auf T_i der y ausgibt, und damit wird auch f dieses y in Z. 5 ausgeben. Gleichzeitig ist damit $(i, x, 1^n) \notin U$ und Z. 7 niemals erreicht. Es gilt also $\mathit{set}\text{-}f(i, x, 1^n) = \mathit{set}\text{-}T_i(x)$.
- Oder es gilt $\mathit{set}\text{-}T_i(x) = \emptyset$. Dann wird jeder Rechenweg der Länge $\leq n$ von $T_i(x)$ ablehnen, und f definitiv nicht in Z. 5 akzeptieren. Andererseits gilt dann $(i, x, 1^n) \in U$ und Z. 7 wird auf mindestens einem Rechenweg von f erreicht. Es gilt also $\mathit{set}\text{-}f(i, x, 1^n) = \{\varepsilon\}$.

Damit ist klar, dass $f \in \mathrm{NPMV}_t$. Wir zeigen nun, dass f auch NPMV_t-vollständig ist. Sei hierfür g eine beliebige Multifunktion aus NPMV_t. Dann existiert auch ein i sodass der nichtdeterministische Transduktor T_i diese Multifunktion g in berechnet, und dabei terminiert $T_i(x)$ in $\leq |x|^i + i$ vielen Schritten.

Nun gilt nach obiger Beobachtung schon dass

$$\mathit{set}\text{-}g(x) = \mathit{set}\text{-}T_i(x) \neq \emptyset \implies \mathit{set}\text{-}f(\underbrace{i, x, 1^{|x|^i+i}}_{h(x)}) = \mathit{set}\text{-}T_i(x) = \mathit{set}\text{-}g(x)$$

und $h(x) = (i, x, 1^{|x|^i+i})$ realisiert die Reduktion von g auf f, wie gewünscht. □

Satz 4.23. *Es gelten die in Abbildung 4.1 abgebildeten Implikationen und Äquivalenzen.*

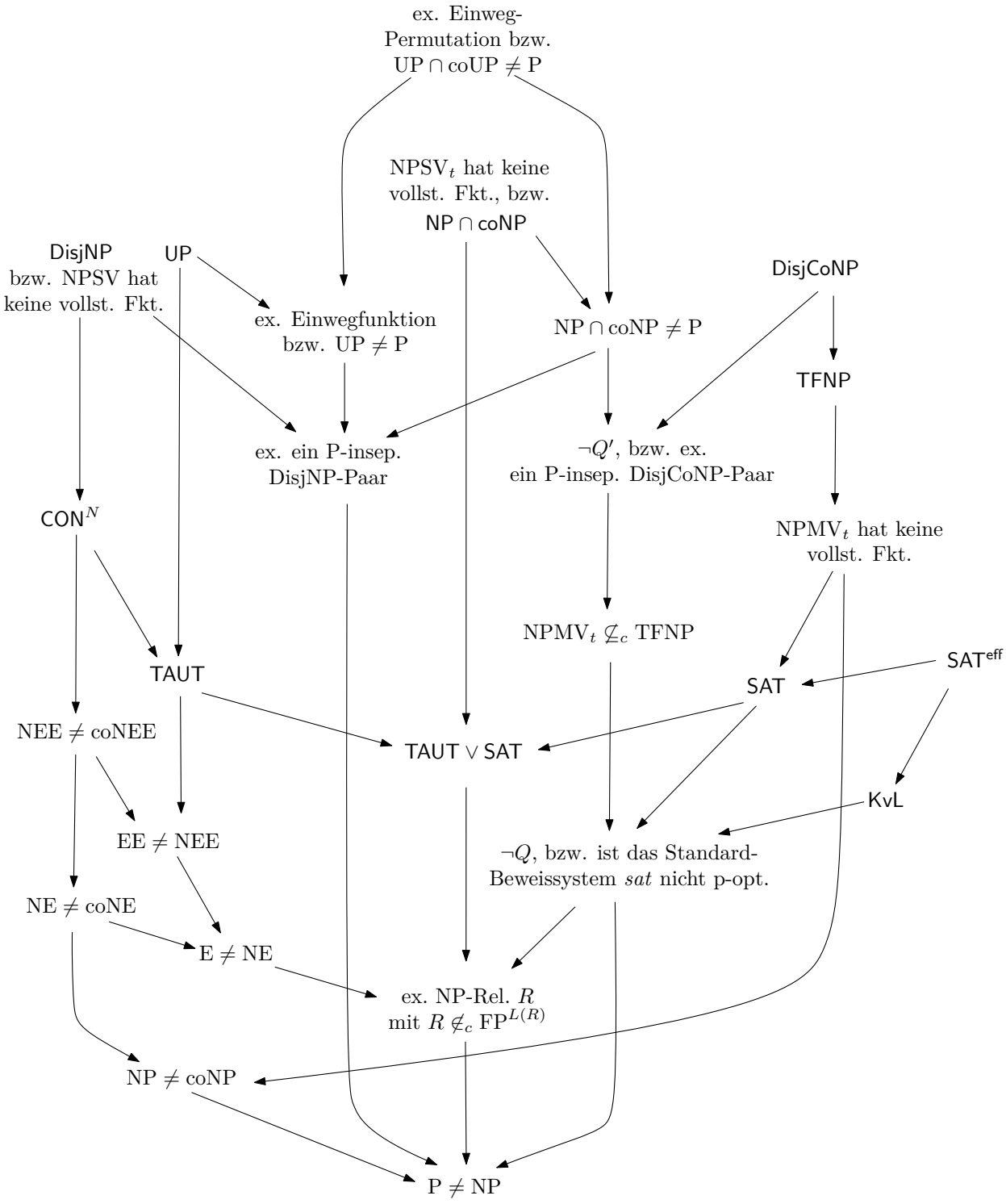

Abbildung 4.1: Bekannte (relativierenden) Implikationen zwischen den betrachteten Hypothesen und weiteren Aussagen. Satz 4.23 gibt Belegstellen für jede dieser Implikationen an.

Beweis. Es gelten die notiertern Äquivalenzen:
1. $\neg\mathsf{Q} \Leftrightarrow$ *sat* ist p-optimal, nach Satz 4.12.
2. NP $\neq$ coNP $\Leftrightarrow$ existiert keine NP-harte Funktion in TFNP, nach Satz 4.21.
3. $\neg\mathsf{Q}' \Leftrightarrow \exists$ P-inseparierbares DisjCoNP-Paar, nach Fortnow und Rogers (2002).
4. NP $\cap$ coNP $\neq$ P $\Leftrightarrow$ NPSV$_t \not\subseteq$ FP, nach Selman (1994).
5. UP $\neq$ P $\Leftrightarrow \exists$ Einwegfunktionen, nach Grollmann und Selman (1988, Thm. 10).
6. UP $\cap$ coUP $\neq$ P $\Leftrightarrow \exists$ Einwegpermutationen, nach Homan und Thakur (2003).
7. $\mathsf{NP} \cap \mathsf{coNP} \Leftrightarrow$ NPSV$_t$ hat keine vollständige Funktion, nach Beyersdorff, Köbler und Messner (2009, Prop. 3).
8. $\mathsf{DisjNP} \Leftrightarrow$ NPSV hat keine vollständige Funktion, nach Glaßer, Selman und Sengupta (2005, Thm. 9).

Es gelten die eingezeichneten Implikationen:
1. $\mathsf{DisjNP} \Rightarrow \mathsf{CON}^N$ nach Köbler, Messner und Torán (2003).
2. $\mathsf{UP} \Rightarrow \mathsf{TAUT}$ nach Köbler, Messner und Torán (2003).
3. $\mathsf{CON}^N \Rightarrow$ NEE $\neq$ coNEE nach Köbler, Messner und Torán (2003).
4. NP $\cap$ coNP $\neq$ P $\Rightarrow\neq \mathsf{Q}' \Rightarrow$ NPMV$_t \not\subseteq_c$ TFNP $\Rightarrow\neq \mathsf{Q}$ nach Fenner u. a. (2003).
5. E $\neq$ NE $\Rightarrow \exists$ NP-Relation die nicht auf Entscheidung reduzierbar ist, nach Impagliazzo und Sudan (private Kommunikation berichtet von Bellare und Goldwasser 1994, Abschn. 1.5.4).
6. UP $\neq$ P $\Rightarrow \exists$ P-inseparierbares DisjNP-Paar, nach Grollmann und Selman (1988, Thm. 5).
7. $\mathsf{NP} \cap \mathsf{coNP} \Rightarrow \mathsf{TAUT} \vee \mathsf{SAT}$ nach Beyersdorff, Köbler und Messner (2009).
8. NPMV$_t$ hat keine vollständige Funktion $\Rightarrow \mathsf{SAT}$ nach Beyersdorff, Köbler und Messner (2009, Thm. 25).
9. NPMV$_t$ hat keine vollständige Funktion $\Rightarrow$ NP $\neq$ coNP nach Satz 4.21.
10. $\mathsf{SAT}^{\mathsf{eff}} \Rightarrow \mathsf{SAT}, \mathsf{SAT}^{\mathsf{eff}} \Rightarrow \mathsf{KvL}$, nach Satz 4.20.
11. $\mathsf{KvL} \Rightarrow \neg\mathsf{Q}$, nach Satz 4.15.
12. $\neg\mathsf{Q} \Rightarrow \exists$ NP-Relation die nicht auf Entscheidung reduzierbar ist, denn unter $\neg\mathsf{Q}$ gilt mit Satz 4.12 auch die Negation von 4.12(1), also eine NPTM N mit $L(N) = \Sigma^*$ wobei keine Funktion $g \in$ FP existiert, welche für alle x durch $g(x)$ einen akzeptierenden Rechenweg von $N(x)$ bestimmt. Definiere die NP-Relation R_N mit $(x, \alpha) \in R_N$ genau dann wenn $N(x)$ mit Rechenweg α existiert. Nun gilt nach Vorigem auch $R \notin_c \mathrm{FP} = \mathrm{FP}^{\Sigma^*} = \mathrm{FP}^{L(R)}$.
13. $\mathsf{NP} \cap \mathsf{coNP} \Rightarrow \mathsf{TFNP} \Rightarrow$ NPMV$_t$ hat keine vollständig Funktion, nach Pudlák (2017).
14. NP $\cap$ coNP $\neq$ P $\Rightarrow \exists$ P-inseparierbares DisjNP-Paar, denn wenn alle DisjNP-Paare P-separierbar, dann ist auch für jede Menge $L \in$ NP $\cap$ coNP jeweils das DisjNP-Paar $(L, \overline{L})$ P-separierbar und damit $L \in$ P.
15. $\mathsf{DisjNP} \Rightarrow \exists$ P-inseparierbares DisjNP-Paar; ist klar, denn wenn alle DisjNP-Paare P-separierbar wären, dann wären auch alle Paare $\leq_m^{\mathrm{pp}}$-vollständig.
16. $\mathsf{DisjCoNP} \Rightarrow \exists$ P-inseparierbares DisjCoNP-Paar; ist aus selben Gründen klar.
17. $\mathsf{CON}^N \Rightarrow \mathsf{TAUT}$ klar, weil aus p-Optimalität auch Optimalität folgt.
18. $\mathsf{SAT} \Rightarrow \neg\mathsf{Q}$ klar, denn wenn *sat* p-optimal ist, dann existiert *ein* p-optimales Beweissystem für SAT.
19. $\mathsf{UP} \Rightarrow$ UP $\neq$ P klar.
20. $\mathsf{NP} \cap \mathsf{coNP} \Rightarrow$ NP $\cap$ coNP $\neq$ P klar.
21. $\exists$ P-inseparierbares DisjNP-Paar $\Rightarrow$ P $\neq$ NP klar.
22. UP $\cap$ coUP $\Rightarrow$ UP $\neq$ P, UP $\cap$ coUP $\Rightarrow$ NP $\cap$ coNP $\neq$ P klar.
23. NEE $\neq$ coNEE $\Rightarrow$ NE $\neq$ coNE $\Rightarrow$ NP $\neq$ coNP $\Rightarrow$ P $\neq$ NP klar.
24. NEE $\neq$ coNEE $\Rightarrow$ EE $\neq$ NEE $\Rightarrow$ E $\neq$ NE klar. □

5 Orakel

Abschließend erweitern wir noch unsere Notation, die uns insbesondere bei der Orakelkonstruktion helfen wird. Diese folgt Überlegungen von Dose und Glaßer (2019). Anstelle von Orakeln als Menge zu verstehen, können wir äquivalent Orakel auch als unendlich lange Wörter $u \in \Sigma^\omega$ formulieren, die wir als das Orakel $\{i \mid w[i] = 1\} \subseteq \mathbb{N}$ interpretieren. Mit der obigen Identifikation von Wörtern und natürlichen Zahlen beschreibt nun u sowohl ein Orakel über $\mathbb{N}$ als auch über Σ^*; wir können also z.B. von der relativen Berechnung $M^w(x)$ sprechen. Analog fassen wir endlich lange Wörter $w \in \Sigma^*$ als *partielles* Orakel $\{i \mid w[i] = 1\}$, welches die Zugehörigkeit der Wörter $x < |w|$ festlegt, aber die Zugehörigkeit aller Wörter $y \geq |w|$ noch nicht endgültig festlegt. Auf dieser Idee der endgültigen bzw. noch nicht endgültigen Zugehörigkeit aufbauend können wir auch von *definiten* Berechnungen sprechen: Eine Rechnung $M^w(x)$ ist *definit* wenn auf allen Rechenwegen von $M^w(x)$ nur Orakelfragen gestellt werden, welche eine Länge $< |w|$ haben.

Literatur

Agrawal, Manindra und Somenath Biswas. 1992. „Universal relations“. In: *Proceedings of the Seventh Annual Structure in Complexity Theory Conference*. Seventh Annual Structure in Complexity Theory Conference. Juni 1992, S. 207–220. DOI: 10.1109/SCT.1992.215395.

Agrawal, Manindra, Neeraj Kayal und Nitin Saxena. 2004. „PRIMES Is in P“. In: *Annals of Mathematics* 160.2 (2004), S. 781–793. DOI: 10.4007/annals.2004.160.781.

Baker, Theodore, John Gill und Robert Solovay. 1975. „Relativizations of the P=?NP Question“. In: *SIAM Journal on Computing* 4.4 (Dez. 1975), S. 431–442. DOI: 10.1137/0204037.

Bellare, Mihir und Shafi Goldwasser. 1994. „The Complexity of Decision Versus Search“. In: *SIAM Journal on Computing* 23.1 (Feb. 1994), S. 97–119. DOI: 10.1137/S0097539792228289.

Beyersdorff, Olaf, Johannes Köbler und Jochen Messner. 2009. „Nondeterministic functions and the existence of optimal proof systems“. In: *Theoretical Computer Science* 410.38 (6. Sep. 2009), S. 3839–3855. DOI: 10.1016/j.tcs.2009.05.021.

Buhrman, H., J. Kadin und T. Thierauf. 1998. „Functions Computable with Nonadaptive Queries to NP“. In: *Theory of Computing Systems* 31.1 (1. Feb. 1998), S. 77–92. DOI: 10.1007/s002240000079.

Cook, Stephen A. und Robert A. Reckhow. 1979. „The relative efficiency of propositional proof systems“. In: *The Journal of Symbolic Logic* 44.1 (März 1979), S. 36–50. ISSN: 0022-4812, 1943-5886. DOI: 10.2307/2273702.

Dingel, David. 2022. „Separation der relativierten Vermutungen SAT und TFNP“. Bachelorarbeit. Universität Würzburg, 22. Okt. 2022.

Dose, Titus und Christian Glaßer. 2019. *NP-Completeness, Proof Systems, and Disjoint NP-Pairs*. 050. 2019.

Fenner, Stephen A., Lance Fortnow, Ashish V. Naik und John D. Rogers. 2003. „Inverting onto functions“. In: *Information and Computation* 186.1 (Okt. 2003), S. 90–103. DOI: 10.1016/S0890-5401(03)00119-6.

Fischer, Sophie, Lane Hemaspaandra und Leen Torenvliet. 1995. „Witness-isomorphic reductions and the local search problem“. In: *Mathematical Foundations of Computer Science 1995*. Hrsg. von Jiří Wiedermann und Petr Hájek. Bd. 969. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, 1995, S. 277–287. DOI: 10.1007/3-540-60246-1_134.

Fortnow, Lance und John D. Rogers. 2002. „Separability and one-way functions“. In: *Computational Complexity* 11.3 (1. Juni 2002), S. 137–157. DOI: 10.1007/s00037-002-0173-4.

Glaßer, Christian, Alan L. Selman und Samik Sengupta. 2005. „Reductions between disjoint NP-Pairs“. In: *Information and Computation* 200.2 (1. Aug. 2005), S. 247–267. ISSN: 0890-5401. DOI: 10.1016/j.ic.2005.03.003.

Goldreich, Oded. 2008. *Computational Complexity: a Conceptual Perspective*. Cambridge: Cambridge University Press, 2008. 606 S. ISBN: 978-0-521-88473-0.

Grollmann, Joachim und Alan L. Selman. 1988. „Complexity Measures for Public-Key Cryptosystems“. In: *SIAM Journal on Computing* 17.2 (Apr. 1988). Publisher: Society for Industrial and Applied Mathematics, S. 309–335. DOI: 10.1137/0217018.

Hartmanis, J. und L. Berman. 1976. „On isomorphisms and density of NP and other complete sets“. In: *Proceedings of the eighth annual ACM symposium on Theory of computing*. STOC '76. New York, NY, USA: Association for Computing Machinery, 3. Mai 1976, S. 30–40. DOI: 10.1145/800113.803628.

Homan, Christopher M. und Mayur Thakur. 2003. „One-way permutations and self-witnessing languages“. In: *Journal of Computer and System Sciences* 67.3 (1. Nov. 2003), S. 608–622. DOI: 10.1016/S0022-0000(03)00068-0.

Köbler, Johannes und Jochen Messner. 2000. „Is the Standard Proof System for SAT P-Optimal?“ In: *FST TCS 2000: Foundations of Software Technology and Theoretical Computer Science*. Hrsg. von Sanjiv Kapoor und Sanjiva Prasad. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2000, S. 361–372. DOI: 10.1007/3-540-44450-5_29.

Köbler, Johannes, Jochen Messner und Jacobo Torán. 2003. „Optimal proof systems imply complete sets for promise classes“. In: *Information and Computation* 184.1 (10. Juli 2003), S. 71–92. DOI: 10.1016/S0890-5401(03)00058-0.

Krajíček, Jan und Pavel Pudlák. 1989. „Propositional proof systems, the consistency of first order theories and the complexity of computations“. In: *Journal of Symbolic Logic* 54.3 (Sep. 1989), S. 1063–1079. DOI: 10.2307/2274765.

Lynch, Nancy und Richard J. Lipton. 1978. „On Structure Preserving Reductions“. In: *SIAM Journal on Computing* 7.2 (Mai 1978). Publisher: Society for Industrial and Applied Mathematics, S. 119–126. DOI: 10.1137/0207010.

Megiddo, Nimrod und Christos H. Papadimitriou. 1991. „On total functions, existence theorems and computational complexity“. In: *Theoretical Computer Science* 81.2 (30. Apr. 1991), S. 317–324. DOI: 10.1016/0304-3975(91)90200-L.

Messner, Jochen. 2001. „On the simulation order of proof systems“. Diss. University of Ulm, Germany, 2001.

Pudlák, Pavel. 2017. „Incompleteness in the finite domain“. In: *The Bulletin of Symbolic Logic* 23.4 (2017), S. 405–441. DOI: 10.1017/bsl.2017.32.

Selman, Alan L. 1994. „A taxonomy of complexity classes of functions“. In: *Journal of Computer and System Sciences* 48.2 (Apr. 1994), S. 357–381. DOI: 10.1016/S0022-0000(05)80009-1.