Leichte und schwere Mengen in NP

Jörg Rothe

Institut für Informatik
Friedrich-Schiller-Universität Jena

1 Einleitung

Gerd Wechsung war es, der uns einen neuen Begriff von Zeit und Raum vermittelt hat. Ich war Student im zweiten Studienjahr, als er uns die Arbeitsweise von Turingmaschinen erklärte und die Bedeutung von Zeit (Rechenzeit) und Raum (Speicherplatz) für Turingmaschinen. Im dritten Studienjahr hörten wir eine Vorlesung bei ihm, die damals noch unter dem Namen „Kompliziertheitstheorie“ lief. Da wurde es noch bunter, da war von Reduzierbarkeiten die Rede, von Vollständigkeit, da erfuhren wir, warum manche Probleme effizient lösbar sind und andere nicht.

Was Effizienz und Zeitverständnis im wirklichen Leben bedeuten, demonstrierte uns Professor Wechsung eindrucksvoll, als er nach einer Vorlesung im Hörsaal 2 des Abbeanum – vielleicht um Zeit zu sparen, vielleicht aus jungenhaftem Übermut – kurzerhand den Weg durchs Fenster nahm und nach draußen sprang, anstatt konventionell und umständlich Türen und Treppen zu benutzen. Gern erinnern wir uns auch seines Ausspruchs am Ende einer hochkomplizierten Kompliziertheitstheorie-Vorlesung, als er eine Weile auf die Uhr sah und uns dann bat: „Können Sie mir vielleicht die Zeit sagen? Mir geht es immer so: Wenn ich zu lange über Turingmaschinen gesprochen habe und dann auf die Uhr schaue, dann sehe ich nur Zeiger.“

Die vorliegende Arbeit stellt in der Art eines Übersichtsartikels einige Resultate vor, die in den Jahren meiner Zugehörigkeit zu Gerd Wechsungs Lehrstuhl entstanden sind. An einigen dieser Ergebnisse war er selbst als Koautor beteiligt, und an fast allen Lane A. Hemaspaandra. Beiden bin ich zu tiefem Dank verpflichtet.

Es geht in dieser Arbeit um die folgenden Themen: Zertifikatkomplexität, Einwegfunktionen und Heuristiken versus NP-Vollständigkeit für Graphfärbbarkeit. Jedes dieser Themen hat, auf die eine oder andere Weise, einen Bezug zum $\mathrm{P} \stackrel{?}{=} \mathrm{NP}$ Problem und somit zur Frage, wie und weshalb sich die leichten und die schweren Mengen in NP unterscheiden. Vom Beweis der vorgestellten Ergebnisse wird

in den meisten Fällen abgesehen, zugunsten einer breiteren Darstellung der umliegenden Ergebnisse, des historischen Kontextes und der Motivation. Gern hätte ich noch weitere Themen eingebracht, doch die den Autoren dieses Bandes auferlegte sublogarithmische Platzbeschränkung ließ das nicht zu. Zum Verständnis dieses Textes sind keinerlei besondere Vorkenntnisse erforderlich. Vorausgesetzt, man hat ein Grundlagenbuch oder eine Monographie der Komplexitätstheorie – wie die von Wagner und Wechsung [WW86] – gründlich studiert.

2 Leichte Mengen mit schweren Zertifikaten

Betrachten wir ein Problem in NP. Wenn Sie von Beruf Graph-Färber sind, liegt es nahe, 3-`Colorability` zu betrachten. 3-`Colorability` ist das folgende Entscheidungsproblem: Können die Knoten eines gegebenen Graphen mit drei Farben so gefärbt werden, daß keine zwei benachbarten Knoten dieselbe Farbe erhalten? Angenommen, jemand gibt Ihnen einen 3-färbbaren Graphen G zum Färben, siehe Abbildung 1.[1] Ein *Zertifikat* für die 3-Färbbarkeit von G (m.a.W. ein Zertifikat für „$G \in$ 3-`Colorability`“) kann durch Angabe einer legalen 3-Färbung von G erstellt werden, d.h. durch Angabe einer Abbildung von der Knotenmenge von G in die Farbenmenge (ROT, GRÜN, BLAU), so daß alle benachbarten Knoten verschieden gefärbt sind. Die 3-Färbung, ψ, des Graphen G in Abbildung 1, die gegeben ist durch die drei Farbklassen $\psi^{-1}(\text{GRÜN}) = \{a, g\}$, $\psi^{-1}(\text{ROT}) = \{c, f, h\}$ und $\psi^{-1}(\text{BLAU}) = \{b, d, e\}$, ist ein solches Zertifikat für „$G \in$ 3-`Colorability`“.

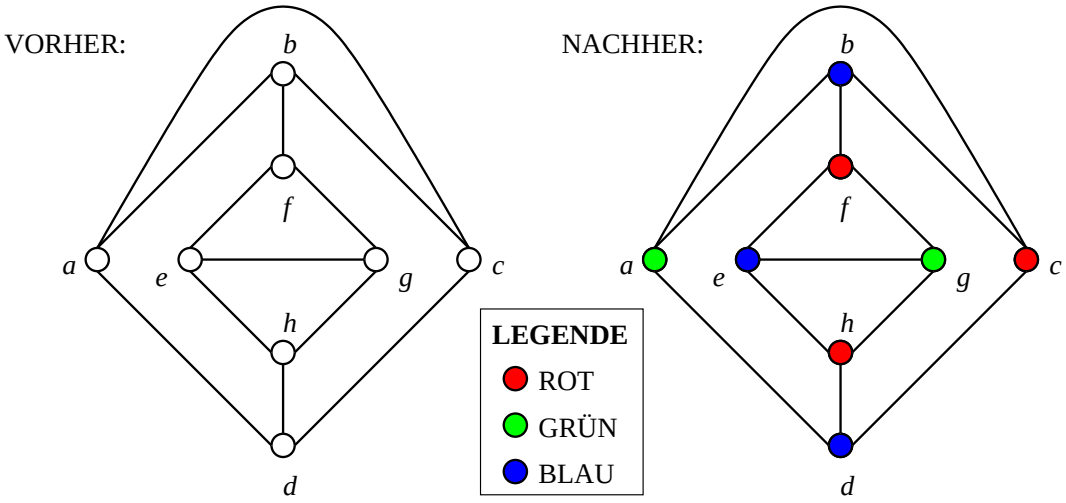

Abbildung 1: Die 3-Färbung ψ des Graphen G

Allgemein bietet sich folgende formale Definition an. Gegeben seien ein Problem $A \in$ NP und eine NP-Maschine N für A (d.h., $A = L(N)$ ist die von N akzeptierte

[1] Kurz vor Redaktionsschluß stellte sich heraus, daß aus drucktechnischen Gründen farbige Abbildungen nicht möglich sind. Abbildung 1 wurde daher um eine Legende bereichert, die die ursprünglich verwendeten Farben erklärt.

Sprache), und $x \in A$ sei eine von N akzeptierte Instanz. Ein *Zertifikat* für „$x \in A$“ ist ein Wort $z \in \Sigma^*$,[2] das einen akzeptierenden Berechnungspfad von N bei Eingabe x kodiert. Andere gebräuchliche Bezeichnungen für „Zertifikat“ sind „Zeuge“ und „Lösung“. Für jedes Zertifikat z für „$x \in A$“ gilt: (a) die Länge von z (bezeichnet mit $|z|$) ist polynomiell in $|x|$, und (b) z bezeugt die Zugehörigkeit von x zu A so, daß dies deterministisch in Polynomialzeit verifiziert werden kann. Die Nützlichkeit von Zertifikaten liegt auf der Hand – bzw. liegt begründet in ihren Eigenschaften (a) und (b). Doch wie kann man Zertifikate in die Hand bekommen, wie sie effizient finden? Wechsung et al. [HRW97a] untersuchen solche NP-Mengen, deren Maschinen schwer bzw. leicht zu berechnende Zertifikate haben.

Die kanonische NP-Maschine für 3-`Colorability` arbeitet bei Eingabe eines Graphen H so: Sie rät alle möglichen Zerlegungen der Knotenmenge von H in drei Farbklassen und verifiziert für jede geratene Zerlegung, ob für jede Farbklasse der Zerlegung gilt, daß keine zwei ihrer Knoten benachbart sind.

Was tun, wenn die kanonische NP-Maschine für 3-`Colorability` gerade zur Reparatur ist (vielleicht ist der Zerlegungsrater defekt)? Der erfahrene Graph-Färber wird in seiner Werkzeugkiste einen unerschöpflichen Vorrat an anderen NP-Maschinen vorfinden, die ebenfalls 3-`Colorability` akzeptieren, wenn auch vielleicht auf mehr oder weniger umständliche Art und Weise.[3] Glücklicherweise gibt es für jedes Problem in NP unendlich viele NP-Maschinen. Die folgende Definition zweier Teilklassen von NP trägt dieser Tatsache Rechnung: Die eine Teilklasse besteht aus solchen NP-Mengen, für die wenigstens eine Maschine mit leicht berechenbaren Zertifikaten *existiert*; die andere besteht aus solchen NP-Mengen, deren *sämtliche* Maschinen stets leicht berechenbare Zertifikate haben müssen.

Im folgenden bezeichne FP die Klasse der in Polynomialzeit berechenbaren Funktionen von Σ^* in Σ^*. Für eine NP-Maschine N und eine Eingabe $x \in \Sigma^*$ sei $\mathrm{ACC}_N(x) \subseteq \Sigma^*$ die Menge aller akzeptierenden Berechnungspfade von N bei Eingabe x. Die Abkürzung „NPTM“ steht für „nichtdeterministische Polynomialzeit-Turingmaschine“.

Definition 2.1 [HRW97a]

1. $\mathrm{EASY}_{\forall}^{\forall}$ *ist die Klasse aller der Mengen* $A \in \mathrm{NP}$*, für die gilt:*

$$(\forall \textit{NPTM } N \textit{ mit } L(N) = A)\,(\exists f_N \in \mathrm{FP})\,(\forall x \in A)\,[f_N(x) \in \mathrm{ACC}_N(x)].$$

2. $\mathrm{EASY}_{\forall}^{\exists}$ *ist die Klasse aller der Mengen* $A \in \mathrm{NP}$*, für die gilt:*

$$(\exists \textit{NPTM } N)\,[L(N) = A \wedge (\exists f_N \in \mathrm{FP})\,(\forall x \in A)\,[f_N(x) \in \mathrm{ACC}_N(x)]].$$

[2] $\Sigma = \{0, 1\}$ ist das zugrundegelegte binäre Alphabet, und Σ^* bezeichnet die Menge aller Wörter über Σ.

[3] Beispiel einer NP-Maschine für 3-`Colorability`, die ohne Zerlegungsrater auskommt: Bei Eingabe H rät sie alle möglichen Reihenfolgen der Knoten von H, wendet den „sequentiellen Algorithmus“ auf jede der geratenen Knotenfolgen an und akzeptiert auf einem solchen Berechnungspfad genau dann, wenn der sequentielle Algorithmus nicht mehr als drei Farben benötigt hat. Der sequentielle Algorithmus, eine bekannte Heuristik zum Färben von Graphen, wird in Abschnitt 4 dieser Arbeit vorgestellt. Siehe Fußnote 10.

Zwei Bemerkungen zur Notation. Der obere Index der EASY-Klassen steht für die Quantifizierung der Maschinen, wie z.B. in „$\exists$NPTM N“; der untere Index, jeweils ein $\forall$, steht für die Quantifizierung der Eingabewörter in „$\forall x \in A$“. In der Arbeit [HRW97a] werden noch zwei weitere EASY-Klassen betrachtet, $\mathrm{EASY}^{\forall}_{\mathrm{io}}$ und $\mathrm{EASY}^{\exists}_{\mathrm{io}}$, bei welchen von den Maschinen lediglich gefordert wird, daß sie für *unendlich viele* Eingaben leicht berechenbare Zertifikate haben müssen. Wir beschränken uns hier aus Platzgründen auf die beiden Klassen aus Definition 2.1, $\mathrm{EASY}^{\forall}_{\forall}$ und $\mathrm{EASY}^{\exists}_{\forall}$. Natürlich wäre es schöner, bei der Einführung neuer Komplexitätsklassen ganz auf Indizes verzichten zu können. Tatsächlich war „NGC“ das Akronym, das die Autoren der Arbeit [HRW97a] intern gebrauchten, um über die Klasse $\mathrm{EASY}^{\forall}_{\forall}$ zu sprechen,[4] und „NGC“ stand für „New Gerd's Class“. Da jedoch noch drei andere Klassen untersucht wurden und die Autoren bestrebt waren, Bezeichnungen wie „TTOFNGC“ für „The Third Of Four New Gerd's Classes“ zu vermeiden, fiel die Wahl schließlich auf $\mathrm{EASY}^{\alpha}_{\beta}$, mit $\alpha \in \{\exists, \forall\}$ und $\beta \in \{\forall, \mathrm{io}\}$.

Es zeigt sich, daß $\mathrm{EASY}^{\exists}_{\forall}$ eigentlich keine neue Klasse ist. Da für jede NPTM N die Menge $\bigcup_{x \in \Sigma^*} \mathrm{ACC}_N(x)$ in P liegt, gilt $\mathrm{EASY}^{\exists}_{\forall} = \mathrm{P}$. Es folgt unmittelbar $\mathrm{EASY}^{\forall}_{\forall} \subseteq \mathrm{P}$. Der Frage, ob diese Inklusion echt ist, werden wir uns später zuwenden; vorerst untersuchen wir die Frage, ob die (triviale) Inklusion $\mathrm{EASY}^{\forall}_{\forall} \subseteq \mathrm{NP}$ echt ist. Es stellt sich heraus, daß $\mathrm{EASY}^{\forall}_{\forall}$, zumindest in diesem Kontext, ebenfalls große Ähnlichkeit mit P hat.

Satz 2.2 [HRW97a] $\mathrm{EASY}^{\forall}_{\forall} \neq \mathrm{NP}$ *genau dann, wenn* $\mathrm{P} \neq \mathrm{NP}$.

Korollar 2.3 [HRW97a] $\mathrm{NP} = \mathrm{EASY}^{\exists}_{\forall}$ *genau dann, wenn* $\mathrm{NP} = \mathrm{EASY}^{\forall}_{\forall}$.

Eine konkrete Interpretation von Korollar 2.3 ist: Damit *jede* NP-Maschine für das Problem 3-`Colorability` stets leicht berechenbare Zertifikate hat, genügt es, daß *irgendeine* NP-Maschine für 3-`Colorability` diese Eigenschaft besitzt.

Daß die *schweren* Mengen in NP schwierig berechenbare Zertifikate haben, erstaunt niemanden, der an die Gültigkeit der Hypothese $\mathrm{P} \neq \mathrm{NP}$ glaubt: Die Schwierigkeit, NP-vollständige Probleme wie 3-`Colorability` zu lösen, rührt gerade daher, daß ihre Zertifikate sehr schwer zu finden sind. Interessanter ist es, die *leichten* Mengen in NP auf diese Eigenschaft zu untersuchen und zu fragen: Haben die NP-Maschinen für leichte Mengen stets auch leicht zu berechnende Zertifikate? Mit anderen Worten, gilt $\mathrm{EASY}^{\forall}_{\forall} = \mathrm{P}$?

Borodin und Demers [BD76] bewiesen bereits vor mehr als zwei Jahrzehnten, daß die Antwort auf diese Frage negativ ist, außer es tritt ein unwahrscheinlich erscheinender Kollaps von Komplexitätsklassen ein. Dieses Resultat soll hier bewiesen werden, denn es ist ein schönes und vielleicht sogar überraschendes Ergebnis: Es sagt, daß auch leichte Mengen in gewissem Sinne schwer sein können. Es war dieses Resultat von Borodin und Demers, das unser Interesse für die EASY-Klassen erweckte und unserer Arbeit [HRW97a] Motivation und historischen Hintergrund lieferte. Apropos, für mich als dem dritten Koautor von [HRW97a] ergab sich noch

[4] Genau genommen bezeichnete NGC die zu $\mathrm{EASY}^{\forall}_{\forall}$ duale Klasse $\mathrm{NP} - \mathrm{EASY}^{\forall}_{\forall}$, die aus den NP-Mengen besteht, für die wenigstens eine NP-Maschine schwer berechenbare Zertifikate hat.

ein weiterer „historischer“ Aspekt: Als ich im Frühjahr 1995 in die Arbeit an diesem Projekt einstieg, existierten bereits Vorarbeiten und Aufzeichnungen meiner beiden Koautoren, die bis ins Jahr 1988 zurückdatierten. 1988, ein Jahr vor dem Fall der Mauer, hatte Lane Hemaspaandra bereits Gerd Wechsung und seine Arbeitsgruppe in Jena zu gemeinsamer Forschung besucht; ich war damals Student im zweiten Studienjahr.

Der Satz von Borodin und Demers wird unten nicht in seiner ursprünglichen Fassung, sondern in unserer Notation angegeben, und die Vereinfachung seines Beweises, die hier präsentiert wird, geht auf Juris Hartmanis zurück.

Satz 2.4 [BD76] *Wenn* $\mathrm{NP} \cap \mathrm{coNP} \neq \mathrm{P}$*, dann* $\mathrm{P} \not\subseteq \mathrm{EASY}^{\forall}_{\forall}$.

Beweis. Es sei L eine Menge in $\mathrm{NP} \cap \mathrm{coNP}$, die nicht in P enthalten ist. N_L und $N_{\overline{L}}$ seien NP-Maschinen, die L bzw. $\overline{L}$ akzeptieren, d.h., $L(N_L) = L$ und $L(N_{\overline{L}}) = \overline{L}$.

Man betrachte nun die folgende NPTM M. Bei Eingabe x rät M nichtdeterministisch, ob $x \in L$ oder $x \in \overline{L}$, wobei simultan dazu Zertifikate für die jeweilige Vermutung geraten werden; d.h., wurde auf einem Berechnungspfad $x \in L$ geraten, so simuliert M die Berechnung von N_L bei Eingabe x, wurde $x \in \overline{L}$ geraten, so simuliert M die Berechnung von $N_{\overline{L}}$ bei Eingabe x. Es gilt:

$$L(M) = L(N_L) \cup L(N_{\overline{L}}) = L \cup \overline{L} = \Sigma^*.$$

Σ^* ist eine Menge in P. Es bleibt zu zeigen, daß unsere Annahme $L \notin \mathrm{P}$ impliziert, daß $\Sigma^* \notin \mathrm{EASY}^{\forall}_{\forall}$. Angenommen, $\Sigma^* \in \mathrm{EASY}^{\forall}_{\forall}$. Da $\Sigma^* = L(M)$, existiert eine FP-Funktion f_M, so daß für jede Eingabe x das Wort $f_M(x)$ ein Zertifikat für „$x \in L$“ ist. Dies ermöglicht die deterministische Entscheidung von L in Polynomialzeit wie folgt: Bei Eingabe x wird $f_M(x)$ berechnet und einfach geprüft, ob die erste nichtdeterministische Verzweigung der Berechnung von M bei Eingabe x – diese Information ist im ersten Bit von $f_M(x)$ kodiert – entweder $x \in L$ oder $x \in \overline{L}$ ist, dann wird entsprechend akzeptiert oder abgelehnt. Somit ist $L \in \mathrm{P}$, ein Widerspruch zu unserer Annahme $L \notin \mathrm{P}$. Folglich gilt $\Sigma^* \notin \mathrm{EASY}^{\forall}_{\forall}$ und somit $\mathrm{P} \not\subseteq \mathrm{EASY}^{\forall}_{\forall}$. ∎

Die Arbeit [HRW97a] liefert eine Reihe neuer Ergebnisse hinsichtlich der Wahrscheinlichkeit, mit der Separationen der Klassen $\mathrm{EASY}^{\alpha}_{\beta}$, mit $\alpha \in \{\exists, \forall\}$ und $\beta \in \{\forall, \mathrm{io}\}$, von Standard-Komplexitätsklassen wie P, NP, $\mathrm{NP} \cap \mathrm{coNP}$ oder der Klasse aller endlichen Mengen zu erwarten sind. Genauer gesagt werden solche Separationen in Zusammenhang zu anderen komplexitätstheoretischen Bedingungen gesetzt und somit Aussagen über die Größe der EASY-Klassen gemacht. Ohne auf Details einzugehen, soll erwähnt werden, daß diese anderen Bedingungen solche Eigenschaften betreffen wie Immunität und Bi-Immunität (siehe [Rog67]), P-printability [HY84], verallgemeinerte Kolmogorov-Komplexität [Har83] (siehe auch [Kol65, Cha66]) und Kollapse von Komplexitätsklassen.

Diese Ergebnisse haben die Form von Implikationen (wie Satz 2.4) oder von Äquivalenzen (wie Satz 2.2) und werden in [HRW97a] als positive Resultate bezeichnet. Es werden auch negative Resultate bewiesen, die zeigen, daß manche der positiven Resultate optimal sind in dem Sinne, daß die Umkehrung der entsprechenden Implikation nicht in allen relativierten Welten gilt. Ein Beispiel für ein solches negatives Resultat ist der folgende Satz.

Satz 2.5 [HRW97a] *Es gibt ein Orakel A, so daß gilt:* $\mathrm{NP}^A \neq \mathrm{P}^A = (\mathrm{EASY}^{\forall}_{\forall})^A$.

Die Implikation $\mathrm{P} \neq \mathrm{EASY}^{\forall}_{\forall} \implies \mathrm{P} \neq \mathrm{NP}$ folgt unmittelbar aus Satz 2.2. Satz 2.5 sagt aber, daß die Umkehrung dieser Implikation nicht in allen Relativierungen gilt. Gleichzeitig verschärft Satz 2.5 das folgende klassische Resultat von Baker, Gill und Solovay [BGS75]: Es gibt ein Orakel B, so daß $\mathrm{NP}^B \neq \mathrm{P}^B = \mathrm{NP}^B \cap \mathrm{coNP}^B$. Dieses Ergebnis sagt, daß die (triviale) Implikation $\mathrm{P} \neq \mathrm{NP} \cap \mathrm{coNP} \implies \mathrm{P} \neq \mathrm{NP}$ nicht robust (d.h. nicht in allen Relativierungen) umkehrbar ist. Da sich, dank des Satzes von Borodin und Demers, die Implikation $\mathrm{P} \neq \mathrm{NP} \cap \mathrm{coNP} \implies \mathrm{P} \neq \mathrm{NP}$ zerlegen läßt in:

$$(2.1) \qquad \left(\mathrm{P} \neq \mathrm{NP} \cap \mathrm{coNP} \implies \mathrm{P} \neq \mathrm{EASY}^{\forall}_{\forall}\right) \wedge \left(\mathrm{P} \neq \mathrm{EASY}^{\forall}_{\forall} \implies \mathrm{P} \neq \mathrm{NP}\right)$$

und da die beiden Implikationen in (2.1) in allen Relativierungen gelten, ist Satz 2.5 tatsächlich stärker als der Satz von Baker, Gill und Solovay. Bemerkenswerterweise ist auch die andere der beiden Implikationen in (2.1) – d.h. die Aussage des Satzes von Borodin und Demers – nicht robust umkehrbar. Naor und Impagliazzo [IN88, Proposition 4.2] (siehe auch [CS93, FR94, FFNR96]) haben ein Orakel C konstruiert, so daß gilt: $(\mathrm{EASY}^{\forall}_{\forall})^C \neq \mathrm{P}^C = \mathrm{NP}^C \cap \mathrm{coNP}^C$. Auch dieses Resultat ist wegen (2.1) eine Verschärfung des Satzes von Baker, Gill und Solovay.

3 Einwegfunktionen

Thomas von Aquino versuchte im 13. Jahrhundert, mit den Mitteln der Vernunft und Logik die Existenz Gottes zu beweisen. Ob ihm das gelungen ist oder nicht, darüber mögen die Theologen streiten. Theologie, Philosophie und Mathematik sind drei Seiten ein und desselben gleichseitigen Dreiecks, sie haben denselben Mittelpunkt und gehen im spitzen Winkel aufeinander zu. Und wenn sich die Theologie mathematischer Methoden und Prinzipien wie der Logik bedient, warum soll sich nicht auch die Mathematik – und insbesondere die Komplexitätstheorie – auf theologische Prinzipien berufen? Einwegfunktionen sind ein Beispiel, wo das besonders gut gelingt. Die Frage ihrer Existenz ist, ganz wie im Falle Gottes, eine Glaubensfrage.

Einwegfunktionen sind Funktionen, die sich leicht berechnen, aber nur schwer invertieren lassen. Im traditionellen Modell der *worst-case* Komplexität versteht man unter „schwer invertierbar", daß kein FP-Algorithmus in der Lage ist, für jedes Wort z aus dem Bild der Funktion eines der Urbilder von z zu berechnen.[5] Dieser Begriff der komplexitätstheoretischen Einwegfunktion geht auf Grollmann und Selman [GS88] und Ko [Ko85] zurück, die die Existenz verschiedener Typen von Einwegfunktionen durch Separationen geeigneter Komplexitätsklassen charakterisierten.

[5] Für die meisten kryptographischen Anwendungen genügt diese Definition nicht. Statt dessen wird das Modell der *average-case* Komplexität zugrundegelegt und gefordert, daß nicht einmal randomisierte Algorithmen in der Lage seien, Einwegfunktionen mit vernünftig beschränkter Fehlerwahrscheinlichkeit zu invertieren. Solche kryptographischen Einwegfunktionen werden hier nicht betrachtet.

Grollmann und Selman [GS88] zeigten, daß es injektive und surjektive Einwegfunktionen (d.h. Einwegpermutationen) genau dann gibt, wenn $P \neq UP \cap coUP$. Einwegfunktionen, die nicht notwendig injektiv oder surjektiv sind, existieren genau dann, wenn $P \neq NP$. Folglich kann man an die Existenz von Einwegfunktionen mit derselben Gewißheit glauben, mit der man an die Gültigkeit der Hypothese $P \neq NP$ glaubt. Mit Euler könnte man, in leichter Abwandlung seines Ausspruchs,[6] sagen: „$P \neq NP$, also existieren Gott und Einwegfunktionen!“

Interessanterweise hat auch die im vorigen Abschnitt definierte Klasse $EASY^{\forall}_{\forall}$ einen engen Bezug zur Existenz von Einwegfunktionen. Fenner et al. [FFNR96] charakterisieren die Existenz surjektiver Einwegfunktionen (welche nicht notwendig injektiv sind) durch die Separation $P \neq EASY^{\forall}_{\forall}$. Wechsung et al. [HRW97b] (siehe auch [RH96]) betrachten die UP und FewP [7] Analoga der Klasse $EASY^{\forall}_{\forall}$, bezeichnet mit $EASY^{\forall}_{\forall}(UP)$ bzw. $EASY^{\forall}_{\forall}(FewP)$, und zeigen, daß Separationen dieser Klassen von P geeignet sind, die Existenz verschiedener anderer Typen von Einwegfunktionen zu charakterisieren. Beispielsweise wird in [HRW97b, RH96] die Existenz von Einwegpermutationen durch jede der folgenden drei Bedingungen charakterisiert: (1) $EASY^{\forall}_{\forall}(UP) \neq P$; (2) $\Sigma^* \notin EASY^{\forall}_{\forall}(UP)$; und (3) $EASY^{\forall}_{\forall}(UP)$ ist nicht unter Komplementbildung abgeschlossen. Eine Konsequenz dieser Charakterisierung der Existenz von Einwegpermutationen und der von Grollmann und Selman ist, daß $EASY^{\forall}_{\forall}(UP) \neq P$ genau dann gilt, wenn $P \neq UP \cap coUP$. Mit anderen Worten, im Falle des UP-Analogs des Satzes von Borodin und Demers (siehe Satz 2.4) gilt sogar die Umkehrung; diese Aussage wurde von Hartmanis and Hemaspaandra [HH88] direkt bewiesen. Auch die Existenz *totaler* Einwegpermutationen wird in [HRW97b, RH96] charakterisiert.

Im folgenden betrachten wir einen speziellen Typ von Einwegfunktionen, die assoziativen Einwegfunktionen, die kürzlich von Rabi und Sherman [RS97] eingeführt wurden. Rabi und Sherman [RS97] präsentierten kryptographische Protokolle für *digital signatures* und für *unauthenticated secret-key agreement*, welche von ihnen selbst und von Rivest – dem wir u.a. das „R“ im berühmten RSA-Kryptosystem zu verdanken haben – und Sherman entwickelt wurden. Die kryptographischen Grundbausteine dieser Protokolle sind starke,[8] totale, kommutative (im Falle von *multi-party secret-key agreement*) und assoziative Einwegfunktionen. Obwohl Rabi und Sherman [RS97] bewiesen, daß assoziative Einwegfunktionen genau dann existieren, wenn $P \neq NP$, ließen sie die Frage offen, ob irgendeine natürliche komplexitätstheoretische Bedingung hinreichend für die Existenz starker, totaler, kommutativer und assoziativer Einwegfunktionen ist. Hemaspaandra und Rothe [HR] beantworten diese Frage.

---

[6] Wörtlich hatte Euler, im Streitgespräch mit Diderot, gesagt (siehe [Sin97]): „$(a + b^n)/n = x$, also existiert Gott, antworten Sie!“ Diderot blieb die Antwort schuldig und reiste aus St. Petersburg ab.

[7] UP [Val76] ist die Teilklasse von NP, deren Maschinen für jede Eingabe höchstens ein Zertifikat erlaubt ist. FewP [All86, AR88] ist die Teilklasse von NP, deren Maschinen höchstens polynomiell viele (in der Eingabegröße) Zertifikate für jede Eingabe erlaubt sind.

[8] Informell gesagt, eine zweistellige Einwegfunktion heißt *stark*, falls sie selbst dann schwer invertierbar ist, wenn eines ihrer Argumente gegeben ist. Siehe Definition 3.6.

Satz 3.1 [HR] *Starke, totale, kommutative und assoziative Einwegfunktionen existieren genau dann, wenn* $\mathrm{P} \neq \mathrm{NP}$.

Der Rest dieses Abschnitts ist dem Beweis von Satz 3.1 gewidmet. Ausgehend von der Annahme $\mathrm{P} \neq \mathrm{NP}$ werden wir skizzieren, wie eine Einwegfunktion, τ, mit den gewünschten Eigenschaften konstruiert werden kann. Die Konstruktion von τ wird wesentlich auf der Zertifikatkomplexität einer Menge in $\mathrm{NP} - \mathrm{P}$ beruhen. Bevor wir uns in die Details des Beweises stürzen können, müssen einige Bezeichnungen und Begriffe definiert und erklärt werden.

Den Definitionsbereich einer Funktion σ bezeichnen wir mit $\mathrm{domain}(\sigma)$ und den Wertebereich von σ mit $\mathrm{image}(\sigma)$. Funktionen sind nicht notwendig total, außer dies wird explizit gesagt. Unter einer *binären* Funktion soll eine zweistellige Funktion verstanden werden. Für binäre Funktionen $\sigma : \Sigma^* \times \Sigma^* \to \Sigma^*$ wird sowohl Präfix- als auch Infixnotation verwendet, d.h., $\sigma(x, y) = x\sigma y$. Um Paare von Wörtern als Wörter zu kodieren, verwenden wir wie üblich eine leicht berechenbare Bijektion, $\langle \cdot, \cdot \rangle : \Sigma^* \times \Sigma^* \to \Sigma^*$, die leicht berechenbare Projektionen hat und in keinem ihrer Argumente monoton fallend ist, wenn das andere Argument fixiert wird. Die folgende Definition von Einwegfunktionen ist die von Rabi und Sherman [RS97] (siehe auch [GS88, HR]) und ist auf den Spezialfall binärer Funktionen zugeschnitten.

Definition 3.2 *Eine binäre Funktion* $\sigma : \Sigma^* \times \Sigma^* \to \Sigma^*$ *heißt* ehrlich, *falls es ein Polynom* p *gibt, so daß für jedes* $z \in \mathrm{image}(\sigma)$ *ein Paar* $(x, y) \in \mathrm{domain}(\sigma)$ *existiert mit* $x\sigma y = z$ *und* $|x| + |y| \leq p(|z|)$.

Eine binäre Funktion σ *ist* FP-invertierbar, *falls es eine totale Funktion* $g \in \mathrm{FP}$ *gibt, so daß für jedes* $z \in \mathrm{image}(\sigma)$ *das Wort* $g(z)$ *eines der Urbilder von* z *unter* σ *ist, d.h.,* $g(z) = \langle x, y \rangle$ *für ein Paar* $(x, y) \in \mathrm{domain}(\sigma)$ *mit* $x\sigma y = z$.

Eine ehrliche Funktion in FP, *die nicht* FP-*invertierbar ist, heißt* Einwegfunktion.

Rabi und Sherman [RS97] definieren den Begriff der Assoziativität binärer Funktionen wie folgt. Diskussion 3.4 erklärt, warum ihr Begriff in Definition 3.3 als *schwache* Assoziativität bezeichnet wird.

Definition 3.3 *Eine binäre Funktion* $\circ : \Sigma^* \times \Sigma^* \to \Sigma^*$ *heißt* schwach assoziativ, *falls für alle* $x, y, z \in \Sigma^*$ *mit* $(x, y), (y, z), (x, y \circ z), (x \circ y, z) \in \mathrm{domain}(\circ)$ *gilt:*

$$x \circ (y \circ z) = (x \circ y) \circ z.$$

Diskussion 3.4 *Assoziativität drückt die Gleichheit zweier 3-stelliger Funktionen aus, die beide aus einer gegebenen binären Funktion hervorgehen. Definition 3.3 liefert einen Begriff der Assoziativität, der für nicht überall definierte (d.h. für nicht-totale) Funktionen nicht natürlich ist. Sind beispielsweise die Paare* (x, y), $(x \circ y, z)$ *und* (y, z) *im Definitionsbereich von* $\circ$, *aber* $(x, y \circ z)$ *ist es nicht, so können „Gleichungen" der Form* „undefiniert $= 1011$" *auftreten, die im Sinne von Definition 3.3 erlaubt sind. Sinnvoller ist es, bei der Definition der Assoziativität nicht notwendig totaler Funktionen zu verlangen, daß beide Seiten obiger Gleichung gemeinsam stehen oder fallen: Entweder sind beide Seiten undefiniert, oder sie sind beide definiert und nehmen denselben Wert an.*

Diese Beobachtung ist nicht neu. Aus der Rekursionstheorie sind zwei verschiedene Interpretationen der „Gleichheit“ nicht-totaler Funktionen bekannt: *schwache Gleichheit* und *vollständige Gleichheit*, siehe Kleene [Kle52]. Kleene schlägt den Gebrauch zweier verschiedener Gleichheitssymbole vor. Hier werden die Symbole „$=_w$“ und „$=_c$“ verwendet, und das folgende Zitat ist entsprechend verändert. Kleene schreibt [Kle52, pp. 327–328]:

> We now introduce "$\psi(x_1, \ldots, x_n) =_c \chi(x_1, \ldots, x_n)$" to express, for particular $x_1, \ldots, x_n$, that if either of $\psi(x_1, \ldots, x_n)$ and $\chi(x_1, \ldots, x_n)$ is defined, so is the other and the values are the same (and hence if either of $\psi(x_1, \ldots, x_n)$ and $\chi(x_1, \ldots, x_n)$ is undefined, so is the other). The difference in the meaning of (i) "$\psi(x_1, \ldots, x_n) =_w \chi(x_1, \ldots, x_n)$" and (ii) "$\psi(x_1, \ldots, x_n) =_c \chi(x_1, \ldots, x_n)$" comes when one of $\psi(x_1, \ldots, x_n)$ and $\chi(x_1, \ldots, x_n)$ is undefined. Then (i) is undefined, while (ii) is true or false according as the other is or is not undefined.

Vollständige Gleichheit ist der natürlichere der beiden Begriffe, und Definition 3.5 liefert einen Begriff der Assoziativität binärer Funktionen, der auf der vollständigen Gleichheit nicht notwendig totaler Funktionen beruht. Es kann gezeigt werden, daß die Ergebnisse der Arbeiten [RS97] und [HR] sogar unter dieser strengeren Definition gelten.

Definition 3.5 [HR] *Es sei $\sigma : \Sigma^* \times \Sigma^* \to \Sigma^*$ eine binäre Funktion. Definiere $\Gamma = \Sigma^* \cup \{\bot\}$ und eine Erweiterung $\widehat{\sigma} : \Gamma \times \Gamma \to \Gamma$ von σ auf Γ wie folgt:*

$$\widehat{\sigma}(a,b) = \begin{cases} \sigma(a,b) & \text{falls } a \neq \bot \text{ und } b \neq \bot \text{ und } (a,b) \in \mathrm{domain}(\sigma) \\ \bot & \text{sonst.} \end{cases}$$

Wir sagen, σ ist assoziativ, *falls für alle $x, y, z \in \Sigma^*$ gilt: $(x\widehat{\sigma}y)\widehat{\sigma}z = x\widehat{\sigma}(y\widehat{\sigma}z)$. Wir sagen, σ ist* kommutativ, *falls für alle $x, y \in \Sigma^*$ gilt: $x\widehat{\sigma}y = y\widehat{\sigma}x$, i.e., $x\sigma y =_c y\sigma x$.*

Jede assoziative Funktion ist schwach assoziativ; die Umkehrung gilt jedoch im allgemeinen nicht. Somit sind diese beiden Begriffe tatsächlich verschieden. Im folgenden steht „AOWF“ für „assoziative Einwegfunktion“.

Rabi and Sherman [RS97] führen auch den Begriff *starker* Einwegfunktionen ein. Das sind binäre Einwegfunktionen, die selbst dann schwer invertierbar sind, wenn eines ihrer Argumente gegeben ist. Hier ist die formale Definition (siehe [HR]).

Definition 3.6 *Eine binäre Einwegfunktion $\sigma : \Sigma^* \times \Sigma^* \to \Sigma^*$ heißt* stark, *falls weder* (a) *noch* (b) *gilt:*

(a) *Es gibt eine totale Funktion $g_1 \in \mathrm{FP}$, so daß für jedes $x \in \mathrm{image}(\sigma)$ und jedes $v \in \Sigma^*$ gilt: Wenn $\sigma(v, w) =_c x$ für ein $w \in \Sigma^*$, dann $\sigma(v, g_1(\langle v, x\rangle)) =_c x$.*

(b) *Es gibt eine totale Funktion $g_2 \in \mathrm{FP}$, so daß für jedes $x \in \mathrm{image}(\sigma)$ und jedes $w \in \Sigma^*$ gilt: Wenn $\sigma(v, w) =_c x$ für ein $v \in \Sigma^*$, dann $\sigma(g_2(\langle w, x\rangle), w) =_c x$.*

Beweisskizze von Satz 3.1. Es genügt zu zeigen, daß unter der Voraussetzung $\mathrm{P} \neq \mathrm{NP}$ binäre Funktionen existieren, die zugleich totale, kommutative, assoziative und starke Einwegfunktionen sind.

Es sei $\mathrm{P} \neq \mathrm{NP}$ angenommen, A sei eine Menge in $\mathrm{NP} - \mathrm{P}$ und M sei eine NPTM, die A akzeptiert. Ohne Beschränkung der Allgemeinheit nehmen wir an, daß für jedes Wort $x \in A$ und für jedes Zertifikat z für „$x \in A$" gilt, daß $|z| = p(|x|) > |x|$ für ein streng monoton wachsendes Polynom p, das von M abhängt. Für Wörter $u, v, w \in \Sigma^*$ bezeichne $\min(u, v)$ das lexikographisch kleinere Wort von u und v, und $\min(u, v, w)$ bezeichne das lexikographisch kleinste Wort von u, v und w.

Definiere die binäre Funktion $\sigma : \Sigma^* \times \Sigma^* \to \Sigma^*$ wie folgt:

$$\sigma(a,b) = \begin{cases} \langle x, \min(z_1, z_2)\rangle & \text{falls } (\exists x \in \Sigma^*)\,(\exists z_1, z_2 \in \mathrm{ACC}_M(x)) \\ & \qquad [a = \langle x, z_1\rangle \ \wedge \ b = \langle x, z_2\rangle] \\ \langle x, x\rangle & \text{falls } (\exists x \in \Sigma^*)\,(\exists z \in \mathrm{ACC}_M(x))\,[(a = \langle x, x\rangle \wedge \\ & \qquad b = \langle x, z\rangle) \ \vee \ (a = \langle x, z\rangle \ \wedge \ b = \langle x, x\rangle)] \\ \text{nicht definiert} & \text{sonst.} \end{cases}$$

Was ist die Intuition hinter der Definition von σ? Um die Assoziativität von σ zu sichern, wird die Anzahl der Zertifikate um eins reduziert, und zwar auf eine bestimmte, sorgfältige Weise, die in „Fall 1" unten genauer beschrieben wird.

Außerdem wird sich zeigen, daß σ selbst dann schwer invertierbar ist, wenn eines der Argumente von σ gegeben ist. Weshalb? Die Intuition ist, daß zur Inversion von σ Information über die Zertifikate für Elemente von A erforderlich ist. Unsere Annahme $A \notin \mathrm{P}$ garantiert aber, daß die Extraktion dieser Information nicht effizient machbar ist.

Der Beweis in [HR] erfolgt in zwei Schritten. Im ersten Schritt wird gezeigt, daß die oben definierte Funktion σ eine starke, kommutative AOWF ist. Da σ i.a. keine totale Funktion ist, wird im zweiten Schritt gezeigt, wie σ zu einer totalen Funktion τ erweitert werden kann, ohne dabei eine der anderen erforderlichen Eigenschaften einzubüßen. Wir beschränken uns hier auf den ersten Schritt.

Daß σ ehrlich ist, folgt sofort aus der Definition von σ. Daß σ eine Funktion in FP ist, kann wie folgt begründet werden: Bei Eingabe eines Wortes $\langle a, b\rangle$ wird geprüft, ob das entsprechende Paar (a, b) in $\mathrm{domain}(\sigma)$ ist, und wenn ja, welches der Wörter $\langle x, x\rangle$ oder $\langle x, z\rangle$ für geeignete $x \in \Sigma^*$ und $z \in \mathrm{ACC}_M(x)$ als Funktionswert $\sigma(a, b)$ auszugeben ist. Hier wird die Annahme gebraucht, daß für jedes Wort $x \in A$ und für jedes Zertifikat z für „$x \in A$" gilt, daß $|z| = p(|x|) > |x|$. Diese Annahme sichert, daß es keine Mehrdeutigkeit gibt, wenn bestimmt werden soll, ob a und b die Form $\langle x, x\rangle$ oder $\langle x, z\rangle$ für ein potentielles Zertifikat z für „$x \in A$" haben. Ob Wörter der Form $\langle x, z\rangle$ für ein potentielles Zertifikat z tatsächlich ein Zertifikat $z \in \mathrm{ACC}_M(x)$ enthalten, kann leicht verifiziert werden, da $\bigcup_{x \in \Sigma^*} \mathrm{ACC}_M(x)$ eine Menge in P ist. Diese Eigenschaft impliziert auch, daß $\mathrm{domain}(\sigma)$ eine Menge in P ist.

Nun zeigen wir, daß σ selbst dann nicht FP-invertierbar ist, wenn eines der Argumente gegeben ist. Angenommen, es gibt eine totale Funktion $g_2 \in \mathrm{FP}$ mit folgender Eigenschaft: Für jedes w im Bild von σ und für jedes zweite Argument b, für das ein Wort $a \in \Sigma^*$ mit $\sigma(a, b) =_c w$ existiert, gilt $\sigma(g_2(\langle b, w\rangle), b) =_c w$.

Dann könnte A, im Widerspruch zu unserer Annahme $A \notin \mathrm{P}$, folgendermaßen in Polynomialzeit entschieden werden:

> Bei Eingabe x wird das Wort $g_2(\langle\langle x, x\rangle, \langle x, x\rangle\rangle)$ berechnet, als ein Paar $\langle d, e\rangle$ interpretiert, und x wird genau dann akzeptiert, wenn $d = x$ und $e \in \mathrm{ACC}_M(x)$.

Ein analoger Beweis kann für den Fall eines fixierten ersten Arguments von σ geführt werden. Somit erfüllt σ keine der beiden Eigenschaften (a) und (b) in Definition 3.6, d.h., σ ist eine starke Einwegfunktion.

Es folgt der Beweis der Assoziativität von σ. Es sei $\widehat{\sigma}$ die Erweiterung von σ aus Definition 3.5. Fixiere drei beliebige Wörter aus Σ^*, $a = \langle a_1, a_2\rangle$, $b = \langle b_1, b_2\rangle$ und $c = \langle c_1, c_2\rangle$. Die Zahl k gebe an, wieviele der drei Wörter a_2, b_2, and c_2 in $\mathrm{ACC}_M(a_1)$ sind. Gilt beispielsweise $a_2 = b_2 = c_2 \in \mathrm{ACC}_M(a_1)$, dann ist $k = 3$. Um zu zeigen, daß

$$(a\widehat{\sigma}b)\widehat{\sigma}c = a\widehat{\sigma}(b\widehat{\sigma}c) \tag{3.2}$$

gilt, unterscheiden wir zwei Fälle.

Fall 1: Es gilt $[a_1 = b_1 = c_1 \ \wedge \ \{a_2, b_2, c_2\} \subseteq \{a_1\} \cup \mathrm{ACC}_M(a_1)]$. Wie bereits erwähnt, reduziert σ, und somit $\widehat{\sigma}$, die Anzahl der Zertifikate um eins. Insbesondere konserviert $\widehat{\sigma}$ das lexikographische Minimum, falls beide Argumente ein Zertifikat für „$a_1 \in A$“ enthalten. Enthält genau eines der Argumente ein Zertifikat für „$a_1 \in A$“, so hat $\widehat{\sigma}$ den Wert $\langle a_1, a_1\rangle$. Falls keines der beiden Argumente ein Zertifikat für „$a_1 \in A$“ enthält, ist σ nicht definiert, und somit hat $\widehat{\sigma}$ den Wert $\perp$. Daraus folgt:

- Ist $k \in \{0, 1\}$, dann gilt: $(a\widehat{\sigma}b)\widehat{\sigma}c = \perp = a\widehat{\sigma}(b\widehat{\sigma}c)$.
- Ist $k = 2$, dann gilt: $(a\widehat{\sigma}b)\widehat{\sigma}c = \langle a_1, a_1\rangle = a\widehat{\sigma}(b\widehat{\sigma}c)$.
- Ist $k = 3$, dann gilt: $(a\widehat{\sigma}b)\widehat{\sigma}c = \langle a_1, \min(a_2, b_2, c_2)\rangle = a\widehat{\sigma}(b\widehat{\sigma}c)$.

In jedem dieser drei Fälle ist Gleichung (3.2) erfüllt.

Fall 2: Es gilt nicht Fall 1. Dann gilt entweder $[a_1 \neq b_1 \ \vee \ a_1 \neq c_1 \ \vee \ b_1 \neq c_1]$ oder $[a_1 = b_1 = c_1 \ \wedge \ \{a_2, b_2, c_2\} \not\subseteq \{a_1\} \cup \mathrm{ACC}_M(a_1)]$. Nach Definition von σ folgt in beiden Fällen $(a\widehat{\sigma}b)\widehat{\sigma}c = \perp = a\widehat{\sigma}(b\widehat{\sigma}c)$, und Gleichung (3.2) ist erfüllt.

Damit ist die Assoziativität von σ bewiesen. Die Kommutativität von σ folgt sofort aus der Definition. Folglich ist σ eine starke, kommutative AOWF, wie behauptet. ∎

Rabi und Sherman [RS93] fragen auch, ob aus jeder gegebenen Einwegfunktion eine starke, totale, kommutative und assoziative Einwegfunktion konstruiert werden kann. Der Beweis von Satz 3.1 gibt auch auf diese Frage eine positive Antwort. Es ist bekannt (siehe [GS88]), wie man aus einer beliebigen Einwegfunktion eine Menge in NP − P erhält. Der Beweis von Satz 3.1 zeigt konstruktiv, wie aus einer

beliebigen Menge in NP − P eine starke, totale, kommutative und assoziative Einwegfunktion kreiert werden kann. Ganz konkret ist unter der Annahme P ≠ NP z.B. 3-`Colorability` eine Menge in NP − P, und die Konstruktion von τ, der totalen Erweiterung von σ im Beweis oben, kann mittels einer konkreten – z.B. der kanonischen – NP-Maschine für 3-`Colorability` deterministisch in Polynomialzeit ausgeführt werden.

Abschließend seien zwei weitere Resultate der Arbeit [HR] erwähnt. Obwohl Rabi und Sherman [RS97] bewiesen, daß keine totale AOWF injektiv sein kann, wird in [HR] gezeigt, daß eine nicht-totale, injektive AOWF genau dann existiert, wenn P ≠ UP. Außerdem wird in [HR] ein Gegenbeispiel zu einer Konstruktion gegeben, von der Rabi und Sherman [RS97] behaupten, sie konvertiere jede nicht-totale assoziative Einwegfunktion in eine totale. Genauer gesagt wird gezeigt, daß ein Beweis dieser Behauptung sofort UP = NP beweisen würde.

4 Heuristiken versus Vollständigkeit

Die vorigen beiden Abschnitte behandelten $\mathrm{EASY}_{\forall}^{\forall}$ und $\mathrm{EASY}_{\forall}^{\forall}(\mathrm{UP})$, die Klassen der Mengen in NP bzw. UP, deren sämtliche Maschinen leicht berechenbare Zertifikate haben. Diese Mengen sind so leicht, daß sie in P liegen, und unter vernünftigen komplexitätstheoretischen Annahmen gibt es sogar in P Mengen, die schwerer sind als sie. Dieser Abschnitt beschäftigt sich mit schweren Mengen in NP, genauer gesagt mit NP-vollständigen Mengen. Eine Menge A in NP ist NP-vollständig, falls jede Menge B aus NP in Polynomialzeit auf A many-one reduziert werden kann, d.h., es gibt eine Funktion $f \in \mathrm{FP}$, so daß für alle $x \in \Sigma^*$ gilt: $x \in B \iff f(x) \in A$.

NP-Vollständigkeitsresultate sind besonders deshalb so ärgerlich, weil sie die Unmöglichkeit ausdrücken, praktisch wichtige Probleme effizient zu lösen. Nehmen wir das Beispiel 3-`Colorability`. Einem Graph-Färber würde es die Arbeit erheblich erleichtern, könnte er effizient entscheiden, ob ein gegebener Graph 3-färbbar ist oder nicht. Da 3-`Colorability` jedoch NP-vollständig ist [Kar72] (siehe auch [Sto73, GJS76, GJ79]), bleibt ihm nichts übrig, als alle Färbungsmöglichkeiten durchzuprobieren, und das bedeutet: Überstunden und nicht mehr als drei Tage Urlaub im Jahr. Eine äußerst unbefriedigende Situation. Schlaue Graph-Färber wenden deshalb gern heuristische Algorithmen an, die schnell sind und in vielen Fällen korrekte Lösungen liefern. Doch nun stellt sich die Frage: *Für welche Eingaben kann eine gegebene Heuristik das Problem lösen*?

Betrachten wir noch einmal den Beispielgraphen G in Abbildung 1. G ist ein zusammenhängender Graph mit maximalem Grad 3,[9] und G ist nicht der K_4, der vollständige Graph mit 4 Knoten. Alle diese Eigenschaften können effizient getestet werden. 1941 hat Brooks [Bro41] bewiesen, daß ein zusammenhängender Graph mit maximalem Grad 3 genau dann 3-färbbar ist, wenn er nicht der K_4 ist. Deshalb ist G 3-färbbar, und die 3-Färbbarkeit von G kann effizient verifiziert werden, weil G so einfach strukturiert ist. Ebenso können Graphfärbungsheuristiken 3-`Colorability` effizient lösen, *falls* der gegebene Graph eine bestimmte, der jeweiligen Heuristik

[9] Der *Grad* eines Knoten v, bezeichnet mit $deg(v)$, ist die Anzahl seiner Nachbarknoten.

genehme Struktur hat. *Doch wie schwer ist es, für einen fixierten heuristischen Algorithmus $\mathcal{A}$ und einen gegebenen Graphen G zu erkennen, ob $\mathcal{A}$ bei Eingabe G die 3-Färbbarkeit von G korrekt entscheiden kann*? Die Arbeit [Rot98] studiert diese Frage für eine Anzahl prominenter Heuristiken zum Färben von Graphen. Es wird – vermutlich zum Ärger der Graph-Färber – gezeigt, daß die Einschränkung des Problems 3-`Colorability` auf solche Graphen, für die sich die fixierte Heuristik gut zur Entscheidung der 3-Färbbarkeit eignet, immer noch NP-vollständig ist.

Für Färbbarkeitsprobleme sind zahlreiche Heuristiken bekannt. Typischerweise besteht eine solche Heuristik aus zwei Teilen. Im ersten Teil wird eine geeignete Reihenfolge der Knoten des Graphen berechnet. Im zweiten Teil wird der eigentliche Färbungsalgorithmus ausgeführt, um die Knoten des Graphen in der gegebenen Reihenfolge zu färben. Eine der simpelsten und grundlegendsten Färbungsprozeduren ist der sogenannte sequentielle Algorithmus,[10] den wir hier mit `SEQ` bezeichnen. Für jeden Graphen gilt: Er ist genau dann 3-färbbar, wenn ihn der sequentielle Algorithmus für mindestens eine der $n!$ möglichen Reihenfolgen seiner Knoten 3-färbt. Das ist der Grund, weshalb die in Fußnote 3 angegebene NP-Maschine 3-`Colorability` akzeptiert.

Die lokale Aktion des sequentiellen Algorithmus erscheint ganz vernünftig, doch *global* scheitert er kläglich, wenn die gegebene Reihenfolge der Knoten unglücklich gewählt ist. Johnson [Joh74] gab eine Folge $G_3, \ldots, G_m, \ldots$ von 2-färbbaren Graphen an, deren Größe linear in m ist, doch für jedes $m \geq 3$ braucht der sequentielle Algorithmus bei Eingabe der Knoten von G_m in einer unglücklichen Reihenfolge mindestens m Farben. Das heißt, für diese Knotenfolge erreicht `SEQ` den schlechtestmöglichen Approximationsratio der chromatischen Zahl.

Liegt das vielleicht daran, daß die entsprechende Knotenfolge absichtlich – und unnötigerweise – ungeschickt gewählt wurde? Johnsons Resultate legen nahe, daß das nicht der Fall ist. Für eine Reihe von z.T. sehr cleveren Heuristiken, die entwickelt wurden, um eine geschickte Reihenfolge der Knoten zu finden, beweist Johnson analoge Aussagen über die Unfähigkeit des sequentiellen Algorithmus, die chromatische Zahl zu approximieren, selbst wenn er auf solche „geschickt“ gewählten Knotenfol-

[10] Angenommen, die Farben werden durch positive natürliche Zahlen repräsentiert und die Knoten des Eingabegraphen sind in der Reihenfolge $v_1, v_2, \ldots, v_n$ gegeben. Der sequentielle Algorithmus durchläuft die Knoten in dieser Reihenfolge und ordnet dabei jedem Knoten v_i die kleinste zur Verfügung stehende Farbe zu, d.h. die kleinste Farbe, die bisher noch keinem Nachbarknoten von v_i zugewiesen wurde. In Abschnitt 2 wurde der sequentielle Algorithmus bereits im Beispiel einer NP-Maschine für 3-`Colorability` erwähnt. Bitte führen Sie folgende Schritte aus:

1. Wenn Sie sich an dieses Beispiel noch erinnern, verlassen Sie die gegenwärtige Fußnote bitte *jetzt*, ohne sie zu Ende zu lesen. Gehen Sie zurück in den Haupttext.

2. Lesen Sie Schritt 2 bereits zum zweiten Mal? Dann haben Sie ein schlechtes Kurzzeitgedächtnis (siehe [Sac85, Part I, Section 2]) und sind in Gefahr, in eine Endlosschleife zu geraten: Fußnote 10 → Fußnote 3 → Fußnote 10 → Fußnote 3 → $\cdots$. In diesem Fall ist hier der Notausgang: Sie sollten ganz von vorn beginnen und diesmal die Fußnoten ignorieren... falls Sie sich das merken können.

   Lesen Sie Schritt 2 jedoch zum ersten Mal, dann gehen Sie zu Schritt 3.

3. Gehen Sie zurück zu Fußnote 3, und lesen Sie dort noch einmal nach.

gen angewendet wird. Tatsächlich zeigten Feige and Kilian [FK96] vor kurzem, daß unter der Annahme NP $\neq$ ZPP kein deterministischer Polynomialzeit-Algorithmus die chromatische Zahl in einem Faktor von $\mathcal{O}(n^{1-\epsilon})$ approximieren kann, für jede fixierte Konstante $\epsilon > 0$. In den letzten Jahren wurden eine Anzahl weiterer beeindruckender Inapproximierbarkeitsergebnisse erreicht, die diese von Johnson und anderen initiierte Forschungsrichtung krönen.

Eine einfache Prozedur zum Finden geeigneter Knotenfolgen ist die Ordnung der Knoten nach *fallendem Grad*, wobei die Prozedur bei mehreren Knoten gleichen Grades die Möglichkeit der nichtdeterministischen Auswahl hat. Diese Prozedur ist jedoch in gewissem Sinne statisch, da sie nur vom Ausgangsgraphen abhängt. Das heißt, wenn eine Teilmenge der Knoten bereits geordnet und die komplementäre Teilmenge noch zu ordnen ist, kann diese Prozedur nicht auf eine neue Situation reagieren, die neue Anforderungen an die zu berechnende Knotenfolge mit sich bringen kann.

Eine flexiblere Methode ist die rekursive *smallest-last* (kurz `SL`) Ordnungsprozedur von Matula et al. [MMI72], die dynamisch wie folgt vorgeht. Gegeben sei ein Graph G mit n Knoten; $V(G)$ bezeichne die Knotenmenge von G. Wähle einen Knoten von kleinstem Grad als den letzten Knoten, v_n. Für $i > 1$ seien $v_i, \ldots, v_n$ die bereits geordneten Knoten. Wähle einen Knoten von kleinstem Grad im Teilgraphen von G, der durch die Knotenmenge $V(G) - \{v_i, \ldots, v_n\}$ induziert wird, als den nächsten Knoten, v_{i-1}. Fahre in dieser Weise induktiv rückwärts fort, bis alle Knoten geordnet sind. Wie man sieht, hat auch die `SL`-Ordnungsprozedur nichtdeterministische Auswahlmöglichkeiten.

Kombiniert man den sequentiellen Algorithmus mit der `SL`-Ordnungsprozedur zu einer Heuristik, bezeichnet mit `SL-SEQ`, so kann die folgende Einschränkung des Problems 3-`Colorability` definiert und hinsichtlich der Komplexität untersucht werden.

> **Entscheidungsproblem**: `SL-SEQ-3-Colorability`
>
> **Instanz**: Ein Graph G.
>
> **Frage**: Gibt es eine Folge von nichtdeterministischen Verzweigungen (bezüglich der Auswahl unter Knoten kleinsten Grades) bei der `SL`-Ordnung der Knotenmenge von G, so daß der `SEQ`-Algorithmus, bei Durchlauf der Knoten von G in dieser Reihenfolge, den Graphen G legal 3-färbt?

Satz 4.1 [Rot98] `SL-SEQ-3-Colorability` *ist* NP*-vollständig.*

Auf den vollständigen Beweis des Satzes wird hier verzichtet, nur die Idee soll kurz skizziert werden. Die obere Schranke, `SL-SEQ-3-Colorability` $\in$ NP, ist klar. Um die untere Schranke zu beweisen, wird die Konstruktion, mit der Stockmeyer ([Sto73], siehe auch [GJS76]) 3-`SAT` auf 3-`Colorability` reduzierte, geeignet modifiziert. Der Schlüssel dafür ist das folgende Lemma. Die Intuition hinter Lemma 4.2 ist die folgende.

Angenommen, die `SL`-Ordnung eines gegebenen Graphen E' wird gerade berechnet, E ist der Teilgraph von E', der durch die noch zu ordnende Knotenmenge

induziert wird, und zwei Knoten, u und v, haben einen Grad in E, so daß die SL-Ordnungsprozedur u vor v plaziert. Nehmen wir weiter an, daß die Struktur des Graphen E' einen Tausch von u und v in der Knotenordnung erfordert, damit der sequentielle Algorithmus eine legale 3-Färbung von E' berechnen kann, falls eine existiert. Der Zweck von Lemma 4.2 ist zu zeigen, wie u und v ihren Platz in der SL-Ordnung vertauschen können.

Natürlich erklärt Lemma 4.2 nur einen lokalen Teil der Gesamtkonstruktion im Beweis von Satz 4.1, in dem es vielfach angewendet wird. Global muß darauf geachtet werden, daß die Anwendung des Lemmas an jeder Stelle konsistent ist mit der an jeder anderen Stelle. Grob gesagt ist Lemma 4.2 eine Art Blindenhund für den SEQ-Algorithmus, und die verschiedenen Größen der Graphen $D_{u,s}$ aus Lemma 4.2 werden für die verschiedenen Anwendungen des Lemmas in der Gesamtkonstruktion so gewählt, daß der SEQ-Algorithmus keinen Fehltritt macht.

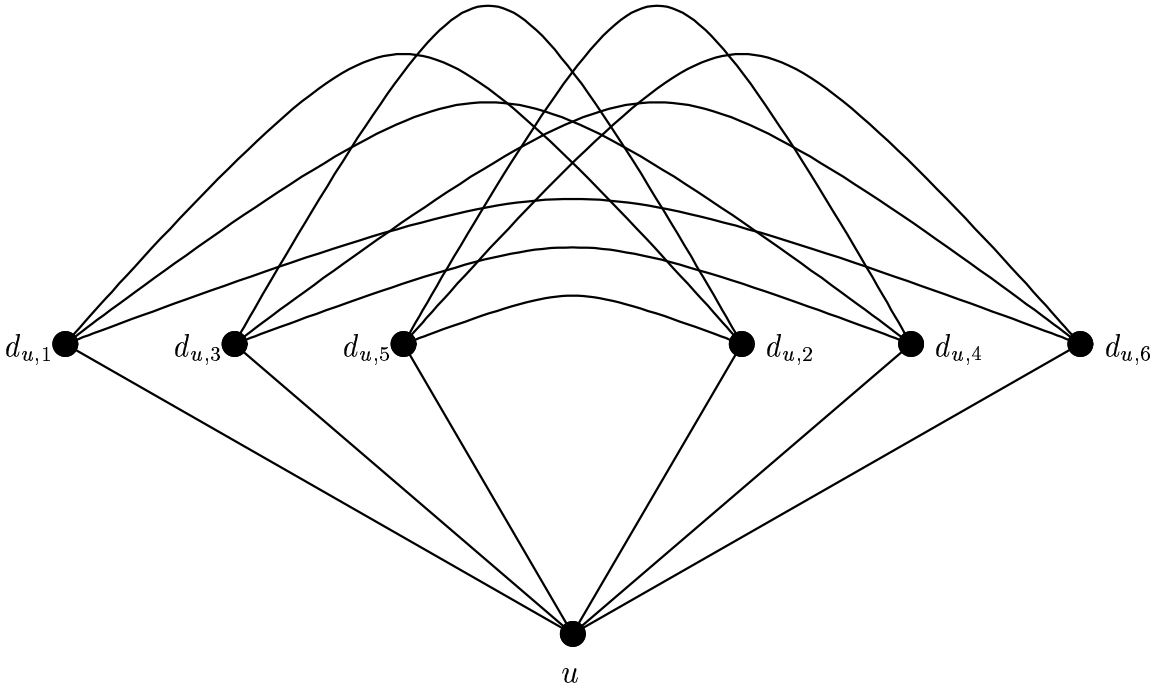

Abbildung 2: Graph $D_{u,4}$ für Lemma 4.2

Lemma 4.2 [Rot98] *Es sei E ein gegebener Graph, u und v seien Knoten in E, so daß $deg(v) > deg(u) > 0$ in E und $s = deg(v)$. Es existiert ein Graph $D_{u,s}$ mit $V(D_{u,s}) = \{u\} \cup \{d_{u,i} \mid 1 \leq i \leq 2(s-1)\}$, so daß $V(E) \cap V(D_{u,s}) = \{u\}$ und im Vereinigungsgraphen, $E \cup D_{u,s}$, gilt:*

(i) *$deg(u) > deg(v)$.*

(ii) *Die Knotenmenge $V(D_{u,s}) \cup \{v\}$ kann wie folgt SL-geordnet werden:*
$u, d_{u,1}, d_{u,2}, \ldots, d_{u,2(s-1)}, v$.

(iii) *Bei Durchlauf der Knoten von $D_{u,s}$ in dieser Reihenfolge findet der Algorithmus* `SEQ` *eine legale 3-Färbung von $D_{u,s}$, egal mit welcher Farbe $i \in \{1,2,3\}$ er startet, um u zu färben.*

Um Lemma 4.2 zu beweisen, verwenden wir eine Beweismethode, die sich bei den Studenten in Professor Wechsungs Vorlesungen stets größter Beliebtheit erfreute. Diese Methode zeigt, daß Formalismus nur als ein Geländer dienen soll, an dem man sich in der Not festhalten kann, nicht aber als ein festgefügtes Mauerwerk, das den Blick auf die Intuition und die Ideen verstellt. Das heißt, wir führen den

Beweis von Lemma 4.2 am Beispiel. Zum Beweis von Lemma 4.2 blicke man 30 Sekunden intensiv auf Abbildung 2. ∎

Nun werden noch einige verwandte Resultate und offene Fragen erwähnt. Satz 4.1 sagt, daß eine bestimmte, von einer fixierten Heuristik induzierte Einschränkung des Problems 3-`Colorability` NP-vollständig bleibt. Offene Fragen ergeben sich in mehr als einer Hinsicht. Zum einen kann man dieselbe Frage natürlich auch für andere Heuristiken untersuchen. Aufgrund des jahrhundertealten theoretischen Interesses an den Problemen der Graphfärbbarkeit (siehe Appel und Haken [AH77a, AH77b]) sowie ihrer praktischen Bedeutsamkeit (siehe Garey und Johnson [GJ79]) sind gerade für das Färben von Graphen zahlreiche Heuristiken entwickelt worden.

In [Rot98] werden analoge NP-Vollständigkeitsresultate für Einschränkungen des Problems 3-`Colorability` erhalten, die von sieben verschiedenen Heuristiken induziert werden, zu denen z.B. der Algorithmus von Wood [Woo69] gehört. Doch es gibt noch viele weitere heuristische Algorithmen; allein in Johnsons Arbeit [Joh74] werden 13 prominente Heuristiken für das Färben von Graphen untersucht, wie zum Beispiel der *Approximate-Maximum-Independent-Set* Algorithmus.

Hier ist noch ein anderer Aspekt. Satz 4.1 bezieht sich auf eine Einschränkung des *Entscheidungsproblems* 3-`Colorability`. Man kann auch nach der Komplexität des von einer fixierten Heuristik induzierten *Optimierungsproblems* für Graphfärbbarkeit fragen. Wie schwer ist es, für eine fixierte Heuristik $\mathcal{H}$ und einen gegebenen Graphen G zu erkennen, ob $\mathcal{H}$ die chromatische Zahl von G exakt bestimmen kann? Dieselbe Frage kann man für das *Approximierbarkeitsproblem* stellen: Wie schwer ist es, für eine fixierte Heuristik $\mathcal{H}$, eine fixierte Konstante r und einen gegebenen Graphen G zu erkennen, ob $\mathcal{H}$ die chromatische Zahl von G in einem Faktor von r approximieren kann?

Für das Problem `Independent Set` und den *Minimum-Degree-Greedy* Algorithmus, eine bekannte Heuristik zum Finden maximaler unabhängiger Mengen in einem Graphen, wurden alle diese Fragen vor kurzem gelöst. Bodlaender, Thilikos und Yamazaki [BTY97] zeigten, daß das entsprechende Entscheidungsproblem NP-vollständig bleibt. Hemaspaandra und Rothe [HR98] zeigten, daß das Approximierbarkeitsproblem, und somit insbesondere das Optimierungsproblem, vollständig ist für $\mathrm{P}_{||}^{\mathrm{NP}}$, die Klasse der Probleme, die mittels parallelem Zugriff auf ein NP-Orakel in Polynomialzeit gelöst werden können.

Die Klasse $\mathrm{P}_{||}^{\mathrm{NP}}$ ist kürzlich wieder stark in den Blickpunkt des Interesses getreten und hat sich als sehr wichtig für die Bestimmung der Komplexität natürlicher

Probleme erwiesen, siehe [HHR97b]. Das oben erwähnte $P_{||}^{NP}$-Vollständigkeitsresultat [HR98] sowie jedes der im folgenden zitierten Resultate basieren auf Techniken, die bereits vor mehr als einem Jahrzehnt in einer exzellenten Arbeit des Wechsung-Schülers Wagner [Wag87] eingeführt wurden.

Hemaspaandra, Hemaspaandra und Rothe [HHR97a] zeigen, daß das Problem, den Sieger in einem gegebenen Dodgson-Wahlsystem zu bestimmen, $P_{||}^{NP}$-vollständig ist. Das Dodgson-Wahlsystem wurde 1876 von Charles L. Dodgson eingeführt, dem Autor von „Alice's Adventures in Wonderland", der besser bekannt ist unter seinem Pseudonym, Lewis Carroll. Das Resultat in [HHR97a] beantwortet eine offene Frage von Bartholdi, Tovey und Trick [BTT89]. Hemaspaandra und Wechsung [HW97] zeigen, daß das Problem `Minimum Equivalent Expression` (kurz `MEE`; siehe [GJ79]) $P_{||}^{NP}$-schwer ist, wodurch die vorher bekannte triviale untere Schranke – `MEE` ist coNP-schwer – beträchtlich verbessert wird. Ob `MEE` in $P_{||}^{NP}$ und somit $P_{||}^{NP}$-vollständig ist, bleibt weiterhin offen. Die beste bekannte obere Schranke für `MEE` ist die triviale, NP^{NP}. Varianten des Problems `MEE` hatten ursprünglich die Einführung der Polynomialzeit-Hierarchie [MS72, Sto77] motiviert, und die Bestimmung der präzisen Komplexität dieses Problems, und seiner Varianten, ist seit langem eine wichtige offene Frage.

Literatur

[AH77a] K. Appel and W. Haken. Every planar map is 4-colorable – 1: Discharging. *Illinois J. Math*, 21:429–490, 1977.

[AH77b] K. Appel and W. Haken. Every planar map is 4-colorable – 2: Reducibility. *Illinois J. Math*, 21:491–567, 1977.

[All86] E. Allender. The complexity of sparse sets in P. In *Proceedings of the 1st Structure in Complexity Theory Conference*, pages 1–11. Springer-Verlag *Lecture Notes in Computer Science #223*, June 1986.

[AR88] E. Allender and R. Rubinstein. P-printable sets. *SIAM Journal on Computing*, 17(6):1193–1202, 1988.

[BD76] A. Borodin and A. Demers. Some comments on functional self-reducibility and the NP hierarchy. Technical Report TR 76-284, Cornell Department of Computer Science, Ithaca, NY, July 1976.

[BGS75] T. Baker, J. Gill, and R. Solovay. Relativizations of the P=?NP question. *SIAM Journal on Computing*, 4(4):431–442, 1975.

[Bro41] R. Brooks. On coloring the nodes of a network. *Proc. Cambridge Philosophical Society*, 37:194–197, 1941.

[BTT89] J. Bartholdi III, C. Tovey, and M. Trick. Voting schemes for which it can be difficult to tell who won the election. *Social Choice and Welfare*, 6:157–165, 1989.

[BTY97] H. L. Bodlaender, D. Thilikos, and K. Yamazaki. It is hard to know when greedy is good for finding independent sets. *Information Processing Letters*, 61:101–106, 1997.

[Cha66] G. Chaitin. On the length of programs for computing finite binary sequences. *Journal of the ACM*, 13:547–569, 1966.

[CS93] P. Crescenzi and R. Silvestri. Sperner's lemma and robust machines. In *Proceedings of the 8th Structure in Complexity Theory Conference*, pages 194–199. IEEE Computer Society Press, 1993.

[FFNR96] S. Fenner, L. Fortnow, A. Naik, and J. Rogers. On inverting onto functions. In *Proceedings of the 11th Annual IEEE Conference on Computational Complexity*, pages 213–222. IEEE Computer Society Press, May 1996.

[FK96] U. Feige and J. Kilian. Zero knowledge and the chromatic number. In *Proceedings of the 11th Annual IEEE Conference on Computational Complexity*, pages 278–287. IEEE Computer Society Press, 1996.

[FR94] L. Fortnow and J. Rogers. Separability and one-way functions. In *Proceedings of the 5th International Symposium on Algorithms and Computation*, pages 396–404. Springer-Verlag *Lecture Notes in Computer Science #834*, August 1994.

[GJ79] M. Garey and D. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman and Company, 1979.

[GJS76] M. Garey, D. Johnson, and L. Stockmeyer. Some simplified NP-complete graph problems. *Theoretical Computer Science*, 1:237–267, 1976.

[GS88] J. Grollmann and A. Selman. Complexity measures for public-key cryptosystems. *SIAM Journal on Computing*, 17(2):309–335, 1988.

[Har83] J. Hartmanis. Generalized Kolmogorov complexity and the structure of feasible computations. In *Proceedings of the 24th IEEE Symposium on Foundations of Computer Science*, pages 439–445. IEEE Computer Society Press, 1983.

[HH88] J. Hartmanis and L. Hemachandra. Complexity classes without machines: On complete languages for UP. *Theoretical Computer Science*, 58:129–142, 1988.

[HHR97a] E. Hemaspaandra, L. Hemaspaandra, and J. Rothe. Exact analysis of Dodgson elections: Lewis Carroll's 1876 voting system is complete for parallel access to NP. *Journal of the ACM*, 44(6):806–825, 1997.

[HHR97b] E. Hemaspaandra, L. Hemaspaandra, and J. Rothe. Raising NP lower bounds to parallel NP lower bounds. *SIGACT News*, 28(2):2–13, June 1997.

[HR] L. Hemaspaandra and J. Rothe. Creating strong, total, commutative, associative one-way functions from any one-way function in complexity theory. *Journal of Computer and System Sciences*. To appear. Available as: University of Rochester Department of Computer Science Technical Report TR-98-688, May 1998.

[HR98] E. Hemaspaandra and J. Rothe. Recognizing when greed can approximate maximum independent sets is complete for parallel access to NP. *Information Processing Letters*, 65(3):151–156, February 1998.

[HRW97a] L. Hemaspaandra, J. Rothe, and G. Wechsung. Easy sets and hard certificate schemes. *Acta Informatica*, 34(11):859–879, 1997.

[HRW97b] L. Hemaspaandra, J. Rothe, and G. Wechsung. On sets with easy certificates and the existence of one-way permutations. In *Proceedings of the Third Italian Conference on Algorithms and Complexity*, pages 264–275. Springer-Verlag *Lecture Notes in Computer Science #1203*, March 1997.

[HW97] E. Hemaspaandra and G. Wechsung. The minimization problem for boolean formulas. In *Proceedings of the 38th IEEE Symposium on Foundations of Computer Science*, pages 575–584. IEEE Computer Society Press, November 1997.

[HY84] J. Hartmanis and Y. Yesha. Computation times of NP sets of different densities. *Theoretical Computer Science*, 34(1/2):17–32, 1984.

[IN88] R. Impagliazzo and M. Naor. Decision trees and downward closures. In *Proceedings of the 3rd Structure in Complexity Theory Conference*, pages 29–38. IEEE Computer Society Press, June 1988.

[Joh74] D. Johnson. Worst case behavior of graph coloring algorithms. In *Proceedings of the 5th Southeastern Conference on Combinatorics, Graph Theory, and Computing*, pages 513–527, February/March 1974.

[Kar72] R. Karp. Reducibilities among combinatorial problems. In R. Miller and J. Thatcher, editors, *Complexity of Computer Computations*, pages 85–103, 1972.

[Kle52] S. C. Kleene. *Introduction to Metamathematics*. D. van Nostrand Company, Inc., New York and Toronto, 1952.

[Ko85] K. Ko. On some natural complete operators. *Theoretical Computer Science*, 37(1):1–30, 1985.

[Kol65] A. Kolmogorov. Three approaches for defining the concept of information quantity. *Prob. Inform. Trans.*, 1:1–7, 1965.

[MMI72] D. Matula, G. Marble, and J. Isaacson. Graph coloring algorithms. In R. Read, editor, *Graph Theory and Computing*, pages 109–122. Academic Press, 1972.

[MS72] A. Meyer and L. Stockmeyer. The equivalence problem for regular expressions with squaring requires exponential space. In *Proceedings of the 13th IEEE Symposium on Switching and Automata Theory*, pages 125–129, 1972.

[RH96] J. Rothe and L. Hemaspaandra. Characterizations of the existence of partial and total one-way permutations. Technical Report Math/Inf/96/7, Friedrich-Schiller-Universität Jena, Institut für Informatik, Jena, Germany, April 1996.

[Rog67] H. Rogers, Jr. *The Theory of Recursive Functions and Effective Computability*. McGraw-Hill, 1967.

[Rot98] J. Rothe. Heuristics versus completeness for graph coloring. Technical Report Math/Inf/98/29, Friedrich-Schiller-Universität Jena, Institut für Informatik, Jena, Germany, November 1998.

[RS93] M. Rabi and A. Sherman. Associative one-way functions: A new paradigm for secret-key agreement and digital signatures. Technical Report CS-TR-3183/UMIACS-TR-93-124, Department of Computer Science, University of Maryland, College Park, Maryland, 1993. Available online at http://www.cs.umbc.edu/pub/REPORTS/cs-93-18.ps.

[RS97] M. Rabi and A. Sherman. An observation on associative one-way functions in complexity theory. *Information Processing Letters*, 64(2):239–244, 1997.

[Sac85] O. Sacks. *The Man Who Mistook His Wife For a Hat*. Summit Books Simon & Schuster, Inc., New York, 1985.

[Sin97] S. Singh. *Fermat's Last Theorem*. John Lynch, 1997.

[Sto73] L. Stockmeyer. Planar 3-colorability is NP-complete. *SIGACT News*, 5(3):19–25, 1973.

[Sto77] L. Stockmeyer. The polynomial-time hierarchy. *Theoretical Computer Science*, 3(1):1–22, 1977.

[Val76] L. Valiant. The relative complexity of checking and evaluating. *Information Processing Letters*, 5(1):20–23, 1976.

[Wag87] K. Wagner. More complicated questions about maxima and minima, and some closures of NP. *Theoretical Computer Science*, 51:53–80, 1987.

[Woo69] D. Wood. A technique for coloring a graph applicable to large scale time-tabling problems. *The Computer Journal*, 12:317–319, 1969.

[WW86] K. Wagner and G. Wechsung. *Computational Complexity*. D. Reidel Publishing Company, 1986. Distributors for the U.S.A. and Canada: Kluwer Academic Publishers.