Aus dem Institut für Theoretische Informatik
der Medizinischen Universität zu Lübeck
Direktor: Prof. Dr. math. R. Reischuk

Average- und Median-Komplexitätsklassen

Inauguraldissertation
zur Erlangung der Doktorwürde
der Medizinischen Universität zu Lübeck
aus der technisch-naturwissenschaftlichen Fakultät

Vorgelegt von
Christian Schindelhauer
aus Lübeck

Lübeck 1996

Berichterstatter:	Prof. Dr. math. R. Reischuk
	Prof. Dr. rer. nat. R. Lasser
	Prof. Dr. rer. nat. U. Schöning
Tag der Prüfung:	14.05.1996
zum Druck genehmigt:	14.05.1996

gezeichnet Prof. Dr. ing. Dr. med. habil. S. Pöppl
Dekan der technisch-naturwissenschaftlichen Fakultät.

Für meine Ehefrau Barbara
in Liebe

und meine Eltern Wilfriede und Wilfried

Danksagung

Meinem Lehrer Rüdiger Reischuk danke ich an dieser Stelle für die Hilfe und Unterstützung, die er mir zukommen ließ.

Dank für Diskussionen, Anregungen und Kritik gebührt meinen Kollegen aus Darmstädter und Lübschen Zeiten: Karin Genther, Michael Goedecke, Andreas Jakoby, Maciej Liskiewicz, Stephan Weis, Rolf Wiehagen und Thomas Zeugmann.

Ohne den von meinen Eltern erwiesenen Rückhalt wäre diese Arbeit nicht möglich gewesen.

Besonders danke ich noch meiner Lektorin in Personalunion mit meiner Ehefrau für ihre Geduld.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Kapitel 1

Problemstellung und Einleitung

1.1 Problemstellung

Die Standardmaße für *average*-Betrachtungen im Bereich der Komplexitätstheorie sind der Erwartungswert und das Levinsche Durchschnittsmaß. Diese bergen gewisse Unzulänglichkeiten: So unterstützt der Erwartungswert keine Übertragung von komplexitätstheoretischen Ergebnissen anderer Maschinenmodelle, wie sie durch die Technik der Simulation sonst möglich ist. Das Levinsche Maß bietet dies, kann dagegen nur unpräzise klassifizieren.

Dieser Sachverhalt wurde schon in der Diplomarbeit des Autors [Schi 91] diskutiert. Ein Ergebnis dieser Arbeit war die Definition des Maßes **Av**, welches sowohl präzise ist als auch Simulationsergebnisse unterstützt. Die zugehörige Komplexitätsklasse **AvTime(T,C)** mit Zeitschranke T und Menge C von Wahrscheinlichkeitsverteilungen besteht aus allen Sprachen L, die für jede Wahrscheinlichkeitsverteilung μ aus C eine Maschine M besitzen, die L akzeptiert und deren Zeitfunktion time_M bezüglich μ $\mathrm{Av}(T)$ -beschränkt ist.

Als Komplexitätsklasse der Wahrscheinlichkeitsverteilungen wurde die Klasse **rankable** statt der bisher bekannten Klassen **computable** [Levin 86] und **sampleable** [BCGL 89] betrachtet, da rankable dem Av-Maß am besten angepaßt ist.

Mit Hilfe dieser Klasse wurde in der Diplomarbeit die Zeithierarchie

$$\mathrm{AvTime}(o(T),\, V\text{–rankable}) \;\subset\; \mathrm{AvTime}(T,\, V\text{–rankable})$$

bewiesen. Für große Schranken $V = T \cdot \mathrm{ExL}$ und $T \in \mathrm{POL}$ wurde dort gezeigt, daß gilt

$$\mathrm{AvTime}(T,\, T \cdot \mathrm{ExL}\text{–rankable}) \;=\; \mathrm{DTime}(T)\;.$$

Ausgehend von diesen Ergebnissen sollen in der vorliegenden Arbeit weitere Ansätze verfolgt werden. So soll das Arsenal der Durschnittsmaße erweitert werden um ein dem Erwartungswert stark verwandtes Maß, welches wir **Eav** nennen werden. Auch ein dem Median verwandtes Maß, **Med** genannt, soll gefunden werden, das den besonderen Notwendigkeiten der Komplexitätstheorie genügt.

Ferner soll untersucht werden, wie sich Av, Eav und Med bezüglich elementarer Operatoren wie z.B. lineare Abbildungen, Summe oder Produkt verhält. Zusätzlich sollen diese drei Durchschnittsmaße miteinander verglichen werden.

Aus den Durchschnittsmaßen Eav und Med ergeben sich analoge Komplexitätsklassen wie bereits für Av. Für diese sollen Klassifikationen gefunden werden. Insbesondere soll die Existenz von Hierarchien der Form

$$\mathrm{AvTime}(T,\, V\text{–rankable}) \;\subset\; \mathrm{AvTime}(T,\, o(V)\text{–rankable})$$

untersucht werden.

Die Komplexitätsklassen basierend auf Med können bei geeigneter Definition auch nicht berechenbare Probleme enthalten. So sollen insbesondere zwei nicht-berechenbare Probleme, das Halte-Problem der Turing-Maschinen und das Kolmogoroff-Komplexitätsproblem, mit Hilfe dieser Median-Zeit-Klasse neu bewertet werden.

Die aus Av, Eav und Med resultierenden deterministischen polynomiellen Zeit-Klassen **AvDis** $\boldsymbol{\mathcal{P}}$, **EavDis** $\boldsymbol{\mathcal{P}}$ und **MedDis** $\boldsymbol{\mathcal{P}}$ bilden Alternativen zu der Klasse polynomiell zeit-berechenbarer Probleme $\boldsymbol{\mathcal{P}}$. Diese Klassen sollen untereinander in Beziehung gesetzt werden.

Zuletzt soll untersucht werden, inwieweit diese Klassen polynomiell zeit-beschränkte Reduktionen unterstützen, wie diese aussehen und ob es vollständige Probleme in $\mathcal{NP}$ bezüglich solcher Reduktionen gibt.

1.2 Einleitung

In dieser Arbeit untersuchen wir das durchschnittliche Laufzeitverhalten deterministischer Algorithmen. Dabei modellieren wir Algorithmen mittels deterministischer Turing-Maschinen. Das folgende Beispiel soll die Fragestellung verdeutlichen:

Beispiel:
Die Sprache der **Palindrome** $\boldsymbol{L_{\mathrm{Pal}}}$ ist die Menge aller Zeichenketten (Wörter), die vorwärts wie rückwärts gelesen gleich lauten, zum Beispiel `ANNA`,

TAT oder OTTO. Wir wollen nun herausfinden, wieviel zeitlicher Aufwand notwendig ist, für ein gegebenes Wort w zu entscheiden, ob es ein Palindrom ist.

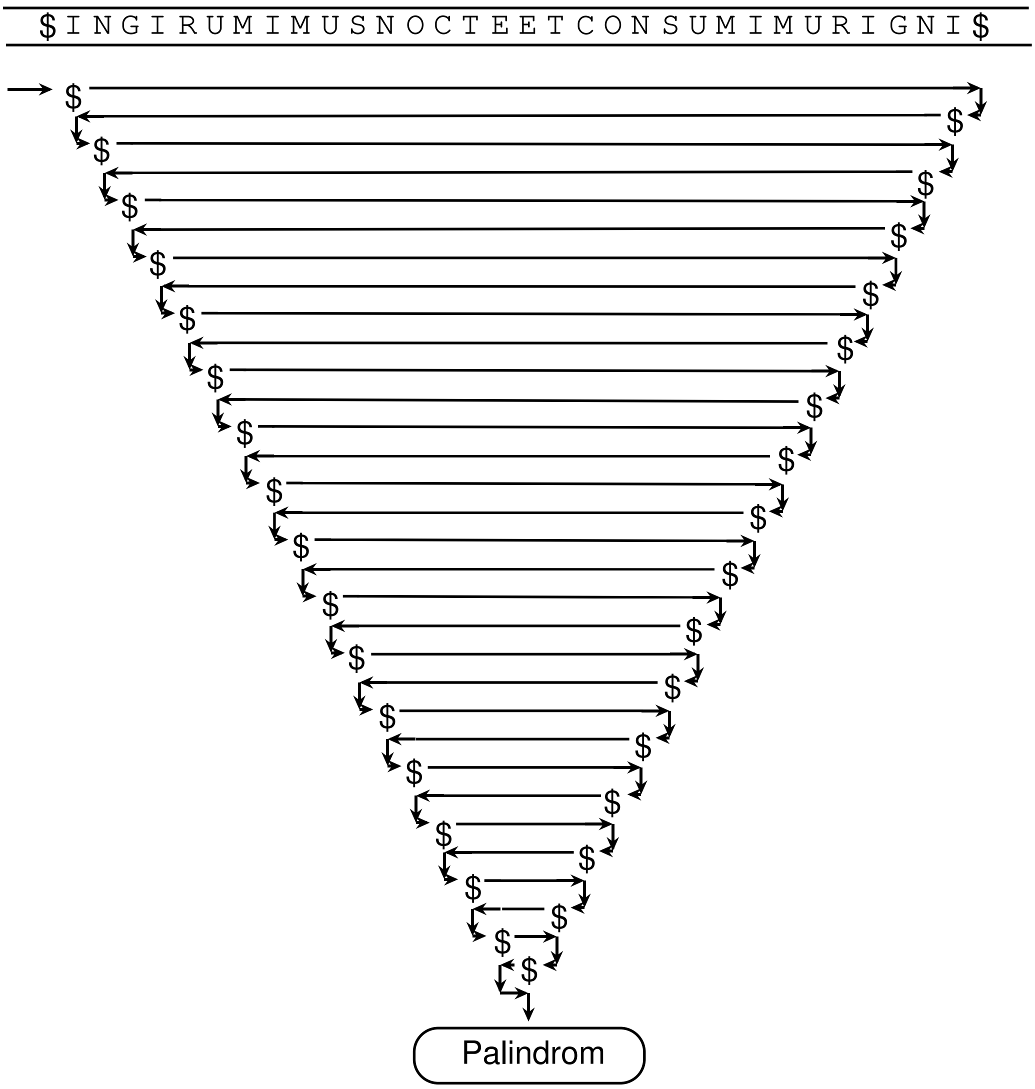

Abbildung 1.1: Eine 1-Band-Turing-Maschine kann ein Palindrom in quadratischer Laufzeit erkennen. Diese hier merkt sich ein Zeichen der linken Seite, überschreibt es mit \$ und vergleicht es mit dem auf der rechten Seite. Dieses wird dann mit \$ überschrieben, der Schreib- und Lese-Kopf fährt nach links und beginnt von Neuen.

Abbildung 1.1 skizziert die Kopf-Bewegungen einer 1-Band-Turing-Maschine, die diese Frage in quadratisch vielen Schritten bezogen auf die Länge der

Eingabe beantwortet. Diese Zeit-Aussage bezieht sich auf die *worst-case*-Laufzeit einer solchen Maschine. Der schlimmste Fall (*worst-case*) wäre für diesen Algorithmus ein Palindrom als Eingabe:

`INGIRUMIMUSNOCTEETCONSUMIMURIGNI`

Hier benötigt die Maschine mindestens quadratische Laufzeit. Für das Entscheidungsproblem der Sprache L_{Pal} kann sogar gezeigt werden, daß jede 1-Band-Turing-Maschine mindestens quadratische Laufzeit benötigt (siehe [Reis 90]).

Palindrome sind, wie jeder weiß, in der Praxis selten, und quadratische Laufzeit ist für viele Eingaben gar nicht notwendig. Eine Eingabe folgender Gestalt

`TNHOLEBREMMINEDREWRESELEMASKREMFUA`

kann unsere Turing-Maschine mit einem Überstreifen des Kopfes über die Eingabe verwerfen, wie Abbildung 1.2 zeigt.

Abbildung 1.2: Für diese Eingabe benötigt dieselbe 1-Band-Turing-Maschine wie in Abbildung 1.1 lineare Zeit, um die Palindrom-Eigenschaft zu testen.

Wäre jedes Eingabewort der Länge n gleich wahrscheinlich, so hätte man bei $1/26$ aller Eingabemöglichkeiten Laufzeit n und bei $\frac{25}{26} \cdot \frac{1}{26}$ die Laufzeit $\leq 3 \cdot n$. Mit Wahrscheinlichkeit $(\frac{25}{26})^k \cdot \frac{1}{26}$ ergibt sich die Laufzeit $\leq (2k + 1) \cdot n$. Man kann sich leicht davon überzeugen, daß die erwartete Laufzeit linear ist.

Man kann aber auch eine Wahrscheinlichkeitsverteilung angeben, die jedem Algorithmus im Durchschnitt das schlechtestmögliche Zeitverhalten abverlangt. Hier kommen nur Palindrome vor, andere Zeichenketten haben Wahrscheinlichkeit 0. Jeder korrekte Algorithmus hätte dann mindestens quadratische Laufzeit.

Eine Verteilung, die die *worst-case*-Laufzeit bis auf einen konstanten Faktor auch zur *average*-Laufzeit macht, nennen wir **bösartig**. Sicher macht eine *average-case*-Betrachtung keinen Sinn, wenn man es mit solchen Verteilungen zu tun hat.

Meist gelingt es nicht vollständig, die *average*-Laufzeit zur *worst-case*-Laufzeit zu machen. Wir werden diesen Begriff nicht formal exakt fassen und sprechen auch dann von bösartig, wenn die *average*-Laufzeit der *worst-case*-Laufzeit nur annähernd entspricht.

In obigem Beispiel hat jemand, der die Verteilung festlegt, anscheinend Kenntnis, ob ein Wort ein Palindrom ist oder nicht. Ein Algorithmus, der diese Entscheidung zu treffen hat, und somit auch der Algorithmus, der die bösartige Verteilung generiert, benötigen wiederum quadratische Zeit, wenn man 1-Band-Turing-Maschinen benutzt.

Ist dies notwendig? Welche Komplexität hat im allgmeinen eine bösartige Verteilung? Diese und verwandte Fragen werden wir im Verlauf dieser Arbeit untersuchen. Dabei werden wir gleich zu Beginn feststellen, daß die Definition eines geeigneten Komplexitätsmaßes für den Durchschnitt einiger Anstrengungen bedarf.

Beispiel:
Betrachten wir hierzu den Erwartungswert. Angenommen wir hätten für ein Problem nachgewiesen, daß es eine 2-Band-Turing-Maschine gibt, die dieses Problem in erwartet linearer Zeit löst. Nun wird aber zusätzlich nach einer Lösung mittels 1-Band-Turing-Maschinen gefragt. Wir erinnern uns daran, daß eine Simulation der Berechnung einer 2-Band-Maschine, die Zeit $T(n)$ benötigt, in Zeit $T(n)^2$ von einer 1-Band-Turing-Maschine simuliert werden kann, wenn für alle Eingabelängen n gilt $T(n) \geq n$.

Die Vermutung, dieses Problem könnte eine 1-Band-Turing-Maschine in erwartet quadratischer Zeit lösen, wäre etwas voreilig.

Nehmen wir weiter an, daß mit Wahrscheinlichkeit $2^{-n/2}$ die 2-Band-Maschine Zeit $2^{n/2}$ benötigt und ansonsten linear beschränkt ist, wenn n die Länge einer Eingabe ist. Für die 2-Band-Maschine ergibt das eine erwartete Zeit von

$$2^{n/2} \cdot 2^{-n/2} + n \cdot \left(1 - 2^{-n/2}\right) \;\leq\; 1 + n \ .$$

Für die 1-Band-Maschine wird aus der linearen Laufzeit quadratische und aus $2^{n/2}$ wird 2^n . Ihr Erwartungswert ist damit exponentiell:

$$2^{n} \cdot 2^{-n/2} + n^2 \cdot \left(1 - 2^{-n/2}\right) \;\geq\; 2^{n/2} \ .$$

Leonid Levin [Levin 86] untersuchte aus diesem Grund ein mit dem Erwartungswert verwandtes Durchschnittsmaß, das die Abgeschlossenheit unter Polynomial-Zeit-Simulationen gewährleistet. Wir folgen dieser Vorgehensweise, die ausführlich in [John 84a, Gure 91a] dargestellt ist.

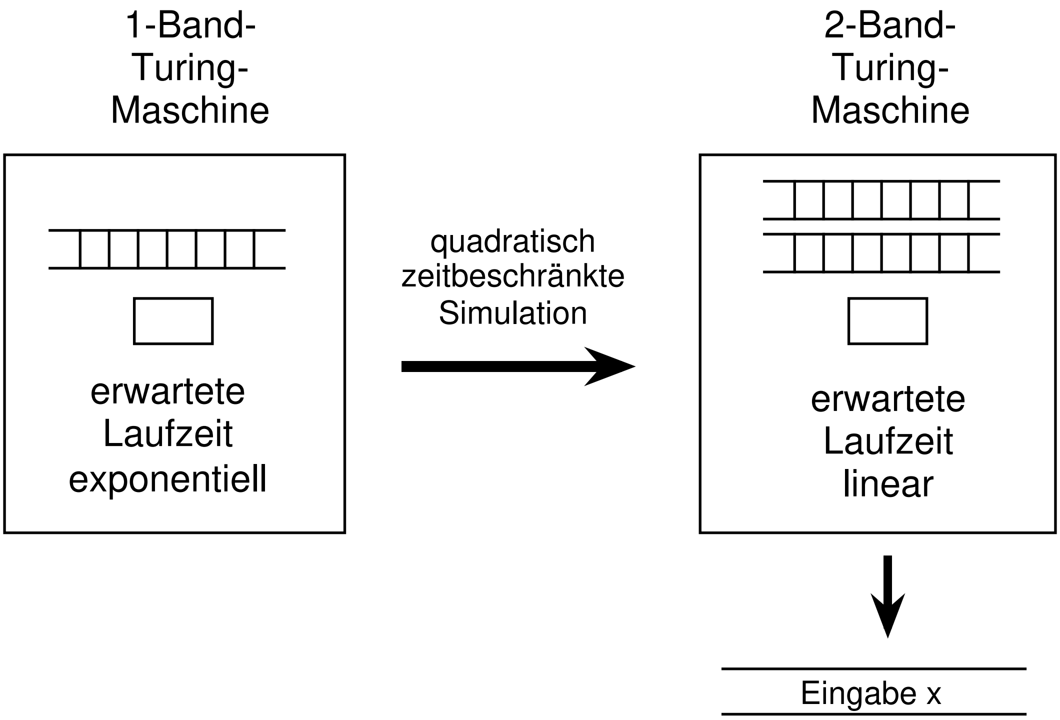

Abbildung 1.3: Im Erwartungswert erhalten Polynomial-Zeit-Simulationen keine polynomiellen Zeitschranken, hier dargestellt am Beispiel der Simulation einer 2-Band-Turing-Maschine durch eine 1-Band-Turing-Maschine.

Der klassische Ansatz der Durchschnittskomplexität einer Maschine M ist der Erwartungswert bezüglich einer **lokalen** (= finiten) Wahrscheinlichkeitsverteilung μ_n für Eingaben x der Länge n:

$$\boldsymbol{E_{\mu_n}(f)} := \sum_{|x|=n} \mu_n(x) \cdot f(x) \ .$$

In dieser Arbeit setzen wir für $f(x)$ die Laufzeit $\text{time}_M(x)$ einer Turing-Maschine M auf Eingabe x. Wir betrachten die Folge $\mu_0, \mu_1, \mu_2, \ldots$ von Wahrscheinlichkeitsverteilungen, wobei $\mu_i : \Sigma^i \to [0;1]$ eine Verteilung über Zeichenketten der Länge i ist. Die Maschine M ist nun **erwartet $\boldsymbol{T}$-zeitbeschränkt** für eine Schranke $T : \mathbb{N} \to \mathbb{N}$, falls $\boldsymbol{\forall n \ E_{\mu_n}(f) \leq T}$, was eine andere Schreibweise ist für

$$\forall n \quad \sum_{|x|=n} \mu_n(x) \cdot \frac{f(x)}{T(|x|)} \ \leq \ 1 \ .$$

Wir kennen jetzt bereits ein Beispiel (siehe Abbildung 1.3) dafür, daß aus einer linearen Beschränkung von f im Erwartungswert nicht folgt, daß f^2 im Erwartungswert polynomiell beschränkt ist. Somit sind Techniken wie polynomial-zeitbeschränkte Simulationen anderer Maschinenmodelle nicht mehr anwendbar.

Auf solche Techniken will man jedoch nicht verzichten und so erscheint es annehmbar, den Erwartungswert samt seiner einfache Interpretation fallen zu lassen, um die Abgeschlossenheitseigenschaft bezüglich solcher Transformationen wieder zu erreichen.

Wendet man die inverse Funktion von T auf Zähler und Nenner des Bruchs $\frac{f(x)}{T(|x|)}$ an, erhält man so

$$\forall n > 0 \quad \sum_{|x|=n} \mu_n(x) \cdot \frac{T^{-1}(f(x))}{|x|} \;\leq\; 1 \;.$$

Natürlich ist dieses Maß nicht der Erwartungswert, es besitzt aber die gewünschten Abgeschlossenheitseigenschaften.

Beispiel:
Überprüfen wir das am zuvor gezeigten Beispiel. Für die 2-Band-Maschine ergibt sich wiederum lineare Komplexität. Eine 1-Band-Maschine, die diese simuliert, bräuchte nun nur noch quadratische Komplexität, da gilt

$$2^{-n/2} \cdot \frac{\sqrt{2^n}}{n} \;+\; \left(1 - 2^{-n/2}\right) \frac{\sqrt{n}}{n} \quad\leq\quad 1 \;.$$

Die Wahrscheinlichkeit einer Zeichenkette x ist bezüglich einer lokalen Verteilung nur im Rahmen des zugehörigen Zufallsexperiments μ_n der entsprechenden Länge $n = |x|$ interpretierbar. Für eine andere Wortlänge findet ein unabhängiges Zufallsexperiment statt. Zeichenketten verschiedener Längen sind kaum vergleichbar.

Eine elegante Alternative hierzu ist eine **globale** Wahrscheinlichkeitsverteilung. Hier steht der über alle Eingabelängen definierten Sprache eine wiederum über alle Eingabelängen definierte Wahrscheinlichkeitsverteilung gegenüber.

Die lokalen Wahrscheinlichkeitsverteilungen $[\mu]_n$ sind jetzt als bedingte Wahrscheinlichkeiten $[\mu]_n(x) := \mathrm{Prob}_\mu[x \mid |x| = n]$ beschrieben. Im Gegensatz zu lokalen Wahrscheinlichkeitsverteilungen können wir jetzt über verschiedene Eingabelängen mitteln.

Levin betrachtete also Verteilungen μ über die gesamte Menge aller Eingaben und verlangte, daß gilt

$$\sum_x \mu(x) \cdot \frac{T^{-1}(f(x))}{|x|} \;\leq\; 1 \;.$$

Wenn dies gilt, nennen wir (f, μ) **Levin-*average*-*T*-zeitbeschränkt** (**Lav**). Dabei setzte er voraus, daß x nicht die leere Zeichenkette λ ($|\lambda| = 0$) annimmt. Wir folgen dieser Betrachtungsweise und betrachten nur Wahrscheinlichkeitsverteilungen μ, welche die leere Zeichenkette λ nicht gewichten ($\mu(\lambda) = 0$). Da damit nur eine einzige Eingabezeichenkette ausgeschlossen wird, stellt dies keine Einschränkung hinsichtlich später betrachteter Komplexitätsklassen dar.

Definition 1

$$(f,\mu) \in \mathbf{Lav(T)} \quad :\Longleftrightarrow \quad E_{\mu(x)}\left(\frac{T^{-1}(f(x))}{|x|}\right) \;\leq\; 1\ .$$

Dieser Ansatz wurde intensiv untersucht in [BCGL 89, Gure 91b].

An dieser Stelle wollen wir den Faden aufgreifen und ein neues Element in die Betrachtungen einbringen: die **Präzision**.

Definition 2 *Ein Maß $M(T)$ bezüglich einer Komplexitätsschranke T ist* ***präzise****, falls für alle Wahrscheinlichkeitsverteilungen μ und für alle Funktionen $f : \Sigma^* \to \mathbb{N}$ mit $f(x) \leq T(|x|)$ gilt*

$$(f,\mu) \in M(T)$$

und andererseits für alle Funktionen $g : \Sigma^* \to \mathbb{N}$ mit

$$\forall k \in \mathbb{N} \quad g(x) \geq_{ae} k \cdot T(|x|), \qquad d.h. \quad g \in \omega(T)$$

aber gilt

$$(g,\mu) \notin M(T)\ .$$

Eine andere Definition könnte sein: Ein Maß ist präzise, wenn eine Zeitfunktion f, die für alle Eingaben genau T ist, nicht einer asymptotisch kleineren Schranke $T' \in o(T)$ (d.h. $\forall k \in \mathbb{N}^+ \quad k \cdot f \leq_{ae} g$) zugeordnet werden kann.

Beispiel:
Nehmen wir eine **uniforme** Wahrscheinlichkeitsverteilung μ_{uni}, die den binären Zeichenketten $x \in \{0,1\}^* \setminus \{\lambda\}$ die Wahrscheinlichkeiten

$$\boldsymbol{\mu_{\mathbf{uni}}(x)} \quad := \quad 6/\pi^2 \cdot |x|^{-2} \cdot 2^{-|x|}$$

zuweist. Eine Turing-Maschine, die genau n^2 Schritte auf jeder Eingabe der Länge n benötigt, würde nun $\text{Lav}\,(O(\mathcal{N}^{1+\epsilon}))$ -zeitbeschränkt sein für beliebiges $\epsilon > 0$ ($\mathcal{N}$ steht hier für die identische Funktion in $\mathbb{N}$).

$$\sum_x \frac{6 \cdot |x|^{\frac{2}{1+\epsilon}}}{\pi^2 \cdot |x|^2 \cdot 2^{|x|}} \;\leq\; k\ ,$$

für eine geeignet gewählte Konstante k. Tabelle 2.1 zeigt weitere Beispiele dieses Phänomens.

Der Grund für den Verlust der Präzision ist der Einfluß des funktionalen Wachstums von $\mu(x)$ auf die Zeitschranke T. Es kann durch immer bessere Wahl der Verteilungen vermindert werden; gelöst wird es auf diese Weise nie (siehe Lemma 2).

Wir werden im nächsten Kapitel präzise und aussagefähige Maße definieren.

1.3 Notationen

Zuerst legen wir einige grundlegende Notationen fest, wobei wir uns im wesentlichen an der in [Reis 90] orientieren. Diese Zusammenstellung umfaßt alle zum Verständnis notwendigen Begriffe.

1.3.1 Allgemeines

Die Menge der natürlichen Zahlen $0, 1, 2, \ldots$ bezeichnen wir mit $\mathbb{N}$. $\mathbb{N}^+ := \mathbb{N} \setminus \{0\}$ sei die Menge aller positiven Zahlen, $\mathbb{Z}$ die Menge aller ganzen Zahlen. $\mathbb{R}$ steht für die Menge der reellen Zahlen. Die Menge der Booleschen Zahlen besteht ausschließlich aus dem Symbol **0** für falsch und **1** für wahr. Funktionen mit Booleschem Wertebereich werden **Prädikate** genannt.

Die Potenzmenge $\mathbf{Pot(A)}$ einer Menge A ist die Menge aller Teilmengen von A. Wenn Menge A eine Teilmenge von Menge B ist, schreiben wir $A \subseteq B$ oder auch $B \supseteq A$. Falls A eine echte Teilmenge von B ist (also A ist Teilmenge von B und beide Mengen sind ungleich), schreiben wir $A \subset B$ oder $B \supset A$. Wir benutzen die Intervall-Schreibweise $\boldsymbol{[a;b]}$ für die Menge $\{x \mid a \leq x \leq b\}$, $\boldsymbol{(a;b)} := \{x \mid a < x < b\}$ für das offene Intervall und $\boldsymbol{[a;b)} := \{x \mid a \leq x < b\}$, $\boldsymbol{(a;b]} := \{x \mid a < x \leq b\}$ für die halboffenen Intervalle.

In Zusammenhang mit natürlichen Zahlen werden wir das Symbol ∞ benutzen. Es steht für eine Zahl, die größer ist als jede natürliche Zahl. Andere Operationen als Vergleiche mit natürlichen Zahlen werden nicht mit dieser Zahl dargestellt.

Für eine Funktion $f : \mathcal{A} \to \mathcal{B}$ und ein Prädikat $P : \mathcal{A} \to \{0,1\}$ bezeichne

$$\boldsymbol{\{f(x) \mid P(x)\}} \quad := \quad \{y \mid \exists x : \ f(x) = y \text{ und } P(x)\} \ .$$

Genauso werden wir in einem Summen-Ausdruck folgende Vereinfachung benutzen, wenn klar ist, welche Variable (in der Regel x) an das Summen-Konstrukt gebunden ist.

$$\sum_{P(x)} t(x) \quad := \quad \sum_{x \in \{y \mid P(y)\}} t(x) \ .$$

Die Definitionsmenge $\mathcal{A}$ einer Funktion $f : \mathcal{A} \to \mathcal{B}$ nennen wir $\mathbf{dom(f)} := \mathcal{A}$ (*domain*). Für undefinierte Ausdrücke $f(c)$, also wenn $c \notin \mathrm{dom}(f)$ gilt, verwenden wir folgende Schreibweise: $\boldsymbol{f(c) = \perp}$.

Für eine Funktion $f : \mathcal{A} \to \mathcal{C}$ und eine Menge $\mathcal{B} \subseteq \mathcal{A}$ definieren wir $\boldsymbol{f(\mathcal{B})}$ als

$$\boldsymbol{f(\mathcal{B})} \quad := \quad \{y \mid \exists x \in \mathcal{B} : \ f(x) = y\} \ .$$

1.3.2 Funktionen und Wachstumsklassen

Wird die Funktion $f : \mathcal{A} \to \mathcal{B}$ auf die Funktion $g : \mathcal{B} \to \mathcal{C}$ angewendet, so bezeichne $f \circ g : \mathcal{A} \to \mathcal{C}$ die resultierende Funktion $\boldsymbol{f \circ g(n)} \;:=\; f(g(n))$.

Die Maximumsfunktion $\mathbf{max} : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ oder $\mathbf{max} : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ bildet (x, y) auf den größeren der beiden Werte ab. Das Maximum einer endlichen Menge $\mathcal{A}$ werde ebenfalls damit ausgedrückt: $\mathbf{max}(\mathcal{A}) \;:=\; x$ genau dann, wenn $x \in \mathcal{A}$ und $\forall y \in \mathcal{A} \quad x \geq y$. Für Funktionen f, g bezeichnet $\mathbf{max}[\boldsymbol{f,g}]$ die Funktion $n \mapsto \max(f(n), g(n))$.

Entsprechende notationelle Vereinbarungen treffen wir für die Minimumsfunktion **min**.

Wird aus dem Kontext klar, daß das Ergebnis einer Operation eine natürliche Zahl sein soll und steht an dieser Stelle eine positive reelle Zahl r, so wurde die untere Gauß-Klammer-Funktion $\lfloor \boldsymbol{r} \rfloor$ der Übersicht halber weggelassen. Sie bezeichnet den größten ganzzahligen Wert, der nicht größer als r ist. Die obere Gauß-Klammer-Funktion $\lceil \boldsymbol{r} \rceil$ bezeichnet dagegen den kleinsten ganzzahligen Wert, der nicht kleiner als r ist.

Eine Funktion $f : A \to B$ ist **monoton zunehmend**, gdw. $x > y \implies f(x) \geq f(y)$. Eine zweistellige Funktion $f : A \times B \to C$ wird **monoton zunehmend** genannt, wenn für alle Konstanten $k_1 \in A$ und $k_2 \in B$ die Funktionen $f_1 : x \mapsto f(k_1, x)$ und $f_2 : x \mapsto f(x, k_2)$ monoton zunehmend sind. Analog vereinbaren wir **monoton abnehmende** Funktionen.

Komplexitätsschranken sind monoton zunehmende Abbildungen $T : \mathbb{N} \to \mathbb{N}$. **Zeitschranken** T als Komplexitätsschranken sind größer als die **identische Funktion** $\boldsymbol{\mathcal{N}}$ mit der Zuordnung $n \mapsto n$. Wie bereits oben angesprochen, bezeichnet deshalb die Funktion $T(\alpha\mathcal{N})$ für eine Konstante $\alpha > 0$ die Funktion $n \mapsto T(\lfloor \alpha \cdot n \rfloor)$.

Um verschiedene Funktionen (zumindest teilweise) ordnen zu können, benutzen wir die **O-Notation** mittels der Terme $\boldsymbol{O}$, $\boldsymbol{o}$, $\boldsymbol{\Omega}$, $\boldsymbol{\omega}$ und $\boldsymbol{\Theta}$, siehe Tabelle 1.1.

Die Exponentialfunktion **exp** ist hier definiert als $\mathbf{exp\ n} := 2^n$. $\log_2 : \mathbb{R}^+ \to \mathbb{R}$ ist der Logarithmus zur Basis 2. Wir benötigen in dieser Arbeit das in der Informatik übliche Äquivalent $\mathbf{log} : \mathbb{N} \to \mathbb{N}$, definiert als

$$\mathbf{log(n)} \quad := \quad \begin{cases} 0, & n = 0 \\ \lfloor \log_2(n) \rfloor + 1, & n > 0\ . \end{cases}$$

Tabelle 1.2 zeigt Funktionen-Mengen von besonderem Interesse.

Für eine Funktion f, die nicht notwendigerweise injektiv ist, definieren wir die **Umkehrfunktion** $\boldsymbol{f^{[-1]}}$ durch

$$\boldsymbol{f^{[-1]}(n)} \quad := \quad \min\{m \mid f(m) \geq n\}\ .$$

Für Funktionen $f, g : \mathbb{N} \to \mathbb{N}$ und eine Menge $\mathcal{G}$ von Funktionen werde definiert:

$$\boldsymbol{f} \ \leq_{ae} \ \boldsymbol{g} \quad :\Longleftrightarrow \quad \exists n_0 \in \mathbb{N} \quad \forall n \geq n_0 : \quad f(n) \leq g(n) \ .$$

$$\begin{aligned}
\boldsymbol{O(g)} &:= \{f \mid \exists k \in \mathbb{N} \quad f \leq_{ae} k \cdot g\} \ , \\
\boldsymbol{o(g)} &:= \{f \mid \forall k \in \mathbb{N}^+ \quad k \cdot f \leq_{ae} g\} \ , \\
\boldsymbol{\omega(g)} &:= \{f \mid \forall k \in \mathbb{N}^+ \quad k \cdot g \leq_{ae} f\} \ , \\
\boldsymbol{\Omega(g)} &:= \{f \mid \exists k \in \mathbb{N} \quad g \leq_{ae} k \cdot f\} \ , \\
\boldsymbol{\Theta(g)} &:= O(g) \quad \cap \quad \Omega(g) \ .
\end{aligned}$$

$$\begin{aligned}
\boldsymbol{O(\mathcal{G})} &:= \bigcup_{g \in \mathcal{G}} O(g) \quad , \quad \boldsymbol{o(\mathcal{G})} := \bigcup_{g \in \mathcal{G}} o(g) \ , \\
\boldsymbol{\Omega(\mathcal{G})} &:= \bigcup_{g \in \mathcal{G}} \Omega(g) \quad , \quad \boldsymbol{\omega(\mathcal{G})} := \bigcup_{g \in \mathcal{G}} \omega(g) \ , \\
\boldsymbol{\Theta(\mathcal{G})} &:= \bigcup_{g \in \mathcal{G}} \Theta(g) \quad .
\end{aligned}$$

Für eine Menge $\mathcal{G}$ von Komplexitätsschranken bedeutet $\boldsymbol{f \leq \mathcal{G}}$, daß es eine Funktion $g \in \mathcal{G}$ gibt mit $f \leq g$.

Tabelle 1.1: Die O-Notation zum Vergleich asymptotisch verschieden wachsender natürlicher Funktionen.

$$\begin{aligned}
\mathbf{Lin}(T) &:= \Theta(T) \ , & \mathbf{LIN} &:= \Theta(\mathcal{N}) \ , \\
\mathbf{Pol}(T) &:= \bigcup_k O(T^k) \ , & \mathbf{POL} &:= \mathrm{Pol}(\mathcal{N}) \ , \\
\mathbf{Log}(T) &:= \Theta(\log T) \ , & \mathbf{LOG} &:= \mathrm{Log}(\mathcal{N}) \ , \\
\mathbf{PLog}(T) &:= \mathrm{Pol}(\mathrm{Log}(T)) \ , & \mathbf{PLOG} &:= \mathrm{PLog}(\mathcal{N}) \ , \\
\mathbf{LLog}(T) &:= \Theta(\log\log T) \ , & \mathbf{LLOG} &:= \mathrm{LLog}(\mathcal{N}) \ , \\
\mathbf{ExL}(T) &:= \exp(\Theta(T)) \ , & \mathbf{EXL} &:= \mathrm{ExL}(\mathcal{N}) \ , \\
\mathbf{EExL}(T) &:= \exp^{[2]}(\Theta(\mathcal{N})) \ , & \mathbf{EEXP} &:= \exp^{[2]} \ .
\end{aligned}$$

Tabelle 1.2: Wichtige Klassen von Funktionen.

$$\begin{aligned} T(T^{[-1]}(n)) &\geq n \ , \\ T^{[-1]}(T(n)) &\leq n \ , \\ T^{[-1]}(T(n)+1) &> n \ . \end{aligned}$$

Falls T_1, T_2 monoton zunehmend sind und $c \in \mathbb{R}^+$, gilt

$$\begin{aligned} (T_1 \circ T_2)^{[-1]} &= T_2^{[-1]} \circ T_1^{[-1]} \ , \\ \lceil c \cdot T \rceil^{[-1]} &= \left\lceil \frac{T^{[-1]}}{c} \right\rceil \ , \\ (T+c)^{[-1]} &= \max\left[0; T^{[-1]} - c\right] \ . \end{aligned}$$

Falls T streng monoton zunehmend ist, gilt

$$T^{[-1]}(T(n)) = n \ .$$

Tabelle 1.3: Nützliche Eigenschaften von $T^{[-1]}$.

Tabelle 1.3 zeigt elementare Eigenschaften dieser für die vorliegende Arbeit grundlegenden Definition. In Anlehnung an diese Notation bezeichne $\boldsymbol{f^{[n]}}$ die n-fache Iteration der Funktion f oder $f^{[-1]}$, je nach Vorzeichen von n.

$$\begin{aligned} \boldsymbol{f^{[0]}} &:= \mathcal{N} \ , \\ \boldsymbol{f^{[n]}} &:= f \circ f^{[n-1]} \ , \qquad \text{falls } n > 0, \\ \boldsymbol{f^{[n]}} &:= f^{[-1]} \circ f^{[n+1]} \ , \quad \text{falls } n < 0. \end{aligned}$$

Die iterierte Exponentialfunktion **itexp** und die zugehörige iterierte Logarithmusfunktion **itlog** werden mittels dieser Notation definiert als

$$\begin{aligned} \mathbf{itexp(n)} &:= \exp^{[n]}(1) \ , \\ \mathbf{itlog(n)} &:= \mathrm{itexp}^{[-1]}(n) \ . \end{aligned}$$

1.3.3 Zeichenketten

Die endliche Menge $\boldsymbol{\Sigma}$ besteht aus mindestens zwei Zeichen. Im allgemeinen sind dies beliebige, allerdings werden diese häufig auf die **binären Zeichen** „0“ und „1“ eingeschränkt. Die Menge aller Zeichenketten einschließlich der leeren Zeichenkette $\boldsymbol{\lambda}$ sei $\boldsymbol{\Sigma^*}$. $\boldsymbol{\Sigma^n}$ ist die Menge aller Zeichenketten der Länge n, für eine Zeichenkette x bezeichnet $\boldsymbol{|x|}$ die Länge. Eine Teilmenge $L \subseteq \Sigma^*$ wird **Sprache** genannt. Eine

endliche Sprache ist eine endliche Teilmenge von Σ^*. **Komplexitätsklassen** sind Mengen von Sprachen, die durch ein Komplexitätskriterium beschrieben werden.

Die charakteristische Funktion $\boldsymbol{\chi_L}$ einer Sprache ist definiert durch

$$\boldsymbol{\chi_L(x)} \quad := \quad \begin{cases} 1, & x \in L\ , \\ 0, & x \notin L\ . \end{cases}$$

bin: $\mathbb{N} \to \{0,1\}^*$ bezeichnet die Standardumwandlung von natürlichen Zahlen in binäre Zeichenketten (z.B. bin(215) = „11010111“). Die Ordnungsfunktion **ord** : $\Sigma^* \to \mathbb{N}$ gibt die lexikographische Ordnungszahl der binären Zeichenkette an. Sie definiert sich zum Beispiel für binäre Zeichenketten rekursiv aus

$$\begin{aligned} \mathbf{ord}(\boldsymbol{\lambda}) &\quad := \quad 1\ , \\ \mathbf{ord}(\boldsymbol{x}\mathbf{0}) &\quad := \quad 2 \cdot \mathrm{ord}(x)\ , \\ \mathbf{ord}(\boldsymbol{x}\mathbf{1}) &\quad := \quad 2 \cdot \mathrm{ord}(x) + 1\ . \end{aligned}$$

Somit ist $\mathrm{ord}(\mathrm{bin}(n)) = 2 \cdot n + 1$.

Die **lexikographische Ordnung** $\boldsymbol{x \leq y}$ der Zeichenketten ergibt sich gemäß der Ordnungszahl:

$$\boldsymbol{x \leq y} \quad :\Longleftrightarrow \quad \mathrm{ord}(x) \leq \mathrm{ord}(y)\ .$$

Der Vorgänger einer Zeichenkette $x \in \Sigma^*$ gemäß dieser Ordnung werde mit $\boldsymbol{x - 1}$ bezeichnet; der Nachfolger mit $\boldsymbol{x + 1}$. $\boldsymbol{\Sigma^{\leq x}}$ beschreibt für eine Zeichenkette x die Menge aller Zeichen, die lexikographisch nicht größer als x sind. Entsprechend steht $\boldsymbol{L^{\leq x}}$ für eine Sprache L

$$\boldsymbol{L^{\leq x}} \quad := \quad L\ \cap\ \Sigma^{\leq x}\ .$$

Dagegen steht die Notation $\boldsymbol{\Sigma^{\leq n}}$ für natürliche Zahlen n für folgendes:

$$\begin{aligned} \boldsymbol{\Sigma^{\leq n}} &\quad := \quad \{x \in \Sigma^* \mid |x| \leq n\}\ , \\ \boldsymbol{\Sigma^{\geq n}} &\quad := \quad \{x \in \Sigma^* \mid |x| \geq n\}\ , \\ \boldsymbol{L^{= n}} &\quad := \quad L\ \cap\ \Sigma^n\ , \\ \boldsymbol{L^{\leq n}} &\quad := \quad L\ \cap\ \Sigma^{\leq n}\ . \end{aligned}$$

1.3.4 Turing-Maschinen

Algorithmen werden hier mittels **Turing-Maschinen** modelliert. Grundsätzlich betrachten wir **deterministische Turing-Maschinen** (**DTM**). Die Ausgabe-Funktionen werden aufgezählt durch $\boldsymbol{\Phi_i^k}$, wobei k die Anzahl der Argumente ist. Die k Eingabe-Zeichenketten einer Turingmaschine werden durch das Symbol # getrennt, das nicht zur Eingabe-Zeichenmenge Σ gezählt wird. Für die Aufzählung Φ_i^1 verwenden wir auch noch die Schreibweise $\boldsymbol{M_i}$. Für eine Turingmaschine M sei $M : \Sigma^* \to \Sigma^*$ die Ausgabe-Funktion. M produziert auf Eingabe x die Ausgabe $M(x)$ und hält. Falls M nicht hält, ist $M(x)$ nicht definiert. **dom(M)** bezeichnet somit die Menge aller Eingaben, auf denen die Ausgabe-Funktion definiert ist und für die M somit hält.

In einigen Fällen werden wir **nicht-deterministische Turing-Maschinen** (**NTM**) betrachten. Alle betrachteten Maschinen haben nur zwei Arbeitsbänder, was für geeignete Aufzählungen garantiert, daß eine **universelle Turing-Maschine** nur mit konstantem multiplikativem Zeitverlust existiert. Genauer gesagt setzen wir voraus, daß eine Maschine U jede Maschine M_i auf Eingabe $x\#\text{bin}(i)$ in höchstens $O(i \cdot \text{time}_{M_i}(x))$ Schritten simulieren kann.

Dabei beschreibt $\boldsymbol{\text{time}_M(x)}$ die Anzahl der Zustandsübergänge von M auf Eingabe x, bis M in einen Endzustand gerät, falls M eine deterministische Turing-Maschine ist. Handelt es sich bei M um eine nicht-deterministische Turingmaschine, bezeichnet $\boldsymbol{\text{time}_M(x)}$ die Anzahl der Zustandsübergänge von M auf Eingabe x auf dem kürzesten akzeptierenden Berechnungspfad. Falls kein solcher existiert und daher M verwirft, bezeichnet $\text{time}_M(x)$ die Länge des längsten Berechnungspfads.

Für deterministische Maschinen werden wir noch eine andere Zeitfunktion definieren (siehe Seite 94).

Handelt es sich bei der von M berechneten Funktion um einen **Akzeptor**, das heißt M berechnet oder entscheidet eine Sprache L, so entspricht die Ausgabe-Funktion von M der charakteristischen Funktion von L

$$M(x) \quad = \quad \chi_L(x) \; .$$

Obwohl wir Algorithmen mit Turingmaschinen modellieren, werden wir in Beweisen, die die Beschreibung eines Algorithmus notwendig machen, diese durch eine prozedural orientierte Programmiersprache angeben.

Eine Funktion $f : \mathcal{N} \to \mathcal{N}$ ist in Zeit T **konstruierbar**, falls es eine deterministische Turingmaschine M gibt, die auf Eingabe 1^n die Ausgabe $\text{bin}(f(n))$ innerhalb der Zeitschranke $T(n)$ berechnet. Eine Funktion T ist **zeitkonstruierbar**, falls die Funktion T in Zeit T konstruiert werden kann. Alle betrachteten Zeitschranken in dieser Arbeit werden als zeitkonstruierbar vorausgesetzt.

1.3.5 Wahrscheinlichkeitstheorie

Hier werden ausschließlich **Wahrscheinlichkeitsfunktionen** $\mu : \mathcal{A} \to [0;1]$ über der Eingabe-Menge $\mathcal{A} \subseteq \Sigma^*$ betrachtet. $\boldsymbol{\mu}$ selber steht für die **Dichtefunktion**, was bedeutet $\sum_x \mu(x) = 1$. In dieser Arbeit betrachten wir aus Rücksicht auf das Levinsche Maß nur Wahrscheinlichkeitsverteilungen μ mit $\mu(\lambda) = 0$, wenn nicht anders vereinbart. Die Wahrscheinlichkeit $\mathbf{Prob}_{\boldsymbol{\mu}}[\boldsymbol{B}]$ eines Ereignisses $B \subseteq \mathrm{dom}(\mu)$ wird beschrieben durch

$$\mathbf{Prob}_{\boldsymbol{\mu}}[\boldsymbol{B}] \quad := \quad \sum_{x \in B} \mu(x) \; .$$

Die **Verteilungsfunktion** $\boldsymbol{\mu^*} : \mathcal{A} \to [0,1]$ definiert sich aus der Dichtefunktion als

$$\boldsymbol{\mu^*(x)} \quad := \quad \sum_{y \leq x} \mu(x) \; .$$

Für eine **globale Wahrscheinlichkeitsverteilung** $\mu : \Sigma^* \to [0;1]$ werde die **lokale Wahrscheinlichkeitsverteilung** $[\boldsymbol{\mu}]_{\boldsymbol{n}} : \Sigma^n \to [0;1]$ für jedes $n \in \mathbb{N}$ mit $\mathrm{Prob}_\mu(\Sigma^{=n}) > 0$ definiert als

$$[\boldsymbol{\mu}]_{\boldsymbol{n}}(\boldsymbol{x}) \quad := \quad \frac{\mu(x)}{\sum_{|x|=n} \mu(x)} \; .$$

Eine Wahrscheinlichkeitsverteilung ist **finit** oder **endlich**, wenn die Anzahl aller Ereignisse mit positiver Wahrscheinlichkeit endlich ist.

Der **Erwartungswert** $\boldsymbol{E_\mu(f)}$ einer Funktion $f : \Sigma^* \to \mathbb{R}$ bezüglich einer Wahrscheinlichkeitsverteilung μ ist definiert als

$$\boldsymbol{E_\mu(f)} \quad := \quad \sum_{x \in \mathrm{dom}(\mu)} f(x) \cdot \mu(x) \; .$$

Kapitel 2

Durchschnittsmaße

2.1 Das Av-Maß

Levins Maß kann die Problematik der Simulationen im *average-case* lösen, es kann aber nur zwischen polynomiellem und überpolynomiellem Zeitverhalten unterscheiden. Dieses in der Einleitung angesprochene Problem läßt sich zwar durch geschickte Wahl der Wahrscheinlichkeitsverteilung verkleinern indem man, wie in Tabelle 2.1 gezeigt, Σ^n anstelle eines Gesamtgewichts von n^{-2}, Gewichte proportional zu $n^{-1} \cdot \log^{-2} n$ oder noch besser proportional zu

$$n^{-1} \cdot \log^{-1} n \cdot \ldots \cdot (\log^{[k-1]} n)^{-1} \cdot (\log^{[k]} n)^{-2}$$

gibt ($\log^{[k]}$ bezeichnet den k-fach iterierten Logarithmus).

Es ergibt sich aber immer ein Eintrag des funktionalen Verhaltens der Wahrscheinlichkeitsverteilung in das Zeitverhalten, da sich folgendes zeigen läßt.

Lemma 1 *Für eine monoton abnehmende positive Funktion $u : \Sigma^* \to [0;1]$ und eine reellwertige Funktion $h : [0,1] \to [0,1]$ mit einer in $(0,1)$ monoton abnehmenden ersten Ableitung h', gilt für $u(x) \neq u(x-1)$*

$$h'(u(x-1)) \;\leq\; \frac{h(u(x)) - h(u(x-1))}{u(x) - u(x-1)} \;\leq\; h'(u(x)) \; .$$

Beweis: Nach dem Mittelwertsatz existiert ein $z_0 \in (0;1)$, so daß

$$h'(z_0) \;=\; \frac{h(u(x)) - h(u(x-1))}{u(x) - u(x-1)} \; .$$

Da h' monoton zunehmend ist, gilt $h'(u(x-1)) \leq h'(z_0) \leq h'(u(x))$. ∎

μ	T_1	T_2
$\frac{1}{2^{2\cdot\lvert x\rvert}}$	$O(1)$	$\mathrm{LOG}^2 \cdot \mathrm{LLOG}^3$
$\frac{c_1}{2^{\lvert x\rvert}\cdot\lvert x\rvert^2}$	$\mathcal{N}\cdot\mathrm{LOG}^2$	$\mathcal{N}\cdot\mathrm{LOG}^2$
$\frac{c_2}{2^{\lvert x\rvert}\cdot\lvert x\rvert\cdot\log^2\lvert x\rvert}$	$\frac{\mathcal{N}^2\cdot\mathrm{LLOG}^2}{\log\mathcal{N}}$	$\frac{\mathcal{N}^2\cdot\mathrm{LLOG}^3}{\log^2\mathcal{N}}$

Tabelle 2.1: Beispiele zur mangelnden Präzision herkömmlicher Maße. T_1 erfüllt Bedingung $E_\mu\left(\frac{\mathcal{N}^2}{T_1}\right) \leq 1$, während T_2 der Anforderung $(\mathcal{N}^2, \mu) \in \mathrm{Lav}\,(T_2)$ genügt. Man beachte, daß bei diesen Maßen immer eine Wahl der Schranken möglich ist, so daß $T_1, T_2 \in o(\mathcal{N}^2)$ gilt.

Lemma 2 *Für jede monoton abnehmende positive Wahrscheinlichkeitsverteilung* $\mu : \Sigma^* \to [0,1]$ *gibt es eine Verteilung* $\tilde{\mu}$ *mit*

$$\lim_{x\to\infty} \frac{\tilde{\mu}(x)}{\mu(x)} = \infty .$$

Beweis: $\mu^*(x) := \sum_{z\leq x} \mu(z)$ sei die Verteilungsfunktion von μ bezüglich der lexikographischen Ordnung der Zeichenketten Σ^*. Wir definieren eine neue Wahrscheinlichkeitsfunktion $\tilde{\mu}$, wie in Abbildung 2.1 gezeigt, durch die Verteilungsfunktion

$$\tilde{\mu}^*(x) := 1 - \sqrt{1-\mu^*(x)} .$$

Daraus ergibt sich die Dichtefunktion durch $\tilde{\mu}(x) := \tilde{\mu}^*(x) - \tilde{\mu}^*(x-1)$. Nach Lemma 1 erhalten wir für $h(x) := 1-\sqrt{1-x}$ und $u(x) := \mu^*(x)$ die Ungleichung

$$\frac{\tilde{\mu}(x)}{\mu(x)} \geq \frac{\mu^*(x-1)}{\sqrt{1-\mu^*(x-1)}} .$$

Dabei strebt $\frac{\mu^*(x)}{\sqrt{1-\mu^*(x)}}$ ins Unendliche. ∎

Levins *average-case*-Maß kann also für keine noch so gut gewählte Wahrscheinlichkeitsverteilung μ präzise sein. Man kann zu μ nämlich immer eine noch langsamer gegen 0 konvergierende Wahrscheinlichkeitsverteilung $\tilde{\mu}$ definieren. Der Quotient $\mu/\tilde{\mu}$ kann nun die Abweichung von der Präzision ermöglichen. Dies war wohl der Grund, daß Levin nur die Oberklasse ***polynomial on the average*** definierte, welche für moderat abnehmende Wahrscheinlichkeitsverteilungen nur zwischen polynomiell und überpolynomiell sicher unterscheiden konnte.

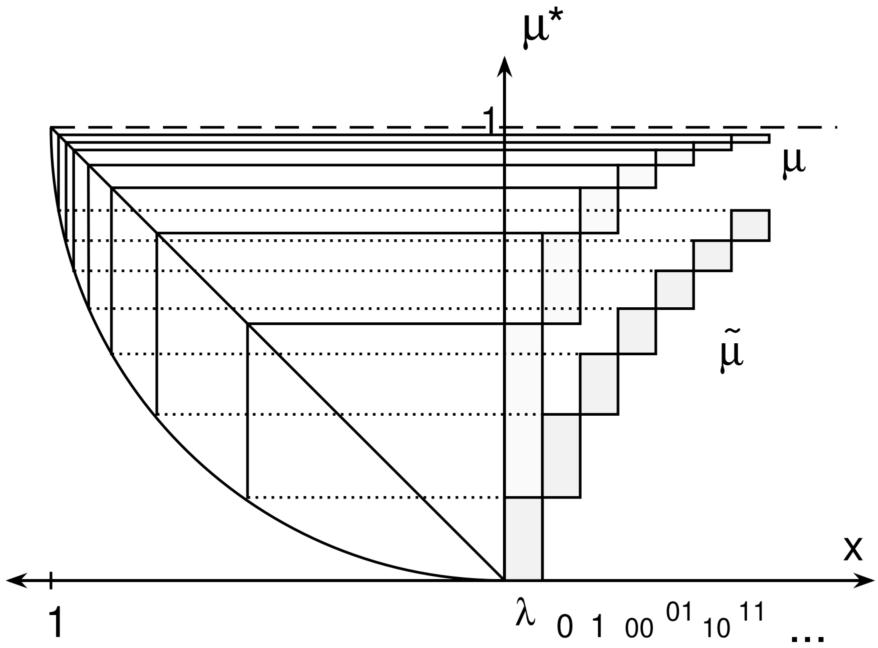

Abbildung 2.1: Die Dichtefunktion $\tilde{\mu}$, dargestellt durch die Flächen der dunkler schraffierten Rechtecke, strebt asymptotisch langsamer gegen 0 als μ.

Definition 3 *Eine Funktion* $f : \Sigma^* \to \mathbb{N}$ *gehört zur Klasse* ***Lav(POL) (Levin average polynomial)*** *bezüglich der Wahrscheinlichkeitsverteilung* μ *gdw. für eine Zahl* k *gilt*

$$E_\mu \left(\frac{f(x)^{1/k}}{|x|} \right) < \infty \; .$$

Wir stellen nun ein strengeres Durchschnittsmaß vor. Die Idee ist, alle Verteilungen $\tilde{\mu}$ gleichzeitig zu betrachten, die, ordnet man Zeichenketten gemäß ihrer Wahrscheinlichkeit, sich in der Rangfolge der Zeichenketten nicht unterscheiden; zum Beispiel wenn $\mu(10001) < \mu(11)$ gilt, dann muß gelten $\tilde{\mu}(10001) \leq \tilde{\mu}(11)$. Wir sprechen in diesem Fall von einer ***Average*-Beschränkung** durch T, falls die Funktion für alle möglichen Verteilungen dieser Art Levin-*average*-beschränkt ist.

Dann ist nur die Reihenfolge der Eingaben hinsichtlich abnehmender Wahrscheinlichkeiten von Bedeutung. Deswegen definieren wir für eine Verteilung μ ihre korrespondierende Rangfunktion $\mathbf{rank}_\mu$, wie in Abbildung 2.2 gezeigt, durch

Definition 4

$$\mathbf{rank}_\mu(\mathbf{x}) \;:=\; \begin{cases} |\{z \in \Sigma^* \mid \mu(z) \geq \mu(x)\}| \, , & \textit{falls } \mu(x) > 0 \, , \\ \infty \, , & \textit{falls } \mu(x) = 0 \, . \end{cases}$$

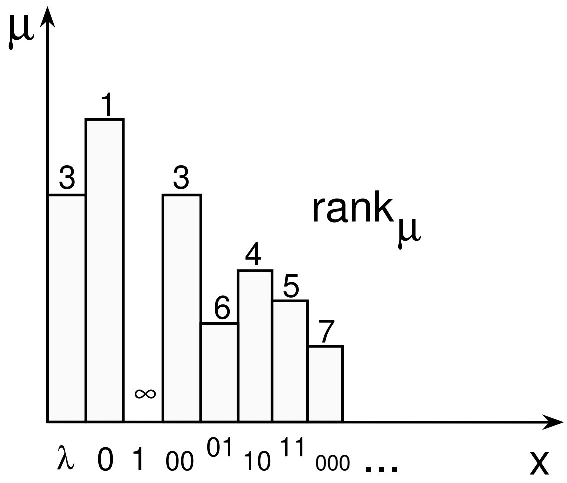

Abbildung 2.2: Der Rang $\text{rank}_\mu(x)$ einer Zeichenkette x in einer Wahrscheinlichkeitsverteilung μ ist die Anzahl aller Zeichenketten, die mindestens ebenso wahrscheinlich sind.

Hierbei spielt ∞ *die Rolle einer Zahl, die größer ist als jede natürliche Zahl. Eine Turing-Maschine, die die Rangfunktion berechnet, gibt ein spezielles Symbol für diesen Fall aus.*

Für eine Klasse spezieller Wahrscheinlichkeitsverteilungen ergeben sich so sehr einfache Rangfunktionen (siehe Abbildung 2.3).

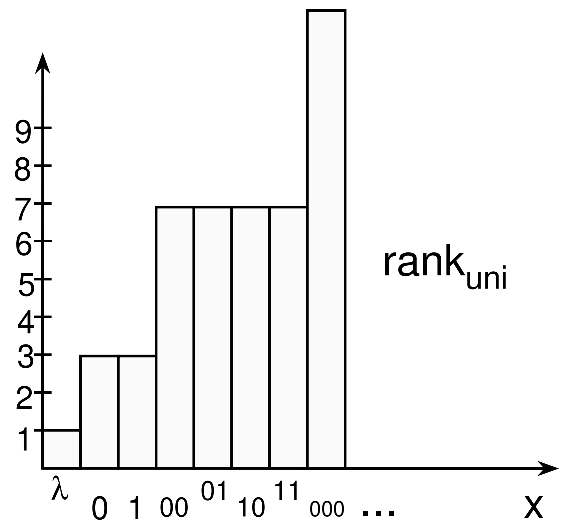

Abbildung 2.3: Die uniforme Rangfunktion rank_{uni} über der Menge aller binären Zeichenketten.

Definition 5 ***Uniforme Wahrscheinlichkeitsverteilungen*** *gewichten Eingaben gleicher Länge gleich. Kürzere Eingaben erhalten eine größere Wahrscheinlichkeit als längere. Zu diesen Verteilungen korrespondiert die* ***uniforme Rangfunktion*** *über der Eingabemenge* Σ^* *, definiert durch*

$$\mathbf{rank_{uni}(x)} \;:=\; \left|\Sigma^{\leq |x|}\right| \;.$$

Wahrscheinlichkeitsverteilungen $\boldsymbol{\mu_S}$ *, die nur auf Teilmengen* $S \subseteq \Sigma^*$ *uniform sind und sonst die Wahrscheinlichkeit* 0 *annehmen, heißen* ***S-uniform****. Deren Rangfunktion werde* $\mathbf{rank_S}$ *genannt:*

$$\mathbf{rank}_{\boldsymbol{S}}\boldsymbol{(x)} \;:=\; \mathrm{rank}_{\mu_S}(x) \;=\; \begin{cases} |S^{\leq |x|}| \;, & \textit{falls } x \in S \;, \\ \infty \;, & \textit{sonst} \;. \end{cases}$$

Für eine Rangfunktion ρ *bezeichne* $\boldsymbol{\rho^{\mathrm{inj}}}$ *diejenige injektive Rangfunktion, die bestimmt wird durch*

$$\boldsymbol{\rho^{\mathrm{inj}}(x)} \;:=\; \begin{cases} \rho(x) \;-|\{z \mid z > x \text{ und } \rho(x) = \rho(z)\}| \;, & \textit{falls } \rho(x) < \infty \;, \\ \infty \;, & \textit{sonst} \;. \end{cases}$$

ρ^{inj} wird auch **injektivierte Rangfunktion** von ρ genannt. Betrachtet man die injektivierte uniforme Rangfunktion $\mathrm{rank}_{\mathrm{uni}}^{\mathrm{inj}}$, so ergibt sich

$$\mathrm{rank}_{\mathrm{uni}}^{\mathrm{inj}}(x) \;=\; \mathrm{ord}(x) \;.$$

Analog ergibt sich für die injektivierten S-uniformen Wahrscheinlichkeitsverteilungen (siehe Abbildung 2.4)

$$\mathrm{rank}_S^{\mathrm{inj}}(x) \;=\; \begin{cases} |S^{\leq x}| \;, & \text{falls } x \in S \;, \\ \infty \;, & \text{sonst} \;. \end{cases}$$

Wir studieren diese Begriffe an folgenden Beispielen.

Beispiel:
Betrachten wir zuerst die Wahrscheinlichkeitsverteilung μ_1, definiert durch

$$\mu_1(x) \;:=\; \begin{cases} \frac{c}{\mathrm{ord}(z)^2} \;, & \text{falls } x = z0 \;, \\ \frac{c}{2^{\mathrm{ord}(z)}} \;, & \text{sonst} \;, \end{cases}$$

wobei c so gewählt wird, daß $\sum_x \mu_1(x) = 1$ gilt. Man sieht sofort, daß μ_1 alle Zeichenketten, die auf 1 enden, wesentlich unwahrscheinlicher macht als solche, die auf 0 enden.

Die Rangfunktion $\rho_1 \;:=\; \mathrm{rank}_{\mu_1}$ berechnet sich nun wie folgt:

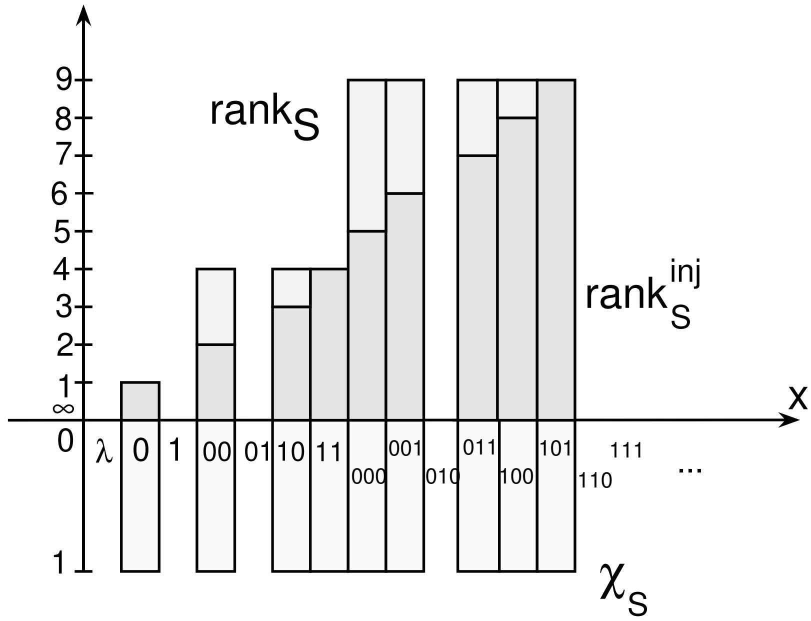

Abbildung 2.4: Die S-uniforme Rangfunktion rank_S und die injektivierte uniforme Rangfunktion $\mathrm{rank}_S^{\mathrm{inj}}$ werden hier der charakteristischen Funktion χ_S von S gegenübergestellt.

1. Fall: $x = z0$

   Der Rang ergibt sich aus der Ordnungszahl der Zeichenketten, die auf 0 enden, addiert um die Anzahl der anderen Zeichenketten, die wahrscheinlicher sein können.

$$\begin{aligned} \rho_1(x) &= \mathrm{ord}(z) + |\{k \mid 2^k \leq \mathrm{ord}(z)^2\}| \\ &= \mathrm{ord}(z) + 2 \cdot \log_2(\mathrm{ord}(z)) \ . \end{aligned}$$

2. Fall: $x \neq z0$

   Aufgrund der geringen Wahrscheinlichkeiten für solche x wird der Rang hauptsächlich durch die Zeichenketten, die auf 0 enden, bestimmt.

$$\begin{aligned} \rho_1(x) &= \mathrm{ord}(z) + |\{k \mid k^2 \leq 2^{\mathrm{ord}(z)}\}| \\ &= \mathrm{ord}(z) + \exp(\mathrm{ord}(z)/2) \ . \end{aligned}$$

Die Rangfunktion ist daher

$$\rho_1(x) = \begin{cases} \mathrm{ord}(z) + 2 \cdot \log_2(\mathrm{ord}(z)) \ , & \text{falls } x = z0 \ , \\ \mathrm{ord}(z) + \exp(\mathrm{ord}(z)/2) \ , & \text{sonst} \ . \end{cases}$$

Gehen wir nun den umgekehrten Schritt und fragen uns, wie man aus der Kenntnis der Rangfunktion eine Wahrscheinlichkeitsverteilung erhalten kann.

Wir betrachten hier die injektivierte L_{Pal}-uniforme Rangfunktion, wobei L_{Pal} die Menge aller binären Palindrome bezeichnet. Die Rangfunktion ergibt sich daher aus

$$\mathrm{rank}_{\mathrm{Pal}} = \begin{cases} |L_{\mathrm{Pal}}^{\leq |x|}| , & \text{falls } x \in S , \\ \infty , & \text{sonst} . \end{cases}$$

Dabei gilt für $|L_{\mathrm{Pal}}^{=n}| = 2^{\lceil \frac{n}{2} \rceil}$. Die injektivierte Rangfunktion $\mathrm{rank}_{\mathrm{Pal}}^{\mathrm{inj}}$ berechnet sich dagegen durch

$$\mathrm{rank}_{\mathrm{Pal}}^{\mathrm{inj}} = \begin{cases} |L_{\mathrm{Pal}}^{\leq x}| , & \text{falls } x \in S , \\ \infty , & \text{sonst} . \end{cases}$$

Seien $\mathrm{praefix}(s, n)$ die Zeichenkette bestehend aus den ersten n Buchstaben von s. Dann ergibt sich

$$|L_{\mathrm{Pal}}^{\leq x}| = |L_{\mathrm{Pal}}^{<|x|}| + \mathrm{ord}(\mathrm{praefix}(x, \lceil |x|/2 \rceil)) - \exp(\lceil |x|/2 \rceil) .$$

Aus jeder Rangfunktion ρ erhält man jetzt passende Wahrscheinlichkeitsverteilungen, indem man für eine streng monoton abnehmende Folge $f(i)$ mit $\sum_{i \in \mathbb{N}} f(i) = 1$ die Rangfunktion einsetzt: $\sum_x f(\rho(x))$. Somit wären geeignete Wahrscheinlichkeitsverteilungen

$$\begin{aligned} \mu_2(x) &= \frac{c_2}{\rho(x)^2} , \\ \mu_3(x) &= \frac{c_3}{\rho(x) \cdot \log(\rho(x))^2} , \\ \mu_4(x) &= \frac{c_4}{\rho(x) \cdot \log(\rho(x)) \cdot \mathrm{llog}(\rho(x))^2} , \end{aligned}$$

wobei c_2, c_3, c_4 geeignete Konstanten sind.

Rangfunktionen definieren also Äquivalenzklassen von Wahrscheinlichkeitsverteilungen. Man kann diese aber auch erhalten, indem man ein anderes Mittel zu Hilfe nimmt: **monotone Transformationen**.

Definition 6 *Eine reellwertige monotone Abbildung* $m : [0,1] \to [0,1]$ *wird* ***monotone Transformation*** *der Wahrscheinlichkeitsverteilung* μ *genannt, gdw.* $m(\mu)$ *wiederum eine Wahrscheinlichkeitsverteilung ist, d.h.* $\sum_x m(\mu(x)) = 1$.

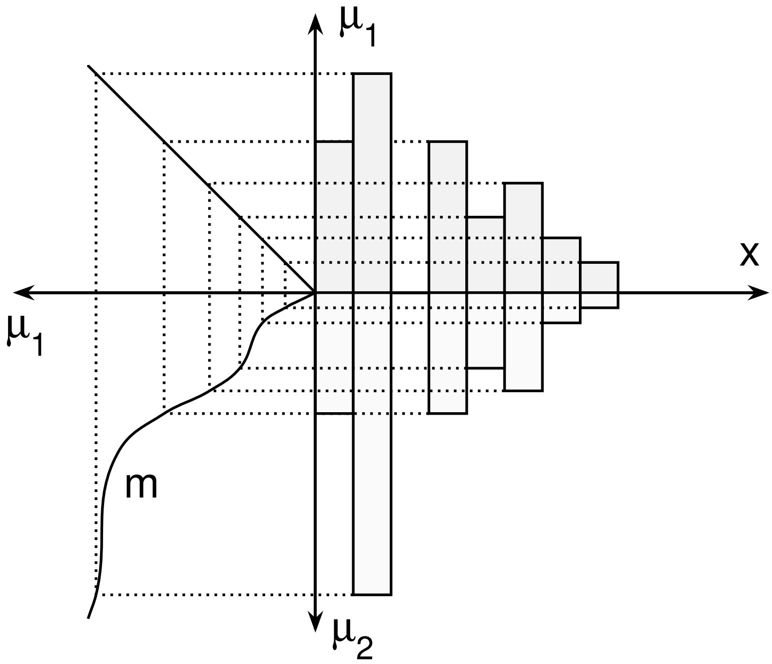

Abbildung 2.5: Die Wahrscheinlichkeitsverteilung $\mu_2(x) = m(\mu_1(x))$ entsteht aus $\mu_1(x)$ durch die monotone Transformation m.

Eine monotone Transformationen m zerstört nicht die Ordnung einer Wahrscheinlichkeitsverteilung μ (siehe Abbildung 2.5), da gilt

$$\mu(x) > \mu(y) \quad \Longrightarrow \quad m(\mu(x)) \geq m(\mu(y)) \ .$$

Somit erhalten wir durch folgende Definition das oben vorgeschlagene *average*-Maß.

Definition 7 *Die Menge* **Av(T)** *(***average T***) enthält alle Paare* (f, μ) *bestehend aus einer Funktion* $f : \Sigma^* \to \mathbb{N}$ *und einer Wahrscheinlichkeitsverteilung* μ*, so daß für alle monotonen Transformationen* m *von* μ *gilt*

$$(f, m(\mu)) \in \mathrm{Lav}\,(T) \ , \quad d.h. \quad E_{m(\mu)}\left(\frac{T^{[-1]} \circ f(x)}{|x|}\right) \leq 1 \ .$$

Wegen des Allquantors über alle montone Transformationen ist die obige Bedingung für $\mathrm{Av}\,(T)$ schwer zu überprüfen. Aber es gibt eine äquivalente und zugleich praktischere Charakterisierung von $\mathrm{Av}\,(T)$. Hierfür betrachten wir den Sonderfall von **Schwellwertfunktionen** $\mathtt{thr}_l : [0,1] \to [0,1]$ als monotone Transformationen, wobei wir für $l = \mathrm{rank}_\mu(x)$ definieren $\mathtt{thr}_l(z) := 1/l$, falls $z \geq \mu(x)$, andernfalls ist $\mathtt{thr}_l(z) := 0$.

Proposition 1

$$(f,\mu) \in \mathrm{Av}(T) \qquad \Longleftrightarrow \qquad \forall l \quad \sum_x \mathtt{thr}_l(\mu(x)) \, \frac{T^{[-1]}(f(x))}{|x|} \;\leq\; 1 \; .$$

Beweis: „$\Rightarrow$“ folgt aus der Definition.
„$\Leftarrow$“: Sei $x_1,\ x_2,\ \ldots$ eine geordnete Aufzählung aller Zeichenketten z mit $\mu(z) \neq 0$ derart, daß $\mu(x_i) \geq \mu(x_{i+1})$. Für $a^n := (a_1, \ldots, a_n)$ mit $a_i := T^{[-1]}(f(x_i))/|x_i|$ werde angenommen $(f,\mu) \notin \mathrm{Av}(T)$. Dann gibt es eine natürliche Zahl n mit $\mu(x_n) > \mu(x_{n+1})$, so daß für eine monotone Transformation m gilt

$$\sum_{i=1}^{n} m(\mu(x_i)) \cdot a_i > 1 \; .$$

Um durch eine Wahl von m den größten Wert der Summe $\sum_{i=1}^{n} m(\mu(x_i)) \cdot a_i$ zu erhalten, müssen wir ein lineares Optimierungsproblem lösen. Die Lösung besteht aus einer Menge von Vektoren, die einen ($n-r$)-dimensionalen Simplex beschreiben. Hierbei beschreibt r die Anzahl der Indizes $i < n$, für die gilt $\mu(x_i) = \mu(x_{i+1})$.

Die Extremstellen sind $(\underbrace{1/i, \ldots, 1/i}_{i\text{-mal}}, 0, \ldots, 0)$. Die Lösung ist die Menge aller Punkte, die größtmögliche Entfernung haben von der $(n-1)$-dimensionalen Hyperebene, die den Ursprung $(0, \ldots, 0)$ beinhaltet und den Normalenvektor a besitzt. Diese Menge enthält einen Extrempunkt des Simplex. Daher gibt es eine ganze Zahl l mit $a_1 = a_2 = \cdots = a_l = 1/l$, durch die die obige Summe ihr Maximum erreicht. ∎

Als direkte Folgerung dieser Proposition erhalten wir, daß eine Durchschnittsschranke berechnet werden kann, ohne eine einzige monotone Transformation zu betrachten. Es ist nur noch die Menge $\boldsymbol{X}_{\boldsymbol{\mu}}^{\boldsymbol{\leq l}}$ der l wahrscheinlichsten Zeichenketten der Wahrscheinlichkeitsverteilung μ interessant:

$$\boldsymbol{X}_{\boldsymbol{\mu}}^{\boldsymbol{\leq l}} \quad := \quad \{x \mid \mathrm{rank}_\mu(x) \leq l\} \; .$$

Wenn die Rangfunktion $\rho = \mathrm{rank}_\mu$ einer Wahrscheinlichkeitsverteilung μ ist, bezeichne analog $\boldsymbol{X}_{\boldsymbol{\rho}}^{\boldsymbol{\leq l}}$ die Menge

$$\boldsymbol{X}_{\boldsymbol{\rho}}^{\boldsymbol{\leq l}} \quad := \quad \{x \mid \rho(x) \leq l\} \; .$$

Korollar 1

$$(f,\mu) \in \mathrm{Av}(T) \qquad \Longleftrightarrow \qquad \forall l \sum_{x \in X_\mu^{\leq l}} \frac{T^{[-1]}(f(x))}{|x|} \;\leq\; l \; .$$

Beweis: Für eine Schwellwertfunktion $\mathtt{thr}_l$ erhält man

$$\sum_x \mathtt{thr}_l(\mu(x)) \; \frac{T^{[-1]}(f(x))}{|x|} \;=\; \sum_{x \in X_\mu^{\leq l}} \frac{T^{[-1]}(f(x))}{l \cdot |x|} \; .$$

Deswegen folgt Korollar 1 aus Proposition 1. ∎

Ab jetzt werden wir nur noch diese einfachere Charakterisierung benutzen. Weil diese Bedingung nur die Rangfolge des Eingaberaums bezüglich der Verteilung berücksichtigt, erhalten wir

Korollar 2 *Falls* μ_1, μ_2 *Wahrscheinlichkeitsverteilungen sind mit gleicher Rangfunktion* $\mathrm{rank}_{\mu_1} = \mathrm{rank}_{\mu_2}$ *, gilt*

$$(f, \mu_1) \in \mathrm{Av}\,(T) \qquad \Longleftrightarrow \qquad (f, \mu_2) \in \mathrm{Av}\,(T) \; .$$

Falls $\mathrm{rank}_{\mu_2} \leq \mathrm{rank}_{\mu_1}$ *, gilt*

$$(f, \mu_1) \in \mathrm{Av}\,(T) \qquad \Longrightarrow \qquad (f, \mu_2) \in \mathrm{Av}\,(T) \; .$$

Eine Rangfunktion ρ repräsentiert also nicht nur eine ganze Äquivalenzklasse von Wahrscheinlichkeitsverteilungen, also genau jene Verteilungen μ mit $\mathrm{rank}_\mu = \rho$, sondern kann auch ohne jede Einschränkung direkt zur *average-case*-Analyse herangezogen werden. Aus diesem Grund werden wir in der weiteren Arbeit nicht unterscheiden zwischen dem Paar (f, μ) , welches eine Verteilung enthält und dem Paar (f, ρ) , welches auf die entsprechende Rangfunktion $\rho = \mathrm{rank}_\rho$ verweist.

Die folgende Proposition gibt eine weitere nützliche Darstellung dieser Mengen und dient zur Vereinfachung bestimmter Abschätzungen.

Proposition 2

$$(f, \mu) \in \mathrm{Av}\,(T) \qquad \Longleftrightarrow \qquad \forall l \sum_{x \in X_\mu^{\leq l}} \min\{\alpha_x \mid f(x) \leq T(\alpha_x \cdot |x|)\} \;\; \leq \; l \; .$$

Beweis: Nach Korollar 1 gilt

$$(f, \mu) \in \mathrm{Av}\,(T) \qquad \Longleftrightarrow \qquad \forall l \sum_{x \in X_\mu^{\leq l}} \frac{T^{[-1]}(f(x))}{|x|} \; \leq \; l \; .$$

Nach der Definition der Inversen von T gilt

$$\Longleftrightarrow \qquad \forall l \sum_{x \in X_\mu^{\leq l}} \frac{\min\{m \mid f(x) \leq T(m)\}}{|x|} \; \leq \; l \; .$$

Nun ersetzen wir m durch $\alpha_x \cdot |x|$

$$\Longleftrightarrow \qquad \forall l \sum_{x \in X_\mu^{\leq l}} \min\{\alpha_x \mid f(x) \leq T(\alpha_x \cdot |x|)\} \ \leq \ l \ .$$

∎

Bevor wir uns mit diesem Maß intensiver beschäftigen werden, untersuchen wir ein verwandtes Maß.

2.2 Das Eav-Maß

Bis jetzt betrachteten wir den Erwartungswert nur bezüglich lokaler Wahrscheinlichkeitsverteilungen μ_n. Eine Funktion $f : \mathbb{N} \to \mathbb{N}$ wird **lokal T-erwartet** genannt, wenn gilt

$$\forall n \quad E_{\mu_n}\left(\frac{f}{T}\right) \ \leq \ 1 \ .$$

Nach Definition des Erwartungswerts bedeutet dies

$$\forall n \quad \sum_{|x|=n} \mu_n(x) \, \frac{f(x)}{T(|x|)} \ \leq \ 1 \ .$$

Wir bevorzugen aber eine Definition, die globale Wahrscheinlichkeitsverteilungen berücksichtigt.

$$E_\mu\left(\frac{f}{T}\right) \ \leq \ 1$$

oder in der Summenschreibweise

$$\sum_x \mu(x) \cdot \frac{f(x)}{T(|x|)} \ \leq \ 1 \ .$$

Beispiel:
Wir betrachten die uniforme Wahrscheinlichkeitsverteilung $\mu_{\text{uni}} := 6/\pi^2 \cdot |x|^{-2} \cdot 2^{-|x|}$ und die Zeitfunktion $f(x) := |x|^2$. Diese Funktion ist aber im Erwartungswert sogar durch eine Schranke $T \in O(\mathcal{N}^{1+\epsilon})$ beschränkt, so daß gilt

$$E_{\mu_{\text{uni}}}(f/T) \ \leq \ 1 \ .$$

Weitere Beispiele dieses Phänomens zeigt Tabelle 2.1.

Dieses Maß ist aus denselben Gründen nicht präzise wie das Levinsche Durchschnittsmaß. Wir benutzen wieder den Begriff der monotonen Transformationen, um die gewünschte Präzision zu erhalten.

Definition 8 *Die Menge* **Eav(T)** *(***expected average T***) enthält alle Paare* (f, μ) *bestehend aus einer Funktion* $f : \Sigma^* \to \mathbb{N}$ *und einer Wahrscheinlichkeitsverteilung* μ *, bei denen für alle monotone Transformationen* m *von* μ *gilt*

$$E_{m(\mu)}\left(\frac{f}{T}\right) \leq 1 \ .$$

Wir können diese Bedingung folgendermaßen vereinfachen.

Proposition 3

$$(f, \mu) \in \mathrm{Eav}\,(T) \qquad \Longleftrightarrow \qquad \forall l \sum_{x \in X_\mu^{\leq l}} \frac{f(x)}{T(|x|)} \leq l \ .$$

Beweis: „$\Rightarrow$" nach Definition.
„$\Leftarrow$": verläuft genauso wie die Beweise von Proposition 1 und Korollar 1, wenn man $\frac{T^{[-1]}(f(x))}{|x|}$ durch $\frac{f(x)}{T(|x|)}$ ersetzt. ∎

Diese präzise Charakterisierung mittels Eav ist dem ursprünglichen lokalen Erwartungswert näher verwandt als der oben angeführte Zwischenschritt $E_\mu(f/T) \leq 1$. So kann man leicht einsehen, daß für die uniforme globale Wahrscheinlichkeitsverteilung μ_{uni} und ihr lokales Äquivalent $[\mu_{\mathrm{uni}}]_n$ gilt

$$\forall n \;\; E_{[\mu_{\mathrm{uni}}]_n}(f) \leq T(n) \quad \Longrightarrow \quad (f, \mu_{\mathrm{uni}}) \in \mathrm{Eav}\,(T) \ .$$

Die Äquivalenz ist nicht gegeben, immerhin gilt aber für Eingabealphabet Σ

$$\forall n \;\; E_{[\mu_{\mathrm{uni}}]_n}(f) \leq \frac{|\Sigma|}{|\Sigma| - 1} \cdot T(n) \quad \Longleftarrow \quad (f, \mu_{\mathrm{uni}}) \in \mathrm{Eav}\,(T) \ .$$

Betrachtet man Komplexitätsklassen, so ergibt sich später für viele Wahrscheinlichkeitsverteilungen die Äquivalenz zwischen Eav und lokalen Erwartungswert (siehe Theorem 4).

2.3 Eigenschaften der Average-Maße

Wie verhalten sich nun die hier konstruierten Maße gegenüber elementaren Operationen?

Lemma 3 *Seien* $(f, \mu) \in \mathrm{Eav}\,(T_f)$, $(f_i, \mu) \in \mathrm{Eav}\,(T_i)$ *und seien* $c \in \mathbb{R}^+, n \in \mathbb{N}$, *dann gilt*

$$\begin{aligned} (c \cdot f, \mu) &\in \mathrm{Eav}(c \cdot T_f + 1) \ , \\ \left(\sum_{i=1}^{n} f_i, \mu\right) &\in \mathrm{Eav}(n \cdot \textstyle\sum_{i=1}^{n} T_i) \ . \end{aligned}$$

Beweis:

1. $(c \cdot f, \mu) \in \mathrm{Eav}\,(c \cdot T_f + 1)$
   Wir werden beweisen, daß $(\lfloor c \cdot f \rfloor, \mu) \in \mathrm{Eav}\,(\lceil c \cdot T_f \rceil)$. Dafür betrachten wir für alle l

$$\sum_{x \in X_\mu^{\leq l}} \frac{\lfloor c \cdot f(x) \rfloor}{\lceil c \cdot T_f(|x|) \rceil} \ \leq \ \sum_{x \in X_\mu^{\leq l}} \frac{c \cdot f(x)}{c \cdot T_f(|x|) \cdot l} \ \leq \ l \ ,$$

   da $(f, \mu) \in \mathrm{Eav}\,(T_f)$.

2. $(\sum_{i=1}^{n} f_i, \mu) \in \mathrm{Eav}\,(n \cdot \sum_{i=1}^{n} T_i)$
   Zuerst beachte man, daß grundsätzlich gilt

$$\frac{\sum_{i=1}^{n} f_i(x)}{\sum_{i=1}^{n} T_i(|x|)} \leq \sum_{i=1}^{n} \frac{f_i(x)}{T_i(|x|)} \ .$$

   Daraus folgt

$$\sum_{x \in X_\mu^{\leq l}} \frac{\sum_{i=1}^{n} f_i(x)}{\sum_{i=1}^{n} T_i(|x|)} \ \leq \ \sum_{x \in X_\mu^{\leq l}} \ \sum_{i=1}^{n} \frac{f_i(x)}{T_i(|x|)} \ \leq \ n \cdot l \ .$$

∎

Hieraus folgt sofort

Korollar 3 *Seien* $(f, \mu) \in \mathrm{Eav}\,(T_f)$, $(g, \mu) \in \mathrm{Eav}\,(T_g)$ *und sei* $k \in \mathbb{N}$. *Dann gilt*

$$\begin{aligned} (f + g, \mu) &\in \mathrm{Eav}(2 \cdot T_f + 2 \cdot T_g) \ , \\ (f + k, \mu) &\in \mathrm{Eav}(2 \cdot T_f + 2 \cdot k) \ . \end{aligned}$$

Für Addition und lineare Abbildungen gelten also ähnliche Gesetzmäßigkeiten, wie wir sie für den Erwartungswert kennen. In der Einleitung wurde schon gezeigt, daß andere Operatoren scheitern. So gibt es eine Funktion f und eine Wahrscheinlichkeitsverteilung μ, so daß $(f,\mu) \in \mathrm{Eav}(\mathcal{N}^2)$ aber auch $(f^2,\mu) \notin \mathrm{Eav}(\mathrm{POL})$ gilt.

Das Av-Maß dagegen bietet eine Vielzahl von Abgeschlossenheitseigenschaften.

Lemma 4
a) Für eine monotone Funktion $g : \mathcal{N} \to \mathcal{N}$ gilt:

$$(f,\mu) \in \mathrm{Av}(T) \qquad \Longrightarrow \qquad (g \circ f, \mu) \in \mathrm{Av}(g \circ T) \ .$$

b) Falls $g > 0$ streng mononton zunehmend ist, gilt sogar

$$(f,\mu) \in \mathrm{Av}(T) \qquad \Longleftrightarrow \qquad (g \circ f, \mu) \in \mathrm{Av}(g \circ T) \ .$$

Beweis: a) Nach Proposition 2,

$$\begin{aligned}
(f,\mu) \in \mathrm{Av}(T) \quad &\Longrightarrow \quad \forall l \sum_{x \in X_\mu^{\leq l}} \min\{\alpha_x \mid f(x) \leq T(\alpha_x \cdot |x|)\} \ \leq \ l \\
&\Longrightarrow \quad \forall l \sum_{x \in X_\mu^{\leq l}} \min\{\alpha_x \mid g(f(x)) \leq g(T(\alpha_x \cdot |x|))\} \ \leq \ l \\
&\Longrightarrow \quad (g \circ f, \mu) \in \mathrm{Av}(g \circ T) \ .
\end{aligned}$$

b) folgt ähnlich durch Anwendung durch die monotone Funktion $g^{[-1]}$ auf $g \circ f$ und $g \circ T$. Man beachte, daß für streng monotone Funktionen g gilt $g^{[-1]} \circ g = \mathcal{N}$. ∎

Korollar 4 *Für beliebige Konstanten $c, c_1 \in \mathbb{R}$, $c_2 \in \mathbb{R}^+$ gilt*

$$\begin{aligned}
(f,\mu) \in \mathrm{Av}(T) \quad &\Longrightarrow \quad (c_1 \cdot f + c_2, \mu) \in \mathrm{Av}(c_1 \cdot T + c_2) \quad \text{und} \\
& \quad (f^c, \mu) \in \mathrm{Av}(T^c) \\
g \in \mathrm{Pol}(f) \quad \text{und} \quad (f,\mu) \in \mathrm{Av}(T) \quad &\Longrightarrow \quad (g,\mu) \in \mathrm{Av}(\mathrm{Pol}(T)) \\
g \in \mathrm{Pol}(f) \quad \text{und} \quad (f,\mu) \in \mathrm{Av}(\mathrm{POL}) \quad &\Longrightarrow \quad (g,\mu) \in \mathrm{Av}(\mathrm{POL})
\end{aligned}$$

Summen oder Produkte von Komplexitätsschranken abzuschätzen, ist etwas aufwendiger. Zu diesem Zweck wird die Maximum-Funktion $\max : \mathcal{N} \times \mathcal{N} \to \mathcal{N}$ herangezogen.

Lemma 5 *Für $0 < \beta < 1$ gilt*

$$\begin{aligned}
(f,\mu) \in \mathrm{Av}(T_f) \text{ und } (g,\mu) \in \mathrm{Av}(T_g) \quad &\Longrightarrow \\
(\max[f,g],\mu) \quad &\in \quad \mathrm{Av}\Big(\max[T_f(\tfrac{\mathcal{N}}{\beta}), T_g(\tfrac{\mathcal{N}}{1-\beta})]\Big)
\end{aligned}$$

Beweis: Sei $D \; := \; \max[T_f(\frac{\mathcal{N}}{\beta}), T_g(\frac{\mathcal{N}}{1-\beta})]$. Dann gilt,

$$\begin{aligned}(f,\mu) \in \mathrm{Av}(T_f) \quad &\Longrightarrow \quad \forall l \sum_{x \in X_\mu^{\leq l}} \min\left\{\alpha_x \mid f(x) \leq T_f\left(\frac{\alpha_x}{\beta} \cdot |x|\right)\right\} \;\leq\; l \cdot \beta \\ &\Longrightarrow \quad \forall l \sum_{x \in X_\mu^{\leq l}} \min\{\alpha_x \mid f(x) \leq D(\alpha_x \cdot |x|)\} \;\leq\; l \cdot \beta \; .\end{aligned}$$

Ähnlich gilt

$$(g,\mu) \in \mathrm{Av}\,(T_g) \qquad \Longrightarrow \qquad \forall l \sum_{x \in X_\mu^{\leq l}} \min\{\alpha_x \mid g(x) \leq D(\alpha_x \cdot |x|)\} \;\leq\; l \cdot (1-\beta) \; .$$

Somit gilt

$$\forall l \sum_{x \in X_\mu^{\leq l}} \min\{\alpha_x \mid g(x) \leq D(\alpha_x \cdot |x|) \;\text{ und }\; f(x) \leq D(\alpha_x \cdot |x|)\} \;\leq\; l \; ,$$

was wegen Proposition 2 impliziert, daß $(\max[f,g],\mu) \in \mathrm{Av}\,(D)$. ∎

Unter Berücksichtigung von $f+g \leq 2 \cdot \max[f,g]$, resp. $f \cdot g \leq \max[f^2, g^2]$ erhalten wir für die Summe und das Produkt

Korollar 5

$$\begin{aligned}(f,\mu) \in \mathrm{Av}(T_f) \quad \text{und} \quad (g,\mu) \in \mathrm{Av}(T_g) \quad \Longrightarrow& \\ (f+g,\mu) \quad &\in \quad \mathrm{Av}(2 \cdot \max[T_f(2 \cdot \mathcal{N}), T_g(2 \cdot \mathcal{N})]) \\ \text{und} \quad (f \cdot g,\mu) \quad &\in \quad \mathrm{Av}(\max[T_f(2 \cdot \mathcal{N})^2, T_g(2 \cdot \mathcal{N})^2]) \; .\end{aligned}$$

Damit folgt also auch $(f+g,\mu) \in \mathrm{Av}\,(2 \cdot (T_f + T_g) \circ (2 \cdot \mathcal{N}))$. Tabelle 2.2 faßt die Ergebnisse dieses Abschnittes zusammen. Eine ausführliche Darstellung gibt Tabelle 9.1.

2.4 Eav versus Av

Das Av-Maß ist das schwächere Maß für $T \geq \mathcal{N}$. Algorithmen mit guter erwarteter Laufzeit $(t,\mu) \in \mathrm{Eav}\,(T)$ „vererben“ demnach ihr Laufzeitverhalten diesem neuen Maß $(t,\mu) \in \mathrm{Av}\,(T(2 \cdot \mathcal{N}))$. Andererseits behalten untere Schranken, die für das Av-Maß bestimmt werden, ihre Gültigkeit auch für das dem Erwartungswert nahen Eav-Maß.

	Eav	Av
$c \cdot f$	$c \cdot T_f + 1$	$c \cdot T_f$
$f + g$	$2 \cdot (T_f + T_g)$	$2 \cdot (T_f(2 \cdot \mathcal{N}) + T_g(2 \cdot \mathcal{N}))$
$f \cdot g$	—	$T_f(2 \cdot \mathcal{N})^2 + T_g(2 \cdot \mathcal{N})^2$
f^k	—	T_f^k

Tabelle 2.2: Das Verhalten von Eav und Av gegenüber elementarer Abbildungen. Die Spaltenüberschrift beschreibt das gewählte Durchschnittsmaß ($(f,\mu) \in \mathrm{Av}(T_f)$ oder $\mathrm{Eav}(T_f)$; analog g).

Lemma 6 *Sei T eine Zeitschranke, so daß $T/\mathcal{N}$ $(n \mapsto \frac{T(n)}{n})$ monoton zunehmend ist, dann gilt*

$$\mathrm{Eav}(T) \subseteq \mathrm{Av}(T(2 \cdot \mathcal{N})) \ .$$

Beweis: Sei $(f,\mu) \in \mathrm{Eav}(T)$. Somit gilt

$$\forall l \qquad \sum_{x \in X_\mu^{\leq l}} \frac{f(x)}{T(|x|)} \ \leq \ l \ .$$

Um obiges zu beweisen, muß man zeigen, daß gilt

$$\forall l \qquad \sum_{x \in X_\mu^{\leq l}} \frac{T^{[-1]}(f(x))}{|x|} \ \leq \ 2 \cdot l \ .$$

Hierfür teilt man die Eingaben in die beiden Teilmengen

$$\begin{aligned} I_1 &:= \{x \mid f(x) \ > \ T(|x|)\} \ , \\ I_2 &:= \{x \mid f(x) \ \leq \ T(|x|)\} \ . \end{aligned}$$

Wenn $f(z) > T(|z|)$, dann ist $T^{[-1]}(f(z)) \ \geq \ T^{[-1]}(T(|z|)+1) \ \geq \ |z|$. Unter der Voraussetzung, daß $g(n) := T(n)/n$ monoton zunehmend ist, ergibt sich für solche Zeichenketten x

$$\begin{aligned} g(T^{[-1]}(f(z))) &\geq g(|z|) && \Longleftrightarrow \\ \frac{T(T^{[-1]}(f(z)))}{T^{[-1]}(f(z))} &\geq \frac{T(|z|)}{|z|} && \Longrightarrow \\ \frac{f(z)}{T^{[-1]}(f(z))} &\geq \frac{T(|z|)}{|z|} && \Longleftrightarrow \\ \frac{f(z)}{T(|z|)} &\geq \frac{T^{[-1]}(f(z))}{|z|} \ . \end{aligned}$$

Dann gilt

$$\sum_{x \in I_1 \cap X_\mu^{\leq l}} \frac{T^{[-1]}(f(x))}{|x|} \leq \sum_{x \in I_1 \cap X_\mu^{\leq l}} \frac{f(x)}{T(|x|)} \leq l \; .$$

Für $x \in I_2$ gilt

$$\frac{T^{[-1]}(f(x))}{|x|} \leq 1 \; .$$

Deswegen gilt

$$\sum_{x \in I_2 \cap X_\mu^{\leq l}} \frac{T^{[-1]}(f(x))}{|x|} \leq l$$

∎

Betrachtet man Platzschranken oder parallele Algorithmen, so ist es notwendig, auch Komplexitätschranken zu betrachten, die langsamer wachsen als lineare. Vorauszusetzen ist dabei für das Av-Maß immer, daß diese nicht durch eine konstante Funktion beschränkt sind. Eine schlüssige Ergänzung der Definition wäre die Forderung, daß die betrachtete Funktion diese konstante Schranke niemals überschreitet. Bisherige Arbeiten [JRS 94, JRSW 94] zeigten jedoch, daß selbst für einfachste Problemstellungen dieser Fall nicht eintritt.

Lemma 7 *Sei T eine Funktion, so daß die Funktion $\mathcal{N}/T$ $(n \mapsto \frac{n}{T(n)})$ monoton zunehmend ist, dann gilt*

$$\mathrm{Av}\,(T) \subseteq \mathrm{Eav}\,(2 \cdot T) \; .$$

Beweis: Sei $(f, \mu) \in \mathrm{Av}\,(T)$. Somit gilt

$$\forall l \qquad \sum_{x \in X_\mu^{\leq l}} \frac{T^{[-1]}(f(x))}{|x|} \leq l \; .$$

Um das obige Lemma zu beweisen, muß man folgendes zeigen.

$$\forall l \qquad \sum_{x \in X_\mu^{\leq l}} \frac{f(x)}{T(|x|)} \leq 2 \cdot l \; .$$

Hierfür teilt man die Eingaben wiederum in die beiden Teilmengen

$$\begin{aligned} I_1 &:= \{x \mid f(x) > T(|x|)\} \; , \\ I_2 &:= \{x \mid f(x) \leq T(|x|)\} \; . \end{aligned}$$

Wenn $f(z) > T(|z|)$, dann gilt für $g(n) := n/T(n)$ die Ungleichung $g(T^{[-1]}(f(z))) \geq g(|z|)$, da g nach Voraussetzung monoton zunehmend ist. Damit ist

$$\frac{T^{[-1]}(f(z))}{T(T^{[-1]}(f(|z|)))} \geq \frac{|z|}{T(|z|)} .$$

Weiter gilt, da $T(T^{[-1]}(f(|z|))) \geq f(|z|)$,

$$\frac{T^{[-1]}(f(z))}{|z|} \geq \frac{f(|z|)}{T(|z|)} .$$

Dann gilt

$$\sum_{x \in I_1 \cap X_{\mu}^{\leq l}} \frac{f(x)}{T(|x|)} \leq \sum_{x \in I_1 \cap X_{\mu}^{\leq l}} \frac{T^{[-1]}(f(x))}{|x|} \leq l .$$

Für $x \in I_2$ gilt

$$\frac{f(x)}{T(|x|)} \leq 1 .$$

Deswegen gilt

$$\sum_{x \in I_2 \cap X_{\mu}^{\leq l}} \frac{f(x)}{T(|x|)} \leq l$$

∎

Der Erwartungswert wird hier also schwächer als das Av-Maß. In [JRS 94] konnte dieser Effekt folgendermaßen ausgenutzt werden: Nachdem der Erwartungswert für eine breite Palette von Schaltkreisentwürfen die konstante Funktion $O(1)$ war, konnte die Wahl des besseren Schaltkreises durch das Av-Maß vorgenommen werden, wobei hier eine untere Schranke im Bereich von llog nachgewiesen werden konnte.

2.5 Median als Maß

Von *Average*-Maßen wird normalerweise erwartet, daß man wenige langsame Laufzeiten von Algorithmen „wegmitteln" kann. Dabei wissen wir immer noch nicht, wie groß die Werte einer Funktion f sein dürfen, damit sie im *average* zum Beispiel noch quadratisch beschränkt ist. Wie verhält es sich zwischen diesem Extrem und $f = \mathcal{N}^2$? Wie häufig darf zum Beispiel kubische Laufzeit auftreten?

Wir werden uns hier ein neuartiges Durchschnittsmaß erarbeiten, das diese Frage formal fassen wird und hilft, sie zu beantworten. Betrachten wir zuerst ein Beispiel.

Beispiel:
Wir wissen von einer Funktion $t : \Sigma^* \to \mathbb{N}$ ausschließlich, daß

$$(t, \mu_{\text{uni}}) \in \text{Eav}(\mathcal{N}^2) \; .$$

Wie oft kann diese Funktion nun höheres polynomielles Verhalten haben, also zum Beispiel $t(x) = |x|^{10}$? Wir untersuchen dafür die mögliche Anzahl aller Eingaben bezogen auf die l wahrscheinlichsten Eingaben $X^{\leq l}_{\mu_{\text{uni}}}$ mit $t(x) > |x|^{10}$. Eine einfache Rechnung führt zu

$$|\{x \in X^{\leq l}_{\mu_{\text{uni}}} \mid t(x) > |x|^{10}\}| \;\leq\; \frac{l}{\log^8 l} \; .$$

Dieser Anteil wird natürlich kleiner, wenn man die exponentielle Funktion betrachtet.

$$|\{x \in X^{\leq l}_{\mu_{\text{uni}}} \mid t(x) > \exp(|x|)\}| \;\leq\; \log^2 l \; .$$

Für größere Funktionen ergibt sich sogar

$$|\{x \in X^{\leq l}_{\mu_{\text{uni}}} \mid t(x) > \exp(|x|)^2\}| \;=\; 0 \; .$$

Betrachtet man dagegen eine Funktion $t : \Sigma^* \to \mathbb{N}$ mit

$$(t, \mu_{\text{uni}}) \;\in\; \text{Av}(\mathcal{N}^2) \; ,$$

so erhält man

$$\begin{aligned}
|\{x \in X^{\leq l}_{\mu_{\text{uni}}} \mid t(x) > |x|^{10}\}| \;&\leq\; \frac{l}{\log^5 l} \; , \\
|\{x \in X^{\leq l}_{\mu_{\text{uni}}} \mid t(x) > \exp(|x|)\}| \;&\leq\; \sqrt{l} \cdot \log l \; , \\
|\{x \in X^{\leq l}_{\mu_{\text{uni}}} \mid t(x) > \exp(|x|)^2\}| \;&\leq\; \log^2 l \; .
\end{aligned}$$

Der mitunter drastische Unterschied ermöglicht interessante Einblicke in das Av- und Eav-Maß. Siehe hierzu auch Tabelle 2.3.

Den Term $|\{x \in X^{\leq l}_{\mu} \mid t(x) > T(|x|)\}|$ wird unser neues Maß bewerten. Dieser Term ist eng verwandt mit dem Begriff des Medians in der Wahrscheinlichkeitstheorie. In vielen statistischen Untersuchungen wird der Median dem Erwartungswert vorgezogen, meist wegen seiner Unempfindlichkeit gegenüber Ausreißern.

Der Median müßte hier eigentlich eine Zeichenkette bezeichnen, da wir Wahrscheinlichkeiten über Zeichenketten betrachten. Für globale Wahrscheinlichkeitsverteilungen ist dies unmöglich, denn wo sollte in einem unendlichen Suchraum eine Zeichenkette zu finden sein, deren Wert genau in der Mitte liegt?

Wir gehen von einer Wahrscheinlichkeitsverteilung μ aus und betrachten wieder die Rangfunktion $\rho := \mathrm{rank}_\mu$ dieser Verteilung. Ziel dieser Untersuchungen soll nicht etwa sein, eine Zeichenkette aus dieser nicht endlichen Menge zu benennen, die mittelmäßiges Verhalten aufweist: Es geht vielmehr darum zu charaktersieren, ob die Laufzeiten $t(x)$ bezüglich der Reihenfolge ρ der Zeichenketten immer zu 50% unterhalb der Zeitschranke $T(|x|)$ liegen. Dabei verlangen wir dies für alle endlichen Teilmengen der Größe l, wobei wir die l am wahrscheinlichsten Eingaben aus Σ^* untersuchen, also

$$\forall l \quad \frac{|\{\, x \in X_\mu^{\leq l} \mid t(x) > T(|x|)\,\}|}{l} \quad \leq 50\% \ .$$

Diese 50%-Schranke ist aber gerade für die Betrachtung von Laufzeiten für Algorithmen uninteressant. Einen potentiellen Abnehmer eines Software-Produkts würde kaum die Versicherung zum Kauf anregen, daß die Hälfte aller Eingaben in quadratischer Laufzeit bearbeitet werden, von den anderen Eingaben aber nichts bekannt ist. Andererseits würde ein Interessent es in Kauf nehmen, wenn die Wahrscheinlichkeit für das Auftreten einer Zeitüberschreitung mit der eines Blitzschlages in den entsprechenden Rechner abgeschätzt werden kann.

Wir beschränken also die Häufigkeit dieser Ausreißer durch eine Komplexitätsschranke $a : \mathbb{N} \to \mathbb{N}$, die im Gegensatz zu Zeitschranken höchstens linear wächst. Wir sprechen gemäß folgender Definition dann von einer **Median-Beschränkung von (t,μ) durch T mit Häufigkeit a**.

Definition 9 *Die Menge* **Med(T,a)** *enthält alle Paare (f,μ) bestehend aus einer Funktion $f : \Sigma^* \to \mathbb{N}$ und einer Wahrscheinlichkeitsverteilung μ, für die gilt*

$$\forall l \in \mathbb{N} \quad |\{x \in X_\mu^{\leq l} \mid f(x) > T(|x|)\}| \ \leq \ a(l) \ .$$

Median-Beschränkung im ursprünglichen Sinne ergibt sich für die Häufigkeitsfunktion $a := \mathcal{N}/2$.

Dieses Maß unterscheidet sich fundamental von den *Average*-Maßen Av und Eav. Deswegen führen wir es nicht unter diesem Begriff, sondern prägen hiermit den gleichwertigen Begriff **Median-Maß**. Dieser Unterschied zeigt sich insbesondere bei den Abgeschlossenheitseigenschaften. Hier gilt

Lemma 8 *Für die monoton zunehmende Funktion $f : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$, die Häufigkeitsfunktionen a_1, a_2 und die Zeitschranken T_1, T_2 gilt die Implikation*

$$(t_1,\mu) \in \mathrm{Med}(T_1,a_1) \text{ und } (t_2,\mu) \in \mathrm{Med}(T_2,a_2) \implies$$
$$(f(t_1,t_2),\mu) \in \mathrm{Med}(f(T_1,T_2),a_1+a_2) \ .$$

Beweis: $f(t_1(x), t_2(x)) > f(T_1(|x|), T_2(|x|))$ impliziert $t_1(x) > T_1(|x|)$ oder $t_2(x) > T_2(|x|)$. Daraus folgt für alle $l \in \mathbb{N}$

$$\begin{aligned} &|\{x \in X_\mu^{\leq l} \mid f(t_1(x), t_2(x)) > f(T_1(|x|), T_2(|x|))\}| \\ &\qquad \leq \quad |\{x \in X_\mu^{\leq l} \mid t_1(x) > T_1(|x|)\}| \\ &\qquad \quad + \qquad |\{x \in X_\mu^{\leq l} \mid t_2(x) > T_2(|x|)\}| \\ &\qquad \leq \quad a_1(l) + a_2(l) \end{aligned}$$

∎

Akzeptiert man also die Verdoppelung der Häufigkeit, so ist der Median praktisch gegenüber allen Operationen wie etwa linearen Abbildungen, Addition, Multiplikation und Potenzierung abgeschlossen. Hierzu siehe auch Tabelle 9.1. Um ein Gefühl für die Bedeutung der Häufigkeit zu bekommen, ziehen wir folgendes Beispiel heran.

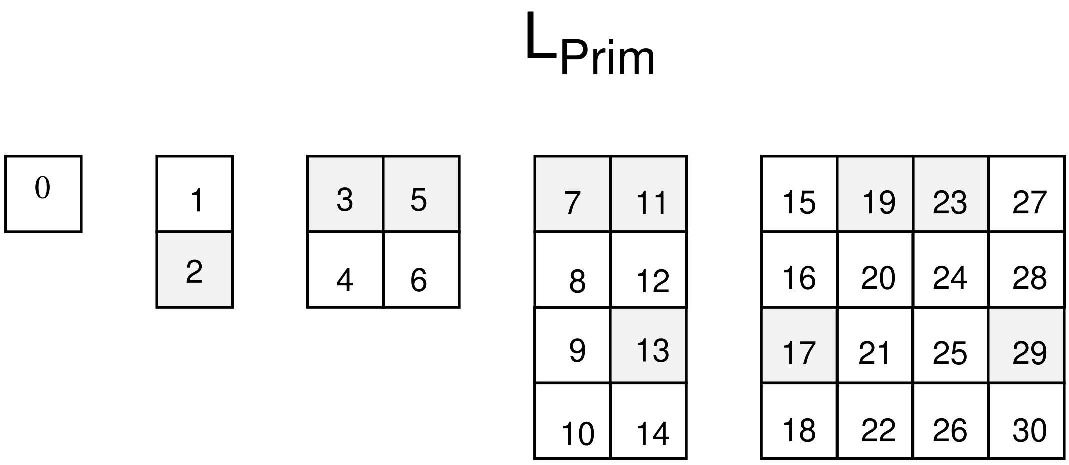

Abbildung 2.6: Die Menge der Primzahlen L_{Prim} im Bereich $[0; 30]$.

Beispiel:
Für eine Sprache $L \subseteq \Sigma^*$ und eine Funktion $a : \mathbb{N} \to \mathbb{N}$ ist der Term (man beachte $\text{rank}_{\text{uni}}^{\text{inj}} = \text{ord}$)

$$(\chi_L, \text{rank}_{\text{uni}}^{\text{inj}}) \ \in \ \text{Med}(0, a)$$

äquivalent zu der Aussage

$$\forall x \quad |L^{\leq x}| \ \leq \ a(\text{ord}(x)) \ .$$

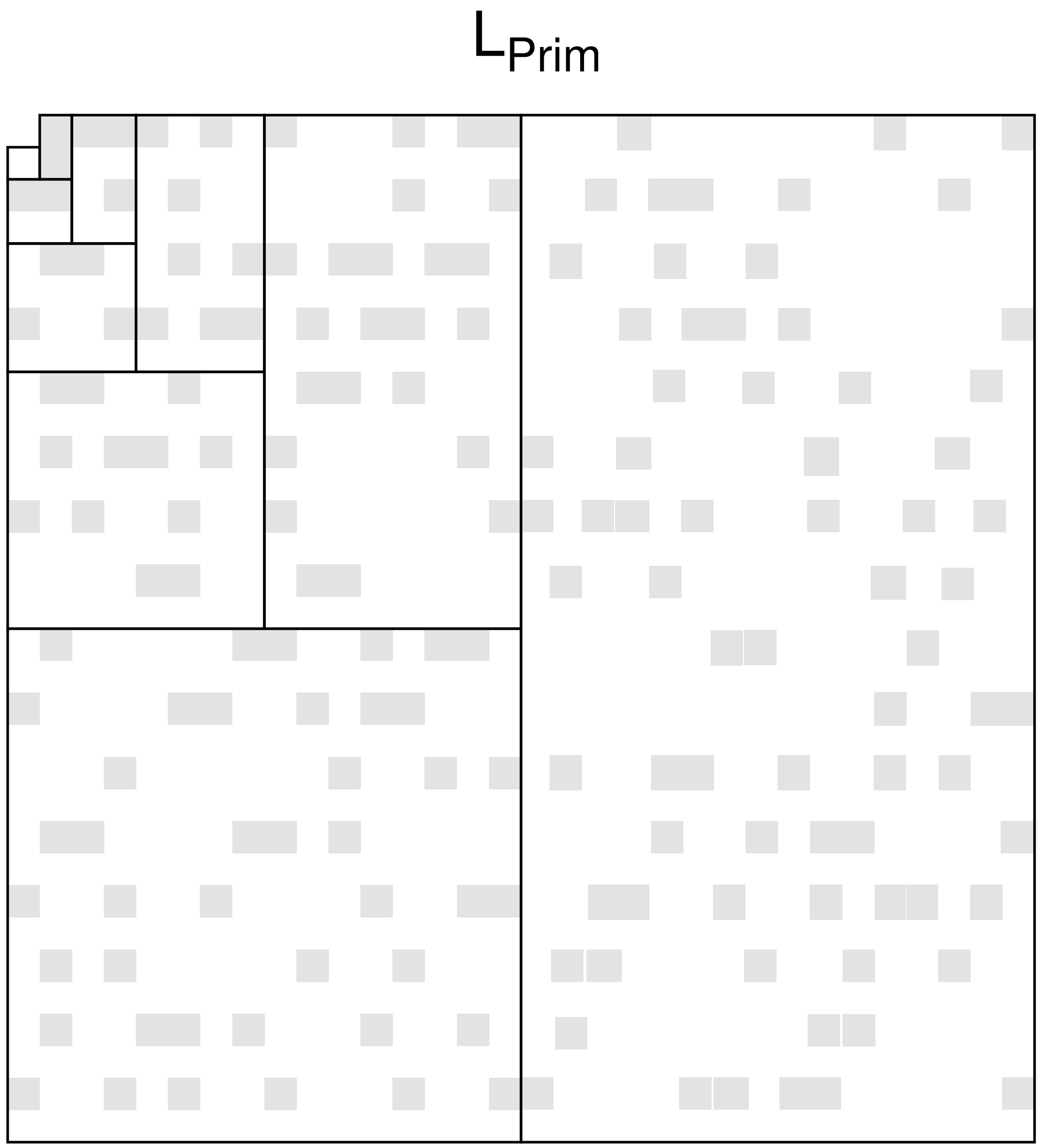

Abbildung 2.7: Die Menge der Primzahlen L_{Prim} im Bereich $[0;510]$. Die Zahlen in den Rechtecken sind wie in Abbildung 2.6 angeordnet.

Betrachten wir zuerst die Sprachen $\boldsymbol{L}_{\textbf{Prim}}$ der binären Zeichenketten, deren Ordnungszahl eine **Primzahl** darstellt.

$$\boldsymbol{L}_{\textbf{Prim}} \quad := \quad \{x \mid \operatorname{ord}(x) \text{ ist Primzahl}\} \; .$$

Der Satz von Tschebyscheff besagt nun, daß für die Anzahl $\boldsymbol{\pi(n)}$ der Primzahlen unter der Schranke n zwei positive Konstanten a, A existieren, so daß für alle $n > 2$ gilt

$$a < \frac{\pi(n) \cdot \ln n}{n} < A .$$

Daraus folgt sofort

$$\left| L_{\mathrm{Prim}}^{\leq x} \right| \in \Theta\left(\frac{\mathrm{ord}(x)}{\log(\mathrm{ord}(x))} \right) .$$

Das heißt, der Anteil der Primzahl-Zeichenketten in Σ^* wird durch eine Häufigkeitsfunktion $a \in \Theta\left(\frac{\mathcal{N}}{\log \mathcal{N}}\right)$ bestimmt.

Wenn man nun die Abbildungen 2.6 und 2.7 betrachtet, bemerkt man, daß diese Anzahl doch recht beträchtlich ist. Hier ist jedes Element $x \in L_{\mathrm{Prim}}$ als Kästchen dargestellt. Die von oben links nach unten rechts größer werdenden Rechtecke umrahmen die Mengen $\{\lambda\}$, $\{0,1\}$, $\{0,1\}^2$, usw.

Die Häufigkeit der Primzahlen erweist sich als sehr hilfreich, wenn man Primzahlgeneratoren zum Beispiel für kryptograpische Anwendungen konstruieren muß. Um nämlich eine Primzahl mit n Ziffern zu erhalten, muß man im Erwartungswert nur $O(n)$ Zufallszahlen aus Σ^n würfeln und auf ihre Primzahl-Eigenschaft testen.

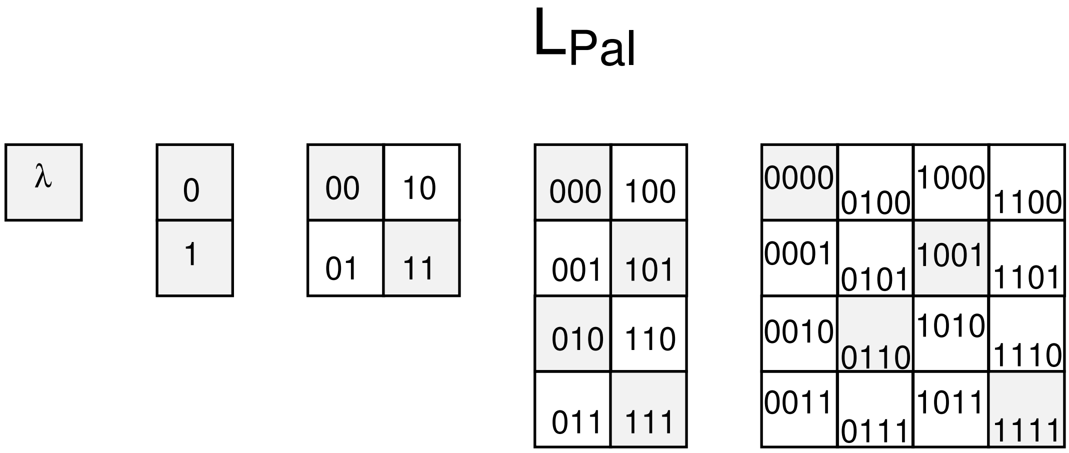

Abbildung 2.8: Die Sprache der binären Palindrome L_{Pal} mit höchstens vier Zeichen.

Für unsere Palindrom-Sprache $\boldsymbol{L_{\mathrm{Pal}}}$ gilt dagegen

$$\left| L_{\mathrm{Pal}}^{\leq x} \right| \in \Theta\left(\sqrt{\mathrm{ord}(x)} \right) .$$

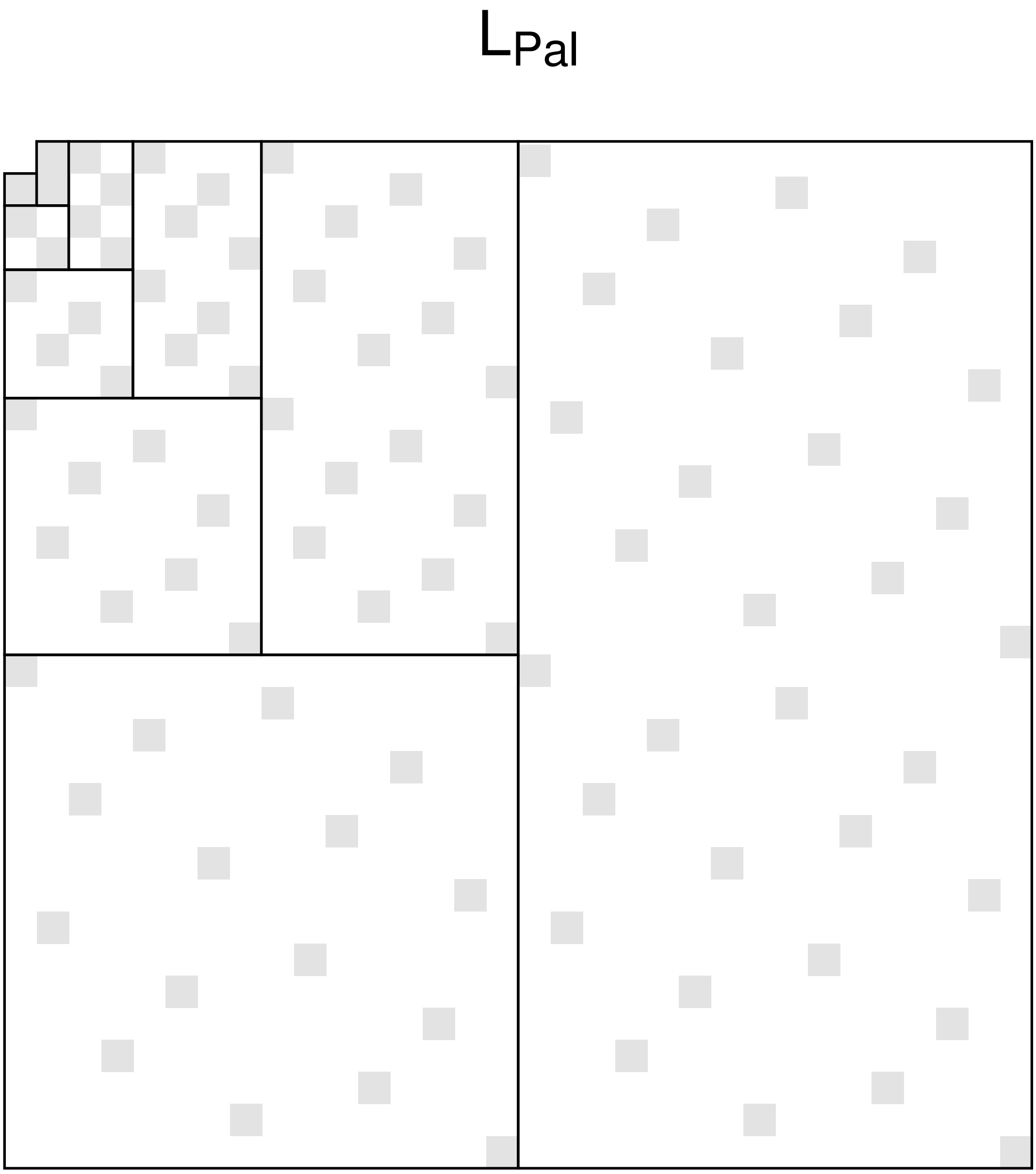

Abbildung 2.9: Die Sprache der binären Palindrome L_{Pal} mit höchstens 10 Zeichen. Die Darstellungsmethode entspricht der in Abbildung 2.8.

Die Häufigkeitsfunktion a im Median-Maß kann daher mit $a \in \Theta(\sqrt{\mathcal{N}})$ bestimmt werden. Zwar kann man ohne Schwierigkeiten unzählige Palindrome finden, und ihre Zahl von $O(2^{n/2})$ in einem $2^{n/2} \times 2^{n/2}$-Feld ist durchaus beträchtlich. Betrachtet man aber die Abbildungen 2.8 bzw. 2.9 und wählt ein Feld zufällig aus, so erhält man mit hoher Wahrscheinlichkeit kein schwarzes Feld, das ein Palindrom markiert[1].

Diese Überlegungen, dann aber bezogen auf Algorithmen-Laufzeiten, werden wir in Kapitel 5 erneut aufgreifen.

Das Medianmaß läßt, wie eingangs angesprochen, interessante Einblicke in das Av- und Eav-Maß zu. Wir untersuchen nun folgende Fragestellung allgemeiner: Wenn eine Funktion Eav (T_1) -beschränkt ist bezüglich einer Verteilung μ, wie groß kann dann der Anteil höherer Laufzeiten sein? Eine Teilantwort hinsichtlich der uniformen Verteilung haben wir im ersten Beispiel dieses Abschnitts schon vorweggenommen. Betrachten wir nun im folgenden wieder für μ die korrespondierende Rangfunktion $\rho := \mathrm{rank}_\mu$.

Wir wollen nun also mit einer Funktion a die maximale Häufigkeit von Funktionswerten $t(x) \geq T_2(|x|)$ (für $T_2 \geq T_1$) abschätzen, wenn ein Paar (t, ρ) Eav (T_1) - oder Av (T_1) -beschränkt ist. t sei nicht bekannt und so setzen wir ohne die Häufigkeit zu reduzieren voraus, daß $t(x)$ nur die Werte 0 oder $T_2(|x|) + 1$ annimmt (unter der Voraussetzung T_2 nimmt monoton zu).

Betrachten wir einen Häufigskeitwert $a(l)$, dann ist obige Fragestellung äquivalent zu: Wie mächtig kann eine Teilmenge X der l wahrscheinlichsten Eingaben $X_\rho^{\leq l}$ maximal werden, ohne die Bedingung $\sum_{x \in X} f(|x|) \leq l$ zu verletzen.

Für $f(|x|)$ wählt man den Term $\frac{T_2(|x|)+1}{T_1(|x|)}$, wenn das Eav-Maß untersucht wird; für das Av-Maß $\frac{T_1^{[-1]}(T_2(|x|)+1)}{|x|}$. Die Häufigkeitsfunktion $\boldsymbol{b_\rho^f}$ liefert somit eine obere Schranke für die gesuchte Häufigkeit.

Definition 10 *Für eine Funktion* $f : \mathbb{N} \to \mathbb{N}$ *und eine Rangfunktion* ρ *definieren wir*

$$\boldsymbol{b_\rho^f(l)} \quad := \quad \max\left\{ m \;\middle|\; \exists X \subseteq X_\rho^{\leq l} \text{ mit } |X| = m \text{ und } \sum_{x \in X} f(|x|) \leq l \right\}$$

Diese Schranke ist aber noch nicht optimal. Schließlich benutzen wir für ein festes l nur die Aussage

$$\sum_{x \in X} f(|x|) \leq l \;.$$

Dabei gilt für jedes $l' \leq \max(\rho(X))$

$$\sum_{x \in X \cap X_\rho^{\leq l'}} f(|x|) \leq l' \;,$$

[1] Der Leser kann dieses Experiment durchführen, indem er mehrmals einen Bleistift auf die Abbildung fallen läßt.

da, wenn man für f den geeigneten Term substituiert, sich die entsprechende Eav- oder Av-Beschränkung ergibt. Die Häufigkeitsfunktion $\boldsymbol{a_\rho^f}$ berücksichtigt diesen Umstand und ergibt so eine bessere Schranke:

Definition 11 *Für eine Funktion $f : \mathbb{N} \to \mathbb{N}$ und eine Rangfunktion ρ definieren wir*

$$\begin{aligned} \boldsymbol{a_\rho^f(l)} \quad := \quad & \max\{m \mid \exists X \subseteq X_\rho^{\leq l} \text{ mit } |X| = m \text{ und} \\ & \forall I \subseteq X \sum_{x \in I \cap X_\rho^{\leq \max(\rho(I))}} f(|x|) \leq \max(\rho(I))\} \ . \end{aligned}$$

Der Spezialfall $I = X$ ergibt die Bedingung der Häufigkeitsfunktion b_ρ^f. Somit gilt

Proposition 4 *Für alle Wahrscheinlichkeitsverteilungen ρ und Funktionen $f : \mathbb{N} \to \mathbb{N}$ gilt*

$$a_\rho^f \ \leq \ b_\rho^f \ .$$

Mit der Häufigkeitsfunktion a_ρ^f gelingt uns somit eine Beantwortung der eingangs gestellten Frage, denn es gilt

Theorem 1 *Für Zeitschranken T_1 und T_2 gilt*

$$(t, \rho) \in \text{Eav}\,(T_1) \implies (t, \rho) \in \text{Med}\left(T_2, a_\rho^{(T_2+1)/T_1}\right) \ .$$

Beweis: Sei $f \ := \ (T_2 + 1)/T_1$. Angenommen es gelte $(t, \rho) \notin \text{Med}\left(T_2, a_\rho^{(T_2+1)/T_1}\right)$.

Dann existieren ein $l \in \mathbb{N}$ und eine Menge $X \subseteq X_\rho^{\leq l}$ mit Mächtigkeit $|X| > a_\rho^f(l)$, so daß gilt $\forall x \in X \quad t(x) > T_2(|x|)$.

Da $|X| > a_\rho^f(l)$, gibt es aber eine Teilmenge $I \subseteq X$, so daß gilt

$$\sum_{x \in I \cap X_\rho^{\leq \max(\rho(I))}} f(|x|) \ > \ \max(\rho(I)) \ .$$

Sei nun $l' \ := \ \max(\rho(I))$, dann gilt

$$\sum_{x \in X_\rho^{\leq l'}} \frac{t(x)}{T_1(|x|)} \quad \geq \quad \sum_{x \in I \cap X_\rho^{\leq l'}} \frac{t(x)}{T_1(|x|)} \quad \geq \quad \sum_{x \in X_\rho^{\leq l'}} \frac{T_2(|x|) + 1}{T_1(|x|)} \quad \geq \quad l' \ .$$

Damit ist $(t, \rho) \notin \text{Eav}\,(T_1)$. ∎

Für das Av-Maß erhält man eine ähnliche Beziehung.

Theorem 2 *Für $T_1^{[-1]}(T_2+1)/\mathcal{N}$ monoton zunehmend gilt*

$$(t,\rho) \in \mathrm{Av}(T_1) \implies (t,\rho) \in \mathrm{Med}\left(T_2, a_\rho^{T_1^{[-1]}(T_2+1)/\mathcal{N}}\right) .$$

Beweis: Dieser Beweis ist analog zu dem in Theorem 1. Nur ist der Term $(T_2+1)/T_1$ mit $T_1^{[-1]}(T_2+1)/\mathcal{N}$ zu ersetzen. ∎

Bedingt durch die Kombination aus Maximumsfunktion und Quantisierung über Teilmengen ist der Term a_ρ^f für Auswertungen relativ unhandlich. Gute Ergebnisse erhält man auch mit der Häufigkeitsfunktion $\boldsymbol{c_\rho^f}$.

Definition 12 *Für eine Funktion $f : \mathbb{N} \to \mathbb{N}$ und eine Rangfunktion ρ definieren wir*

$$\boldsymbol{c_\rho^f(l)} \quad := \quad \sum_n \min\left(\frac{\max(\rho(X_\rho^{\leq l} \cap \Sigma^n))}{f(n)}, |X_\rho^{\leq l} \cap \Sigma^n|\right) .$$

Hierbei betrachten wir jede Eingabelänge n einzeln. So ergeben sich jeweils zwei obere Schranken: Zum einen die Anzahl aller dort betrachteten Zeichenketten $|X_\rho^{\leq l} \cap \Sigma^n|$. Andererseits die maximale Anzahl von Zeichenketten, um die *Average*-Schranke allein durch Betrachtung von Eingaben dieser Länge zu sprengen.

$c_{\rho,f}$ ist wiederum eine obere Schranke für die Funktion $a_{\rho,f}$.

Lemma 9 *Für eine Funktion $f : \mathbb{N} \to \mathbb{N}$ und eine Rangfunktion ρ gilt*

$$a_\rho^f \ \leq \ c_\rho^f .$$

Beweis: Für gegebenes f, ρ und $l \in \mathbb{N}$ betrachten wir eine Menge $X \subseteq X_\rho^{\leq l}$ mit $|X| = a_\rho^f(l)$ und der Eigenschaft

$$\forall I \subseteq X \quad \sum_{x \in I \cap X_\rho^{\leq \max(\rho(I))}} f(|x|) \leq \max(\rho(I)) .$$

Betrachten wir nun eine feste Eingabelänge n und sei $I \ := \ X \cap \Sigma^n$. Dann gilt

$$|I \cap X_\rho^{\leq l}| \ \leq \ |X_\rho^{\leq l} \cap \Sigma^n| ,$$

zusätzlich gilt aber auch

$$|I \cap X_\rho^{\leq l}| \cdot f(n) \ \leq \ \max(\rho(X_\rho^{\leq l} \cap \Sigma^n)) .$$

Somit gilt

$$|X \cap \Sigma^n| \;\le\; \min\left(\frac{\max(\rho(X_\rho^{\le l} \cap \Sigma^n))}{f(n)}, |X_\rho^{\le l} \cap \Sigma^n|\right) \;.$$

und damit $|X| \le c_\rho^f(l)$. ∎

Alle drei Häufigkeitsfunktionen a_ρ^f, b_ρ^f und c_ρ^f beschränken also die Anzahl von Funktionswerten größer als T von *average*-beschränkten Paaren (t,ρ), denn es gilt nachstehendes Korollar.

Korollar 6 *Für Zeitschranken T_1 und T_2 gilt*

$$\begin{aligned}(t,\rho) \in \mathrm{Eav}(T_1) \;&\Longrightarrow\; (t,\rho) \in \mathrm{Med}\left(T_2, b_\rho^{(T_2+1)/T_1}\right) \;,\\ (t,\rho) \in \mathrm{Eav}(T_1) \;&\Longrightarrow\; (t,\rho) \in \mathrm{Med}\left(T_2, c_\rho^{\rho,(T_2+1)/T_1}\right) \;.\end{aligned}$$

Für $T_1^{[-1]}(T_2+1)/\mathcal{N}$ monoton zunehmend gilt

$$\begin{aligned}(t,\rho) \in \mathrm{Av}(T_1) \;&\Longrightarrow\; (t,\rho) \in \mathrm{Med}\left(T_2, b_\rho^{T_1^{[-1]}(T_2+1)/\mathcal{N}}\right) \;,\\ (t,\rho) \in \mathrm{Av}(T_1) \;&\Longrightarrow\; (t,\rho) \in \mathrm{Med}\left(T_2, c_\rho^{T_1^{[-1]}(T_2+1)/\mathcal{N}}\right) \;.\end{aligned}$$

Für Wahrscheinlichkeitsverteilungen μ mit ausgedünnter Menge an positiv gewichteten Zeichenketten, zum Beispiel $|\{x \in X_\mu^{\le l} \mid \mu(x) > 0\}| \le \mathrm{PLog}(l)$, ergeben sich schon mit b_ρ^f brauchbare Aussagen.

Ist diese Menge aber sehr dicht, wie zum Beispiel bei der uniformen Verteilung, dann ist die Häufigkeitsfunktion c_ρ^f vorzuziehen. Dort gilt folgendes.

Lemma 10 *Für eine Funktion f und die uniforme Wahrscheinlichkeitsverteilung μ_{uni} über dem Alphabet Σ gilt*

$$c^f_{\mathrm{rank}_{\mathrm{uni}}}(l) \;\le\; \frac{|\Sigma|}{|\Sigma|-1} \cdot \sum_{n=1}^{\log l - 1} \frac{|\Sigma|^n}{f(n)} \;.$$

Beweis: folgt direkt aus Definition 12. ∎

Für viele Verteilungen kann man so eine einfache Beziehung zwischen Eav, Av und Med erhalten. Für die uniforme erhält man aufgrund des letzten Lemmas folgendes Korollar.

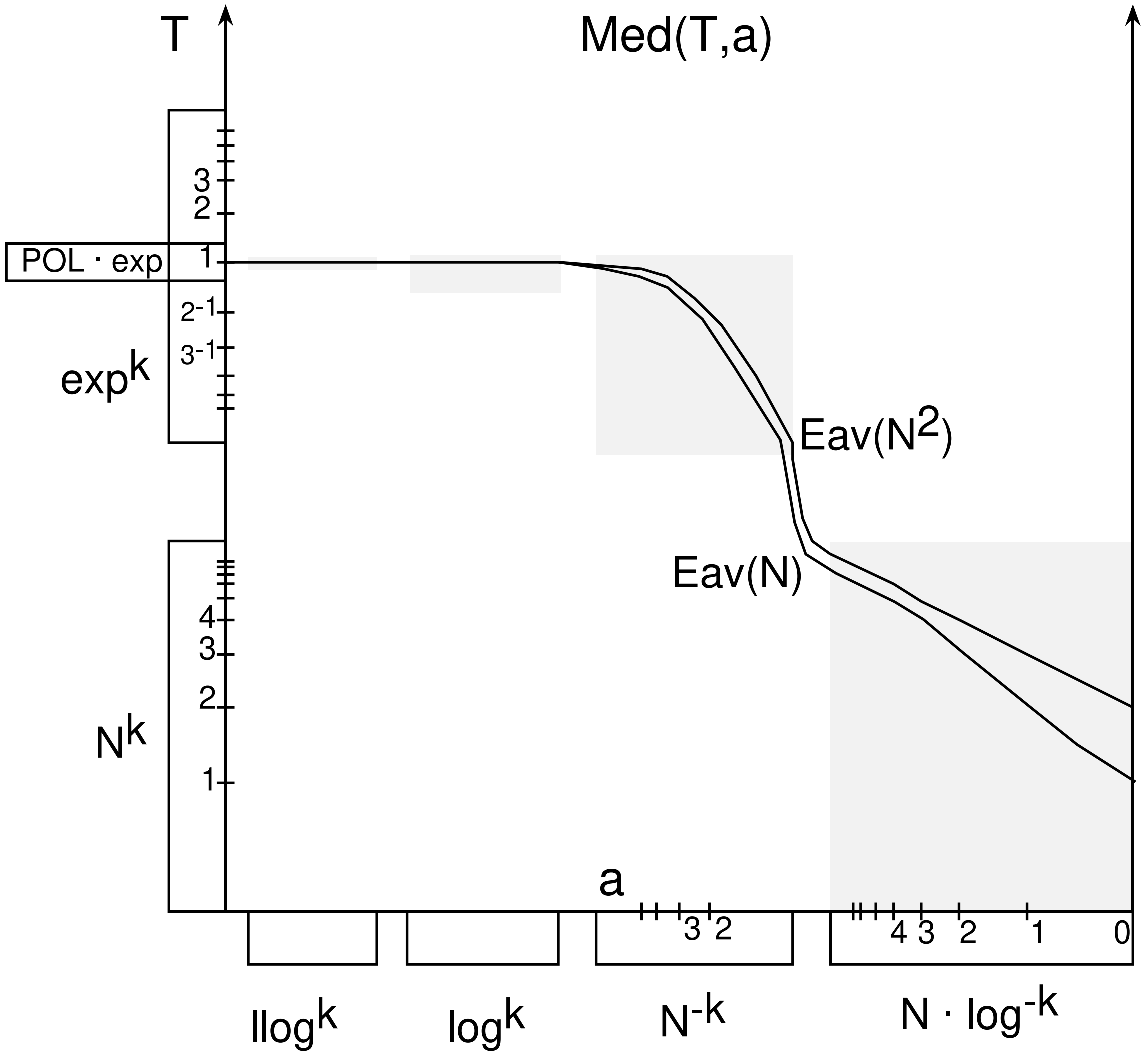

Abbildung 2.10: Jeder Punkt im Diagramm korrespondiert mit einer Klasse $\mathrm{Med}\,(T,a)$. Ist ein Punkt (a,T) unterhalb der mit $\mathrm{Eav}\,(\mathcal{N}^2)$ beschrifteten Kurve, so gilt für alle Funktionen $f:\Sigma^*\to\mathbb{N}$ die Implikation $(f,\mu_{\mathrm{uni}})\in\mathrm{Eav}\,(\mathcal{N}^2)\ \Rightarrow\ (f,\mu_{\mathrm{uni}})\in\mathrm{Med}\,(T,a)$.

Korollar 7 *Für* $T_2\geq T_1$ *und die uniforme Rangfunktion* $\mathrm{rank}_{\mathrm{uni}}$ *über dem Alphabet* Σ *gilt*

$$(t,\mu_{\mathrm{uni}})\in\mathrm{Eav}(T_1)\quad\Longrightarrow\quad(t,\mu_{\mathrm{uni}})\in\mathrm{Med}\left(T_2,\frac{|\Sigma|}{|\Sigma|-1}\cdot\sum_{n=1}^{\log\mathcal{N}-1}\frac{|\Sigma|^n\cdot T_1(n)}{T_2(n)}\right)\ ,$$

$$(t,\mu_{\mathrm{uni}})\in\mathrm{Av}(T_1)\quad\Longrightarrow\quad(t,\mu_{\mathrm{uni}})\in\mathrm{Med}\left(T_2,\frac{|\Sigma|}{|\Sigma|-1}\cdot\sum_{n=1}^{\log\mathcal{N}-1}\frac{|\Sigma|^n\cdot\exp n\cdot n}{T_1^{[-1]}(T_2(n))}\right)\ .$$

Eine erschöpfende Übersicht für binäres Eingabealphabet liefert Tabelle 2.3. Abbildungen 2.10 und 2.11 zeigen diesen Sachverhalt noch einmal graphisch.

Jeder Punkt (a,T) der Abbildung 2.10 steht für eine Klasse $\mathrm{Med}(T,a)$. Für jeden Punkt (a,T) unterhalb der oberen Linie gilt: Ist das Paar (f,μ_{uni}) quadratisch im *expected average* ($\mathrm{Eav}\,(\mathcal{N}^2)$) beschränkt, ist es auch $\mathrm{Med}(T,a)$ beschränkt.

Die entsprechende Abbildung 2.11 für das Av-Maß zeigt, daß das Av-Maß gerade im Bereich großer Laufzeiten wesentlich größere Häufigkeiten als im Eav-Maß zuläßt. In den Abbildungen 2.10 und 2.11 deuten die schraffierten Flächen die Bereiche an, durch die die Kurven von Eav(POL) und Av(POL) verlaufen.

Für die *average-case*-Analyse eines Algorithmus liefert also eine einzige Aussage $(t,\rho)\notin \mathrm{Med}\,(T,a)$ eine entsprechende untere Schranke im Av- und Eav-Maß.

Gehen wir jetzt den umgekehrten Weg, indem wir annehmen, wir hätten für eine gewichtete Funktion (t,μ) eine große Vielfalt verschiedener Medianabschätzungen bereits gewonnen ($(t,\mu)\in\mathrm{Med}\,(T_i,a_i)$). Kann man hieraus eine obere Schranke für das Av- oder das Eav-Maß ableiten?

Beispiel:
Betrachten wir einen Algorithmus, der auf Eingabe x entweder quadratische Laufzeit $|x|^2$ oder exponentielle Laufzeit $2^{|x|}$ benötigt. Der Anteil der exponentiellen Werte ist bezüglich der uniformen Wahrscheinlichkeitsverteilung nicht allzu hoch. Es gelte

$$\frac{|\{x\in X^{\leq l}_{\mu_{\mathrm{uni}}} \mid t(x)=\exp|x|\}|}{l} \leq \frac{1}{\sqrt{l}}\ .$$

Somit lassen sich mit dem Median-Maß die nachstehenden Aussagen treffen:

$$(t,\mu_{\mathrm{uni}})\in\mathrm{Med}(\exp,0)\ ,$$

$$(t,\mu_{\mathrm{uni}})\in\mathrm{Med}(\mathcal{N},\sqrt{\mathcal{N}})\ ,$$

$$(t,\mu_{\mathrm{uni}})\in\mathrm{Med}(0,\mathcal{N})\ .$$

Verwendet man nur diese drei Aussagen, so können bereits diese oberen Durchschnittsschranken bestimmt werden:

$$(t,\mu_{\mathrm{uni}})\in\mathrm{Eav}\left(\sqrt{\exp}\right)\ ,$$

$$(t,\mu_{\mathrm{uni}})\in\mathrm{Av}\,(\mathcal{N}^2)\ .$$

T_1	T_2		$(t, \mu_{\text{uni}}) \in \text{Eav}(T_1)$	$(T_2, \mu_{\text{uni}}) \in \text{Eav}(T_1)$
			$\Longrightarrow \quad (t, \mu_{\text{uni}}) \in \text{Med}(T_2, a)$	
			a	a
$\mathcal{N}$	$\mathcal{N}^{k+1}$	$k > 0$	$\frac{\mathcal{N}}{\log^k \mathcal{N}}$	$\frac{\mathcal{N}}{\log^k \mathcal{N}}$
	$\exp^k$	$0 < k < 1$	$\mathcal{N}^{1-k} \cdot \text{LOG}$	$\mathcal{N}^{1-k} \cdot \text{LOG}$
	$\mathcal{N}^k \cdot \exp$	$k < 1$	LOG^{2-k}	LOG^{2-k}
		$k = 1$	LLOG	LLOG
		$k > 1$	$O(1)$	$O(1)$
$\mathcal{N}^m$ $(m > 1)$	$\mathcal{N}^{m+k}$	$k > 0$	$\frac{\mathcal{N}}{\log^k \mathcal{N}}$	$\frac{\mathcal{N}}{\log^{\frac{k}{m}} \mathcal{N}}$
	$\exp^k$	$k < 1,\ k < m$	$\mathcal{N}^{1-k} \cdot \text{LOG}^m$	$\mathcal{N}^{1-\frac{k}{m}} \cdot \text{LOG}$
		$k = 1,\ k < m$	LOG^{m+1}	
		$k > 1,\ k < m$	$O(1)$	
		$k > 1,\ k = m$		LOG^2
		$k > m$		$O(1)$
	$\mathcal{N}^l \cdot \exp^k$	$k < 1,\ k < m$	$\text{LOG}^{m-l} \cdot \mathcal{N}^{1-k}$	$\text{LOG}^{1-\frac{l}{m}} \cdot \mathcal{N}^{1-\frac{k}{m}}$
		$k = 1,\ l < m,$	LOG^{m-l+1}	
		$k = 1,\ l = m$	LLOG	
		$k = 1,\ m < l$	$O(1)$	
		$1 < k < m$		
		$l < k = m$		$\text{LOG}^{2-\frac{l}{m}}$
		$k = m = l$		LLOG
		$k = m < l$		$O(1)$
		$k > m$		
$\exp^m$	$\mathcal{N}^l \cdot \exp^k$	$k = m,\ l < 1$	LOG^{2-k}	$\Theta(\mathcal{N})$
		$k = m,\ l = 1$	LLOG	
		$k = m,\ l > 1$	$O(1)$	
		$k > m$		
	$\exp(\exp^k)$	$0 < k < 1$		$\text{LOG} \cdot \mathcal{N}^{1-k}$
		$k = 1$		LOG^2
		$k > 1$		$O(1)$

Tabelle 2.3: Für ausgewählte *average*-Beschränkungen ergeben sich diese Beschränkungen im Median-Maß bezüglich uniformer Wahrscheinlichkeitsverteilungen.

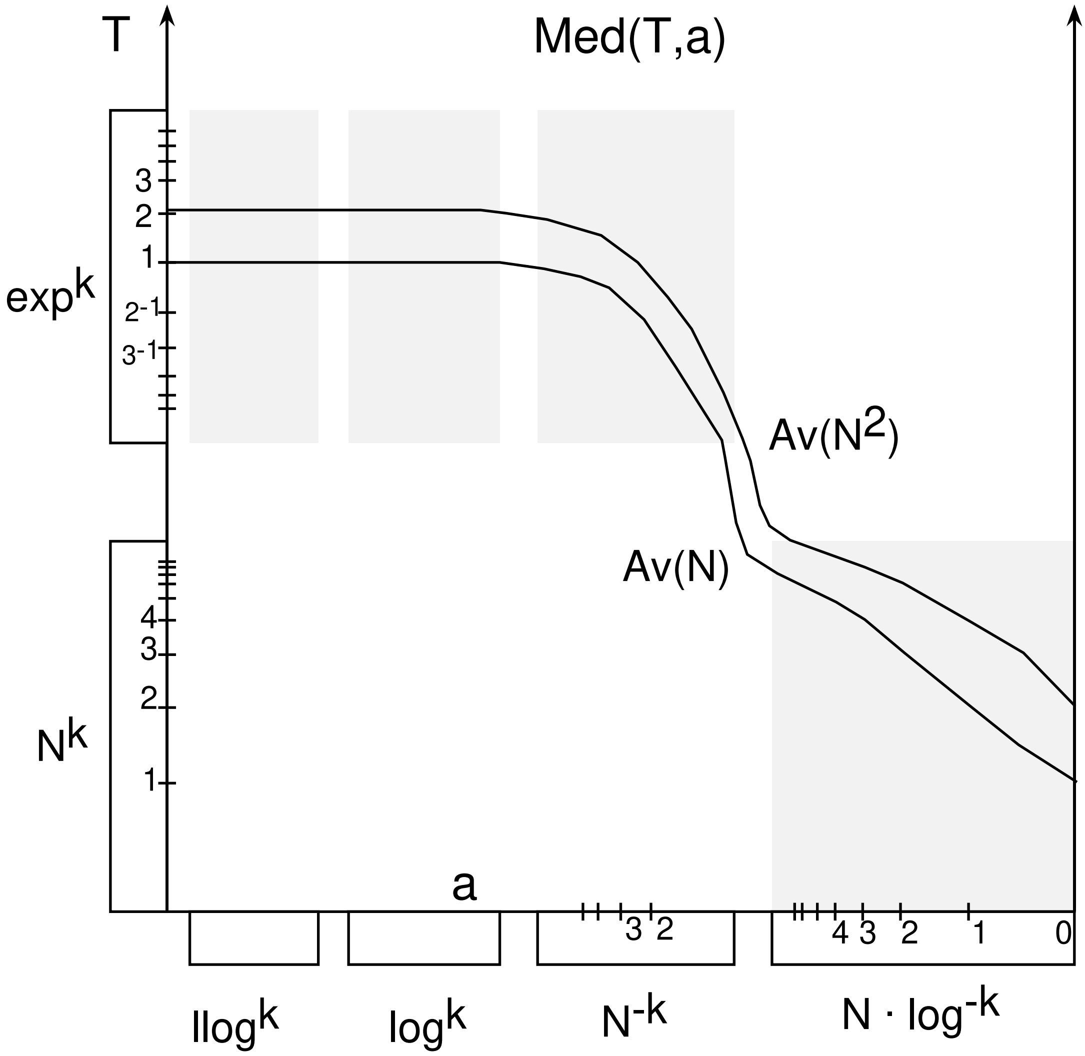

Abbildung 2.11: In diesem Diagramm werden Av-Maß und Med-Maß bezüglich uniformer Wahrscheinlichkeitsverteilungen in Beziehung gebracht. Ein Punkt (a, T) im Diagramm steht für eine Klasse $\mathrm{Med}(T, a)$. Ist ein Punkt unterhalb der Kurve $\mathrm{Av}(\mathcal{N}^2)$, so gilt $(f, \mu_{\mathrm{uni}}) \in \mathrm{Av}(\mathcal{N}^2) \Rightarrow (f, \mu_{\mathrm{uni}}) \in \mathrm{Med}(T, a)$.

Um diese Überlegungen für allgemeine Rangfunktionen ρ fortführen zu können, definieren wir die Funktion $\boldsymbol{\nu_\rho} : \mathbb{N} \to \mathbb{N}$, wobei $\nu_\rho(n)$ den größten endlichen Rang korrespondierend zu einer Eingabe aus $\Sigma^{\leq n}$ bezeichnet.

Definition 13 *Für eine Rangfunktion ρ werde $\nu_\rho : \mathbb{N} \to \mathbb{N}$ definiert als*

$$\boldsymbol{\nu_\rho(n)} \;:=\; \max_{|x| \leq n \text{ und } \rho(x) \neq \infty} \rho(x) \ .$$

Mit Hilfe dieser Funktion können, sofern man passende Median-Abschätzungen $(t,\rho) \in \mathrm{Med}\,(T_i, a_i)$ kennt, obere Av- und Eav-Schranken bewiesen werden.

Theorem 3 *Für Häufigkeitsfunktionen $a_0 = \mathcal{N}$, $a_p = 0$, $a_i/\mathcal{N}$ monoton abnehmend und monoton zunehmende Funktionen $T_0 = 0$, $T_1 \le T_2 \le \ldots \le T_p$ gelte*

$$(t,\rho) \in \mathrm{Med}(T_i, a_i), \quad \textit{für alle } i \in \{0,\ldots,p\}\,.$$

Dann folgt

$$\begin{aligned}(t,\rho) &\in \mathrm{Eav}\Big(p\cdot \max\nolimits_{i<p}\ \tfrac{T_{i+1}\cdot a_i(\nu_\rho)}{\nu_\rho}\Big)\ ,\\ (t,\rho) &\in \mathrm{Av}\bigg(\max\nolimits_{i<p}\ T_{i+1}\circ\Big[\tfrac{\nu_\rho}{p\cdot a_i(\nu_\rho)}\Big]^{[-1]}\bigg)\ .\end{aligned}$$

Beweis: Gegeben sei ein beliebiges $l \in \mathbb{N}$. Seien die Eingabe-Mengen A_i definiert als

$$A_i \ := \ \{x \in X_\rho^{\le l} \mid T_i(|x|) < t(x) \le T_{i+1}(|x|)\}\ .$$

Die Größe dieser Mengen ist beschränkt durch $|A_i| \le a_i(l)$, da $(t,\rho) \in \mathrm{Med}(T_i, a_i)$.

1. $(t,\rho) \in \mathrm{Eav}\Big(p\cdot \max_{i<p}\ \frac{T_{i+1}\cdot a_i(\nu_\rho)}{\nu_\rho}\Big)$.

   Sei $\tilde T \ := \ p\cdot\max_{i<p}\ \frac{T_{i+1}\cdot a_i(\nu_\rho)}{\nu_\rho}$. Dann gilt
   $$\forall i<p \quad \forall x\in A_i \quad \tilde T(|x|) \ \ge\ \frac{p\cdot T_{i+1}(|x|)\cdot a_i(l)}{l}\ ,$$
   da $\mathcal{N}/a_i$ monoton zunehmend ist und $\nu_\rho(|x|) \ge l$ für $\rho(x) = l$. Somit gilt
   $$\begin{aligned}\sum_{\rho(x)\le l}\frac{t(x)}{\tilde T(|x|)} \ &= \ \sum_{0\le i<p}\sum_{x\in A_i}\frac{t(x)}{\tilde T(|x|)}\\ \le \ \sum_{0\le i<p}\sum_{x\in A_i}\frac{T_{i+1}(x)}{\tilde T(|x|)} \ &\le \ \sum_{0\le i<p}\sum_{x\in A_i}\frac{T_{i+1}(x)\cdot l}{p\cdot T_{i+1}(|x|)\cdot a_i(l)} \quad \le \quad l\ .\end{aligned}$$

2. $(t,\rho) \in \mathrm{Av}\bigg(\max_{i<p}\ T_{i+1}\circ\Big[\frac{\nu_\rho}{p\cdot a_i(\nu_\rho)}\Big]^{[-1]}\bigg)$. Sei $\tilde T \ := \ \max_{i<p}\ T_{i+1}\circ\Big[\frac{\nu_\rho}{p\cdot a_i(\nu_\rho)}\Big]^{[-1]}$.
   Dann gilt $\forall i<p \quad \forall x \in A_i$
   $$\tilde T^{[-1]}(T_{i+1}(|x|)) \ \le\ \frac{\nu_\rho(|x|)}{p\cdot a_i(\nu_\rho(|x|))}\cdot T_{i+1}^{[-1]}(T_{i+1}(|x|)) \ \le\ \frac{l\cdot|x|}{p\cdot a_i(l)}\ ,$$
   da $\mathcal{N}/a_i$ monoton zunehmend ist und $\nu_\rho(|x|) \ge l$ für $\rho(x) = l$. Somit gilt
   $$\begin{aligned}\sum_{\rho(x)\le l}\frac{\tilde T^{[-1]}(t(x))}{|x|} \ &= \ \sum_{0\le i<p}\sum_{x\in A_i}\frac{\tilde T^{[-1]}(t(x))}{|x|}\\ \le \ \sum_{0\le i<p}\sum_{x\in A_i}\frac{\tilde T^{[-1]}(T_{i+1}(x))}{|x|} \ &\le \ \sum_{0\le i<p}\sum_{x\in A_i}\frac{l\cdot|x|}{p\cdot a_i(l)\cdot|x|} \quad \le \quad l\ .\end{aligned}$$

■

Für den interessanten Spezialfall uniformer Rangfunktionen erhält man unter Berücksichtigung von $\nu_{\mathrm{rank}_{\mathrm{uni}}}(n) = \max(\mathrm{ord}(\Sigma^n))$ folgendes.

Korollar 8 *Sei $t \leq T_p$ und für Schranken $a_0 = \mathcal{N}$, $a_p = 0$, $a_i/\mathcal{N}$ monoton abnehmend und monoton zunehmenden Funktionen $T_1 \leq T_2 \leq \ldots \leq T_p$ gelte*

$$(t, \mu_{\mathrm{uni}}) \in \mathrm{Med}(T_i, a_i), \quad \textit{für alle } i \in \{1, \ldots, p\}.$$

Dann folgt für binäres Eingabealphabet

$$\begin{aligned} (t, \mu_{\mathrm{uni}}) &\in \mathrm{Eav}\Big(p \cdot \max\nolimits_{i<p}\ T_{i+1} \cdot \tfrac{a_i(2\cdot\exp)}{2\cdot\exp}\Big), \\ (t, \mu_{\mathrm{uni}}) &\in \mathrm{Av}\bigg(\max\nolimits_{i<p}\ T_{i+1} \circ \Big[\tfrac{2\cdot\exp}{p\cdot a_i(2\cdot\exp)}\Big]^{[-1]}\bigg). \end{aligned}$$

Betrachten wir hierzu Abbildung 2.12. Um zu zeigen, daß $(t, \rho) \in \mathrm{Av}(O(\mathcal{N}^2))$, genügt es eine Treppenfunktion zu konstruieren, die die skizzierte Linie niemals überschreitet. Jeder Kreis hat dabei als Koordinaten (T, a), die Zeit- und Häufigkeitsfunktion der oberen Schranke

$$(t, \mu_{\mathrm{uni}}) \in \mathrm{Med}(T, a).$$

Man sieht sofort, daß in dem skizzierten Beispiel für eine quadratische *average*-Abschätzung die beiden mittleren Median-Schranken überflüssig sind.

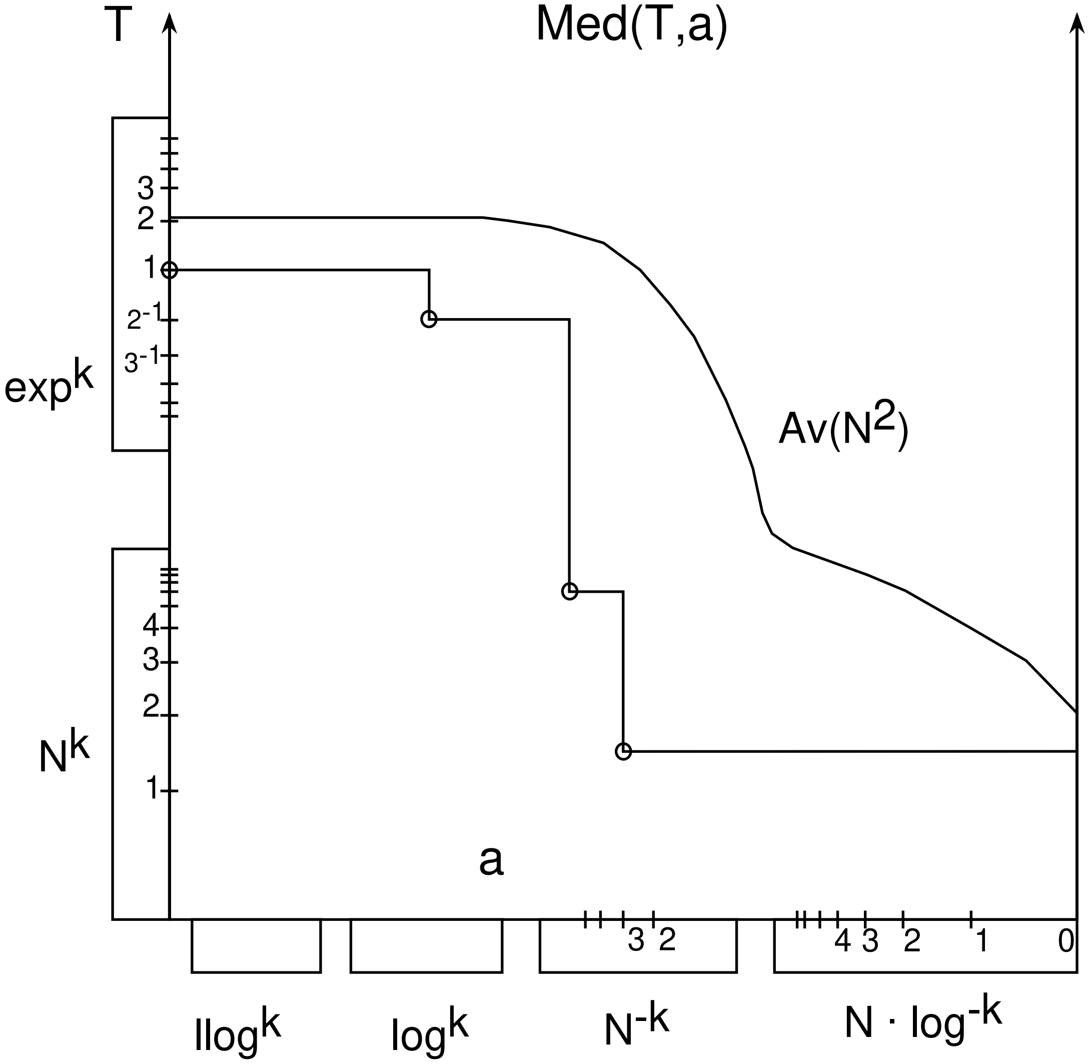

Abbildung 2.12: Die vier Kreise stehen für obere Schranken Med(T_i, a_i) . Verläuft die resultierende Treppenfunktion unterhalb der mit Av$(\mathcal{N}^2)$ beschrifteten Kurve, dann ist jedes Paar (f, μ_{uni}) mit dieser Median-Beschränkung auch quadratisch im *average* beschränkt.

Kapitel 3

Average-Komplexitätsklassen

3.1 AvTime und EavTime

Komplexitätsklassen sind Mengen von Problemen vergleichbarer Komplexität. Dabei betrachten wir in dieser Arbeit ausschließlich Entscheidungsprobleme , d.h. für eine fest vorgegebene Sprache L soll ein Algorithmus feststellen, ob $x \in L$ gilt. Die klassischen *worst-case*-Komplexitätsklassen, auf die wir uns in dieser Arbeit beziehen wollen, sind Mengen von Sprachen.

Wie wir bereits gesehen haben, ist im *average* (unabhängig von dem verwendeten Durchschnittsmaß) der Einfluß der Wahrscheinlichkeitsverteilung gravierend. So liegt es nahe, für *Average*-Komplexitätsklassen eine Sprache zusammen mit der zugrundegelegten Wahrscheinlichkeitsverteilung als das zu lösende Problem aufzufassen.

Dieses Paar aus Entscheidungsproblem und Wahrscheinlichkeitsverteilung nennen wir **gewichtetes Problem**. Sie dienen zur Definition der ***Average*-Komplexitätsklassen**.

Definition 14

$$\begin{aligned}
\mathbf{AvDisTime(T)} &:= \{(L,\mu) \mid \exists M \in DTM\colon\ L(M) = L \text{ und } (\mathrm{time}_M, \mu) \in \mathrm{Av}(T)\} \\
\mathbf{AvDis}\boldsymbol{P} &:= \bigcup_{T \in \mathrm{POL}} \mathrm{AvDisTime}(T)\ , \\
\mathbf{AvDisTime(T)} &:= \{(L,\mu) \mid \exists M \in DTM\colon\ L(M) = L \text{ und } (\mathrm{time}_M, \mu) \in \mathrm{Eav}(T)\} \\
\mathbf{EavDis}\boldsymbol{P} &:= \bigcup_{T \in \mathrm{POL}} \mathrm{EavDisTime}(T)\ .
\end{aligned}$$

So wie man $\mathcal{P}$ als Klasse effizient lösbarer *worst-case*-Probleme ansieht, stehen nun die Klassen $\mathrm{AvDis}\,\mathcal{P}$ und $\mathrm{EavDis}\,\mathcal{P}$ für die Klasse der im *average* effizient berechenbaren gewichteten Probleme.

Versucht man EavDisTime mit bislang untersuchten *average*-Komplexitätsklassen zu vergleichen, bietet sich eine Klasse an, die sich durch Verwendung des **lokal** gewichteten Erwartungswert ergibt. Diese Klasse beinhaltet genau dann ein Problem L, wenn ein Algorithmus M für alle Eingabelängen n in erwarteter Zeit T bzgl. der lokalen Wahrscheinlichkeitsverteilung das Problem löst. Das heißt, für alle n gilt

$$E_{[\mu]_n}(\mathrm{time}_M) \;\leq\; T(n)$$

für einen Akzeptor M von L.

Diese Klasse beinhaltet sicher weniger Probleme als EavDisTime, da die Betrachtung lokaler Wahrscheinlichkeitsverteilungen den schon im vorherigen Kapitel beobachteten ausgleichenden Effekt über verschiedene Eingabelängen hinweg nicht bieten kann.

Solch ein lokal T-erwartet beschränktes Problem ist immer in $\mathrm{EavDisTime}(T)$ enthalten. Damit stellt offensichtlich der lokale Erwartungswert härtere Anforderungen an das Zeitverhalten eines Algorithmus.

Dennoch gibt es Klassen von Verteilungen, für die beide Maße sich nicht unterscheiden. Dies gilt zum Beispiel für die uniformen Verteilungen μ_{uni} mit der Rangfunktion $\mathrm{rank}_{\mathrm{uni}}(x) \;:=\; |\Sigma^{\leq|x|}|$, aber auch für alle S-uniformen Verteilungen, solange S **füllig** ist, also für fast alle n nicht zu wenig Zeichenketten der Länge n besitzt.

Definition 15 *Eine Menge S aus Zeichenketten heiße* ***füllig****, falls*

$$\exists c \in \mathbb{R}^+ \qquad c \cdot |S^{=n}| \;\geq_{ae}\; |S^{<n}| \;.$$

Theorem 4 *Für eine S-uniforme Wahrscheinlichkeitsverteilung, wobei S füllig ist, gibt es ein $c > 0$, so daß für alle Zeitschranken $T \geq 4 \cdot \mathcal{N}/c$ gilt*

$$(L, \mu_S) \in \mathrm{EavDisTime}(T) \iff \begin{array}{l} \exists DTM\ M \ \ mit\ L(M) = L \quad und \\ \forall n \quad E_{[\mu_S]_n}(\mathrm{time}_M) \leq T \;. \end{array}$$

Beweis: „$\Longrightarrow$“:
Sei $L \in \mathrm{EavTime}(T, \{\mu_S\})$. Wir betrachten nun eine Turingmaschine M mit $L(M) = L$ und $(\mathrm{time}_M, \mu_S) \in \mathrm{Eav}\,(T)$. Somit gilt

$$\begin{aligned} \forall l \sum_{\mathrm{rank}_S(x) \leq l} \frac{\mathrm{time}_M(x)}{T(|x|)} \;\leq\; l \;\Longrightarrow\; \forall n \quad & \sum_{x \in S^{\leq n}} \frac{\mathrm{time}_M(x)}{T(|x|)} \;\leq\; \left|S^{\leq n}\right| \\ \Longrightarrow\; \exists c, n_0\, \forall n \geq n_0 \quad & \sum_{x \in S^{=n}} \frac{\mathrm{time}_M(x)}{|S^{=n}|} \;\leq\; c \cdot T(n) \;. \end{aligned}$$

Durch lineare Beschleunigung um den Faktor $2/c$ und Speicherung aller Ergebnisse auf Eingaben aus $\Sigma^{\leq n_0}$ im endlichen Gedächtnis kann man eine Turingmaschine M' mit $L(M') = L(M)$ erhalten, deren Laufzeit sich verringert auf

$$\mathrm{time}_{M'}(x) \ \leq \ \max\left\{\frac{c}{2}\cdot \mathrm{time}_M(x), (1+c)\cdot |x|\right\} \ .$$

Da T mindestens so stark wächst wie $4\cdot \mathcal{N}/c$, kann man nun direkt zeigen, daß

$$\forall n \qquad \sum_{|x|=n} \frac{\mathrm{time}_{M'}(x)}{|S^{=n}|} \ \leq \ T(n) \ ,$$

was gleichbedeutend ist mit M' ist erwartet T-zeitbeschränkt.

„$\Longleftarrow$“:
Sei M eine DTM, so daß $L(M) = L$ und $\forall n\ E_{[\mu_S]_n}(\mathrm{time}_M) \leq T$. Somit gilt

$$\begin{aligned}
\forall n \quad \sum_{x\in S^{=n}} \frac{\mathrm{time}_M(x)}{|S^{=n}|} \ \leq \ T(n) \quad & \Longleftrightarrow \quad \forall n \quad \sum_{x\in S^{=n}} \frac{\mathrm{time}_M(x)}{T(n)} \ \leq \ |S^{=n}| \\
& \Longrightarrow \quad \forall n \quad \sum_{x\in S^{\leq n}} \frac{\mathrm{time}_M(x)}{T(n)} \ \leq \ |S^{\leq n}| \\
& \Longrightarrow \quad \forall l \quad \sum_{\mathrm{rank}_S(x)\leq l} \frac{\mathrm{time}_M(x)}{T(n)} \ \leq \ l \\
& \Longleftrightarrow \quad (\mathrm{time}_M, \mu_S) \ \in \ \mathrm{Eav}(T) \ .
\end{aligned}$$

∎

Korollar 9 *Für alle* $T \geq \alpha\mathcal{N}$ *gilt*

$$(L, \mu_{\mathrm{uni}}) \in \mathrm{EavDisTime}(T) \quad \Longleftrightarrow \quad \begin{array}{l} \exists DTM\ M\ \mathit{mit}\ L(M) = L \quad \mathit{und} \\ \forall n\ E_{[\mu_{\mathrm{uni}}]_n}(\mathrm{time}_M) \leq T \ . \end{array}$$

Der formale Unterschied zwischen *worst-case*-Komplexitätsklassen und den Klassen aus gewichteten Problemen ist der Unterschied zwischen Problem und einem Paar Problem–Wahrscheinlichkeitsverteilung. Ein Ziel sollte es aber sein, *worst-case* und *average-case* als Komplexitätsklassen direkt zu vergleichen. Aus diesem Grund erscheint der folgende Ansatz vielversprechend. Wir betrachten die *average*-Komplexität eines Entscheidungsproblems der Sprache L bezüglich aller Wahrscheinlichkeitsverteilungen einer Menge C. Nur wenn jedes der gewichteten Probleme (L, μ) für $\mu \in C$ Av-beschränkt (bzw. Eav-beschränkt) ist, so liegt L in der entsprechenden Komplexitätsklasse **AvTime** (bzw. **EavTime**).

Definition 16 *Sei T eine Komplexitätsschranke und C eine Menge von Wahrscheinlichkeitsverteilungen, bzw. Rangfunktionen. Dann sei*

$$\begin{aligned}\mathbf{AvTime(T,C)} &:= \{L \mid \forall \mu \in C \quad (L,\mu) \in \mathrm{AvDisTime}(T)\}\ ,\\ \mathbf{EavTime(T,C)} &:= \{L \mid \forall \mu \in C \quad (L,\mu) \in \mathrm{EavDisTime}(T)\}\ .\end{aligned}$$

Das nächste Kapitel wird sich mit diesen Klassen näher auseinandersetzen.

Aufgrund der Definition der Durchschnittsmaße Av und Eav ergibt sich für lineares T, daß AvTime und EavTime die gleichen Komplexitätsklassen definieren. Für größere Schranken T ist bekannt, daß $\mathrm{Av}\,(T(2\cdot\mathcal{N}))$ schwächer beschränkt als $\mathrm{Eav}\,(T)$. Der direkte Schluß aus Lemma 7 ergibt für $T/\mathcal{N}$ monoton zunehmend

$$\mathrm{EavTime}(T,C) \subseteq \mathrm{AvTime}(T(2\cdot\mathcal{N}),C)\ .$$

Dieses Ergebnis läßt sich aber durchaus noch verbessern, wenn man die besonderen Eigenschaften von Turing-Maschinen berücksichtigt.

Theorem 5 *Für beliebiges $\delta,\epsilon > 0$ und $T \geq (1/\delta+\epsilon)\,\mathcal{N}$, wobei zusätzlich die Funktion $T/\mathcal{N}$ monoton zunehmend ist und für alle Klassen von Verteilungen C gilt*

$$\mathrm{EavTime}(T,C) \subseteq \mathrm{AvTime}(T(\mathcal{N}\cdot(1+\delta)),C)\ .$$

Beweis: Für $L \in \mathrm{AvTime}(T,C)$ und $\mu \in C$ sei M eine DTM mit $L(M) = L$ und $(\mathrm{time}_M,\mu) \in \mathrm{Eav}\,(T)$. Durch lineare Beschleunigung erhält man eine Maschine M' für L mit $\mathrm{time}_{M'}(x) \leq \delta\cdot\mathrm{time}_M(x)$ für alle Eingaben, auf denen M mindestens Zeit $T(|x|)$ benötigt. Wir teilen die Eingabe in die beiden Teilmengen

$$\begin{aligned}I_1 &:= \{x \mid \mathrm{time}_{M'}(x) > T(|x|)\}\ ,\\ I_2 &:= \{x \mid \mathrm{time}_{M'}(x) \leq T(|x|)\}\ .\end{aligned}$$

Für $x \in I_1$ gilt (siehe hierzu Beweis von Lemma 6)

$$\frac{\mathrm{time}_{M'}(x)}{T(|x|)} \geq \frac{T^{-1}(\mathrm{time}_{M'}(x))}{|x|}\ .$$

Somit gilt

$$\begin{aligned}\delta\cdot l &\geq \sum_{x\in I_1\cap X_\mu^{\leq l}} \frac{\mathrm{time}_{M'}(x)}{T(|x|)} \geq \sum_{x\in I_1\cap X_\mu^{\leq l}} \frac{\mathrm{time}_{M'}(x)}{T(|x|)}\\ &\geq \sum_{x\in I_1\cap X_\mu^{\leq l}} \frac{T^{-1}(\mathrm{time}_{M'}(x))}{|x|}\ .\end{aligned}$$

Für $x \in I_2$ gilt

$$\frac{T^{-1}(\mathrm{time}_{M'}(x))}{|x|} \leq 1 \ .$$

Deswegen ist

$$\sum_{x \in I_1 \cap X_\mu^{\leq l}} \frac{T^{-1}(\mathrm{time}_{M'}(x))}{|x|} \leq l \qquad \text{und}$$

$$\sum_{x \in X_\mu^{\leq l}} \frac{T^{-1}(\mathrm{time}_{M'}(x))}{|x|} \leq l \cdot (1+\delta) \ .$$

∎

Setzt man für C ein-elementige Mengen ein, ergibt sich daraus

Korollar 10 *Für beliebiges $\delta, \epsilon > 0$ und $T \geq (1/\delta + \epsilon)\ \mathcal{N}$, wobei zusätzlich die Funktion $T/\mathcal{N}$ monoton zunehmend ist, gilt*

$$\mathrm{EavDisTime}(T) \subseteq \mathrm{AvDisTime}(T(\mathcal{N} \cdot (1+\delta))) \ .$$

Bei der Definition der Klassen $\mathrm{AvTime}(T, C)$ und $\mathrm{EavTime}(T, C)$ ist offengeblieben, wie die Menge der Verteilungen C zu wählen ist. Wählt man sie zu klein, also z.B. als endliche Menge, so erhält man nur eine unrepräsentative *average*-Klasse, die das mannigfaltige Auftauchen eines bestimmten Problems in der Praxis kaum zu charakterisieren vermag.

Wählt man C zu mächtig, gar als die Menge aller möglichen Wahrscheinlichkeitsverteilungen, ist zu befürchten, daß eine Wahrscheinlichkeitsverteilung darunter sein kann, bezüglich der die *average*-Zeit der *worst-case*-Zeit entspricht (Wir werden später sehen, daß genau dies der Fall ist).

Um einen Mittelweg zu finden, wird in dieser Arbeit folgender Ansatz zugrundegelegt:

Turing-Maschinen sollen in die Lage versetzt werden, Wahrscheinlichkeitsverteilungen zu berechnen. Man kann sich leicht vorstellen, daß je mehr Zeit die Turing-Maschine zur Verfügung hat, desto mächtiger wird die Menge aller berechenbaren Verteilungen. Dabei ist auch zu erwarten, daß diese Wahrscheinlichkeitsverteilung irgendwann gutes *average*-Laufzeitverhalten unmöglich macht.

3.2 Rankable

Doch wie kann eine Turing-Maschine eine Wahrscheinlichkeitsverteilung berechnen? Bisherige Ansätze ließen zum Beispiel eine Turing-Maschine die Verteilungsfunktion $\mu^*(x) := \sum_{z \leq x} \mu(x)$ berechnen.

Definition 17 *[Levin 86, Gure 90, BCGL 89] Eine Wahrscheinlichkeitsverteilung μ ist in der Klasse* ***T*-computable** *, falls eine Turing-Maschine auf Eingabe x die Binärdarstellung von $\mu^*(x)$ in Zeit $T(|x|)$ ausgibt.*

Von besonderem Interesse ist die Komplexitätsklasse **POL-computable**. Für sie wird in der Fachliteratur auch die abweichende Definition benutzt, in der nur gefordert wird, daß die ersten m Nachkomma-Stellen von $\mu^*(x)$ für eine Eingabe $|x|$ in Zeit $\mathrm{Pol}(|x| \cdot m)$ berechenbar sind.

Ein anderer Ansatz [BCGL 89] benutzt probabilistische Turing-Maschinen [Rabin 76]. Diese sind in der Lage, in jedem Berechnungsschritt eine Münze zu werfen.

Definition 18 *[BCGL 89] Eine Wahrscheinlichkeitsverteilung μ ist* ***T*-sampleable**, *falls eine probabilistische Turing-Maschine M ohne Eingabe die Ausgabe $x \in \Sigma^n$ mit Wahrscheinlichkeit $\mu(x)$ generiert, d.h.*

$$\mathrm{Prob}[M(\lambda) = x] \;=\; \mu(x)$$

und dabei nur $T(n)$ viele Schritte benötigt.

Beide Definitionen beziehen sich auf explizite Wahrscheinlichkeitswerte. Eine Änderung dieser expliziten Werte führt zu gewichtigen Änderungen der Algorithmen und sicher auch zu drastischen Auswirkungen auf die Laufzeit. In unseren *average*-Maßen aber hinterläßt auch eine beliebig starke Änderung dieser Werte keine Spuren, solange nur die Rangfunktion gleich bleibt.

Da wir mit Hilfe der Rangfunktionen eine vollständige Charakterisierung hinsichtlich aller hier eingeführten Maße besitzen, liegt es nahe, die Zeit zu beschränken, um den Rang einer Eingabe zu berechnen. Die zugehörige Klasse nennen wir dann ***T*-rankable**, wenn T die Zeitschranke ist.

Definition 19 rankable(T) *sei die Menge aller Wahrscheinlichkeitsverteilungen μ bzw. Rangfunktionen ρ, für die es eine Turing-Maschine M gibt, die auf Eingabe x die Ausgabe $\mathrm{bin}(\rho(x))$ in Zeit $T(|x|)$ berechnet.*

Beispiel:
Erinnern wir uns an die Rangfunktionen $\mathrm{rank}_{\mathrm{Pal}}$ und $\mathrm{rank}_{\mathrm{Pal}}^{\mathrm{inj}}$ für Palindrom-Sprachen, die wir auf Seite 23 schon besprochen haben.

$$\mathrm{rank}_{\mathrm{Pal}}(x) \;=\; \begin{cases} \sum_{i \leq |x|} 2^{\lceil \frac{i}{2} \rceil} \;, & \text{falls } x \in L_{\mathrm{Pal}} \;, \\ \quad \infty \;, & \text{sonst} \;. \end{cases}$$

$$\mathrm{rank}_{\mathrm{Pal}}^{\mathrm{inj}}(x) \;=\; \begin{cases} \sum_{i < |x|} 2^{\lceil \frac{i}{2} \rceil} + \mathrm{ord}(\mathrm{praefix}(x, \lceil \frac{|x|}{2} \rceil)) & \\ \qquad\qquad - \exp(\lceil \frac{|x|}{2} \rceil) \;, & \text{falls } x \in L_{\mathrm{Pal}} \;, \\ \qquad\qquad \infty \;, & \text{sonst} \;. \end{cases}$$

Diese beiden Rangfunktionen können von einer 1-Band-Turing-Maschine in quadratischer Zeit berechnet werden, wobei ein Großteil der Zeit allein zur Entscheidung der Palindrom-Eigenschaft der Eingabe x benötigt wird.

Eine 2-Band-Turing-Maschine dagegen kann beide in linearer Zeit berechnen, so daß gilt

$$\mathrm{rank}_{\mathrm{Pal}},\ \mathrm{rank}_{\mathrm{Pal}}^{\mathrm{inj}} \quad \in \quad \mathrm{LIN}\text{–rankable} .$$

Dagegen kann die Rangfunktion ρ_1 von Seite 22

$$\rho_1(x) \ = \ \begin{cases} \mathrm{ord}(z) + 2 \cdot \log_2(\mathrm{ord}(z)) \ , & \text{falls } x = z0 \ , \\ \mathrm{ord}(z) + \exp(\mathrm{ord}(z)/2) \ , & \text{sonst} \ . \end{cases}$$

von keiner Turing-Maschine in Zeit $o(\exp(\mathcal{N}))$ berechnet werden. Dies resultiert alleine aus den großen Rängen für $x = z1$, die mit geringen Wahrscheinlichkeiten korrespondieren. Hier gilt insbesondere für jede Wahrscheinlichkeitsverteilung μ_1 mit $\mathrm{rank}_{\mu_1} = \rho_1$ für eine Eingabe der Form $x = z1$

$$\mu_1(x) \ \leq \ \frac{1}{2^{2^{|x|-1}}}$$

(Die Abschätzung $\mathrm{rank}_\mu(x) \cdot \mu(x) \leq 1$ gilt für alle x mit $\mu(x) \neq 0$).

Eine Turing-Maschine, die die Verteilungsfunktion von μ_1 gemäß der Definition *computable* berechnet, müßte demnach mindestens $2^{|x|/2}$ Binär-Stellen ausgeben. Die Wahrscheinlichkeitsverteilung μ_1 ist folglich nicht POL-*computable* enthalten.

μ_1 ist auch nicht POL-*sampleable*. Um eine so kleine Wahrscheinlichkeit wie $\frac{1}{2^{2^{n-1}}}$ zu erzeugen, müßte man mindestens 2^{n-1} Münzen werfen. Damit kann die probabilistische Turing-Maschine nicht mehr polynomiell in der Länge $n = |x|$ beschränkt sein.

Alle drei Maße für Wahrscheinlichkeitsverteilungen scheinen bei ρ_1 an ihre Grenzen zu stoßen.

3.3 Describable

Dieser Effekt erwächst aus dem Aufwand, kleine Wahrscheinlichkeiten zu beschreiben. Die Komplexitätsklasse **V-describable** umfaßt alle Wahrscheinlichkeitsverteilungen, deren positive Wahrscheinlichkeiten nach unten beschränkt sind.

Definition 20 **V-describable** *sei die Menge aller Wahrscheinlichkeitsverteilungen* μ *, deren Rangfunktion* $\rho = \mathrm{rank}_\mu$ *für alle Zeichenketten* x *die folgende Eigenschaft besitzt*

$$\log \rho(x) \ \leq \ V(|x|) \quad \textit{oder} \quad \rho(x) = \infty \ .$$

Beispiel:
Die Wahrscheinlichkeitsverteilungen korrespondierend zu den Rangfunktionen $\mathrm{rank}_{\mathrm{Pal}}$ und $\mathrm{rank}_{\mathrm{Pal}}^{\mathrm{inj}}$ sind daher in der Klasse LIN–describable.

Eine Wahrscheinlichkeitsverteilung mit Rangfunktion ρ_1 (siehe Seite 22) ist dagegen nicht in der Klasse POL–describable, da es Eingaben x gibt mit $\rho_1(x) \geq \exp(\mathrm{ord}(x)/2)$.

Die Größe des Ranges $\mathrm{rank}_\mu(x)$ steht in einem engen Zusammenhang mit der Komplexität der Wahrscheinlichkeitsverteilung.

Proposition 5 *Für eine Zeitschranke T gilt*

$$\begin{aligned} T\text{–rankable} &\subseteq T\text{–describable} , \\ T\text{–computable} &\subseteq T\text{–describable} , \\ T\text{–sampleable} &\subseteq T\text{–describable} . \end{aligned}$$

Beweis:

1. T–rankable $\subseteq$ T–describable:

   Sei $\rho \in T$–rankable eine Rangfunktion, dann gibt es eine Turing-Maschine, die $\mathrm{bin}(\rho(x))$ in Zeit $T(|x|)$ ausgibt. Somit ist die Länge der Ausgabe $|\mathrm{bin}(\rho(x))| = \log \rho(x)$ für $\rho(x) \neq \infty$ beschränkt durch $T(|x|)$.

2. T–computable $\subseteq$ T–describable:

   Sei $\mu \in T$–computable. Die korrespondierende Turing-Maschine gibt nun auf Eingabe x den Wert $\mu^*(x)$ in Zeit $T(|x|)$ aus. Der Wert $\mu^*(x-1)$ wird ebenfalls innerhalb dieser Zeitspanne ausgegeben. Damit ist $\mu(x) = \mu^*(x) - \mu^*(x-1)$ entweder 0 oder $\mu(x) \geq 1/\exp T(|x|)$.

   Für $\mu(x) \neq 0$ ergibt sich aus der Ungleichung

   $$\mu \cdot \mathrm{rank}_\mu(x) \ \leq \ 1$$

   sofort $\log \mathrm{rank}_\mu(x) \leq T(|x|)$.

3. T–sampleable $\subseteq$ T–describable:

   Sei M eine probabilistische Turing-Maschine, die mit Wahrscheinlichkeit $\mu(x)$ die Ausgabe x in Zeit $T(|x|)$ berechnet. Wenn $\mu(x) \neq 0$, dann ist

   $$\mu(x) \geq 1/\exp T(|x|) ,$$

   da in jedem Schritt der Berechnung die Wahrscheinlichkeit für eine bestimmte Ausgabe höchstens halbiert werden kann. Somit folgt wieder $\mu \in T$–describable.

∎

3.4 LavDis$\mathcal{P}$ versus AvDis$\mathcal{P}$

In der Fachwelt wird ein anderer Begriff für effizient lösbare Probleme der hier verwendeten Klassen AvDis$\mathcal{P}$ und EavDis$\mathcal{P}$ vorgezogen. Wir bezeichnen diese Klasse hier **LavDis $\mathcal{P}$**. Sie umfaßt alle gewichteten Probleme, die *average*-polynomiell im Levin'schen Sinne beschränkt sind.

Definition 21

$$\mathbf{LavDis}\,\boldsymbol{\mathcal{P}} \;:=\; \{(L,\mu) \mid \exists M \in \mathrm{DTM} : L(M) = L \quad \text{und} \quad (\mathrm{time}_M, \mu) \in \mathrm{Lav}\,(\mathrm{POL})\,\}$$

Wir wollen versuchen diese Klasse mit AvDis$\mathcal{P}$ in Beziehung zu setzen. Hierzu untersuchen wir, unter welchen Einschränkungen Av (POL) der Menge Lav (POL) entspricht.

Für stark abnehmende Wahrscheinlichkeitsverteilungen definieren Av (POL) und *polynomial on the average* (Lav (POL)) verschiedene Klassen, was auf die mangelnde Präzision des Levin'schen Maßes zurückzuführen ist.

Beispiel:
Betrachten wir hierzu die Funktion $\boldsymbol{t(x)} := \exp|x|$ und $\boldsymbol{\mu(x)} := \frac{c}{\exp\exp|x|}$, für eine Konstante c, so daß $\sum_x \mu(x) = 1$.

$$\sum_x t(x)\cdot\mu(x) \;=\; c\cdot\sum_x \frac{\exp|x|}{\exp\exp|x|} \;\leq\; c\cdot\sum_{n\in\mathbb{N}} \frac{\exp^2 n}{\exp\exp n} \;\leq\; c\;.$$

Somit ist $(t,\mu) \in \mathrm{Lav}\,(\mathrm{LIN}) \subseteq \mathrm{Lav}\,(\mathrm{POL})$. Dagegen ist (t,μ) nicht polynomiell Av-beschränkt, denn es gilt für alle $c_1, c_2 \in \mathbb{N}$:

$$\sum_{x\in X_\mu^{\leq l}} \frac{c_1\cdot t(x)^{\frac{1}{c_2}}}{|x|} \;=\; \sum_{x\in X_\mu^{\leq l}} \frac{c_1\cdot \exp^{\frac{1}{c_2}}|x|}{|x|} \;\geq_{ae}\; l\;.$$

Offensichtlich nimmt μ zu stark ab, um (t,μ) im Levinschen Maße gut genug zu charakterisieren.

Daher betrachten wir moderat abnehmende Wahrscheinlichkeitsverteilungen mit der Eigenschaft **modest**.

Definition 22 *Die Menge* **modest** *umfaßt alle Wahrscheinlichkeitsverteilungen* μ*, so daß für alle* x *mit* $\mu(x) \neq 0$ *gilt*

$$\mu(x)\cdot\mathrm{rank}_\mu(x)\cdot\mathrm{PLog}(\mathrm{rank}_\mu(x)) \;\geq\; 1\;.$$

Beschränkt man sich nun auf Wahrscheinlichkeitsverteilungen mit den Eigenschaften *modest* und POL–describable, so unterscheiden sich AvDis$\mathcal{P}$ und LavDis$\mathcal{P}$ nicht mehr. Diese Einschränkung verhindert nur extreme, für die Praxis untaugliche Wahrscheinlichkeitsverteilungen.

Theorem 6 *Für Wahrscheinlichkeitsverteilungen* $\mu \in$ modest $\cap$ POL–describable *und Funktionen* $f : \Sigma^* \to \mathbb{N}$ *gilt:*

$$(f, \mu) \in \mathrm{Lav}\,(\mathrm{POL}) \iff (f, \mu) \in \mathrm{Av}\,(\mathrm{POL}) \ .$$

Beweis: „$\Leftarrow$“: Es gilt $(f, \mu) \in \mathrm{Av}\,(\mathrm{POL})$. Damit gibt es Konstanten $c_1, c_2 \in \mathbb{R}^+$, so daß für alle monotonen Transformationen m von μ gilt

$$\sum_x \frac{c_1 \cdot f(x)^{c_2}}{|x|} \cdot m(\mu(x)) \ \leq \ 1 \ .$$

Wählt man die identische Funktion $\mathcal{N} : x \mapsto x$ als monotone Transformation m, so ergibt sich

$$\sum_x \frac{c_1 \cdot f(x)^{c_2}}{|x|} \cdot \mu(x) \ \leq \ 1 \ ,$$

was gleichbedeutend ist mit $(f, \mu) \in \mathrm{Lav}\,(\mathrm{POL})$.

„$\Rightarrow$“: Gilt $(f, \mu) \in \mathrm{Lav}\,(\mathrm{POL})$, dann gibt es Konstanten $c_1, c_2 \in \mathbb{R}^+$, so daß

$$\sum_x \frac{c_1 \cdot f(x)^{c_2}}{|x|} \cdot \mu(x) \ \leq \ 1 \ .$$

Nun ist $\mu \in$ modest. Daher existieren Konstanten $c_4, c_5 \in \mathbb{R}^+$, so daß gilt

$$c_4 \cdot \mu(x) \cdot \mathrm{rank}_\mu(x) \cdot \log^{c_5} \mathrm{rank}_\mu(x) \ \geq \ 1 \ .$$

Da $\mu \in$ POL–describable, gilt für eine geeignete Konstante $c_3 \in R^+$ und für $|x| > 1$ mit $\mu(x) \neq 0$

$$\log \mathrm{rank}_\mu(x) \ \leq \ |x|^{c_3} \ .$$

Wir werden zeigen, daß $(f, \mu) \in \mathrm{Av}\left((k_1 \cdot \mathcal{N})^{k_2}\right)$ für $k_1 := \max\left(2; \frac{2 \cdot c_4}{c_1}\right)$ und $k_2 := \frac{c_3 \cdot c_5 + 1}{c_2}$.

Betrachten wir dazu folgende Mengen $X_1 := \{x \mid f(x) \leq |x|^{k_2}\}$ und $X_2 := \{x \mid f(x) > |x|^{k_1}\}$.

1. $f(x) \leq |x|^{k_2}$

   Damit gilt sofort $\frac{f(x)^{\frac{1}{k_2}}}{|x|} \ \leq \ 1$.

2. $f(x) > |x|^{k_2}$

   Somit ist
$$f(x) \quad \geq \quad |x|^{k_2} \quad \geq \quad \log^{\frac{k_2}{c_3}} \operatorname{rank}_\mu(x) \ .$$
   Damit gilt für ein $l \in \mathbb{N}$ und für alle x mit $\operatorname{rank}_\mu(x) \leq l$
$$\begin{aligned} \frac{f(x)^{\frac{1}{k_2}}}{|x|} \quad &\leq \quad \frac{f(x)^{\frac{c_3 \cdot c_5 + 1}{k_2}}}{f(x)^{\frac{c_3 \cdot c_5}{k_2}} \cdot |x|} \quad \leq \quad \frac{f(x)^{c_2}}{\log^{c_5} \operatorname{rank}_\mu(x) \cdot |x|} \\ \leq \quad \frac{f(x)^{c_2}}{\log^{c_5} \operatorname{rank}_\mu(x) \cdot |x|} \cdot \frac{l}{\operatorname{rank}_\mu(x)} \quad &\leq \quad c_4 \cdot \frac{f(x)^{c_2}}{|x|} \cdot \mu(x) \cdot l \end{aligned}$$

Daraus resultiert für alle $l \in \mathbb{N}$
$$\begin{aligned} \sum_{x \in X_\mu^{\leq l}} \frac{f(x)^{\frac{1}{k_2}}}{k_1 |x|} \quad &\leq \quad \sum_{x \in X_\mu^{\leq l} \cap X_1} \frac{f(x)^{\frac{1}{k_2}}}{k_1 |x|} + \sum_{x \in X_\mu^{\leq l} \cap X_2} \frac{f(x)^{\frac{1}{k_2}}}{k_1 |x|} \\ &\leq \quad \frac{l}{2} + \sum_{x \in X_\mu^{\leq l} \cap X_2} \frac{c_4}{k_2} \cdot \frac{f(x)^{c_2}}{|x|} \cdot \mu(x) \cdot l \ \leq \ l \ . \end{aligned}$$
∎

Somit entspricht die *Average*-Komplexitätsklasse LavDis$\mathcal{P}$ der hier vorgeschlagenen Klasse AvDis$\mathcal{P}$, solange man sich auf polynomiell beschreibbare (POL-*describable*) und moderate abnehmende (*modest*) Wahrscheinlichkeitsverteilungen beschränkt.

Korollar 11 *Für Wahrscheinlichkeitsverteilungen* $\mu \in \text{modest} \cap \text{POL}$–describable *und Sprachen* $L \subseteq \Sigma^*$ *gilt:*
$$(L, \mu) \in \text{LavDis}\,\mathcal{P} \iff (L, \mu) \in \text{AvDis}\,\mathcal{P} \ .$$

Kapitel 4

Average-Hierarchien

4.1 Beziehungen zwischen Worst Case und Average Case

4.1.1 Uniforme Wahrscheinlichkeitsverteilungen

Nähert man sich dem Vergleich der Konzepte *worst-case* und *average* zum ersten Mal, erscheint es sinnvoll, zuerst Überlegungen für einfache Wahrscheinlichkeitsverteilungen anzustellen. Eine einfache und zugleich harmonische Verteilung ist die uniforme. Beschränkt man aber die komplexitätstheoretischen Betrachtungen auf solche, ist die genaueste Beziehung zwischen *worst-case* und *average-case*, die man erhalten kann, nur ein exponentieller Sprung.

Theorem 7

$$\begin{aligned} \mathrm{EavTime}(T, \{\mu_{\mathrm{uni}}\}) &\subseteq \mathrm{DTime}(T \cdot \mathrm{EXL}) \; , \\ \mathrm{AvTime}(T, \{\mu_{\mathrm{uni}}\}) &\subseteq \mathrm{DTime}(T \circ \mathrm{EXL}) \; . \end{aligned}$$

Beweis: Für $L \in \mathrm{EavTime}(T, \{\mu_{\mathrm{uni}}\})$ sei M eine erwartet T-zeitbeschränkte Turing-Maschine mit $L(M) = L$. Für alle Zeichenketten z muß nun gelten

$$\frac{\mathrm{time}_M(z)}{T(|z|)} \quad \leq \quad \sum_{x \leq z} \frac{\mathrm{time}_M(x)}{T(|x|)} \quad \leq \quad \mathrm{rank}_{\mathrm{uni}}(z) \; .$$

Dadurch kann für jede Eingabe z der *worst-case* abgeschätzt werden durch

$$\mathrm{time}_M(z) \;\leq\; \mathrm{rank}_{\mathrm{uni}}(z) \cdot T(|z|) \;\leq\; \exp(O(|z|)) \cdot T(|z|) \; .$$

Angenommen, daß $L \in \mathrm{AvTime}(T, \{\mu_{\mathrm{uni}}\})$ und M eine Turing-Maschine für M sei, so daß für alle Zeichenketten z gilt

$$\frac{T^{-1}(\mathrm{time}_M(z))}{|z|} \quad \leq \quad \sum_{x \leq z} \frac{T^{-1}(\mathrm{time}_M(x))}{|x|} \quad \leq \quad \mathrm{rank}_{\mathrm{uni}}(z)\ .$$

Dann folgt $\mathrm{time}_M(z) \ \leq\ T(\mathrm{rank}_{\mathrm{uni}}(z) \cdot |z|) \ \leq\ T(\exp O(|z|))$. ∎

Man beachte, daß es für die EavTime-Klassen einen exponentiellen Zuwachs in der *worst-case*-Schranke gibt, wenn die Zeitschranke T ein Polynom ist. Andererseits, wenn T selbst exponentiell ist, fällt der zusätzliche Faktor EXL nur wenig ins Gewicht. Es ist leicht zu zeigen, daß für kleine Zeitschranken exponentielle Zunahmen nicht vermieden werden können.

Theorem 8 *Für eine Zeitschranke $T \geq 2 \cdot \mathcal{N}$ gilt*

$$\mathrm{EavTime}(T, \{\mu_{\mathrm{uni}}\}) \quad \setminus \quad \mathrm{DTime}(o(T \cdot \exp)) \quad \neq \quad \emptyset\ .$$

Beweis: Sei $L_1 \subseteq 1^*$ eine unäre Sprache in

$$\mathrm{DTime}\Big(\exp(\frac{1}{2}T \cdot \exp)\Big) \ \setminus \mathrm{DTime}\Big(\exp(o(T \cdot \exp))\Big)$$

und L definiert durch $L \ := \ \{xy \mid x \in L_1 \text{ und } y = 0^{\exp(|x|)-|x|}\}$. Dann gilt

$$L \in \ \mathrm{DTime}(\tfrac{1}{2} \cdot T \cdot \exp) \setminus \mathrm{DTime}\,(o(T \cdot \exp))\ .$$

Genauer gesagt gibt es eine Maschine M für L, die alle Eingaben nicht in der Form $1^k 0^{2^k - k}$ in Linear-Zeit verwirft und ansonsten Zeit $\frac{1}{2} \cdot T \cdot \exp$ benötigt. Dies ergibt $L \in \mathrm{EavTime}(T, \{\mu_{\mathrm{uni}}\})$, da für alle l gilt

$$\begin{aligned}
&\sum_{x \in X^{\leq l}_{\mu_{\mathrm{uni}}}} \frac{\mathrm{time}_M(x)}{T(|x|)} \\
\leq\ & \sum_{x \in X^{\leq l}_{\mu_{\mathrm{uni}}} \setminus \{1^k 0^{2^k-k} \mid k \in \mathbb{N}\}} \frac{|x|}{T(|x|)} \quad + \sum_{x \in X^{\leq l}_{\mu_{\mathrm{uni}}} \cap \{1^k 0^{2^k-k} \mid k \in \mathbb{N}\}} \frac{1}{2} \cdot \frac{\exp(|x|) \cdot T(|x|)}{T(|x|)} \\
\leq\ & \frac{l}{2} + \frac{1}{2} \sum_{n=1}^{\log l - 1} \exp n \ \leq \ l\ .
\end{aligned}$$

∎

Theorem 7 impliziert somit auch $L \notin \mathrm{EavTime}(o(T), \{\mu_{\mathrm{uni}}\})$.

Die AvTime-Klassen verhalten sich in diesem Zusammenhang vollkommen anders. Für jede Schranke T ist das Aufblähen nicht vernachlässigbar, es ist wirklich exponentiell, wie es auch für die bisher besten bekannten Simulationen zwischen nichtdeterministischen oder probabilistischen Zeitklassen und deterministischen Zeitklassen der Fall ist. Dies mag ein Zeichen dafür sein, daß dieses Konzept für Komplexitätsuntersuchungen geeigneter ist.

Theorem 9 *Für eine Zeitschranke $T \geq 2 \cdot \mathcal{N}$ gilt*

$$\mathrm{AvTime}\Big(T, \{\mu_{\mathrm{uni}}\}\Big) \quad \setminus \quad \mathrm{DTime}\left(o(T \circ \exp \tfrac{\mathcal{N}}{2})\right) \quad \neq \quad \emptyset \; .$$

Beweis: Sei L_1 wiederum eine unäre Sprache aus

$$L_1 \in \mathrm{DTime}(\exp(T \circ \exp \tfrac{\mathcal{N}}{2})) \setminus \mathrm{DTime}(o(\exp(T \circ \exp \tfrac{\mathcal{N}}{2}))) \; ,$$

und wir defineren $L := \{xy \mid x \in L_1 \text{ und } y = 0^{\exp(|x|)-|x|}\}$. Für diese Sprache L gilt daher $L \in \mathrm{DTime}(T \circ \exp \frac{\mathcal{N}}{2}) \setminus \mathrm{DTime}(o(T \circ \exp \frac{\mathcal{N}}{2}))$. Alle Eingaben nicht in der Form $1^k 0^{2^k - k}$ können wiederum in Linear-Zeit von einer geeigneten Maschine M verworfen werden. Die Aussage $L \in \mathrm{AvTime}(T, \{\mu_{\mathrm{uni}}\})$ folgt durch

$$\begin{aligned}
& \sum_{x \in X^{\leq l}_{\mu_{\mathrm{uni}}}} \frac{T^{-1}(\mathrm{time}_M(x))}{|x|} \\
\leq \quad & \sum_{x \in X^{\leq l}_{\mu_{\mathrm{uni}}} \setminus \{1^k 0^{2^k-k} \mid k \in \mathbb{N}\}} \frac{T^{-1}(|x|)}{|x|} \quad + \sum_{x \in X^{\leq l}_{\mu_{\mathrm{uni}}} \cap \{1^k 0^{2^k-k} \mid k \in \mathbb{N}\}} \frac{T^{-1}(T(\exp(|x|/2)))}{|x|} \\
\leq \quad & \frac{l}{2} + \frac{1}{2} \sum_{n=1}^{\log l - 1} \exp n \;\leq\; l \; .
\end{aligned}$$

■

Wegen Theorem 7 gilt nun auch $L \notin \mathrm{AvTime}(T(o(\mathcal{N})), \{\mu_{\mathrm{uni}}\})$.

Theorem 5 zeigte bereits, daß das Av-Maß schwächer ist als das Eav-Maß. Wir zeigen jetzt, daß es sich hier um eine echte Hierarchie handelt.

Korollar 12 *Sei $T \in \omega(\mathcal{N})$ eine Zeitschranke, so daß $T/\mathcal{N}$ monoton zunehmend ist. Dann gilt*

$$\mathrm{EavTime}(T, \{\mu_{\mathrm{uni}}\}) \;\subset\; \mathrm{AvTime}(T(2 \cdot \mathcal{N}), \{\mu_{\mathrm{uni}}\}) \; .$$

Beweis: Wenn man die Gleichung

$$T \circ \exp \tfrac{\mathcal{N}}{2} \;=\; T' \cdot \exp$$

nach T' auflöst, ergibt sich die Zeitschranke $T' \in \omega(T)$. Eine Sprache L werde konstruiert wie in den letzten beiden Beweisen: Für AvTime wird dies bezüglich der Zeitschranke T getan, für die Klassen EavTime bezüglich T'. In beiden Fällen erhält man die gleiche Sprache, für die gilt

$$\begin{aligned}
L &\in \mathrm{AvTime}(T, \{\mu_{\mathrm{uni}}\}) \; , \\
L &\notin \mathrm{EavTime}(o(T'), \{\mu_{\mathrm{uni}}\}) \;\supseteq\; \mathrm{EavTime}(T, \{\mu_{\mathrm{uni}}\}) \; .
\end{aligned}$$

■

4.1.2 Komplexere Wahrscheinlichkeitsverteilungen

Als nächstes werden wir zeigen, daß, sobald man beliebige Wahrscheinlichkeitsverteilungen zuläßt, es keinen Unterschied zwischen dem *average-case* und dem *worst-case* gibt. Zu diesem Zwecke wollen wir eine Rangfunktion konstruieren, die für jede Turing-Maschine M mit $L(M) \notin \mathrm{DTime}(T)$ kleine Ränge (korrespondierend zu hohen Wahrscheinlichkeiten) den Eingaben x mit Berechnungszeiten $\mathrm{time}_M(x) > T$ zuordnet. Bezüglich dieser Wahrscheinlichkeitsverteilung gibt es keinen Unterschied zwischen DTime und EavTime (oder AvTime).

Die Hauptschwierigkeit ist die Kontrolle der Anzahl kleiner Ränge. Falls beispielsweise eine Maschine die Zahl 1 zweimal ausgibt, berechnet sie keine Rangfunktion. Würde man mit expliziten Wahrscheinlichkeiten argumentieren, entspräche dieses Problem einer Maschine, die zwei Ereignissen jeweils die Wahrscheinlichkeit $2/3$ zuweist.

Andererseits dürfen keine Lücken in den Rängen auftreten. So fehlt in der Abbildung $1 \mapsto 2$, $2 \mapsto 2$, $n \mapsto n+1$, für $n \geq 3$ der Rang 3. Eine Lösung dieses Problems wäre, zuerst alle Ränge zu berechnen, die die Maschine für kleinere Ränge ausgegeben hat. Dieser Vorgang erfordert jedoch exponentielle Zeit.

Ein anderer Lösungsansatz führt zu der Betrachtung einer speziellen Orakelsprache $\boldsymbol{H_T}$. Diese Sprache versetzt uns in die Lage, ganze Eingabeintervalle zu kontrollieren.

Definition 23 *Für eine Zeitschranke T werde die Sprache $\boldsymbol{H_T}$ definiert als*

$$\boldsymbol{H_T} := \{(x, 1^i) \mid \exists z \leq x : \mathrm{time}_{M_i}(z) > T(|z|)\} .$$

Abbildung 4.1 veranschaulicht diese Menge H_T. Man beachte, daß H_T in $\mathcal{NP}$ enthalten ist, falls T polynomiell beschränkt und konstruierbar ist. Dann ist H_T auch $\mathcal{NP}$ -vollständig, da das **beschränkte Halteproblem**

$$\texttt{NBH} := \{(x01^t0^i) \mid \mathrm{time}_{M_i}(x) \leq t\}$$

für nicht-deterministische Turing-Maschinen M_i darauf reduziert werden kann.

Sei nun $\Omega(T^2) \leq \boldsymbol{h_T} \leq O(\exp \cdot T)$ eine Zeitschranke, so daß $H_T \in \mathrm{DTime}(\boldsymbol{h_T})$.

Theorem 10 *Für $\delta > 1$ und eine Zeitschranke $T \geq 2 \cdot \delta \cdot \mathcal{N}$ gilt*

$$\begin{aligned} \mathrm{EavTime}(T, h_{\delta \cdot T}\text{–rankable}) &= \mathrm{DTime}(T), \\ \mathrm{AvTime}(T, h_{T(\delta \cdot \mathcal{N})}\text{–rankable}) &\subseteq \mathrm{DTime}(T(\delta \cdot \mathcal{N})). \end{aligned}$$

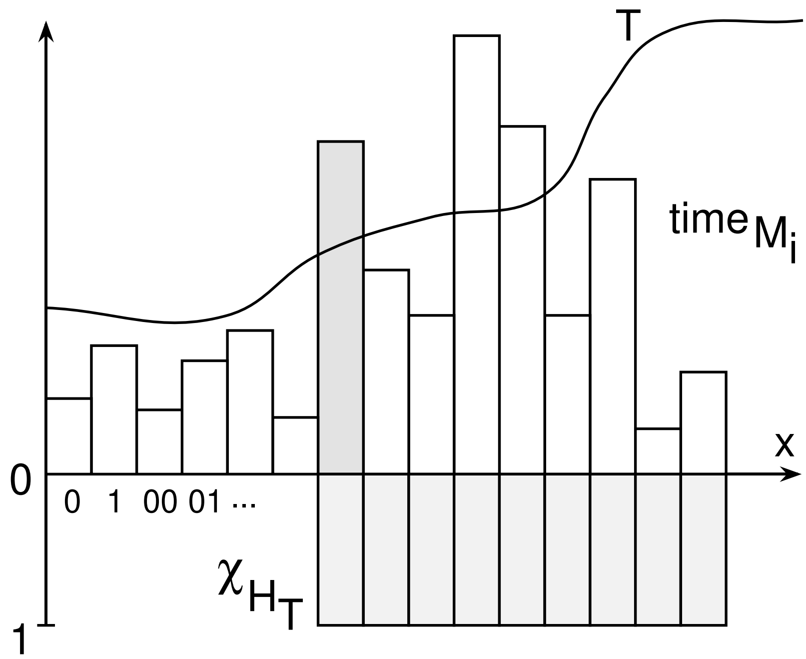

Abbildung 4.1: Die Definition der Sprache H_T .

Beweis:

1. EavTime(T, $h_{\delta \cdot T}$–rankable) $\supseteq$ DTime(T)
   folgt direkt aus der Definition von EavTime.

2. EavTime(T, $h_{\delta \cdot T}$–rankable) $\subseteq$ DTime(T):
   Diese Inklusion wird indirekt bewiesen. Sei L eine Sprache, die nicht in DTime(T) enthalten ist. Es genügt die Existenz einer Wahrscheinlichkeitsverteilung aus $h_{\delta \cdot T}$–rankable zu beweisen, bezüglich deren L nicht im Durchschnitt T zeitbeschränkt ist. Um die Rangfunktion dieser Verteilung zu erhalten, werden wir eine Folge endlicher Mengen $\boldsymbol{X_i}$ konstruieren, deren Vereinigung $\boldsymbol{D}$ alle Eingaben mit endlichem Rang, also positivem Gewicht, bestimmt. So ergibt

   $$\boldsymbol{D} := \bigcup_{i \in \mathbb{N}} X_i$$

   die D-uniforme Wahrscheinlichkeitsverteilung $\boldsymbol{\mu_D}$. Deren Rangfunktion $\mathbf{rank}_{\boldsymbol{D}}$ wiederum bestimmt sich gemäß Definition 5 durch

   $$\mathbf{rank}_{\boldsymbol{D}} \quad := \quad |\{z \in D \mid z \leq x\}| \quad \text{für } x \in D \; .$$

   Die Menge $\boldsymbol{X_i}$ und die Ränge ihrer Elemente erhält man durch den unten definierten Algorithmus `long-time`. Wir werden in Lemma 11 zeigen, daß eine Turing-Maschine die Rangfunktion in Zeit $h_{\delta \cdot T}$ berechnet. Die Idee hinter der Konstruktion der Mengen X_i ist: X_i wird benötigt, um gegen die Maschine M_i zu diagonalisieren. Falls $L(M_i) = L$, werden die folgenden Eigenschaften erfüllt.

(a) $\forall x \in X_i \qquad \mathrm{time}_{M_i}(x) > \delta \cdot T(|x|)$.
Das hieße, daß jede Menge X_i nur Eingaben enthält, für die M_i zu viel Zeit benötigt (ausreichend mehr als die Zeitschranke).

(b) $|X_i| \;>\; (\delta-1)^{-1} \sum_{j=1}^{i-1} |X_j|$:
Diese Mengen werden ausreichend groß gewählt, damit genügend langwierige Berechnungen ein gutes *average*-Verhalten verhindern.

(c) Für $j < l$ gilt $\qquad |x| < |y| \qquad \forall\, x \in X_j \quad \forall\, y \in X_l$.
So nimmt die Rangfunktion monoton zu, und denkbare Probleme mit zu vielen kleinen Rängen oder Löchern in der Rangfolge treten nicht auf.

Nehmen wir an, daß wir für jedes i solch eine Menge X_i konstruiert hätten und daß es eine Maschine M_i gibt, die L in *expected average* Zeit T bezüglich μ_D berechnet. Dies würde voraussetzen, daß

$$\forall l \qquad \sum_{x \in X_{\mu_D}^{\leq l}} \frac{\mathrm{time}_{M_i}(x)}{T(|x|)} \;\leq\; l\;.$$

Sei $l := \sum_{j=1}^{i} |X_j|$. Unter Benutzung der Eigenschaft (b) erhalten wir den Widerspruch:

$$\begin{aligned}\sum_{x \in X_{\mu_D}^{\leq l}} \frac{\mathrm{time}_{M_i}(x)}{T(|x|)} \;&\geq\; \sum_{x \in X_i} \frac{\delta T(|x|)}{T(|x|)} \\ &=\; \delta \cdot |X_i| \;=\; |X_i| \cdot (\delta - 1) + |X_i| \\ &>\; (l - |X_i|) + |X_i| \;=\; l\;.\end{aligned}$$

3. $\mathrm{AvTime}(T,\, h_{T(\delta\cdot\mathcal{N})}\text{-rankable}) \;\subseteq\; \mathrm{DTime}(T(\delta \cdot \mathcal{N}))$: Wie oben betrachte man eine Sprache $L \;\notin\; \mathrm{DTime}(T(\delta\cdot\mathcal{N}))$. Nun ersetzen wir die erste Eigenschaft von X_i durch

   (a') $\forall x \in X_i \quad \mathrm{time}_{M_i} > T(\delta \cdot |x|)$.

   Nehmen wir an, daß es eine Turing-Maschine gibt, die L akzeptiert und für deren Zeitverhalten $(\mathrm{time}_{M_i}, \mu_D) \in \mathrm{Av}\,(T)$ gilt. Das bedeutet

$$\forall l \qquad \sum_{x \in X_{\mu_D}^{\leq l}} \frac{T^{-1}(\mathrm{time}_{M_j}(x))}{|x|} \;\leq\; l\;.$$

   Wiederum erhält man für $l = \sum_{j=1}^{i} |X_j|$ einen Widerspruch

$$\sum_{x \in X_{\mu_D}^{\leq l}} \frac{T^{-1}(\mathrm{time}_{M_i}(x))}{|x|} \;\geq\; \sum_{x \in X_i} \frac{T^{-1}(T(\delta \cdot |x|))}{|x|} \;\geq\; \sum_{x \in X_i} \frac{\delta \cdot |x|}{|x|} \;=\; \delta \cdot |X_i| \;>\; l\;.$$

Die Folge X_i wird wie folgt konstruiert. Wir erinnern an die Definition der iterierten Exponentialfunktion $\mathbf{itexp(n)} := \underbrace{\exp((\cdots\exp(}_{n\text{-mal}}1)\cdots))$ und ihrer inversen Funktion $\mathbf{itlog} := \mathrm{itexp}^{[-1]}$. Wir betrachten die Zeichenketten $\boldsymbol{z_i} := 1^{\mathrm{itexp}(i)}$. Man beachte, daß für jede Zeichenkette x unter Berücksichtigung der lexikographischen Ordnung gilt $z_{\mathrm{itlog}(|x|)-1} < x \leq z_{\mathrm{itlog}(|x|)}$.

Sei M^* eine Turing-Maschine für L. Um die Berechnung der Ränge für jede Maschine M_i effizient vorzunehmen, werden wir eine andere Maschine mit identischem Zeitverhalten für längere Eingaben, aber mit linearer Zeitschranke für kleine Eingaben benutzen. Hierbei ist es irrelevant, ob die Maschine die gleiche Sprache wie M_i akzeptiert. Formal definiert sei $\boldsymbol{c} : \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ eine Funktion, so daß für alle i

$$\mathrm{time}_{M_{c(i,k)}}(x) := \begin{cases} \mathrm{time}_{M_i}(x) & \text{falls } x \geq z_k \ , \\ |x| & \text{falls } x < z_k \ . \end{cases}$$

Man kann sich leicht davon überzeugen, daß solche Indizes $c(i,k)$ der Größe $O(i \cdot k)$ in Zeit $O(i \cdot k)$ berechnet werden können. Unten wird ein rekursiver Algorithmus `long-time` beschrieben, welcher auf Eingabe x das Tupel $(\mathtt{index}, s, j, r)$ ausgibt. Diese Komponenten haben folgende Bedeutung: `index` bezeichnet den Index der Menge X_{index}, in die x eingefügt werden könnte. j zählt die Anzahl der Zeichenketten lexikographisch kleiner als x, die in X_{index} enthalten sind, während s diejenigen in D zählt. r ist gleich $\mathrm{rank}_D(x)$. Durch die letzte Komponente ergibt sich die Rangfunktion. Die anderen Informationen werden in den rekursiven Aufrufen benötigt, um die obigen Eigenschaften zu erreichen.

Um zu überprüfen, ob eine Maschine M_i sich anders verhält als M^* für irgendeine Eingabe z, wird das folgende Unterprogramm `error` benötigt.

`error` (i, x)
für alle z mit $|z| \leq \log\log|x|$
    simuliere M^* und M_i auf Eingabe z für $|x|/(\log|x|)^2$ Schritte;
    <u>if</u> entweder beide akzeptierten oder (beide verwarfen oder hielten nicht)
        <u>then</u> <u>return</u> `true`;
<u>return</u> `false`.

Der Hauptalgorithmus `long-time` sieht folgendermaßen aus:

`long-time` (x)
    <u>if</u> $|x| \leq 1$ <u>then</u> <u>return</u> $(0, 0, 0, \infty)$;
    $(\mathtt{index}, j, s, r) \leftarrow \mathtt{long\text{-}time}(z_{\mathrm{itlog}(|x|)-1})$;

<u>case 1:</u> error $(\texttt{index}, x) = \texttt{true}$ <u>then</u>
<u>if</u> $(c(\texttt{index}, \text{itlog}(|x|-1)), x-1) \in H_{\delta \cdot T}$
<u>then</u> <u>return</u> $(\texttt{index}+1, 0, s+1, \infty)$;
<u>then</u> <u>return</u> $(\texttt{index}+1, 0, s, \infty)$;
<u>case 2:</u> $(c(\texttt{index}, \text{itlog}(|x|-1)), x-1) \in H_{\delta \cdot T}$ <u>then</u>
<u>return</u> $(\texttt{index}, j+1, s+1, \infty)$;
<u>case 3:</u> $\text{time}_{M_{\text{index}}}(x) \leq \delta \cdot T(|x|)$ <u>then</u>
<u>return</u> $(\texttt{index}, j, s, \infty)$;
<u>case 4:</u> $j > \frac{s-j}{\delta-1}$ <u>then</u>
Füge x zur Menge X_i und <u>return</u> $(\texttt{index}+1, 0, s+1, s+1)$;
<u>case 5:</u> <u>else</u>
Füge x in die Menge X_i ein und <u>return</u> $(\texttt{index}, j+1, s+1, s+1)$.

Im Fall 1 kann der Index erhöht werden, da $L(M_i) \neq L$ und man sich um Maschine M_i deswegen nicht mehr zu kümmern braucht. Im zweiten Fall gibt es schon eine Zeichenkette im Intervall $(z_{\text{itlog}(|x|)-1}, z_{\text{itlog}(|x|)}]$, das in $X_{\texttt{index}}$ enthalten ist und deswegen nicht ein weiteres Mal gewählt wird.

In den letzten beiden Fällen benötigt M_i zu viel Zeit auf Eingabe x, und deswegen wird es in $X_{\texttt{index}}$ aufgenommen. Wenn wir nun genügend derartige Zeichenketten haben (4. Fall), wird der Index erhöht, andernfalls müssen wir noch mehr solche Zeichenketten für diese Menge sammeln.

Diese Prozedur wird nicht bei einem Index i steckenbleiben, da das implizieren würde, daß für alle bis auf endlich viele x $\text{time}_{M_i}(x) > \delta \cdot T(|x|)$ gelten würde. Dann würde aber eine andere Maschine L in *worst-case*-Zeit $\delta \cdot T$ berechnen. Mittels linearer Beschleunigung könnte L auch in *worst-case*-Zeit T erkannt werden — ein Widerspruch zur Wahl von L.

Nach Konstruktion erfüllen jetzt auch die Mengen X_i die gewünschten Eigenschaften (a), (b) und (c).

Lemma 11 $\mu_D \in O(h_{\delta \cdot T})$–rankable

Beweis: Sei $d(n)$ die obere Schranke der Laufzeit von `long-time` bei einer Eingabe der Länge n.

1. Die Rechenzeit von $\texttt{long-time}(z_{\text{itlog}(|x|)-1})$ ist beschränkt durch $d(\log|x|)$.

2. $\texttt{test}(\texttt{index}, x)$ kann innerhalb der Zeit $|x| \cdot \texttt{index}^2/\log|x|$ berechnet werden, wobei $\texttt{index} \leq \text{itlog}|x|$.

3. Die Simulation von $\delta \cdot T(|x|)$ Schritten der Maschine M_{index} benötigt Zeit $\delta \cdot T(|x|) \cdot \text{itlog}(|x|)^2 \;\leq\; h_{\delta T}(|x|)$.

4. Nach Definition von $h_{\delta \cdot T}$ ist für eine Simulation des Orakels $h_{\delta \cdot T}(|x|)$ Zeit ausreichend.

Deswegen ist $\text{rank}_D \in O(h_{\delta \cdot T})$–rankable. ∎

Der Beweis für das Av-Maß ist beinahe identisch, lediglich $\delta \cdot T$ muß in der obigen Konstruktion durch $T(\delta \cdot \mathcal{N})$ ersetzt werden. ∎

Miltersen hat für ein anderes Durchschnittsmaß gezeigt, daß es eine Wahrscheinlichkeitsverteilung μ gibt, die bösartig (*malign*) für DTime($\mathcal{N}^k$) bezüglich des erwarteten Zeitmaßes ist. Diese Verteilung kann berechnet werden mittels eines $\Sigma_2^{\mathcal{P}}$–Orakels ([Milt 91]). Das obige Theorem zeigt, daß hier ein $\mathcal{NP}$ -Orakel genügt.

Korollar 13

$$\text{EavTime}(T, C) \;=\; \text{DTime}(T)$$

für eine Zeitschranke $T \geq 2\delta\mathcal{N}$ *(* $\delta > 1$ *) und für eine Menge* C *von Wahrscheinlichkeitsverteilungen, die* $O(T \cdot \log^2 T)$*–rankable sind mit einem Orakel für* $H_{T(\delta \cdot \mathcal{N})}$.

Benutzt man die Tatsache, daß die Zeitschranke h_T höchstens so stark zunimmt wie $T(n) \cdot \exp n$ und daß für polynomielle Schranken die Konstante δ weggelassen werden kann, erhält man

Korollar 14 *Für Zeitschranken* $T, T' \geq 2\delta\mathcal{N}$ *(* $\delta > 1$ *) mit* $T' \in \text{POL}$ *gilt*

$$\begin{aligned} \text{EavTime}(T, (T \cdot \text{EXL})\text{–rankable}) &= \text{DTime}(T)\ , \\ \text{AvTime}(T', (T' \cdot \text{EXL})\text{–rankable}) &= \text{DTime}(T')\ . \end{aligned}$$

Man beachte, daß für das strengere *average*-Maß die Gleichheit nur behauptet wird für polynomielle Schranken T'. Die Motivation, *average*-Maße verschieden vom Erwartungswert zu betrachten, war, das Konzept der Simulationen zu erhalten. Dieses Ziel schien eine Schwächung der Präzision vorauszusetzen. Levins Ansatz ist relativ grob, während unserer sich hier als genauso präzise herausstellen wird wie der *worst-case* in dem interessanten Bereich der polynomiellen Schranken. Für Zeitschranken größer als Polynome wird die Situation etwas anders.

Theorem 11 *Sei* $T \geq \text{EEXP}$ *und* U *die Menge aller Wahrscheinlichkeitsverteilungen, dann gilt*

$$\text{DTime}(T) \subset \text{AvTime}(T, U) \ .$$

Beweis:
Die Beziehung „ $\subseteq$ “ folgt aus der Definition.
Daß die deterministischen Klassen eine echte Teilmenge der *average*-Zeitklassen sind, sieht man wie folgt. Sei $T = \exp^{[2]}$ die zweifach iterierte Exponentialfunktion und $L_E := \{1^{\exp i} \mid i \in \mathbb{N}\}$.

Sei $L \subseteq L_E$ eine Sprache aus $\text{DTime}(T^2) \setminus \text{DTime}(T)$ und μ eine beliebige Wahrscheinlichkeitsverteilung. Seien x, y zwei Elemente aus L_E mit den beiden kleinsten Rängen bezüglich μ. Man kann nun eine Maschine M für L konstruieren, die lineare Zeit für alle Eingaben in $\overline{L_E} \cup \{x, y\}$ und sonst Zeit T^2 benötigt. Das heißt für eine Eingabe x, daß

$$\frac{T^{-1}(\text{time}_M(x))}{|x|} \quad \leq \quad \frac{\log\log|x|}{|x|} \quad \leq \quad 1/2$$

und gleiches für y.

Dann sei für jedes l $\boldsymbol{e_l} := |\{X_\mu^{\leq l} \cap L_E\}|$:

$$\begin{aligned}
\sum_{z \in X_\mu^{\leq l}} \frac{T^{-1}(\text{time}_M(z))}{|z|} &\leq 1 + \sum_{z \in X_\mu^{\leq l} \setminus L_E} \frac{T^{-1}(|z|)}{|z|} + \sum_{1^{\exp j} \in X_\mu^{\leq l} \setminus \{x,y\}} \frac{T^{-1}(\text{time}_M(1^{\exp j}))}{|1^{\exp j}|} \\
&\leq 1 + \sum_{z \in X_\mu^{\leq l} \setminus L_E} 1 + \sum_{1^{\exp j} \in X_\mu^{\leq l} \setminus \{x,y\}} \frac{\log\log(\exp(2 \cdot \exp\exp j))}{\exp j} \\
&= 1 + l - e_l + \sum_{1^{\exp j} \in X_\mu^{\leq l} \setminus \{x,y\}} 1 + \frac{1}{\exp j} \\
&\leq l - e_l + 1 + (e_l - 2) + 1 \quad = \quad l \ .
\end{aligned}$$

Somit ist $L \in \text{AvTime}(T, \mu)$ für alle Wahrscheinlichkeitsverteilungen μ. ∎

Dieser Beweis scheitert für das Eav-Maß, was sich an folgenden Beispiel zeigt.

Beispiel:
Seien L, L_E und M für $T = \text{EEXP}$ wie im obigen Beweis gewählt. Dann gilt für alle $l \geq 4$

$$\sum_{z \in X_\mu^{\leq l}} \frac{\text{time}_M(z)}{T(|z|)} \quad \geq \quad \sum_{z \in X_\mu^{\leq l} \setminus L_E} \frac{T(|z|)^2}{T(|z|)} \quad \geq \quad l \ .$$

Also gilt für die Laufzeit dieser Maschine $(\text{time}_M, \mu) \not\in \text{Eav}(T)$.

Es ist nicht schwierig einzusehen, daß der obige Effekt überhaupt nicht beim Erwartungswert auftreten kann. Die zusätzlichen Simulationseigenschaften des Av-Maßes scheinen doch für sehr große Komplexitätsschranken ein Aufweichen der Präzision nach sich zu ziehen.

4.2 Zeithierarchien des Average Case

Für *average*-Komplexitätsklassen mit einer festen Schranke der Berechenbarkeit der Wahrscheinlichkeitsverteilungen erhalten wir enge Hierarchieergebnisse. Zuerst zeigen wir, daß wir selbst unter der uniformen Wahrscheinlichkeitsverteilung nicht immer ein Problem von höherer *worst-case*-Komplexität lösen können. Dadurch wird Theorem 4.2 in [Milt 91], das eine schwächere Separation im Erwartungswert bewies, substantiell verbessert.

Theorem 12 *Für Zeitschranken T_1, T_2 mit $T_1 \leq o(T_2)$ gilt*

$$\begin{aligned} \mathrm{DTime}(T_2) \ \setminus \ \mathrm{EavTime}(T_1, \{\mu_{\mathrm{uni}}\}) &\neq \emptyset \,, \\ \mathrm{DTime}(T_2) \ \setminus \ \mathrm{AvTime}(T_1, \{\mu_{\mathrm{uni}}\}) &\neq \emptyset \,. \end{aligned}$$

Beweis: Die Funktion $f : \mathbb{N} \to \mathbb{N}$ werde wie folgt definiert.

$$\begin{aligned} \boldsymbol{f(0)} &:= 1 \,, \\ \boldsymbol{n_i} &:= \min \left\{ b \geq f(i) \,\middle|\, \forall j \in [b; 2b+1] \quad \frac{T_2(j)}{T_1(j)} \geq i \right\} \,, \\ \boldsymbol{f(i+1)} &:= \exp(2n_i + 1) \,. \end{aligned}$$

Unter Zuhilfenahme von f definieren wir eine Diagonalisierungssprache $\boldsymbol{L}$ durch

$$\begin{aligned} \boldsymbol{L} \ := \ \{x \mid \text{für } i \text{ mit } f(i) \leq |x| < f(i+1) \text{ gilt:} \\ i \cdot T_1(|x|) \leq T_2(|x|) \quad \text{und} \\ \left[\mathrm{time}_{M_i}(x) > 2 \cdot T_1(|x|) \quad \text{oder} \quad x \notin L(M_i) \right] \} \,. \end{aligned}$$

Diese Definitionen sind wie folgt motiviert.

- Wie in einer herkömmlichen Diagonalisierung gehört eine Zeichenkette x zu L, falls eine korrespondierende Maschine M_i auf Eingabe x länger rechnet als $2 \cdot T_1(|x|)$ Schritte oder verwirft.
- Um zu garantieren, daß $L \in \mathrm{DTime}(T_2)$, sollte eine solche Simulation von M_i nur durchgeführt werden, wenn genügend Zeit vorhanden ist, was heißt $i \cdot T_1(|x|) \leq T_2(|x|)$.

- Um die *average*-Zeitschranke T_1 zu überschreiten, lassen wir den Index i sehr langsam wachsen. Auf diese Weise erhalten wir für jeden Index i mindestens so viele endliche Ränge (relevant für die Diagonalisierung), wie für alle Indizes $1, \ldots, i-1$ zusammen. Erreicht wird das durch das rasche Wachstum von n_i.
- Um das Inverse der resultierenden Funktion f in Zeit $T_2(|x|)$ berechnen zu können, wählen wir $f(i+1)$ als $\exp(2n_i + 1)$.

1. $L \in \mathrm{DTime}(T_2)$:

   Um L durch eine Turing-Maschine M entscheiden zu lassen, berechnen wir zuerst für eine Eingabe x das korrespondierende i, für das $f(i) \le |x| < f(i+1)$ gilt.

   Zu diesem Zweck berechnet die Maschine sukzessive alle $f(j)$ für $j \le |x|$. Dies ist innerhalb der gegebenen Zeitschranke leicht möglich, da f sehr rasch zunimmt ($f \in \omega(\mathrm{itexp})$).

   Für ein gegebenes $f(j)$ wird $f(j+1)$ dadurch berechnet, daß b und j heraufgezählt werden, bis b die Bedingung für n_j erfüllt. Wenn $j > \log|x|$, hält der Algorithmus. Der letzte berechnete Wert von f ist $f(i)$. Der Zeitaufwand ist höchstens $(\log f(j+1) - f(j)) \cdot T_2(\log f(j+1)) \le T_2(j+1)$ für fast alle j.

   Es ist nicht notwendig $f(i+1)$ zu berechnen. Somit wird i in $O(T_2)$ Schritten berechnet. Falls $i \cdot T_1(|x|) > T_2(|x|)$, verwirft die Maschine x, andernfalls simuliert M $2 \cdot T_1(|x|)$ Schritte der Maschine M_i auf Eingabe x. Nach Annahme kann dies in $i \cdot T_1(|x|) \le T_2(|x|)$ Schritten durchgeführt werden. Falls M_i die Eingabe x innerhalb dieser Anzahl von Schritten akzeptiert, verwirft M, andernfalls akzeptiert M.

2. $L \notin \mathrm{EavTime}(T_1, \{\mu_{\mathrm{uni}}\})$:

   Nehmen wir an, daß $L \in \mathrm{EavTime}(T_1, \{\mu_{\mathrm{uni}}\})$. Sei M_i eine Maschine, die L berechnet. Nach Definition der Folge n_i gilt für alle $x \in [n_i, 2n_i+1]$: $i \cdot T_1(|x|) \le T_2(|x|)$. Daher ist die erste Bedingung in der Definition von L immer für solche x erfüllt.

   Nun impliziert $L = L(M_i)$, daß für solche x der Fall $x \notin L(M_i)$ in der Definition nicht auftreten kann, somit $\mathrm{time}_{M_i}(x) > 2 \cdot T_1(|x|)$. Wir behaupten, daß die *average*-Zeitkomplexität von M_i bezüglich der uniformen Wahrscheinlichkeitsverteilung größer ist als T_1 für beide Maße, Eav und Av.

   Andernfalls erhalten wir für das Eav-Maß den nachstehenden Widerspruch. Sei $l = \exp(2n_i + 1) - 1$, dann gilt

$$\sum_{\mathrm{rank}_{\mathrm{uni}}(x) \le l} \frac{\mathrm{time}_{M_i}(x)}{T_1(|x|)} \quad \le \quad l \qquad \Rightarrow$$

$$\begin{aligned}
\sum_{\exp n_i \le \mathrm{rank}_{\mathrm{uni}}(x) < \exp(2n_i+1)} \frac{\mathrm{time}_{M_i}(x)}{T_1(|x|)} &\le l && \Rightarrow \\
\sum_{\exp n_i \le \mathrm{rank}_{\mathrm{uni}}(x) < \exp(2n_i+1)} 2 &\le \exp(2n_i+1) && \Rightarrow \\
2 \cdot (\exp(2n_i+1) - \exp n_i) &\le \exp(2n_i+1) && \Rightarrow \\
\exp(2n_i+1) &\le 2 \cdot \exp n_i \ .
\end{aligned}$$

3. $L \notin \mathrm{AvTime}(T_1, \{\mu_{\mathrm{uni}}\})$:

   Für das Av-Maß erhält man den analogen Widerspruch.

$$\begin{aligned}
\sum_{\mathrm{rank}_{\mathrm{uni}}(x) \le l} \frac{T_1^{-1}(\mathrm{time}_{M_i}(x))}{|x|} &\le l && \Rightarrow \\
\sum_{\exp n_i \le \mathrm{rank}_{\mathrm{uni}}(x) < \exp(n_i+1)} \frac{T_1^{-1}(\mathrm{time}_{M_i}(x))}{|x|} &\le l && \Rightarrow \\
\sum_{\exp n_i \le \mathrm{rank}_{\mathrm{uni}}(x) < \exp(2n_i+1)} \frac{|x|+1}{|x|} &\le \exp(2n_i+1) && \Rightarrow \\
\sum_{\exp n_i \le \mathrm{rank}_{\mathrm{uni}}(x) < \exp(2n_i+1)} 1 + \frac{1}{|x|} &\le \exp(2n_i+1) && \Rightarrow \\
(\exp(2n_i+1) - \exp n_i) \cdot \left(1 + \frac{1}{\exp n_i}\right) &\le \exp(2n_i+1) && \Rightarrow \\
2 \cdot \exp n_i &\le \exp n_i + 1 \ .
\end{aligned}$$

∎

Man beachte nun, daß wenn $V_1 \le V_2$ gilt, dann ist V_1–rankable $\subseteq V_2$–rankable und daher gilt

$$\mathrm{EavTime}(T,\, V_1\text{–rankable}) \supseteq \mathrm{EavTime}(T,\, V_2\text{–rankable}) \ .$$

Somit gilt für $V_1 \ge \mathcal{N}$

$$\begin{aligned}
\mathrm{DTime}(T) &\subseteq \mathrm{EavTime}(T, V_1\text{–rankable}) \\
&\subseteq \mathrm{EavTime}(T, \{\mu_{\mathrm{uni}}\})
\end{aligned}$$

und ähnlich für das andere Maß. Dies impliziert

Korollar 15 *Seien $T_1, V \ge (1+\epsilon)\mathcal{N}$ und $T_2 \in \omega(T_1)$ Zeitschranken. Dann gilt*

$$\begin{aligned}
\mathrm{EavTime}(T_1, V\text{–rankable}) &\subset \mathrm{EavTime}(T_2, V\text{–rankable}) \ , \\
\mathrm{AvTime}(T_1, V\text{–rankable}) &\subset \mathrm{AvTime}(T_2, V\text{–rankable}) \ .
\end{aligned}$$

4.3 Average-Hierarchien bezüglich Verteilungen

Im letzten Abschnitt beobachteten wir, daß wie im *worst-case* jede noch so geringe Zunahme der *average*-Zeitschranke der Turing-Maschinen die Berechnungsstärke erhöht. Jetzt untersuchen wir, wie es sich verhält, wenn die Zeitkomplexität der Wahrscheinlichkeitsverteilungen erhöht wird. Betrachten wir zum Beispiel die Klasse

$$\mathrm{AvTime}(\mathcal{N}^3,\ \mathcal{N}^2\text{–rankable})\ .$$

Verkleinern wir die Menge der Verteilungen auf $\mathcal{N}$–rankable, ist es wohl zu erwarten (wenn auch nicht zwingend), daß bezüglich $\mathcal{N}$–rankable mehr Sprachen in *average*-Zeit $\mathcal{N}^3$ erkannt werden können.

Vergrößert man dagegen die Menge der Verteilungen bis auf EXL–rankable, so wissen wir bereits, daß nun AvTime sich nicht mehr von DTime unterscheidet. Zwischen $\mathcal{N}^2$ und EXL ist aber unklar, wie sich die Zunahme der Komplexität der Verteilungen auf die Größe der Klasse AvTime auswirkt.

Wir werden daher die allgemeine Frage untersuchen, ob $\mathrm{AvTime}(T,\ V\text{–rankable})$ eine echte Teilmenge ist von $\mathrm{AvTime}(T,\ o(V)\text{–rankable})$. Die Lösung dieser Frage wird den technisch schwierigsten Beweis dieser Arbeit erfordern. Darum leisten wir zuerst einige Vorarbeit.

4.3.1 Diagonalisierung bezüglich Mengen von Rangfunktionen

Um die Technik der Diagonalisierung hier verwenden zu können, müssen wir folgendes Problem lösen: V–rankable, die Menge der Rangfunktionen der Zeitkomplexität V, ist nicht aufzählbar, noch ist es deren Komplement, wie folgendes Beispiel zeigt.

Beispiel:
Betrachten wir die Menge $R \subseteq \mathcal{N}$–rankable der Funktion $r_{(i,y)} : \{0,1\} \to \mathcal{N}$, wobei $i \in \mathcal{N}$ und $y \in \Sigma^*$, die definiert werden durch

$$r_{(i,y)}(x)\ :=\ \begin{cases} \frac{\mathrm{ord}(x)}{2}\ , & \text{falls } \mathrm{ord}(x) \text{ gerade,} \\ 1\ , & \text{falls } x \in \{0\}^* \\ & \quad \text{und } \mathrm{time}_{M_i}(y) = |x|, \\ \infty\ , & \text{sonst.} \end{cases}$$

Jede dieser Funktionen ist in linearer Zeit berechenbar, wenn man für „ ∞ “, wie schon früher vereinbart, ein Sondersymbol ausgibt.

$r_{(i,y)}$ ist aber genau dann eine Rangfunktion, wenn M_i auf Eingabe y nicht hält, weil ansonsten Rang 1 doppelt besetzt wäre. Somit ist $R \cap \mathcal{N}$–rankable nicht aufzählbar.

Betrachten wir nun $R' \subseteq \mathcal{N}$–rankable mit ähnlicher Funktion $r'_{(i,y)}$, definiert als

$$r'_{(i,y)}(x) := \begin{cases} 1 + \frac{\mathrm{ord}(x)}{2}\ , & \text{falls } \mathrm{ord}(x) \text{ gerade,} \\ 1\ , & \text{falls } x \in \{0\}^* \\ & \ \text{und } \mathrm{time}_{M_i}(y) = |x|, \\ \infty\ , & \text{sonst.} \end{cases}$$

$r_{(i,y)}$ ist genau dann eine Rangfunktion, wenn M_i auf eine Eingabe y hält. Der Rang 1 wäre ansonsten nicht besetzt. Folglich ist auch $R' \cap \overline{\mathcal{N}\text{–rankable}}$ nicht aufzählbar.

Wir umgehen diese Schwierigkeit durch Betrachtung einer Obermenge der Rangfunktionen, bestehend aus speziellen Funktionen, den **Pseudo-Rangfunktionen**.

Definition 24 *Eine* ***Pseudo-Rangfunktion*** *ist eine Funktion* $r : \Sigma^* \to \mathbb{N} \cup \{\infty\}$ *mit der Eigenschaft*

$$\forall l \in \mathbb{N} \quad |\{x \mid r(x) \leq l\}| \ \leq \ l\ .$$

Ein Beispiel für eine Pseudo-Rangfunktion, die keine Rangfunktion ist, ist die Funktion $n \mapsto n + 3$. Die Ränge 1, 2 und 3 werden niemals belegt. Andererseits werden in Pseudorangfunktionen nie Ränge überbesetzt.

Zwar kann man auch nicht alle Pseudo-Rangfunktionen aufzählen, wie direkt aus dem Beispiel der Menge R folgt. Auch kann man nicht das Komplement der Pseudo-Rangfunktionen aufzählen. Zumindestens kann man eine dynamische Menge von Kandidaten erzeugen, in der jede Maschine, die diese Eigenschaft erfüllt, irgendwann aufgenommen wird und jede Maschine, die diese Eigenschaft nicht erfüllt, ausgeschlossen wird. Dieses gewährleistet ein **Rang-Akkumulator**.

Definition 25 *Für eine Zeitschranke V ist ein* ***V-Rang-Akkumulator*** *eine Turing-Maschine M, die auf Eingabe 1^n die Ausgabe $I_n \subseteq \mathbb{N}$ berechnet. Hierbei muß für I_n gelten*

$$\forall\ i \in \mathbb{N} : \ M_i \text{ berechnet in Zeit } V \text{ eine Pseudo-Rangfunktion} \quad \Longleftrightarrow \quad \exists\ n_0 \quad \forall n \geq n_0 \qquad i \in I_n\ .$$

Diese Definition macht Sinn, denn es gilt

Lemma 12 *Für jede Komplexitätsschranke V existiert ein linear-zeitbeschränkter V-Rang-Akkumulator.*

Beweis: Sei $c : \mathbb{N} \to \mathbb{N}$ eine monotone, nicht beschränkte Funktion, die in linearer Zeit berechnet werden kann. Der folgende Algorithmus `RANG-AKKU` definiert einen V-Rang-Akkumulator, der auf Eingabe 1^n die Ausgabe I_n berechnet.

```
RANG-AKKU(x)
n ← |x| ;
I ← [1 ... c(n)] ;
for all i ∈ I
    for all x ∈ Σ^{≤c(n)}
        if time_{M_i}(x) > V(|x|)
            then I ← I \ {i}
            else R[i, x] ← M_i(x) ;
    for all l ≤ max_{x∈Σ^{≤c(n)}} R[i, x]
        if |{x|R[i, x] ≤ l}| > l then I ← I \ {i} ;
return I .
```

Nehmen wir an, M_j berechne eine Pseudo-Rangfunktion in Zeit $|V|$, dann ist $j \in I_{c^{[-1]}(j)}$. Aber auch für alle $n \geq c^{[-1]}(j)$ gilt in diesem Fall $j \in I_n$, da j niemals aus der Menge I durch den Algorithmus entfernt werden kann.

Würde M_j keine Pseudo-Rangfunktion berechnen, so gäbe es entweder ein x mit $\mathrm{time}_{M_j} > V(|x|)$ oder ein l mit $|\{x|M_j(x) \leq l\}| > l$. Beide Fälle führten für ein genügend großes n zum Ausschluß von j aus I_n.

Damit liegt die Richtigkeit dieses Algorithmus auf der Hand. Eine obere Schranke für den Zeitaufwand auf Eingabe 1^n ist

$$T_{V,c}(n) \;\leq\; O\Big(c(n) \cdot |\Sigma|^{c(n)} \cdot V(c(n))\Big)$$

Für jedes V kann man nun $\boldsymbol{c}$ derart monoton und unbeschränkt definieren, daß $T_{V,c}$ höchstens linear zunimmt und $c(n)$ in Linear-Zeit berechenbar ist. Die folgende Funktion erfüllt diese Eigenschaft:

$$\boldsymbol{c(n)} \;\; := \;\; \min\left\{V^{-1}(\sqrt{n}), \frac{\log n}{3\log|\Sigma|}\right\} . \qquad \blacksquare$$

Um die Klasse AvTime(T, V_1–rankable) von der Klasse AvTime(T, V_2–rankable) für $V_1 \leq o(V_2)$ zu separieren, benötigen wir mindestens eine Rangfunktion ρ mit $\rho \in V_2$–rankable$\setminus V_1$–rankable. Ein besonders langsam zunehmendes Exemplar ρ erhalten wir durch folgendes Lemma.

Lemma 13 *Für alle Komplexitätsschranken* V_1, V_2 *mit* $\mathcal{N} \leq V_1 \leq o(V_2)$ *gibt es eine nicht endliche Sprache* L *, so daß* $\mathrm{rank}_L \in V_2$–rankable *und*

$$\forall \rho \in V_1\text{–rankable} \quad \exists x_0 \forall x \geq x_0 \quad rank_L(x) < \infty \implies \mathrm{rank}_L(x)^2 < \rho(x) \; .$$

Beweis: Wir konstruieren L als unäre Sprache, das heißt als Teilmenge von $\{1\}^*$. Betrachten wir jedoch zuerst die streng monoton zunehmende Folge $1 = n_1 < n_2 < n_3 < \ldots$ mit $n_i \in \mathbb{N}$. Diese sollte folgende Eigenschaften erfüllen:

1. Für alle l gilt $2 \cdot V_2(n_l) \leq V_2(n_{l+1})$.
2. Für jedes $m \in \mathbb{N}$ kann das Element n_l mit $n_l \leq m < n_{l+1}$ in Zeit $O(V_2(m))$ berechnet werden.

Solch eine Folge kann man wie folgt erhalten. Angenommen n_l wurde schon definiert. Dann betrachten wir für $k = 1, 2, 3, \ldots$ die Folge $V_2(2^k \cdot n_l)$ bis zum ersten k, für das gilt $V_2(2^k \cdot n_l) \;\geq\; 2^k \cdot V_2(n_l)$.

Den Einwand, ein solches k würde nicht immer existieren, kann man folgendermaßen widerlegen. Angenommen, für alle k gelte

$$V_2(2^k \cdot n_l) < 2^k \cdot V_2(n_l) \; .$$

Dann gilt für die Summe

$$\begin{aligned}
\sum_{i=1}^{2^m} V_2(i) \;&=\; \sum_{i=1}^{n_l - 1} V_2(i) \;+\; \sum_{i=n_l}^{2^m} V_2(i) \\
&\leq\; O(1) \;+\; \sum_{i=1}^{m} 2^{i-1} \cdot V_2(2^i \cdot n_l) \\
&\leq\; O(1) \;+\; \sum_{i=1}^{m} 2^{i-1} \cdot 2^i \cdot V_2(n_l) \\
&\leq\; O(1) \;+\; 2^{2m} \cdot V_2(n_l) \;\leq\; O((2^m)^2) \; .
\end{aligned}$$

Damit ergibt sich aber der Widerspruch $V_2 \in O(\mathcal{N})$.

Also läßt sich ein solches k, nennen wir es hier $\bar{k}$, immer finden. Die Folge sei damit definiert als $n_{l+1} := 2^{\bar{k}} \cdot n_l$.

Diese Folge erfüllt somit die erste Bedingung. Da V_2 zeitkonstruierbar ist, kann für ein gegebenes n_l der nächste Wert n_{l+1} konstruiert werden in Zeit

$$\sum_{k=1}^{\bar{k}} V_2(2^k \cdot n_l) \;\leq\; \sum_{k=1}^{\bar{k}-1} 2^k \cdot V_2(n_l) + V_2(2^{\bar{k}} \cdot n_l) \;\leq\; 2 \cdot V_2(n_{l+1}) \; .$$

So kann die gesamte Folge $n_1, \ldots, n_{l+1}$ in Zeit $O(V_2(n_{l+1}))$ berechnet werden, und für $m = n_{l+1}$ gilt auch die zweite Bedingung. Für ein beliebiges m berechnen wir zuerst $V_2(m)$ und setzen ein Zeitlimit der Größe $O(V_2(m))$ fest. Wir fangen an, die Folge $n_1, n_2, n_3 \ldots$ zu berechnen, bis entweder n_{l+1} gefunden oder das Zeitlimit ausgeschöpft ist. Für diesen Fall ist der letzte berechnete Wert das gesuchte n_l.

Der folgende Algorithmus `RANKL` berechnet rank_L und definiert so L. Hierfür benutzt er die Folge n_l. Wir definieren also $x_l := 1^{n_l}$.

Auf Eingabe x_l generiert `RANKL` ein Tripel (j, k, r), wobei r den Rang von x_l in L beschreibt. Somit ist $x_l \in L$ gdw. $r < \infty$. j und k sind zusätzliche Parameter, die für dieses rekursive Programm benötigt werden.

`RANKL` (x)
$m \leftarrow |x|$;
<u>if</u> $\forall l \quad n_l \neq m$ oder $x \neq 1^m$ <u>then</u> <u>return</u> $(-, -, \infty)$;
$l \leftarrow$ dasjenige l' mit $x = x_{l'}$;
<u>if</u> $l = 1$
    <u>then</u> <u>return</u> $(1, 1, 1)$
    <u>else</u> $(r, k, \cdot)$:= `RANKL`(x_{l-1});
$I_k \leftarrow$ Ausgabe des V_1-Rank-Akkumulators auf Eingabe 1^k;
$I \leftarrow I_k \cap [1..b]$, wobei $b := \min\left\{r, \sqrt{\frac{V_2(m)}{6 \cdot V_1(m)}}\right\}$;
<u>if</u> $\forall i \in I \quad (\text{time}_{M_i}(x) \leq V_1(|x|)$ und $M_i(x) < \infty) \Rightarrow M_i(x) > (r+1)^2$
    <u>then</u> <u>return</u> $(r+1, k, r+1)$
    <u>else</u> <u>return</u> $(r, k+1, \infty)$.

Wir werden uns nun noch davon überzeugen, daß dieser Algorithmus eine Sprache L konstruiert, die oben genannte Eigenschaften besitzt.

1. $\text{rank}_L \in V_2$–rankable:
   Gehen wir von der Induktionsannahme aus, der Zeitaufwand von `RANKL` auf Eingaben aus $\Sigma^{\leq n_{l-1}}$ sei beschränkt durch V_2.

   Betrachten wir nun $x = 1^m$ mit $m > n_{l-1}$.

   Aufgrund der Konstruktion der Folge n_i ist es möglich, l in Zeit $\frac{1}{6}V_2(m)$ so zu bestimmen, daß $n_{l-1} < m \leq n_l$.

   Für $m = n_l$ ist der Zeitaufwand des rekursiven Aufrufs von `RANKL`(x_{l-1}) beschränkt durch $V_2(n_{l-1}) \leq \frac{1}{2}V_2(n_l)$.

   Die Ausgabe des Rangakkumulators kann auch in Zeit $\frac{1}{6}V_2(n_l)$ erhalten werden.

Für die Laufzeit aller Simulationen der Maschinen, gegeben durch I, wird nun höchstens folgendes benötigt.

$$b^2 \cdot V_1(m) \;\leq\; \frac{V_2(m)}{6 \cdot V_1(m)} \cdot V_1(m) \;=\; \frac{1}{6} V_2(n_l)\ .$$

Dies zeigt, daß rank_L in Zeit V_2 berechnet werden kann.

2. L ist nicht endlich:
`RANKL` berechnet auf eine Eingabe x_l einen Rang $r \neq \infty$.

   Solange der nächste Kandidat $x = x_{l+1}, x_{l+2} \ldots$ keinen endlichen Rang erhält, bleibt die Variable r fest und k wird jedesmal erhöht. Nach der Definition eines V_1-Rang-Akkumulators wird für ein k die Menge $I = I_k \cap [1..b]$ mit $b = \min\left\{r, \sqrt{\frac{V_2(m)}{6 \cdot V_1(m)}}\right\}$ nur noch Indizes von Maschinen enthalten, die Pseudo-Rangfunktionen berechnen.

   Dann können alle Pseudorangfunktionen die Bedingung „Ausgabe“ $> (r+1)^2$ nur endlich oft verletzen, da jetzt r konstant bleibt.

   So muß letztendlich der Rang $r+1$ einer Eingabe $x_{l'}$ zugeordnet werden.

3. $\forall \rho \in V_1$–rankable $\quad \exists x_0 \forall x \geq x_0 \quad rank_L(x) < \infty \;\Rightarrow\; (\mathrm{rank}_L(x))^2 < \rho(x)$.

   Sei j der Index einer V_1-zeitbeschränkten Turing-Maschine M, die ρ berechnet. Da L nicht endlich ist, kann die Variable r groß genug werden, so daß $j \leq \min\left\{r, \sqrt{\frac{V_2(m)}{6 \cdot V_1(m)}}\right\}$. Es wird für eine genügend große Eingabe x_0 die Zahl j als Ergebnis des V_1-Rank-Akkumulators in I aufgenommen, da jede Rangfunktion eine Pseudo-Rangfunktion ist.

   Da j für größere Eingaben $x_l > x_0$ in I vorhanden bleibt, sind alle auszugebenden endlichen Ränge dann kleiner als $\sqrt{M_i(x)} = \sqrt{\rho(x)}$.

∎

Diese eben beschriebene Sprache hat die interessante Eigenschaft, daß ihr Schnitt mit einer beliebigen Sprache $P \in \mathrm{DTime}(T_2)$ in $\mathrm{EavTime}(T_1, V_1\text{–rankable})$ liegt, für ein geeignet gewähltes $T_2 \in \omega(T_1)$. Um dies zu zeigen, erinnern wir uns an die Definition von $\boldsymbol{\nu_\rho}$ auf Seite 48

$$\boldsymbol{\nu_\rho(n)} \quad := \max_{|x| \leq n \text{ und } \rho(x) \neq \infty} \rho(x)$$

und definieren jetzt $\boldsymbol{\nu_L} := \nu_{\mathrm{rank}_L}$ für L-uniforme Wahrscheinlichkeitsverteilungen μ_L

$$\boldsymbol{\nu_L(n)} \quad := \max_{|x| \leq n \text{ und } \mathrm{rank}_L(x) \neq \infty} \mathrm{rank}_L(x)\ .$$

Lemma 14 *Seien T_1, V_1, V_2 Komplexitätsschranken mit $\mathcal{N} \leq V_1 \leq o(V_2)$ und $V_2 \leq O(T_1)$. Dann gibt es eine Sprache $L \in DTime(V_2)$ und eine Komplexitätsschranke $T_2 \in \omega(T_1)$, so daß*

1. $\mu_L \in V_2$–rankable $\setminus V_1$–rankable *und*

2. $P \cap L \in$ EavTime(T_1, V_1–rankable) *, für alle* $P \in$ DTime(T_2) .

Beweis: Sei L eine Sprache wie in Lemma 13 beschrieben. Ferner sei $T_2(n) := T_1(n) \cdot \nu_L(n)$. Überprüfen wir nun die Eigenschaften von L.

1. $\mu_L \in V_2$–rankable$\setminus V_1$–rankable folgt sofort aus Lemma 13.

2. $\forall P \in$ DTime(T_2) $\quad P \cap L \in$ EavTime(T_1, V_1–rankable):

   Sei $\rho \in V_1$–rankable, und sei $P \in$ DTime(T_2). Ein Algorithmus, der auf Eingabe x die Frage $x \in P \cap L$ entscheidet, berechnet folgendes.

   - Die endlich vielen Eingaben x mit $\rho(x) \leq \mathrm{rank}_L(x)^2$ wurden in einer Tabelle im Vorhinein angelegt, um in Linearzeit das Resultat zu erhalten. Eine Turing-Maschine kann diese Eingaben aus dem endlichen Gedächtnis beantworten.
   - Andernfalls wird getestet, ob $x \in L$. Falls nicht, verwirft der Algorithmus. Da $L \in$ DTime(V_2) und nach Voraussetzung $V_2 \leq O(T_1)$, kann diese Frage in Zeit $T_1(|x|)$ entschieden werden (bei Bedarf muß eine linear beschleunigte Turing-Maschine verwendet werden).
   - Im übriggebliebenen Fall gilt $x \in L$, damit $\mathrm{rank}_L(x) < \infty$ und zusätzlich $\rho(x) > \mathrm{rank}_L(x)^2$.

     Hier wird nun ein T_2-zeitbeschränkter Algorithmus benutzt, um zu entscheiden, ob $x \in P$ ist.

   Für die Abschätzung der *average*-Zeitkomplexität der Turing-Maschine M, die diesen Algorithmus ausführt, erhalten wir für alle $l \in \mathbb{N}$

$$\begin{aligned}
\sum_{\rho(x)\leq l} \frac{\mathrm{time}_M(x)}{2\cdot T_1(|x|)} &\leq \sum_{\substack{\rho(x)\leq l \text{ und} \\ \rho(x)\leq \mathrm{rank}_L(x)^2}} \frac{T_1(|x|)}{2\cdot T_1(|x|)} \\
&\quad + \sum_{\mathrm{rank}_L(x)^2<\rho(x)\leq l} \frac{T_2(|x|)}{2\cdot T_1(|x|)} \\
&\leq \frac{l}{2} + \sum_{\mathrm{rank}_L(x)<\sqrt{l}} \frac{T_1(|x|)\cdot \nu_L(|x|)}{2\cdot T_1(|x|)} \\
&\leq \frac{l}{2} + \frac{1}{2}\cdot \sum_{i=1}^{\lfloor\sqrt{l}\rfloor-1} i \leq l ,
\end{aligned}$$

da $\nu_L \leq \mathcal{N}$. Somit ist $(\mathrm{time}_M, \rho) \in \mathrm{Eav}\,(2 \cdot T_1)$. Durch eine lineare Beschleunigung von M kann nun eine Maschine M' gefunden werden, für die gilt $L(M') = L(M) = P \cap L$ und $(\mathrm{time}_{M'}, \rho) \in \mathrm{Eav}\,(T_1)$.

∎

Nun wollen wir den äquivalenten Fall für das Av-Maß untersuchen.

Lemma 15 *Sei* $\delta > 1$ *und* T_1, V_1, V_2 *Komplexitätsschranken mit* $\mathcal{N} \leq V_1 \leq o(V_2)$ *und* $V_2(\delta \cdot \mathcal{N}) \leq O(T_1)$ *. Dann gibt es eine Sprache* $L \in DTime(V_2)$ *und eine Komplexitätsschranke* $T_2 \in \omega(T_1)$ *, so daß*

1. $\mathrm{rank}_L \in$ V_2-rankable $\setminus$ V_1-rankable *und*

2. $P \cap L \in$ AvTime(T_1, V_1-rankable) *, für alle* $P \in \mathrm{DTime}(T_2)$.

Beweis: Die Sprache L sei definiert wie im letzten Beweis. Die Zeitschranke $\boldsymbol{T_2}$ wird hier definiert durch

$$\boldsymbol{T_2(n)} \quad := \quad T_1\left(n \cdot \nu_L(n) \cdot \left(1 - \delta^{-1}\right)\right) \ .$$

Es gilt wiederum $\mathrm{rank}_L \in V_2$-rankable $\setminus$ V_1-rankable .

Es bleibt jetzt noch zu zeigen, daß für alle $P \in \mathrm{DTime}(T_2)$

$$P \cap L \in \ \mathrm{AvTime}(T_1,\ V_1\text{-rankable})\ .$$

Für eine gegebene Sprache $P \in \mathrm{DTime}(T_2)$ und für eine gegebene Rangfunktion $\rho \in$ V_1-rankable konstruieren wir eine Maschine M ähnlich wie im letzten Beweis.

- Für die endlich vielen x mit $\rho(x) \leq \mathrm{rank}_L(x)^2$ wird die Antwort in das endliche Gedächtnis der Maschine kodiert und damit in linearer Zeit erhältlich.

- M testet, ob $x \in L$ und verwirft, falls nicht. Das kostet höchstens Zeit $V_2 \leq O(T_1(\mathcal{N}/\delta))$.

- Im verbleibenden Fall $\mathrm{rank}_L(x) < \infty$ und $\rho(x) > \mathrm{rank}_L(x)^2$ entscheiden wir $x \in P$ in höchstens $T_2(|x|)$ Schritten.

Dann gilt für alle $l \in \mathbb{N}$

$$\begin{aligned}
\sum_{\rho(x)\leq l} \frac{{T_1}^{-1}(\mathrm{time}_M(x))}{|x|} &\leq \sum_{\substack{\rho(x)\leq l \text{ und} \\ \rho(x)\leq \mathrm{rank}_L(x)^2}} \frac{{T_1}^{-1}(T_1(|x|/\delta))}{|x|} \\
&\quad + \sum_{\mathrm{rank}_L(x)^2<\rho(x)\leq l} \frac{{T_1}^{-1}(T_2(|x|))}{|x|} \\
&\leq \frac{l}{\delta} + \sum_{\mathrm{rank}_L(x)\leq\sqrt{l}-1} \frac{\nu_L(|x|)\cdot|x|\cdot(1-\frac{1}{\delta})}{|x|} \\
&\leq \frac{l}{\delta} + \left(1-\frac{1}{\delta}\right)\cdot\sum_{i=1}^{\lfloor\sqrt{l}\rfloor-1} i \;\leq\; l\ .
\end{aligned}$$

Daher ist $(\mathrm{time}_M, \mathrm{rank}_L) \in \mathrm{Av}\,(T_1)$. ∎

4.3.2 Die Separation

Als nächstes zeigen wir, daß die Sprache $P \cap L$ zwar in $\mathrm{EavTime}(T_1,\, V_1\text{–rankable})$ liegt, aber bezüglich rank_L nicht von einer $\mathrm{Eav}\,(T_1)$ -zeitbeschränkten Turing-Maschine berechnet werden kann.

Lemma 16 *Seien $\mathcal{N} \leq V_2 \leq o(T_1)$ und $T_1 \leq o(T_2)$ Komplexitätsschranken. Dann gibt es für alle $\delta > 1$ und für alle nicht endlichen Sprachen $L \in \mathrm{DTime}(V_2)$ eine Sprache $P \in \mathrm{DTime}(T_2)$, für die gilt*

$$(P \cap L, \mathrm{rank}_L) \;\notin\; \mathrm{EavDisTime}(T_1)\ .$$

Beweis: Wir definieren $f : \mathbb{N} \to \mathbb{N}$ rekursiv durch

$$\begin{aligned}
\boldsymbol{f(0)} &:= 1\ , \\
\boldsymbol{n_i} &:= \min\left\{ b \geq f(i) \;\middle|\; \forall\, j \in [b; \nu_L^{-1}(\nu_L(b) + 2^{2b+1})] :\ \frac{T_2(j)}{T_1(j)} \geq i \right\}\ , \\
\boldsymbol{f(i+1)} &:= 2^{2n_i+1}\ .
\end{aligned}$$

Da $T_1 \leq o(T_2)$ gilt, ist die Funktion f wohl definiert.

Außerdem definieren wir eine Diagonalisierungssprache $\boldsymbol{P}$ ähnlich der im Beweis von Theorem 12 durch

$$\begin{aligned}
\boldsymbol{P} \;:=\; \{x \mid\ & \text{für dasjenige } i \text{ mit } f(i) \leq |x| < f(i+1) \text{ gilt:} \\
& i \cdot T_1(|x|) \leq T_2(|x|) \quad \text{und} \\
& [\mathrm{time}_{M_i}(x) > 2 \cdot T_1(|x|) \quad \text{oder} \quad x \notin L(M_i)]\ \}\ .
\end{aligned}$$

P erfüllt die geforderten Eigenschaften, denn es gilt

1. $P \in \mathrm{DTime}(T_2)$.

   Auf Eingabe x berechnen wir zuerst alle $f(j)$ und n_j, die kleiner sind als $\log |x|$. Wegen des starken Wachstums von f und der Zeitkonstruierbarkeit aller Zeitschranken, kann dies in Zeit $T_2(|x|)$ erledigt werden. Wir berechnen dasjenige i mit $f(i) \leq |x| < f(i+1)$ in Zeit $O(|x|)$ unter Benutzung des zuletzt berechneten n_i. Nun testen wir, ob $i \cdot T_1(|x|) \leq T_2(|x|)$. Falls nein, verwerfen wir. Andernfalls simulieren wir $2 \cdot T_1(|x|)$ Schritte der Maschine M_i.

   Wir akzeptieren genau dann, wenn M_i nicht innerhalb dieses Zeitraums akzeptiert. Diese Simulation kostet höchstens $2 \cdot T_1(|x|) \cdot i \leq 2 \cdot T_2(|x|)$ viel Zeit.

2. $(P \cap L, \mathrm{rank}_L) \notin \mathrm{EavDisTime}(T_1)$.

   Für einen Widerspruchsbeweis nehmen wir an, daß M_i die Sprache $P \cap L$ in erwarteter Zeit T_1 bezüglich der Rangfunktion rank_L berechnet.

   Nach Definition von n_i und ν_L gilt für alle i

   $$|\{x \in L|\ f(i) \leq |x| < f(i+1) \text{ und } T_1(|x|) \cdot i \leq T_2(|x|)\}| \geq \nu_L(n_i) + 2^{2n_i+1}\ .$$

   Weil wir Turing-Maschinen durch eine Gödel-Numerierung aufzählen, existieren unendlich viele Maschinenindizes $i_1, i_2 \ldots$ mit $L(M_{i_k}) = P$. Daher ist die Sprache $P \cap L$ nicht endlich.

   Für Eingaben $x \in P \cap L$ muß $x \in L(M_i)$ gelten, und deswegen ist $\mathrm{time}_{M_i} > 2 \cdot T_1(|x|)$.

   Sei $l := \nu_L(n_i) + 2^{2n_i+1}$. Dann gilt $l \leq 2^{2n_i+2}$, da $\nu_L(n_i) \leq n_i$. Die Zeitkomplexität von M_i kann nun abgeschätzt werden durch

   $$\begin{aligned} \sum_{\mathrm{rank}_L(x) \leq l} \frac{\mathrm{time}_{M_i}(x)}{T_1(|x|)} &\geq \sum_{\nu_L(n_i) \leq \mathrm{rank}_L(x) \leq \nu_L(n_i) + 2^{2n_i+1}} \frac{2 \cdot T_1(|x|)}{T_1(|x|)} \\ &= \left(\nu_L(n_i) + 2^{2n_i+1} - \nu_L(n_i) + 1\right) \cdot 2 \\ &\geq 2^{2n_i+2} + 1 \ \geq\ l+1\ , \end{aligned}$$

   was der Annahme widerspricht, M_i ist $\mathrm{Eav}\,(T_1)$ -zeitbeschränkt bezüglich rank_L. ∎

Das folgende Lemma beschreibt den gleichen Sachverhalt für das Av-Maß.

Lemma 17 *Seien $\delta > 1$, $V_2 \geq \mathcal{N}$, $V_2(\delta \cdot \mathcal{N}) \leq T_1$ und $T_1 \leq T_2(o(\mathcal{N}))$ Komplexitätsschranken. Dann gibt es für alle nicht endlichen Sprachen $L \in \mathrm{DTime}(V_2)$ eine Sprache $P \in \mathrm{DTime}(T_2)$, für die gilt*

$$(P \cap L, \mathrm{rank}_L) \notin \mathrm{AvDisTime}(T_1)\ .$$

Beweis: Wir definieren eine Folge $\boldsymbol{n_i}$ und eine Funktion $\boldsymbol{f}$ ähnlich der im letzten Beweis.

$$\begin{aligned}
\boldsymbol{f(0)} &:= 1\,, \\
\boldsymbol{n_i} &:= \min\left\{ b \geq f(i) \;\middle|\; \forall\, j \in [b \ldots \nu_L^{-1}(\nu_L(b) + 2^{2b+1})] : \frac{T_2(j)}{T_1(2 \cdot j)} \geq i \right\}\,, \\
\boldsymbol{f(i+1)} &:= 2^{2n_i+1}\,.
\end{aligned}$$

Hier ist die Bedingung $\frac{T_2(j)}{T_1(j)} \geq i$ durch $\frac{T_2(j)}{T_1(2 \cdot j)} \geq i$ ersetzt worden. Da jetzt $T_1 \leq T_2(o(\mathcal{N}))$, ist f wiederum wohl definiert.

Die Diagonalisierungssprache $\boldsymbol{P}$ ist definiert als

$$\begin{aligned}
P \;:=\; \{x \mid m\text{für dasjenige } i \text{ mit } f(i) \leq |x| < f(i+1) \text{ gilt:} \\
i \cdot T_1(2 \cdot |x|) \leq T_2(|x|) \quad \text{und} \\
[\text{time}_{M_i}(x) > T_1(2 \cdot |x|) \quad \text{oder} \quad x \notin L(M_i)]\ \}\ .
\end{aligned}$$

1. $P \in \text{DTime}(T_2)$.

   Die veränderte Definition von f und die Konstruktion von P ermöglichen es einer Turing-Maschine, analog zum vorherigen Beweis die Sprache P innerhalb der Zeitschranke T_2 zu entscheiden.

2. $(P \cap L, \text{rank}_L) \;\notin\; \text{AvDisTime}(T_1,)$.

   Angenommen es gibt eine Turing-Maschine M_i, die $P \cap L$ in Av-Zeit T_1 bezüglich rank_L berechnet. Dann definieren wir $l := \nu_L(n_i) + 2^{2n_i+1}$ und erhalten den folgenden Widerspruch.

$$\begin{aligned}
\sum_{\text{rank}_L(x) \leq l} \frac{{T_1}^{[-1]}(\text{time}_{M_i}(x))}{|x|} \;&\geq \sum_{\nu_L(n_i) \leq \text{rank}_L(x) \leq \nu_L(n_i) + 2^{2n_i+1}} \frac{{T_1}^{[-1]}(T_1(2 \cdot |x|))}{|x|} \\
&= \left(\nu_L(n_i) + 2^{2n_i+1} - \nu_L(n_i) + 1\right) \cdot 2 \\
&\geq 2^{2n_i+2} + 1 \;\geq\; l + 1\,.
\end{aligned}$$

∎

Dank dieser Vorüberlegungen erhalten wir nun die folgenden Separationen für EavTime und AvTime hinsichtlich der Komplexität der Verteilungsklassen.

Theorem 13 *Sei $\delta > 1$. Dann gilt für alle Komplexitätsschranken T, T', V_1 und V_2 mit $\mathcal{N} \leq V_1 \leq o(V_2)$, $V_2 \leq O(T)$ und $V_2(\delta \cdot \mathcal{N}) \leq O(T')$*

$$\begin{aligned}
\text{EavTime}(T,\ V_2\text{–rankable}) \;&\subset\; \text{EavTime}(T,\ V_1\text{–rankable})\,, \\
\text{AvTime}(T',\ V_2\text{–rankable}) \;&\subset\; \text{AvTime}(T',\ V_1\text{–rankable})\,.
\end{aligned}$$

Beweis: Wir wählen $T = T_1$, wobei T_1 die in den vorherigen Lemmas betrachtete Zeitschranke ist. Die in Lemma 14 und 16 konstruierte Sprache $P \cap L$ ist in der *average*-Klasse EavTime(T_1, V_1–rankable) gemäß Lemma 14 enthalten, aber nicht in EavTime(T, V_2–rankable) nach Lemma 16, da $\text{rank}_L \in V_2$–rankable (Lemma 14). Den analogen Schluß für das Av-Maß erhält man durch Anwenden von Lemma 15 und 17 für $T' = T_1$. ∎

Die etwas ungewohnte Beziehung $\exists \delta > 1\ V_2(\delta \cdot \mathcal{N}) \leq O(T')$ ist für $T' \in$ POL äquivalent zu $V_2 \leq O(T')$. Damit erhalten wir für $T \in$ POL im Av- und Eav-Maß gleiche Separationsresultate.

Korollar 16 *Für Komplexitätsschranken* $\mathcal{N} \leq V_1 \leq o(V_2)$, $V_1 \leq O(T)$ *und* $T \in$ POL

$$\text{AvTime}(T, V_2\text{–rankable}) \subset \text{AvTime}(T, V_1\text{–rankable}) .$$

Abbildung 4.2 und 4.3 geben einen Eindruck der in den letzten beiden Theoremen beschriebenen Hierarchien. Jeder Punkt $\bullet$ im Diagramm präsentiert eine Komplexitätsklasse EavTime(T, V–rankable) respektive AvTime(T, V–rankable). $\bullet \subset \bullet$ heißt echte Teilmengenrelation zwischen den beiden Klassen, $\bullet = \bullet$ Gleichheit und $\bullet\ ?\ \bullet$, daß die Art der Relation noch offen ist.

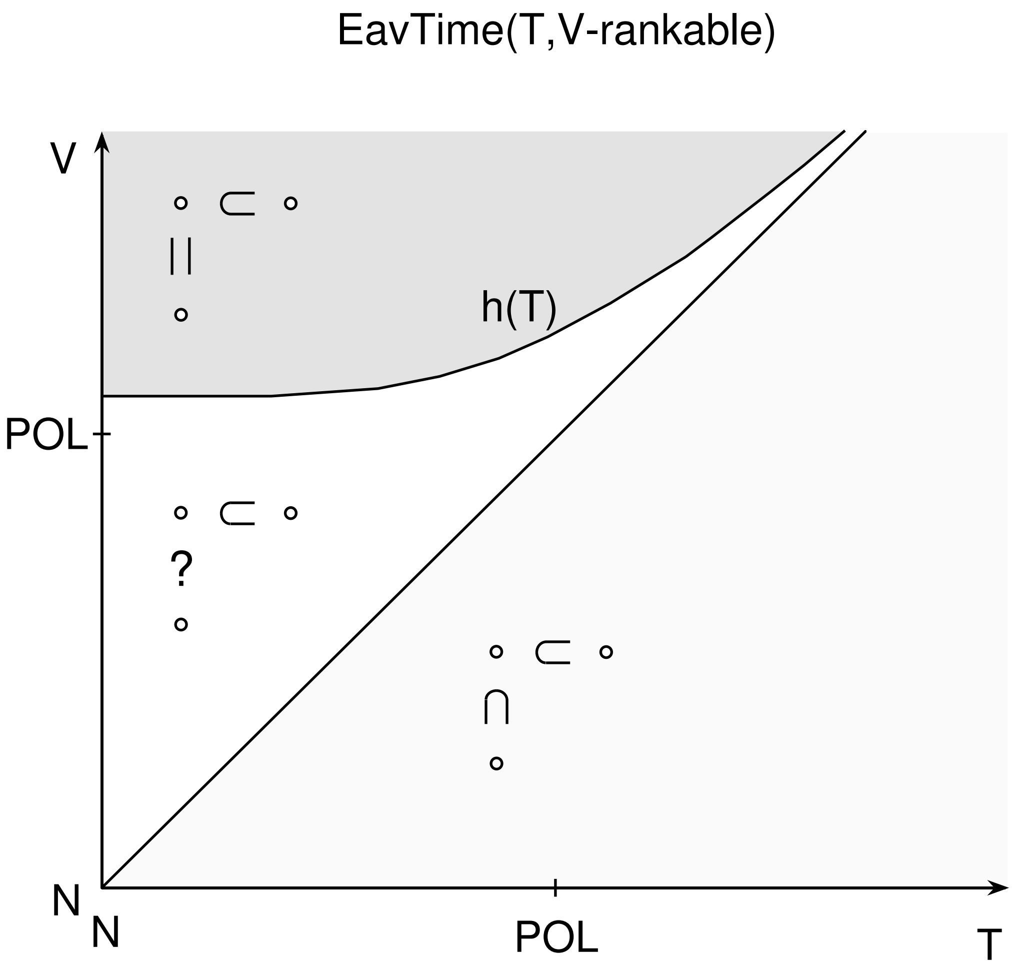

Abbildung 4.2: Die Hierarchien der *expected*-*average*-Komplexitätsklassen.

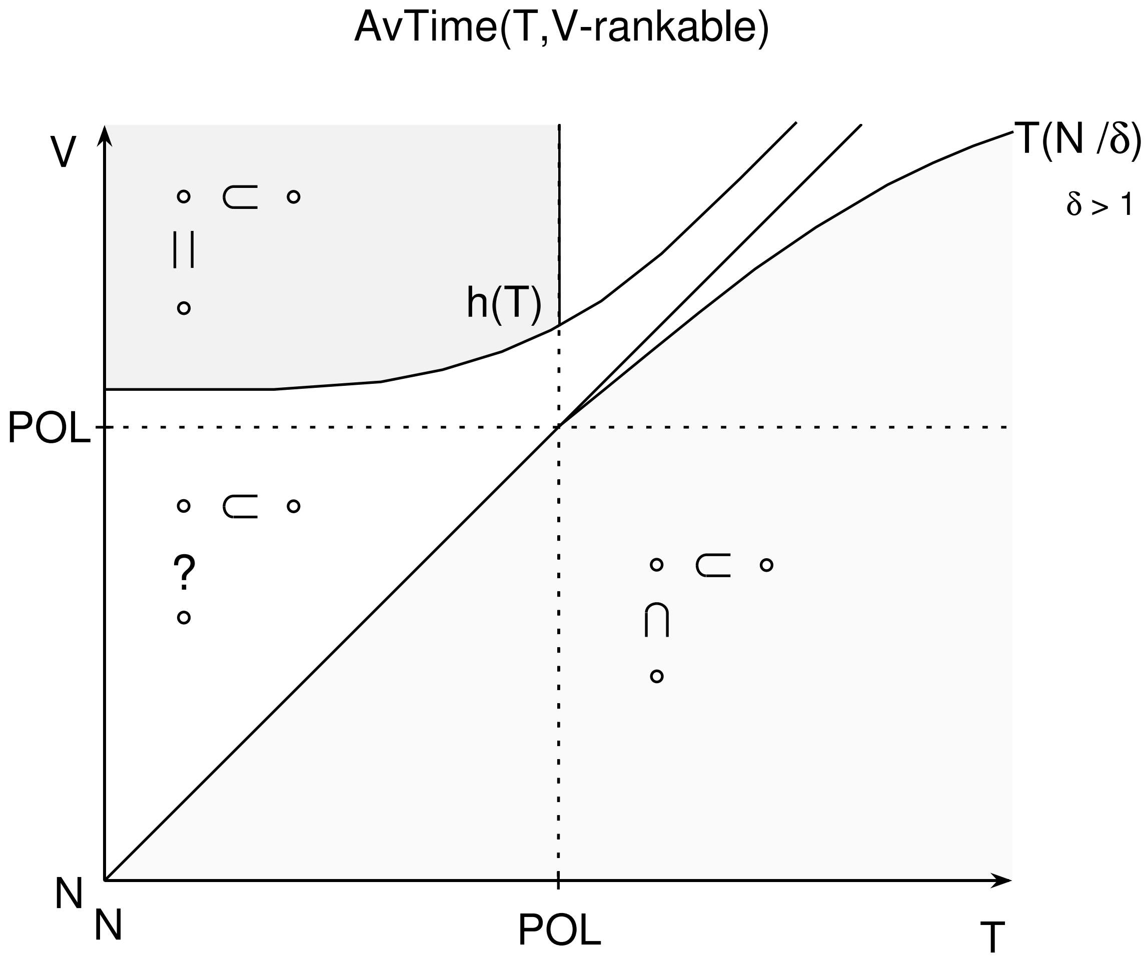

Abbildung 4.3: Die Hierarchien der *average*-Komplexitätsklassen.

Kapitel 5

Median-Komplexitätsklassen

5.1 MedDisTime

In den vorherigen Kapiteln untersuchten wir hauptsächlich das Av- und das Eav-Maß, sowie die dadurch definierten *average*-Komplexitätsklassen. Das Median-Maß wurde dort mit diesen *average*-Maßen verglichen. Es zeigte sich, daß eine eingehende Median-Klassifikation eine *average-case*-Analyse durchaus ersetzen kann.

Darüber hinaus besitzt der Median aber eine eigenständige Bedeutung. Aufgrund der Verwandtschaft zu Av und Eav kann man entsprechende **Median-Komplexitätsklassen** analog der *average*-Komplexitätsklassen definieren.

Definition 26 *Für eine Zeitschranke T und eine Häufigkeitsfunktion $\zeta : \mathbb{N} \to \mathbb{N}$ mit $0 \leq \zeta \leq \mathcal{N}$ definieren wir*

$$\begin{aligned}\mathbf{MedDisTime}(\boldsymbol{T},\boldsymbol{\zeta}) \quad := \quad & \{(L,\mu) \mid \exists M \in \mathrm{DTM} \quad L(M) = L \\ & \qquad \text{und } (\mathrm{time}_M, \mu) \in \mathrm{Med}(T,\zeta)\} \ .\end{aligned}$$

Betrachtet man diese Definition näher, so erkennt man den entscheidenden Unterschied zwischen *average*- und Median-Komplexitätsklassen. Überschreitet hier die Maschine M die Schranke T auf einer Eingabe x, ist es egal, ob sie nun $T(|x|)+1$ oder gar $\mathrm{itexp}(|x|)$ Schritte benötigt. Bei den *average*-Klassen $\mathrm{Av}\,(T)$ oder $\mathrm{Eav}\,(T)$ dagegen wäre eine Laufzeit-Überschreitung von höchstens $\exp(T(|x|))$ oder $\exp(|x|) \cdot T(|x|)$ möglich gewesen.

Könnte dann eine Turing-Maschine in einer Median-Komplexitätsklasse auch auf einer Eingabe unendlich viel Zeit benötigen?

Nicht nach dieser Definition, da explizit gefordert wird: $L(M) = L$. Es wäre aber möglich, wenn man nur fordert $M(x) = \chi_L(x)$ für $M(x) \neq \perp$ und diesbezüglich die Definition der Zeit $\widehat{\mathbf{time}}_M$ einer Turing-Maschine erweitert.

$$\widehat{\mathbf{time}}_M(x) \quad := \quad \begin{cases} \mathrm{time}_M(x)\ , & \text{falls } M(x) \in \{0,1\}\ , \\ \infty\ , & \text{falls } M(x) = \perp\ . \end{cases}$$

Wenn die Turing-Maschine M nun die Zeit T überschreitet, so kann M bezüglich dieser Definition sich auf zwei grundsätzlich verschiedene Weisen verhalten:

1. M rechnet noch eine gewisse Zeit und entscheidet schließlich, ob $x \in L$ und gibt das entsprechende Ergebnis aus.
2. M trifft keine Entscheidung und hält nie.

Beide Fälle fielen bei solch einer Median-Analyse gleich ins Gewicht. Wenn der Algorithmus nicht innerhalb der Zeitschranke T zu einem Ergebnis kommt, wäre es wohl einfacher den Befehl „stürze ab“ ausführen zu lassen, ohne den Versuch zu unternehmen das Problem für diesen Fall zu lösen.

Wenn es aber nun freigestellt ist, in schwierigen Situationen anstatt der Ausgabe 0 oder 1 quasi ein Fragezeichen (durch absichtlich herbeigeführten Absturz) auszugeben, erscheint es doch sinnvoller zu erlauben, eine Ausgabe beliebig zu wählen, auch unter der Gefahr, daß diese dann falsch ist.

Derartige Fehler würden wir dann der Zeitschranke genauso in Rechnung stellen wie eine Laufzeit-Überschreitung. Hierfür definieren wir die Zeitfunktion **time** in Abhängigkeit von der zu akzeptierenden Sprache $L \subseteq \Sigma^*$ oder der zu berechnenden Funktion $f : \Sigma^* \to \Sigma^*$ als

Definition 27 *Für eine Turing-Maschine M und eine Sprache $L \subseteq \Sigma^*$ oder eine Funktion $f : \Sigma^* \to \Sigma^*$ definieren wir*

$$\mathbf{time}_{M,L}(x) \quad := \quad \begin{cases} \mathrm{time}_M(x)\ , & \textit{falls } M(x) = \chi_L(x)\ , \\ \infty\ , & \textit{falls } M(x) \neq \chi_L(x)\ ; \end{cases}$$

$$\mathbf{time}_{M,f}(x) \quad := \quad \begin{cases} \mathrm{time}_M(x)\ , & \textit{falls } M(x) = f(x)\ , \\ \infty\ , & \textit{falls } M(x) \neq f(x)\ . \end{cases}$$

Die diesem Zeitmaß zugehörige Median-Komplexitätsklasse heißt **Med* DisTime**. Wir sprechen in Zukunft hier vom **Med*-Maß** oder **Median-Fehler-Maß**, um dieses von dem fehlerfreien Median-Maß unterscheiden zu können.

Definition 28 *Für eine Zeitschranke* T *und für eine Funktion* $\zeta : \mathbb{N} \to \mathbb{N}$ *mit* $0 \le \zeta \le \mathcal{N}$ *definieren wir*

$$\mathbf{Med^*DisTime}(\boldsymbol{T}, \boldsymbol{\zeta}) \quad := \quad \{(L,\mu) \mid \exists M \in \mathrm{DTM}\ (\mathrm{time}_{M,L}, \mu) \in \mathrm{Med}(T,\zeta)\}\ .$$

Die Hauptargumente in der Diskussion, warum nun solche Komplexitätsklassen, die auch fehlerhaft berechnete Probleme umfassen, von Interesse sind, erinnern an die Begründung von Monte-Carlo- und Las-Vegas-Algorithmen. Betrachten wir also das folgende Beispiel.

Beispiel:
Angenommen, wir hätten Sprachen L_1, L_2, $L_3 \subseteq L_{\mathrm{Pal}}$, wobei L_{Pal} die schon früher betrachtete Sprache aller binären Palindrome ist. Ferner sei L_1 genau in exponentieller Zeit lösbar, L_2 in doppelt exponentieller Zeit und L_3 nicht berechenbar.

Die Lage nach klassischer *worst-case*-Sicht ist klar:

$$\begin{aligned} L_1 &\in \mathrm{DTime}(\mathrm{EXP}) \setminus \mathrm{DTime}(o(\mathrm{EXP}))\ , \\ L_2 &\in \mathrm{DTime}(\mathrm{EEXP}) \setminus \mathrm{DTime}(o(\mathrm{EEXP}))\ , \\ L_3 &\notin \mathcal{REC}\ . \end{aligned}$$

Nehmen wir weiter an, daß dieses Problem ausschließlich im Zusammenhang mit der uniformen Wahrscheinlichkeitsverteilung auftritt. Somit betrachten wir die gewichteten Probleme $P_1 := (L_1, \mu_{\mathrm{uni}})$, $P_2 := (L_2, \mu_{\mathrm{uni}})$ und $P_3 := (L_3, \mu_{\mathrm{uni}})$.

Es ist nicht schwer zu zeigen, daß dann gilt

$$L_1 \in \mathrm{AvDis}\,\mathcal{P}\ , \quad \text{aber auch} \quad L_2 \notin \mathrm{AvDis}\,\mathcal{P}\ .$$

Ersteres folgt aus der hohen Wahrscheinlichkeit für lineare Laufzeit für geeignete Turing-Maschinen M_1 und M_2 (mit $L(M_1) = L_1$, $L(M_2) = L_2$)

$$\begin{aligned} \mathrm{Prob}_{[\mu_{\mathrm{uni}}]_n}[\mathrm{time}_{M_1}(x) \ \le\ 2n] &\ \ge\ 1 - \frac{1}{2^{n/2}}\ , \\ \mathrm{Prob}_{[\mu_{\mathrm{uni}}]_n}[\mathrm{time}_{M_2}(x) \ \le\ 2n] &\ \ge\ 1 - \frac{1}{2^{n/2}}\ . \end{aligned}$$

Diese Wahrscheinlichkeiten implizieren

$$P_1, P_2 \in \mathrm{MedDisTime}(2 \cdot \mathcal{N}, \Theta(\sqrt{\mathcal{N}}))\ .$$

P_1 ist also nur deswegen effizient (in $\mathrm{AvDis}\,\mathcal{P}$) im Gegensatz zu P_2, weil die *worst-case*-Laufzeiten exponentiell beschränkt sind.

Allerdings konvergiert der Erwartungswert dafür, daß eine einzige Laufzeit schlechter als linear erscheint, für beide Sprachen gegen 0. Dies gilt im allgemeinen auch für alle gewichteten Probleme $P \in \mathrm{MedDisTime}(\mathcal{N}, \zeta)$, wenn $\zeta \in o(\mathcal{N})$.

In der Praxis wird das Programm sicher öfter als einmal eingesetzt werden. Jedoch ist es für genügend große n schlicht unmöglich, alle Fälle auszutesten. Realistisch (wenn nicht schon übertrieben) ist eine Aufrufhäufigkeit von $p(n)$, wobei p ein beliebiges Polynom darstellt.

Doch selbst dann ist im allgemeinen, wenn $\zeta \leq \frac{\mathcal{N}}{\log^{\omega(1)}}$, asymptotisch die Wahrscheinlichkeit für eine Laufzeit-Überschreitung 0. Der Grund ist, daß eine polynomielle Wiederholung eines Tests die Wahrscheinlichkeit p nur auf $\mathrm{Pol}(n) \cdot p$ erhöhen kann.

In unseren Beispiel ist es also praktisch unmöglich für große Eingabemengen, eine Laufzeit-Überschreitung zu erhalten. Ist es aber egal, ob dieser praktisch nicht vorkommende Fall nun exponentiell oder doppelt exponentiell ist?

Selbst wenn er auftritt, wird für eine Eingabelänge $n = 100$ kein Software-Anwender jemals erfahren, ob die Laufzeit nun 2^{100} oder $2^{2^{100}}$ war.

Betrachten wir nun P_3. Es gilt

$$P_3 \ \in \ \mathrm{Med}^* \, \mathrm{DisTime}(2 \cdot \mathcal{N}, \Theta(\sqrt{\mathcal{N}})) \ .$$

Dies folgt aus der Existenz einer Maschine M_3, die genau die Sprache der Palindrome akzeptiert und daher die Eigenschaft

$$\mathrm{Prob}_{[\mu_{\mathrm{uni}}]^n}[\mathrm{time}_{M_3,L_3}(x) \ \leq \ 2n] \ \geq \ 1 - \frac{1}{2^{n/2}}$$

besitzt (Anscheinend ist der nicht-berechenbare Anteil in L_3 für die praktische Anwendung nicht interessant).

Tritt aber der Fall $\mathrm{time}_{M_3,L_3}(x) \ \leq \ 2n$ nicht auf, so wären hier die möglichen Konsequenzen härter:

1. M_3 rechnet beliebig lange weiter,
2. M_3 hält überhaupt nicht, oder
3. M_3 gibt einen falschen Wert aus.

Doch obwohl der zweite Fall und der dritte Fall im Med-Maß nicht vorkommen, besteht in der Konsequenz für den Benutzer kein Unterschied:

Das Programm gibt nicht schnell genug die gewünschte Antwort. Jedoch tritt dieser Fall „praktisch“ nie auf.

Die Menge MedDisTime umfaßt nur gewichtete Probleme mit berechenbaren Sprachen. Med*DisTime kann auch nicht-berechenbare Sprachen beinhalten. Wird es also dadurch gelingen, nicht-berechenbare Probleme auch im Rahmen der Komplexitätstheorie als effizient zu klassifizieren? In Abschnitt 5.5 wenden wir uns dieser Frage zu.

An dieser Stelle soll nicht unerwähnt bleiben, daß es sich bei Med*DisTime im allgemeinen um keine abstrakte Komplexitätsklasse handelt. Wenn nämlich $L \notin \mathcal{REC}$, ist für alle Turing-Maschinen M die Funktion $\mathrm{time}_{M,L}$ nicht berechenbar.

Wichtiger erscheint aber die Frage, ob eine Betrachtung dieser Klassen Med*DisTime sich mit der Church'schen These verträgt, die besagt, daß nur das in einem intuitiven Sinne lösbar ist, was auch maschinell berechenbar ist. Im Gegensatz zu vielen bekannten Klassen mit nicht berechenbaren Sprachen wird hier behauptet, daß die Klasse Med*DisTime(T,ζ) für geeignetes T und ζ (siehe Definition 29) für die Praxis relevant sei. Dies ist kein eigentlicher Widerspruch, denn sicher wird eine Turing-Maschine ein nicht-berechenbares Problem nicht wirklich lösen, sondern höchstens den Lösungsraum „approximieren“.

Somit sind die durch Turing-Maschinen definierten Approximationssprachen die eigentlichen betrachteten Probleme dieser Klasse, und nicht die anvisierten nicht-berechenbaren Sprachen.

Wie bei den *average*-Komplextitätsklassen macht es auch hier Sinn, polynomiell zeitberechenbare Problemklassen zu definieren. Diese heißen **MedDis$\mathcal{P}$** und **Med*Dis$\mathcal{P}$** und sind gemäß obiger Überlegungen

Definition 29

$$\begin{aligned}\mathbf{MedDis}\boldsymbol{\mathcal{P}} &:= \mathrm{MedDisTime}\left(\mathrm{POL}, \frac{\mathcal{N}}{\log^{\omega(1)}}\right) ,\\ \mathbf{Med^*Dis}\boldsymbol{\mathcal{P}} &:= \mathrm{Med^*DisTime}\left(\mathrm{POL}, \frac{\mathcal{N}}{\log^{\omega(1)}}\right) .\end{aligned}$$

Um aus den Klassen MedDisTime und Med*DisTime Klassen von Sprachen zu erhalten, definieren wir **MedTime** und **Med*Time** analog zu AvTime und EavTime.

Definition 30 *Für eine Zeitschranke T, eine Häufigkeitsfunktion $\zeta : \mathbb{N} \to \mathbb{N}$ mit $0 \le \zeta \le \mathcal{N}$ und einer Menge von Wahrscheinlichkeitsverteilungen sei definiert*

$$\begin{aligned}\mathbf{MedTime}(\boldsymbol{T},\boldsymbol{\zeta},\boldsymbol{C}) &:= \{L \mid \forall\mu \in C\ (L,\mu) \in \mathrm{MedDisTime}(T,\zeta)\} ,\\ \mathbf{Med^*Time}(\boldsymbol{T},\boldsymbol{\zeta},\boldsymbol{C}) &:= \{L \mid \forall\mu \in C\ (L,\mu) \in \mathrm{Med^*DisTime}(T,\zeta)\} .\end{aligned}$$

5.2 MedDis$\mathcal{P}$ versus AvDis$\mathcal{P}$

Als erstes versuchen wir eine Relation zwischen MedDis$\mathcal{P}$, Med*Dis$\mathcal{P}$ und AvDis$\mathcal{P}$, EavDis$\mathcal{P}$ nachzuweisen.

Hierfür benötigen wir folgendes Lemma, das wir mit Hilfe eines modifizierten Diagonalisierungsarguments beweisen werden. Als erstes wollen wir dabei beweisen, daß es eine Sprache L_2 gibt, die für eine gegebene Zeitschranke T_1, Häufigkeitsfunktionen a und Rangfunktion ρ nicht median-fehler-beschränkt ist, d.h.

$$(L_2, \rho) \;\notin\; \mathrm{Med}^*\,\mathrm{DisTime}(T, a-1)\ .$$

Eine Teilmenge dieser Sprache $L_1 \subseteq L_2$ soll dagegen relativ leicht in Zeit T_ρ zu entscheiden sein, wobei T_ρ die Zeitkomplexität von ρ ist. Das passiert aber nicht etwa dadurch, daß diese Sprache stark ausgedünnt ist. Es soll nämlich für eine Häufigkeitsfunktion $b \in \omega(1)$ gelten

$$\forall l \quad |X_\rho^{\leq l} \cap L_1| \;\geq\; l - a(l) - b(l)\ .$$

Schließlich möchten wir auch nicht, daß die *worst-case*-Zeitkomplexität von L_2 zu groß ist. Deshalb fordern wir für ein $T_{\mathrm{worst}} \in \omega(T)$

$$L_2 \;\in\; \mathrm{DTime}(T_{\mathrm{worst}})\ .$$

Eine direkte Implikation dieser Forderungen ist beispielsweise

$$L_2 \;\in\; \mathrm{MedDisTime}(T_\rho, a+b)\ .$$

Eine Turing-Maschine testet hier zuerst, ob die Eingabe $x \in L_1$ ist. Falls ja, akzeptiert sie. Andernfalls berechnet sie die Sprache L_2 mit Hilfe des T_{worst}-zeitbeschränkten Algorithmus. Dies geschieht höchstens $a+b$ oft.

Lemma 18 *Seien $T \leq o(T_{\mathrm{worst}})$ Zeitschranken, $\rho \in T_\rho$–*rankable *eine injektive Rangfunktion, wobei gilt $T_\rho \leq T$.*

Ferner seien $a, b, c \in \omega(1)$ monoton zunehmende Häufigkeitsfunktionen, so daß gilt $a + b + c \;=\; \mathcal{N}$.

Eine monoton zunehmende Funktion $e : \mathbb{N} \to \mathbb{N}$ sei dabei so gewählt, daß für deren Umkehrfunktion gilt $e^{[-1]} \leq b$. Ferner gelte für

$$\boldsymbol{h(x)} \;:=\; \max\{e^{[j]}(0) \mid e^{[j]}(0) \leq \rho(x)\}\ ,$$

daß die Bedingung $\rho(x) \leq h(x) + c(e(h(x)))$ in Zeit $T_\rho(|x|)$ entscheidbar ist.

Dann gibt es Sprachen $L_1 \subseteq L_2$, so daß gilt

$$L_1 \in \mathrm{DTime}(T_\rho) , \qquad L_2 \in \mathrm{DTime}(T_{\mathrm{worst}}) ,$$
$$\forall l \quad |X_\rho^{\leq l} \cap L_1| \geq c(l) \quad \text{und} \quad (L_2, \rho) \notin \mathrm{Med}^* \mathrm{DisTime}(T, a) .$$

Beweis: Die Folge $x_1, x_2, x_3 \ldots$ von Zeichenketten sei gemäß der Rangfunktion gegeben, das heißt $\forall i \; \rho(x_i) = i$.

Wir werden die Sprachen L_1 und L_2 intervallweise unter Berücksichtigung der Funktionen $r, l : \mathbb{N} \to \mathbb{N}$ definieren. Im ersten Intervall $x_1, \ldots, x_{r(1)}$ werden alle Zeichenketten sowohl in L_1 als auch in L_2 enthalten sein. Im zweiten Intervall $x_{r(1)+1}, \ldots, x_{l(2)}$ sind keine Zeichenketten aus L_1. Ob eine Zeichenkette hieraus in L_2 enthalten ist, entscheidet eine Diagonalbedingung.

Im dritten Intervall $x_{l(2)+1}, \ldots, x_{l(2)+r(2)}$ sind wiederum alle Elemente in L_1 und L_2; im vierten $x_{l(2)+r(2)+1}, \ldots, x_{l(3)}$ entscheidet wieder die Diagonalbedingung für L_2 und so weiter.

Wir definieren nun $\boldsymbol{l}$, $\boldsymbol{r}$: $\mathbb{N} \to \mathbb{N}$ folgendermaßen:

$$\begin{aligned} \boldsymbol{l(i)} &:= e^{[i-1]}(0) , \\ \boldsymbol{r(i)} &:= c(l(i+1)) . \end{aligned}$$

Man beachte, daß dabei

$$l(i+1) - r(i) - l(i) = l(i+1) - c(l(i+1)) - e^{[-1]}(l(i+1)) \geq a(l(i+1)) .$$

Die Sprachen L_1 und L_2 bestehen nur aus Zeichenketten x mit endlichen Rang $\rho(x) < \infty$. Ansonsten definieren wir sie durch ihre charakteristischen Funktionen χ_{L_1} und χ_{L_2} folgendermaßen.

$$\chi_{L_1}(x_i) := \begin{cases} 1 , & \text{falls } \exists j \text{ mit } i \in (l(j); l(j) + r(j)] , \\ 0 , & \text{sonst} ; \end{cases}$$

$$\chi_{L_2}(x_i) := \begin{cases} 1 , & \text{falls } \exists j \text{ mit } i \in (l(j); l(j) + r(j)] , \\ 1 - \chi_{M_j}(x_i) , & \text{für das } j \text{ mit } i \in (l(j) + r(j); l(j+1)] \\ & \quad \text{und } \mathrm{time}_{M_j}(x_i) \leq T(|x_i|), \text{falls es existiert} , \\ 0 , & \text{sonst} . \end{cases}$$

Diese beiden Sprachen erfüllen die gewünschten Eigenschaften:

1. $L_1 \subseteq L_2$ folgt direkt aus der Definition.

2. $L_1 \in \mathrm{DTime}(T_\rho)$.

   Ein Algorithmus M berechnet für Eingabe x zuerst den Rang $\rho(x)$. Falls $\rho(x) = \infty$ verwirft M.

   Sei nun $i := \rho(x)$. Dann ist dasjenige j mit $l(j) \leq i < l(j+1)$ gegeben durch $l(j) = h(x)$. Die Eingabe x ist nun genau dann in L_1 enthalten, falls $i \leq l(j) + r(j)$. Das ist gleichbedeutend mit

   $$\rho(x) \leq h(x) + c(e(h(x))) ,$$

   was in Zeit $T_\rho(|x|)$ entscheidbar ist.

3. $L_2 \in \mathrm{DTime}(T_{\mathrm{worst}})$.

   Zuerst wird der oben beschriebene Algorithmus für L_1 ausgeführt. Falls $x \in L_1$, ist x auch in L_2 enthalten.

   Andernfalls simulieren wir $T(|x|)$ Schritte der Maschine M_j mit Eingabe x. Falls M innerhalb dieser Simulation nicht hält, wird die Eingabe verworfen. Ansonsten gilt $x \in L_2 \;\Leftrightarrow\; M_j(x) = 0$.

   Wählt man die Aufzählung M_j aller Turing-Maschinen geschickt, so kann erreicht werden, daß dieser Algorithmus Zeitkomplexität T_{worst} besitzt.

4. $\forall l \quad |X_\rho^{\leq l} \cap L_1| \;\geq\; c(l)$

   Betrachten wir die Anzahl der „Einser" $g(l) := |X_\rho^{\leq l} \cap L_1|$ der Sprache L_1. Zu beweisen ist also für alle l

   $$g(l) \;\geq\; c(l) .$$

   Dabei wächst die Funktion c mit steigenden l höchstens um einen Schritt, da $\mathcal{N} - c = a + b$ monoton zunehmend ist. Auf den Intervallen $(l(i), l(i) + r(i)]$ wächst g genau linear (also maximal), ansonsten ist g konstant.

   Also genügt es zu zeigen, daß für alle i gilt

   $$g(l(i)) \;\geq\; c(l(i+1)) ,$$

   was aus $g(l(i)) \geq r(i) = c(l(i+1))$ folgt.

5. $(L_2, \rho) \notin \mathrm{Med}^*\,\mathrm{DisTime}(T, a-1)$

   Für eine beliebige Maschine M_i ergeben sich für $l := l(i+1)$ mindestens $l(i+1) - l(i) - r(i) \;\geq\; a(l(i+1)) > a(l) - 1$ viele Zeichenketten, auf denen M_i aufgrund der Diagonalbedingung L_2 nicht berechnet oder auf denen M_i mehr als T Rechenschritte benötigt.

∎

Damit läßt sich zeigen, daß die polynomiellen *average*-Klassen $\mathrm{AvDis}\,\mathcal{P}$ und $\mathrm{EavDis}\,\mathcal{P}$ unvergleichbar sind mit den Klassen $\mathrm{MedDis}\,\mathcal{P}$ und $\mathrm{Med}^*\,\mathrm{Dis}\,\mathcal{P}$.

Theorem 14

$$\mathrm{EavDis}\,\mathcal{P} \;\setminus\; \mathrm{Med}^*\,\mathrm{Dis}\,\mathcal{P} \;\neq\; \emptyset\;,$$
$$\mathrm{MedDis}\,\mathcal{P} \;\setminus\; \mathrm{AvDis}\,\mathcal{P} \;\neq\; \emptyset\;.$$

Beweis:

1. $\mathrm{EavDis}\,\mathcal{P} \;\setminus\; \mathrm{Med}^*\,\mathrm{Dis}\,\mathcal{P} \;\neq\; \emptyset$

   Wir werden eine Diagonalsprache L mit

$$\begin{aligned}(L,\mu_{\mathrm{uni}}) &\notin \mathrm{Med^*DisTime}\left(\mathcal{N}^k, \frac{\mathcal{N}}{2^{k+1}\cdot\log^{k+1}}\right) \text{ und}\\ (L,\mu_{\mathrm{uni}}) &\in \mathrm{EavDisTime(LIN)}\end{aligned}$$

   wie folgt konstruieren. Hierzu betrachten wir die Rangfunktionen $\rho_1, \rho_2 \ldots$ definiert als

$$\boldsymbol{\rho_k(x)} := \begin{cases} \mathrm{ord}(z), & \text{falls } x = z10^{k-1}\;,\\ \infty, & \text{sonst}\;.\end{cases}$$

   Jetzt benötigen wir die Sprachen $L_1, L_2, \ldots$ mit folgenden Eigenschaften

$$\begin{aligned}(L_k,\rho_k) &\in \mathrm{MedDisTime}\left(\mathrm{LIN}, \frac{\mathcal{N}}{2\cdot\log^{k}}\right)\;,\\ (L_k,\rho_k) &\notin \mathrm{Med^*DisTime}\left(\mathcal{N}^k, \frac{\mathcal{N}}{2\cdot\log^{k+1}}\right)\;,\\ L_k &\in \mathrm{DTime}(\mathcal{N}^{k+1}).\end{aligned}$$

   Die Existenz solcher Sprachen folgt aus Lemma 18. Für diese folgt sofort bezüglich der uniformen Wahrscheinlichkeitsverteilung

$$\begin{aligned}(L_k,\mu_{\mathrm{uni}}) &\in \mathrm{MedDisTime}\left(\mathrm{LIN}, \frac{\mathcal{N}}{2^{k+1}\cdot\log^{k}}\right)\;,\\ (L_k,\mu_{\mathrm{uni}}) &\notin \mathrm{Med^*DisTime}\left(\mathcal{N}^k, \frac{\mathcal{N}}{2^{k+1}\cdot\log^{k+1}}\right)\;.\end{aligned}$$

   Die gesuchte Sprache L wird mit Hilfe von L_k definiert als

$$\boldsymbol{\chi_L(x)} \;:=\; \begin{cases} 0\;, & \text{falls } \exists n \text{ mit } 0^n = x,\\ \chi_{L_k}(x)\;, & \text{wobei } x = z10^{k-1}.\end{cases}$$

   Aufgrund der Konstruktion von L aus den Sprachen $L_1, L_2, \ldots$ gilt

$$(L,\mu_{\mathrm{uni}}) \notin \mathrm{Med}^*\,\mathrm{Dis}\,\mathcal{P}\;.$$

   Seien die Sprachen L_k gemäß des Beweises von Lemma 18 gewählt. Dann gibt es eine Turing-Maschine M, die wegen der gleichartigen Definition der Sprachen

$L_1, L_2, \ldots$ sogar für alle k die Eingabe $x = z10^{k-1}$ in Zeit $|x|^{k+1}$ entscheidet. Dieser Algorithmus kann auch in die Lage versetzt werden alle oberen Median-Schranken auf den von ρ_k gewichteten Teilmengen einzuhalten.

Definieren wir nun für festes $l \in \mathbb{N}$ die Mengen $\boldsymbol{A_k}$:

$$\boldsymbol{A_k} \quad := \quad \{x \in X^{\leq l}_{\mu_{\text{uni}}} \mid |x|^k < \text{time}_M(x) \leq |x|^{k+1}\} \ .$$

Damit folgt für deren Mächtigkeit

$$|A_k| \ \leq \ \frac{l}{2^{k+1} \cdot (\log l)^k} \ .$$

Somit gilt für eine Konstante $c \in \mathbb{N}$

$$\begin{aligned} \sum_{x \in X^{\leq l}_{\mu_{\text{uni}}}} \frac{\text{time}_M(x)}{|x|} &= \sum_k \sum_{x \in A_k} \frac{\text{time}_M(x)}{|x|} \ \leq \ \sum_k \sum_{x \in A_k} \frac{c \cdot |x|^{k+1}}{|x|} \\ &\leq \ c \cdot \sum_k \sum_{x \in A_k} (\log l)^k \ \leq \ c \cdot \sum_k \frac{l}{2 \cdot 2^k} \ = \ c \cdot l \end{aligned}$$

Also ist $(L, \text{rank}_{\text{uni}}) \in \text{Eav}\,(\text{LIN})$.

2. $\text{MedDis}\,\mathcal{P} \ \setminus \ \text{AvDis}\,\mathcal{P} \ \neq \ \emptyset$

Die folgende Definition der Sprache L findet durch deren charakteristische Funktion χ_L statt. Dabei ist M_i wiederum eine Aufzählung aller deterministischen Turing-Maschinen und $\text{rank}_{\text{Pal}} := \text{rank}_{L_{\text{Pal}}}$ die Rangfunktion der L_{Pal}-uniformen Wahrscheinlichkeitsverteilung (L_{Pal} ist die Sprache der binären Palindrome).

$$\boldsymbol{\chi_L(x)} \quad := \quad \begin{cases} 0\ , & \text{falls } x \notin L_{\text{Pal}} \text{ oder} \\ & \quad \text{time}_{M_{\text{rank}_{\text{Pal}}(x)}}(x) > \exp\exp(|x|)\ , \\ 1 - M_{\text{rank}_{\text{Pal}}(x)}(x)\ , & \text{falls } x \in L_{\text{Pal}} \text{ und} \\ & \quad \text{time}_{M_{\text{rank}_{\text{Pal}}(x)}}(x) \leq \exp\exp(|x|)\ . \end{cases}$$

Wir haben schon früher gesehen, daß $(\chi_{L_{\text{Pal}}}, \mu_{\text{uni}}) \in \text{Med}\left(0, \Theta(\sqrt{\mathcal{N}})\right)$. Damit gilt

$$(L, \mu_{\text{uni}}) \ \in \ \text{MedDisTime}(\mathcal{N}, \Theta(\sqrt{\mathcal{N}}))\ .$$

Andererseits ist aufgrund der Diagonalbedingung $L \notin \text{DTime}(\text{EEXP})$, und somit gilt nach Theorem 7

$$(L, \mu_{\text{uni}}) \ \notin \ \text{AvDis}\,\mathcal{P}\ . \qquad \blacksquare$$

Aus Lemma 6 ergibt sich als Relation zwischen den Average-Case-Komplexitätsklassen

$$\text{EavDis}\,\mathcal{P} \ \subseteq \ \text{AvDis}\,\mathcal{P}\ .$$

Da nach Definition

$$\mathrm{MedDis}\,\mathcal{P} \subseteq \mathrm{Med}^*\,\mathrm{Dis}\,\mathcal{P}$$

gilt, ist $\mathrm{MedDis}\,\mathcal{P}$ und $\mathrm{Med}^*\,\mathrm{Dis}\,\mathcal{P}$ jeweils sowohl mit $\mathrm{Eav}\,\mathcal{P}$ als auch mit $\mathrm{Av}\,\mathcal{P}$ unvergleichbar

$$\left.\begin{array}{r}\mathrm{EavDis}\mathcal{P}\\ \mathrm{AvDis}\mathcal{P}\end{array}\right\} \not\# \left\{\begin{array}{l}\mathrm{MedDis}\mathcal{P}\\ \mathrm{Med}^*\mathrm{Dis}\mathcal{P}\ .\end{array}\right.$$

Abbildung 5.1 verdeutlicht diese Mengenrelationen.

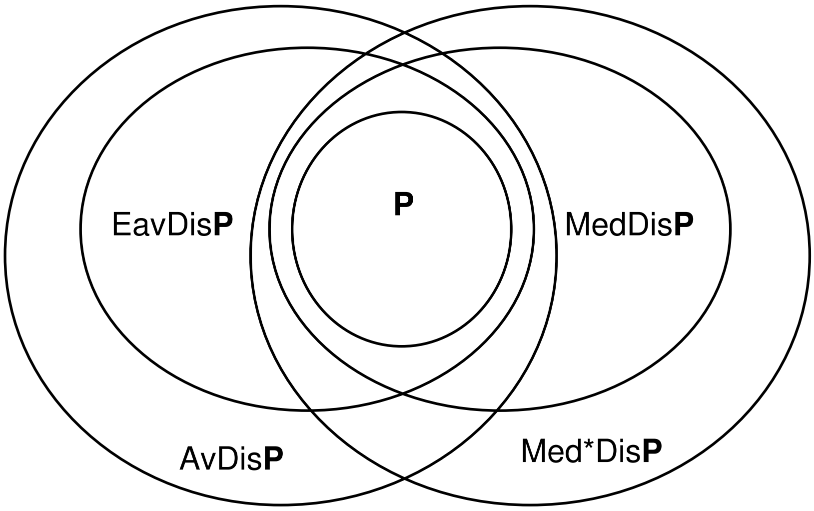

Abbildung 5.1: Die Mengenrelation zwischen den polynomiellen *worst-case*, *average* und Median-Zeitklassen.

5.3 Sparse Mengen

Die Einbeziehung nicht berechenbarer Informationen in eine Komplexitätsklasse ist prinzipiell nichts Neues. In einer ganzen Reihe von Arbeiten [AHHK 92, HOW 92] wurden Reduktionsmengen auf Mengen mit geringer Information betrachtet. Ausgangspunkt waren *tally* oder *sparse* Mengen.

Definition 31 *$\mathcal{SPARSE}$ bezeichnet die Klasse aller Sprachen L mit der Eigenschaft $|L^{\leq n}| \leq \mathrm{Pol}(n)$.*

Anhand dieser nicht berechenbaren Mengen wurden unter anderen die Leistungsfähigkeit verschiedener Arten polynomiell zeitbeschränkter Reduktionen erforscht. Das Bindeglied zwischen diesem Gedankengebäude und dieser Arbeit liefert das folgende Theorem.

Theorem 15 *Für eine Zeitschranke $T \geq \mathcal{N}$ gilt*

$$\begin{aligned}&\{(L_1 \cup L_2) \setminus L_3 \mid L_1 \in \mathrm{DTime}(T) \text{ und } L_2, L_3 \in \mathcal{SPARSE}\} \\ = \quad &\mathrm{Med}^*\,\mathrm{Time}(T, \mathrm{PLOG}, \{\mu_{\mathrm{uni}}\}) \ .\end{aligned}$$

Beweis: „$\subseteq$“: Sei M eine T-zeitbeschränkte Turing-Maschine für L_1. Dann gilt für die Anzahl der Fehler $f(l)$, die M bezüglich der Menge $(L_1 \cup L_2) \setminus L_3$ auf $X^{\leq l}_{\mu_{\mathrm{uni}}}$ macht

$$f(l) \ \leq \ |L_2 \cap X^{\leq l}_{\mu_{\mathrm{uni}}}| + |L_3 \cap X^{\leq l}_{\mu_{\mathrm{uni}}}| \ .$$

Damit ist $(\mathrm{time}_{M,(L_1 \cap L_2) \setminus L_3}, \mu_{\mathrm{uni}}) \in \mathrm{Med}(T, \mathrm{PLOG})$.

„$\supseteq$“: Sei M eine Turing-Maschine für $L \in \mathrm{Med}^*\,\mathrm{Time}(T, \mathrm{PLOG}, \{\mu_{\mathrm{uni}}\})$, so daß gilt

$$(\mathrm{time}_{M,L}, \mu_{\mathrm{uni}}) \in \mathrm{Med}(T, \mathrm{PLOG}) \ .$$

Wir modifizieren M so, daß die resultierende Maschine $\tilde{M}$ auf alle Fälle nach T Schritten hält.

$$\tilde{M}(x) \ := \ \begin{cases} 0, & \text{falls } M(x) = 0 \text{ oder } \mathrm{time}_M(x) > T(|x|) \ , \\ 1, & \text{falls } M(x) = 1 \text{ und } \mathrm{time}_M(x) \leq T(|x|) \ . \end{cases}$$

Wir definieren $L_1 := L(\tilde{M})$, $L_2 := L \setminus L(\tilde{M})$, $L_3 := L(\tilde{M}) \setminus L$. L_2 und L_3 sind *sparse*, da die Anzahl der Fehler von $\tilde{M}$ polynomiell in der Eingabelänge beschränkt ist. ∎

5.4 Median-Hierarchien

Die Median-Komplexitätsklassen liegen ähnlich wie die *average*-Komplexitätsklassen oberhalb der *worst-case*-Klassen, das heißt

$$\mathrm{DTime}(T) \ \subseteq \ \mathrm{MedTime}(T, a, C) \ \subseteq \ \mathrm{Med}^*\,\mathrm{Time}(T, a, C) \ .$$

Sie sind aber unvergleichbar mit den *average*-Komplexitätsklassen wie wir in Theorem 14 bereits gesehen haben. Der Häufigkeitsparameter der Medianklassen wählt zwischen den Extremen oberhalb des *worst-case*. Zum einen charakterisiert $a = O(1)$ die klassischen Komplexitätsklassen

$$\mathrm{DTime}(T) \ = \ \mathrm{MedTime}(T, O(1), \{\rho\}) \ = \ \mathrm{Med}^*\,\mathrm{Time}(T, O(1), \{\rho\}) \ ,$$

wenn $\forall x \ \rho(x) < \infty$. Auf der anderen Seite bedeutet $a = \mathcal{N} - O(1)$ für die fehlerfreie Medianklasse die Menge aller berechenbaren Funktionen

$$\mathrm{MedTime}(T, \mathcal{N} - O(1), \{\rho\}) \ = \ \mathcal{REC} \ .$$

Für die Medianklasse mit $a = \mathcal{N} - O(1)$, welche also fast für alle Eingaben fehlerhafte Ausgaben oder Laufzeitüberschreitungen zuläßt, gilt für die Menge aller Sprachen $\mathrm{Pot}(\Sigma^*)$ und für beliebige Wahrscheinlichkeitsverteilung ρ

$$\mathrm{Med}^*\,\mathrm{Time}(T, \mathcal{N} - O(1), \{\rho\}) \;=\; \mathrm{Pot}(\Sigma^*) \;.$$

Die Häufigkeitsfunktion a in $\mathrm{Med}\,(T, a)$ reagiert auf Veränderungen wesentlich empfindlicher als die Zeitschranke T. Während sich erst für $T_1 \leq o(T_2)$ eine Separation ergibt, ist dies schon für $a_1 \leq a_2 - \omega(1)$ der Fall.

Für $a_1 = a_2 - O(1)$ kann eine neue Turing-Maschine gefunden werden, die diese konstant vielen Eingaben mit der Methode des „endlichen Gedächtnis“ entscheiden kann. Hier gilt also

$$\mathrm{MedDisTime}(T, a) \;=\; \mathrm{MedDisTime}(T, a + O(1)) \;,$$

und gleiches für $\mathrm{Med}^*\,\mathrm{DisTime}$. Ansonsten liefert Lemma 18 folgendes Theorem.

Theorem 16 *Sei $T \geq \mathcal{N}$ eine Zeitschranke, seien $a, b, c \in \Omega(1)$ monoton zunehmende Häufigkeitsfunktionen mit $a+b+c = \mathcal{N}$ und der gleichen Zeitkonstruktionsbedingung bezüglich T_ρ wie in Lemma 18. Für eine injektiven Rangfunktion $\rho \in T_\rho$*–rankable *mit $T_\rho \leq T$ gilt nun*

$$\begin{aligned} \mathrm{MedTime}(T, a, \{\rho\}) &\subset \mathrm{MedTime}(T, a + b, \{\rho\}) \;, \\ \mathrm{Med}^*\mathrm{Time}(T, a, \{\rho\}) &\subset \mathrm{Med}^*\mathrm{Time}(T, a + b, \{\rho\}) \;, \\ \mathrm{MedTime}(T, a + b, \{\rho\}) &\not\subseteq \mathrm{Med}^*\mathrm{Time}(T, a, \{\rho\}) \;. \end{aligned}$$

Beweis: Nach Lemma 18 existiert eine Sprache L, für die gilt

$$\begin{aligned} L &\in \mathrm{MedTime}(T, a + b, \{\rho\}) \text{ und} \\ L &\notin \mathrm{Med}^*\mathrm{Time}(T, a, \{\rho\}) \;. \end{aligned}$$

Aus $\mathrm{MedTime}(T, a, \{\rho\}) \subseteq \mathrm{Med}^*\,\mathrm{Time}(T, a, \{\rho\})$ folgt nun sofort

$$L \notin \mathrm{MedTime}(T, a, \{\rho\}) \;.$$

Genauso läßt sich folgern

$$L \in \mathrm{Med}^*\,\mathrm{Time}(T, a + b, \{\rho\}) \;.$$

∎

5.5 Zwei aussichtslose Fälle

Viele Probleme von hohem praktischen Interesse sind nicht berechenbar. Die Skala reicht hier von automatischen Beweisern in der künstlichen Intelligenz, dem Problem der Software-Verifikation im Software Engineering, der Frage nach dem Ressourcenverbrauch eines Prozesses im Bereich Betriebssysteme bis hin zur bestmöglichen Komprimierung einer Zeichenkette im Bereich der Kodierungstheorie.

Wegen der hohen Relevanz versuchte man immer wieder algorithmische (Teil-) Lösungen anzugeben. Diese sind aber aufgrund der Unlösbarkeit des Problems zum Scheitern verurteilt, wenn man sich der klassischen Sichtweise des *worst-case* bedient.

Charakterisiert man diese Algorithmen aber mit dem Median-Fehler-Maß, ist es nun möglich, die Güte dieser algorithmischen Lösungsversuche sowohl hinsichtlich des Zeitverhaltens als auch hinsichtlich des Anteils fehlerhafter Berechnungen zu bewerten.

Gehen wir von einem gewichteten Problem (L, ρ) aus mit $L \notin \mathcal{REC}$. Der Idealfall wäre eine Klassifizierung durch

$$(L, \rho) \in \mathrm{Med}^* \mathrm{Dis}\,\mathcal{P} \ .$$

Nimmt man einen exponentiell großen Eingaberaum $I := \{x \mid \rho(x) \leq 2^n\}$, würde für ein Polynom $p \in \mathrm{POL}$ bei zufälliger Auswahl eines x aus I die Wahrscheinlichkeit höchstens $\frac{1}{\omega(\mathrm{POL})}$ betragen, daß ein Algorithmus auf Eingabe x die Laufzeit $p(|x|)$ überschreitet oder eine fehlerhafte Antwort gibt. Diese Wahrscheinlichkeit ist sicherlich für genügend große n vernachlässigbar.

Informeller ausgedrückt könnte man auch sagen: Die Maschine macht „praktisch" keine Fehler und bleibt „praktisch" immer innerhalb der Zeitschranke p, wenn sie versucht L zu entscheiden.

Für zwei sehr bekannte nicht berechenbare Problemstellungen wollen wir nun untersuchen, ob dieser Fall hier zutrifft. Als erstes untersuchen wir hierzu das **Halteproblem** der Turing-Maschinen.

Definition 32 *Das* **Halteproblem** *der deterministischen Turing-Maschinen* (HP) *sei definiert durch Instanzen* $(y\#z)$ *mit* $z, y \in \Sigma^*$ *und der Fragestellung: Ist* $z \in \mathrm{dom}(\Phi_i^1)$ *für* $i := \mathrm{ord}(y)$, *oder anders formuliert: Hält die Turing-Maschine* M_i *auf Eingabe der Zeichenkette* z *?*

Um ein gewichtetes Problem mit der Sprache HP zu spezifizieren, benötigen wir noch eine Rangfunktion.

Diese sollte möglichst bösartig hinsichtlich des Med^*-Maßes sein. Das heißt, sie sollte ein schlechtes Median-Fehler-Verhalten provozieren. Folgendes Beispiel zeigt, daß es nicht schwierig ist, eine besonderes gutmütige Rangfunktion anzugeben.

Beispiel:
Für eine Sprache $L \in \mathcal{REC}$ sei M_j ein Akzeptor, welcher auch auf Eingaben $x \notin L$ hält. Nun definieren wir eine Rangfunktion ρ_{nice} mit

$$\boldsymbol{\rho}_{\textbf{nice}}(\boldsymbol{y\#z}) \quad := \quad \begin{cases} \mathrm{ord}(z)\ , & \text{falls } \mathrm{ord}(y) = j, \\ \infty\ , & \text{sonst.} \end{cases}$$

Falls nun $\mathrm{ord}(y) = j$ gilt, muß eine Maschine, die versucht $\mathtt{HP}$ auf Eingabe $(y\#x)$ zu entscheiden, akzeptieren. Falls $\mathrm{ord}(y) \neq j$, ist das Ausgabe- und Laufzeitverhalten der Maschine für das Med^*–Maß nicht von Belang. Es genügt also eine Maschine zu betrachten, die ohne Kenntnis der Eingabe immer nur akzeptiert.

Also gilt $(\mathtt{HP}, \rho_{\text{nice}}) \in \mathrm{Med}^*\,\mathrm{DisTime}(1, 0)$.

Andererseits sollte die Rangfunktion nicht nur bösartig, sondern auch einfach sein. Eine gute Wahl, wie sich gleich herausstellt, ist damit $\boldsymbol{\rho}_{\textbf{diag}}$.

Definition 33

$$\boldsymbol{\rho}_{\textbf{diag}}(\boldsymbol{y\#x}) \quad := \quad \begin{cases} \mathrm{ord}(z)\ , & \textit{falls } y = x, \\ \infty\ , & \textit{sonst.} \end{cases}$$

Durch eine geschickte Modifikation des Diagonalisierungsarguments des klassischen Beweises zur Nicht-Berechenbarkeit von $\mathtt{HP}$ kann nun eine untere Schranke für das Med^*-Maß bestimmt werden. Hierbei spielt die Häufigkeitsfunktion f_j eine wichtige Rolle. Diese beschreibt die Anzahl von Turing-Maschinen, die auf den gleichen Eingaben halten wie Maschine M_j.

Theorem 17 *Es gilt für alle Zeitschranken* T *und für alle Schranken* $a : \mathbb{N} \to \mathbb{N}$ *mit* $\forall j \quad \exists l \quad a(l) < f_j(l)$, *wobei* $f_j(l) := |\{i \mid i \leq l \text{ und } \mathrm{dom}(\Phi_i^1) = \mathrm{dom}(\Phi_j^1)\}|$,

$$(\mathtt{HP}, \rho_{\text{diag}}) \notin \mathrm{Med}^*\,\mathrm{DisTime}(T, a)\ .$$

Beweis: Für einen Widerspruchsbeweis nehmen wir an, daß

$$(\mathtt{HP}, \rho_{\text{diag}}) \in \mathrm{Med}^*\,\mathrm{DisTime}(T, a)$$

und daß M eine Maschine für HP ist, welche bezüglich ρ_{diag} dieser Median-Fehler-Schranke genügt. Nun führen wir die Notation $\boldsymbol{M(i,j)} := M(\text{ord}(i)\#\text{ord}(j))$ ein und betrachten die Funktion

$$\boldsymbol{h(i)} := \begin{cases} 0, & \text{falls } M(i,i) = 0, \\ \perp, & \text{falls } M(i,i) \text{ nicht hält} \\ & \quad \text{oder } M(i,i) = 1. \end{cases}$$

Offensichtlich ist h partiell rekursiv, da M als Turing-Maschine eine partiell rekursive Funktion berechnet. Wir definieren $A := \{i \mid \text{dom}(\Phi_i^1) = \text{dom}(h)\}$ als die Menge aller Maschinen-Indizes mit gleicher Definitionsmenge wie h. Man beachte, daß $\forall j \ \exists l \ a(l) < f_j(l) = |\{i \mid i \leq l \text{ und } i \in A\}|$ gilt.

Wir werden nun nachweisen, daß auf jede Eingabe $(z\#z)$ mit $\text{ord}(z) \in A$ die Maschine M entweder niemals zum Halten kommt oder eine fehlerhafte Ausgabe erzeugt.

Sei also $i \in A$ und $M(i,i)$ halte.

1. Fall: M gibt 1 aus. Diese Ausgabe ist nur korrekt, wenn Maschine M_i auf Eingabe i hält.

   Es gilt aber $M_i(i)$ hält genau dann, wenn $h(i)$ definiert ist, da $i \in A$. Andererseits ist $h(i)$ nicht definiert, falls $M(i,i) = 1$.

2. Fall: M gibt 0 aus. Diese Ausgabe ist korrekt, wenn Maschine M_i auf Eingabe i nicht hält.

   $M_i(i)$ hält genau dann nicht, wenn $h(i)$ undefiniert ist.

   Da $M(i,i) = 0$ gilt, hat aber $h(i)$ den Wert 0 und ist damit definiert.

Die Fallstudie zeigt, daß M auf all diesen Eingaben (i,i) mit $i \in A$ nicht hält oder eine falsche Ausgabe produziert. Der Fehleranteil a wird spätestens überschritten, wenn für j mit $h = \Phi_j$ gilt $a(l) < f_j(l)$. ∎

Man mag sich nun mit dieser unteren Schranke nicht zufrieden geben und auf einen geschlossenen Ausdruck für a drängen. Hierbei ist zu bedenken, daß a sich hier in Abhängigkeit der Art der Aufzählung der Turing-Maschinen ergibt. Die Wahl der Aufzählungsmethode beeinflußt offensichtlich die Med*-Schranke ziemlich stark.

Legen wir aber eine Standardaufzählung der Turing-Maschinen, die beispielsweise das Übergangsdiagramm als binär kodierte Adjazensliste führt, so läßt sich für diese unschwer zeigen, daß $\forall j \ f_j = \Theta(\mathcal{N})$, was sofort impliziert

Korollar 17 *Für eine geeignete Aufzählung der Turing-Maschinen gilt für alle Zeitschranken T*

$$(\texttt{HP}, \rho_{\text{diag}}) \notin \text{Med}^* \, \text{DisTime}(T, o(\mathcal{N})) \ .$$

Viele nicht berechenbare Probleme sind strukturell verwandt mit dem Halteproblem, so daß ähnliche Beweistechniken funktionieren werden. Der Beweis der Nichtberechenbarkeit des **Kolmogoroff-Komplexitätsproblems** hat dagegen eine völlig andere Struktur.

Die Kolmogoroff-Komplexität hat in vielen Naturwissenschaften eine immense Bedeutung. Für einen Überblick über diese Materie sei an dieser Stelle auf [LiVi 93] verwiesen.

Am anschaulichsten erschließt sich der Begriff der Kolmogoroff-Komplexität aus der Kenntnis von Komprimierungsverfahren. Diese Algorithmen sind in der Lage, große Datenmengen, wie sie als Ausgaben von Programmen häufiger anfallen, injektiv so abzubilden, daß „in der Regel" die Ausgabe kleiner ist als die Eingabe. Diese Programme funktionieren nur wenn redundante Information in den Eingabezeichenketten vorliegen; wenn beispielsweise bestimmte Mengen von Zeichen häufiger vorkommen als andere. Auf uniform zufällig gewählten Eingaben müssen sie mit hoher Wahrscheinlichkeit wieder eben so lange Eingaben produzieren.

Im theoretischen Model besteht die komprimierte Ausgabe nicht nur aus der komprimierten Zeichenkette, sondern zusätzlich aus der Gödelnummer des dekomprimierenden Programms. Dieses hat den Vorteil, daß die Kolmogoroff-Komplexität des Komprimierungsverfahrens mitberücksichtigt wird.

Im allgemeinen begnügt man sich bei der Untersuchung der Kolmogoroff-Komplexität mit der Größenordnung der Komprimierung. Wir fragen hier aber explizit nach der besten Komprimierung..

Definition 34 *Das Kolmogoroff-Komplexitätsproblem deterministischer Turing-Maschinen* (`KCP`) *ist definiert durch die Instanz einer binären Zeichenkette* x *und die Aufgabenstellung: Berechne das kleinste Paar* (y, z) *mit Zeichenketten* $y, z \in \{0,1\}^*$*, so daß für die Turing-Maschine* $M_{\text{ord}(y)}$ *gilt:* $M_{\text{ord}(y)}(z) = x$.

Hierbei müssen wir noch spezifizieren, nach welcher Reihenfolge die Paare (y, z) geordnet sind. Zuerst setzen wir für die folgenden Untersuchungen eine totale Ordnung voraus, die durch eine bijektive berechenbare Funktion $\text{ord}_2 : \Sigma^* \times \Sigma^* \to \mathbb{N}^+$ beschrieben wird. Für die folgenden Betrachtungen ist dabei noch die folgende Eigenschaft wichtig. Für konstantes y soll die Ordnungszahl von (y, z) höchstens linear in der Ordnungszahl von z wachsten. Eine solche Ordnung nennen wir **halb-linear**.

Definition 35 *Eine Ordnungsfunktion* $\text{ord}_2 : \Sigma^* \times \Sigma^* \to \mathbb{N}$ *heißt* ***halb-linear*** *genau dann, wenn*

$$\forall y \in \Sigma^* \quad \exists c \in \mathbb{N} \quad \forall z \in \Sigma^* \qquad \text{ord}_2(y, z) \ \leq c \cdot \text{ord}(z) \ .$$

Dies ist keine wesentliche Einschränkung, wie folgende Beispiele zeigen.

Beispiel:
Folgende drei Ordnungsrelationen sind halb-linear:

1. **lexikographische Ordnung**
   Sei $\Sigma := \{0,1,\#\}$. Dann ordnen wir die Zeichenketten $(y\#z)$ lexikographisch, nachdem sie in Gruppen von gleich langen Zeichenketten vorliegen. Die resultierende Ordnungsfunktion werde $\mathrm{ord}_{\mathbf{lex}}$ genannt. Für sie gilt:
   $$\mathrm{ord}_{\mathrm{lex}}(y,z) \;\leq\; 4 \cdot 2^{|y|+\log|y|} \cdot \mathrm{ord}(z)\;.$$
2. **teilweise verdoppelte Kodierung**
   Sei $\mathrm{dop} : \{0,1\}^* \to \{0,1\}^*$ die Funktion, welche jedes Vorkommen eines Bits verdoppelt: $\mathbf{dop}(\boldsymbol{\lambda}) := \lambda$, für $b \in \{0,1\}$ und $s \in \{0,1\}^*$ ist $\mathbf{dop(b\ s)} := bb\ \mathrm{dop}(b\ s)$. Sei $\mathbf{ord_{dop}}(\boldsymbol{y},\boldsymbol{z})$ die Ordnungsfunktion, gegeben durch die Ordnungszahl der Kodierung $\mathrm{dop}(y)\ 10\ y\ z$ innerhalb des Wertebereichs $\{0,1\}^* \setminus \{x_0\ x_1 | x_0 \in \{0\}^* \text{ und } x_1 \in \{1\}^*\}$. Hier gilt:
   $$\mathrm{ord}_{\mathrm{dop}}(y,z) \;\leq\; 16 \cdot 2^{2\cdot|y|} \cdot \mathrm{ord}(z)\;.$$
3. **Kodierung mit Längeninformation**
   Sei $\mathbf{cod_{length}}(\boldsymbol{y},\boldsymbol{z}) := \mathrm{dop}(\mathrm{bin}(|y|))\ 10\ y\ z$ und $\mathbf{ord_{length}}$, die zugehörige Ordnungsfunktion. Dann gilt
   $$\mathrm{ord}_{\mathrm{length}}(y,z) \;\leq\; 4 \cdot 2^{|y|+\log|y|} \cdot \mathrm{ord}(z)\;.$$

Zur Klassifizierung der Median-Fehler-Komplexität benötigen wir die Funktion $\boldsymbol{\kappa_f}$.

Definition 36 *Für eine monoton zunehmende Funktion f beschreibt die Funktion $\boldsymbol{\kappa_f(l)}$ die Anzahl der Zeichen z mit $\mathrm{ord}(z) \leq l$, die sich kleiner als $f(\mathrm{ord}(z))$ komprimieren lassen.*

$$\boldsymbol{\kappa_f(l)} := |\{\mathrm{ord}(z) \leq l \mid \mathrm{ord}_2(\mathtt{KCP}(z)) \leq f(\mathrm{ord}(z))\}|,$$

wobei ord_2 für eine berechenbare halb-lineare totale Ordnungsfunktion steht.

Diese Funktion ermöglicht aber auch Aussagen über die Anzahl komprimierbarer Zeichen in Teilmengen. So erweist sich für jede berechenbare Teilmenge A der Wert $\kappa_{f/c}(|A^{\leq x}|)$ für eine Konstante c als untere Schranke jener Anzahl von Zeichenketten z in $A^{\leq x}$, die man kleiner als $f(\mathrm{ord}(z))$ komprimieren kann.

Lemma 19 *Für* $A \in \mathcal{REC}$ *und eine halb-lineare berechenbare Ordnungsfunktion* $\mathrm{ord}_2 : \Sigma^* \times \Sigma^* \to \mathbb{N}^+$ *gilt*

$$\exists c \in \mathbb{N}\ \ \forall x \in \Sigma^* \quad |\{z \in A^{\leq x}\ |\mathrm{ord}_2(\mathtt{KCP}(z)) \leq f(\mathrm{ord}(z))\}| \ \geq\ \kappa_{f/c}(|A^{\leq x}|)\ .$$

Beweis: Sei M ein Turing-Akzeptor für A. Dann berechne die Turing-Maschine M_k auf Eingabe (y', z') folgendes.

Zuerst simuliert sie Maschine $M_{\mathrm{ord}(y')}$ auf Eingabe z'. Sollte die Simulation terminieren, sei x' das Resultat. Nun berechnet die Maschine für $p := \mathrm{ord}(x')$ das p-te Element aus der Menge A, indem sie in lexikographischer Reihenfolge für alle Zeichenketten so lange die Berechnungen von M ausführt, bis die Anzahl der aus A gefundenen Elemente gleich p ist.

Seien $a_1, a_2, a_3 \ldots$ die lexikographisch angeordneten Elemente von A. Dann gibt es eine Konstante $c > 0$, so daß für alle $z \in \Sigma^*$ gilt:

$$\begin{aligned}
\mathrm{ord}_2(\mathtt{KCP}(a_{\mathrm{ord}(z)})) &\leq \min_{(y',z'):M_k(y',z')=z} \mathrm{ord}_2(\mathrm{ord}^{[-1]}(k), \mathrm{ord}^{[-1]}(\mathrm{ord}_2(y',z')))\\
&\leq c \cdot \min_{(y',z'):M_k(y',z')=z} \mathrm{ord}_2(y',z')\\
&= c \cdot \mathrm{ord}_2(\mathtt{KCP}(z))\ .
\end{aligned}$$

Somit gilt für alle $x \in \Sigma^*$

$$\begin{aligned}
&|\{w \leq A^{\leq x} \mid \mathrm{ord}_2(\mathtt{KCP}(w)) \leq f(\mathrm{ord}(w))\}|\\
&\qquad = |\{\mathrm{ord}(z) \leq |A^{\leq x}| \mid \mathrm{ord}_2(\mathtt{KCP}(a_{\mathrm{ord}(z)})) \leq f(\mathrm{ord}(a_{\mathrm{ord}(z)}))\}|\\
&\qquad \leq |\{\mathrm{ord}(z) \leq |A^{\leq x}| \mid c \cdot \mathrm{ord}_2(\mathtt{KCP}(z)) \leq f(\mathrm{ord}(a_{\mathrm{ord}(z)}))\}|\\
&\qquad \leq |\{\mathrm{ord}(z) \leq |A^{\leq x}| \mid c \cdot \mathrm{ord}_2(\mathtt{KCP}(z)) \leq f(\mathrm{ord}(z))\}|\\
&\qquad = \kappa_{f/c}(|A^{\leq x}|)
\end{aligned}$$

∎

Bei der Wahl der Rangfunktion für das gewichtete Problem zeigt sich diesmal, daß schon die uniforme Rangfunktion (bezüglich einer halb-linearen Ordnungsfunktion) eine hohe untere Median-Fehler-Schranke zur Folge hat.

Der Beweis der unteren Schranke setzt folgende Idee um. Wir versuchen der Turing-Maschine möglichst viele Fehler oder langsame Laufzeiten nachzuweisen. Wenn die Maschine innerhalb der zulässigen Laufzeit Ausgaben länger als f produziert, schließen wir gemäß Lemma 19, daß ein bestimmter Anteil kleiner als f komprimierbar ist. Für die untere Schranke kann man ungesehen annehmen, daß die Maschine sehr wohl Ausgaben kleiner als f richtig und schnell berechnet. Der ausschlaggebene Fehleranteil ergibt sich, wie eine Fallunterscheidung zeigt, für diese zu langen Ausgaben.

Die Funktionenklasse $\mathcal{F}$**Med*DisTime** sei das Analogon zu der Sprachenklassen Med* DisTime.

Theorem 18 *Für eine berechenbare, monoton zunehmende Funktion $f \leq \mathcal{N}$ und für alle Funktionen $a : \mathbb{N} \to \mathbb{N}$ mit*

$$\forall c \quad \exists l \qquad a(l) < \kappa_{f/c}(l - \kappa_f(l))$$

gilt für das Kolmogoroff-Komplexitätsproblem `KCP` *bezüglich einer berechenbaren halblinearen Ordnungsfunktion* ord *und beliebigen Zeitschranken T*

$$(\texttt{KCP}, \mu_{\text{uni}}) \notin \mathcal{F}\text{Med}^* \, \text{DisTime}(T, a) \; .$$

Beweis: Sei M eine Turing-Maschine mit einer zweistelligen Ausgabe. Für $l \in \mathbb{N}$ und $z_l \in \Sigma^*$ mit $\text{ord}(z_l) = l$ definieren wir

$$\begin{aligned}
A_l &:= \{x \leq z_l \mid \text{time}_M(x) \leq T(|x|) \text{ und } \text{ord}_2(M(x)) > f(\text{ord}(x))\} \; , \\
L_l &:= \{x \leq z_l \mid \text{time}_M(x) > T(|x|)\} \; , \\
K_l &:= \{x \leq z_l \mid \text{ord}_2(M(x)) \leq f(\text{ord}(x))\} \; , \\
F_l &:= \{x \leq z_l \mid M(x) \neq \texttt{KCP}(x) \text{ oder } \text{time}_M(x) > T(|x|)\} \; .
\end{aligned}$$

All diese Mengen sind Teilmengen der Menge aller Zeichenketten mit uniformem Rang kleiner oder gleich l. Die Menge A_l beinhaltet alle Eingaben x, welche die Maschine M zwar in Zeit $T(|x|)$ berechnet, wobei aber diese Maschine eine Ausgabe M(x) größer alse $f(\text{ord}(x))$ generiert. Die Menge L_l beschreibt alle Eingaben x, die M nicht in $T(|x|)$ berechnet. In der Menge K_l sind alle Elemente, die M kleiner als f zu komprimieren vorgibt.

Am interessantesten ist die Menge F_l. Sie beschreibt die Menge aller Eingaben, die entweder zu langsam oder fehlerhaft bearbeitet werden.

Die Menge $A := \bigcup_l A_l$ ist berechenbar. Somit kann Lemma 19 angewendet werden und für eine Teilmenge $S_l \subseteq A_l$, definiert als

$$S_l \;:=\; \{x \in A_l \mid \text{ord}_2(\texttt{KCP}(x)) \leq f(\text{ord}(x))\}$$

gilt dann für eine Konstante c

$$|S_l| \;\geq\; \kappa_{f/c}(l) \; .$$

Nach der Definition von S_l ist aber auch $S_l \subseteq F_l$, und es gilt

$$|A_l \cap F_l| \;\geq\; |S_l| \;\geq\; \kappa_{f/c}(l) \; .$$

Betrachten wir nun die Menge $K_l \cap \overline{F_l}$. Diese beinhaltet alle Eingaben, die Maschine M richtig innerhalb der Zeitschranke T berechnet, und deren Kolmogoroff-Komplexität durch f beschänkt sind. Somit gilt gemäß der Definition 36

$$|K_l \cap \overline{F_l}| \;\leq\; \kappa_f(l) \; .$$

Die Mengen A_l, K_l und L_l stellen eine disjunkte Partition der Menge $\{x|x \leq z_l\}$ dar:

$$\begin{array}{rrcl} & |A_l| + |K_l| + |L_l| & = & l \\ \Longrightarrow & |L_l| + |K_l \cap F_l| & = & l - |A_l| - |K_l \cap \overline{F_l}| \; . \end{array}$$

Für die Mächtigkeit von A_l ergeben sich folgende Möglichkeiten:

1. $|A_l| \; \geq \; l - \kappa_f(l)$.

   Also ist die Anzahl $|F_l|$ aller fehlerhaften oder zu langsamen Berechnungen von M nach unten beschränkt durch

   $$|F_l| \; \geq \; |A_l \cap F_l| \; \geq \; \kappa_{f/c}(|A_l|) \; \geq \; \kappa_{f/c}(l - \kappa_f(l)) \; ,$$

   da κ_f monoton zunehmend ist.

2. $|A_l| \; \leq \; l - \kappa_f(l)$.

   Für die Menge F_l gilt nun

   $$\begin{array}{rcl} |F_l| & = & |A_l \cap F_l| + |L_l| + |K_l \cap F_l| \\ & \geq & \kappa_{f/c}(|A_l|) + |L_l| + |K_l \cap F_l| \\ & = & \kappa_{f/c}(|A_l|) + l - |A_l| - |K_l \cap \overline{F_l}| \\ & \geq & \kappa_{f/c}(|A_l|) - |A_l| + l - \kappa_f(l) \; . \end{array}$$

   Da $\kappa_{f/c} - \mathcal{N}$ monoton abnehmend ist, wird der letzte Ausdruck genau dann minmal, wenn gilt $|A_l| = l - \kappa_f(l)$. Das bedeutet für F_l

   $$|F_l| \; \geq \; \kappa_{f/c}(l - \kappa_f(l)) \; . \qquad \blacksquare$$

Die untere Med*-Schranke ergibt sich somit in Abhängigkeit der Funktion κ_f. Diese ist nicht berechenbar, allerdings kann sie zum Beispiel für $f = \mathcal{N}/c$ asymptotisch gut abgeschätzt werden.

Lemma 20 *Für jede Konstante* $c > 1$ *gilt*

$$\kappa_{\mathcal{N}/c} \; \in \; \Theta(\mathcal{N}) \; .$$

Beweis:

1. $\kappa_{\mathcal{N}/c} \; \in \; O(\mathcal{N})$.

   `KCP` ist eine injektive Abbildung von Σ^* auf $\Sigma^* \times \Sigma^*$. Bei $\mathrm{ord}_2 : \Sigma^* \times \Sigma^* \to \mathbb{N} \backslash \{0\}$ handelt es sich um eine surjektive Abbildung, da wir nur totale Ordnungen von Paaren betrachten. Daher ist

   $$\kappa_{\mathcal{N}/c}(l) \; \leq \; \frac{l}{c} \; .$$

2. $\kappa_{\mathcal{N}/c} \;\in\; \Omega(\mathcal{N})$.

   Wir beweisen diese untere Schranke, indem wir zeigen, daß für ein festes y' linear oft eine Eingabe x als (y', w) komprimiert werden kann, so daß $M_{\mathrm{ord}(y')}(w) \;=\; x$ und $c \cdot \mathrm{ord}_2(y', w) \le \mathrm{ord}(x)$ gilt; das heißt also

   $\forall c > 1 \quad \exists x_0, y' \in \Sigma^*,\; c' > 1 \quad \forall x \ge x_0$

$$|\{z \le x \mid c \cdot \min_{w: M_{\mathrm{ord}(y)}(w) = z} \mathrm{ord}_2(y', w) \;\le\; \mathrm{ord}(z)\}| \,.$$

   Diese Eigenschaft ist keine harte Auflage für eine einigermaßen effektive Komprimierungsstrategie. Das folgende Beispiel zeigt einen sehr simplen Komprimierungsalgorithmus, der hierfür schon ausreicht.

   Daraus folgt dann $\exists l_0, c' > 1 \quad \forall l \ge l_0 \qquad \kappa_{\mathcal{N}/c} \;\ge\; \frac{l}{c'}$.

∎

Beispiel:
Auf Eingabe $w \;=\; \mathrm{ord}^{[-1]}(\mathrm{ord}_{\mathrm{length}}(e, v))$ berechnet Algorithmus M_k die Ausgabe $z \;=\; \underbrace{11\cdots 1}_{\mathrm{ord}(e)-mal} \;\; v$. Dabei ist $\mathrm{ord}_{\mathrm{length}}$ die auf Seite 110 definierte, halb-lineare rekursive Ordnungsfunktion. Für y' mit $\mathrm{ord}(y') = k$ und eine geeignete Konstante c' gilt dann für den Komprimierungsfaktor

$$\begin{aligned}
\frac{\mathrm{ord}(z)}{\mathrm{ord}_2(y', w)} &\ge \frac{\mathrm{ord}(1^e v)}{c' \cdot \mathrm{ord}(w)} \\
&\ge \frac{\mathrm{ord}(1^e v)}{c' \cdot \mathrm{ord}_{\mathrm{length}}(e, v)} \\
&\ge \frac{2^{\mathrm{ord}(e)} \cdot \mathrm{ord}(v)}{4c' \cdot 2^{|e| + \log|e|} \cdot \mathrm{ord}(v)} \\
&\ge \frac{1}{4c'} \cdot 2^{\mathrm{ord}(e) - |e| - \log|e|} \,.
\end{aligned}$$

Sei $e_0 \in \Sigma^*$ nun so gewählt, daß $\forall e \ge e_0 \; \frac{\mathrm{ord}(z)}{\mathrm{ord}_2(y', w)} \;\ge\; c$. Dann können alle Zeichenketten, die mit 1^{m_0}, für $m_0 \;:=\; \mathrm{ord}(e_0)$ beginnen, um den gewünschten Faktor c komprimiert werden. Die Anzahl dieser Zeichenketten innerhalb von $\{z \mid \mathrm{ord}(z) \le l\}$ beträgt mindestens $\left\lfloor \frac{l}{|\Sigma|^{m_0}} \right\rfloor$.

Wählen wir in Theorem 18 $f = \mathcal{N}/c$, erhalten wir folgendes Korollar.

Korollar 18 *Für alle Zeitschranken T gilt*

$$(\mathtt{KCP}, \mu_{\mathrm{uni}}) \notin \mathcal{F}\mathrm{Med}^* \, \mathrm{DisTime}(T, o(\mathcal{N}))\;.$$

Das Resumee der Betrachtung dieser beiden gewichteten Probleme $(\texttt{HP}, \rho_{\text{diag}})$ und $(\texttt{KCP}, \mu_{\text{uni}})$ ist, daß unsere optimistischen Erwartungen hinsichtlich $\text{Med}^* \, \text{Dis} \, \mathcal{P}$ enttäuscht wurden.

Sie verursachen für alle Zeitschranken einen extrem starken Anteil von Fehlern oder Laufzeit-Überschreitungen. Denn jeder Algorithmus, der eine mit der Problemgröße m zunehmenden Anzahl $b(m)$ (einzige Bedinung $b \in \omega(1)$) von Testeingaben aus $\{x \mid \rho(x) \leq m\}$ erhält, wird für ein genügend großes m mit mehr als 50-prozentiger Sicherheit einmal jede berechenbare Zeitschranke überschreiten oder einen Fehler produzieren (bei uniformer Auswahl der Testeingaben).

So gesehen bleiben das Halteproblem `HP` und das Kolmogoroff-Komplexitätsproblem `KCP` nicht berechenbar.

Kapitel 6

Reduktionen

6.1 Dominante Wahrscheinlichkeitsverteilungen

Nachdem in den drei vorherigen Kapiteln Komplexitätsklassen im Vordergrund standen, wenden wir uns jetzt noch einmal gewichteten Funktionen (t, ρ) mit $t : \Sigma^* \to \mathbb{N}$ und Rangfunktion ρ zu.

Wie im Kapitel 2 stellen wir die Frage: Wenn $(t_1, \rho_1) \in \left\{ \begin{array}{l} \mathrm{Av}(T) \\ \mathrm{Eav}(T) \\ \mathrm{Med}(T, a) \end{array} \right.$ gilt, was folgt dann für (t_2, ρ_2) bezüglich des Av-, Eav- oder Med-Maßes? Im Kapitel 2 setzten wir $\rho_1 = \rho_2$. Nun untersuchen wir den Fall $t_1 = t_2$ und $\rho_1 \neq \rho_2$.

Motiviert ist diese Betrachtung durch die Suche nach polynomiell zeit-beschränkten Reduktionen zwischen gewichteten Problemen. Zwar waren bislang schon polynomiell zeit-beschränkte *average*-Reduktionen bekannt, allerdings galten diese nur für Levins Maß *average on polynomial*. Für den Erwartungswert waren bisher solche Reduktionen nicht bekannt.

Nun unterscheidet sich das Av-Maß wegen der zusätzlichen Präzision ganz entschieden von der Notation *average on polynomial*. Eine direkte Übertragung scheitert schon an diesen formalen Gründen. Jedoch zeigt sich, daß die von Leonid Levin, Yuri Gurevich, Ben-David et al. [Levin 86, Gure 91a, Gure 91b, BCGL 89] vorgestellten Ideen sich auch auf diese neuartigen Maße Av, Eav und Med anwenden lassen. Die immanente Präzision dieser Maße läßt nun sogar zu, die Auswirkungen wesentlich genauer und allgemeiner zu untersuchen.

Ein fundamentales Konzept im Bereich der Reduktionen von gewichteten Problemen ist das Prinzip der Dominanz von Wahrscheinlichkeitsverteilungen, wie folgendes Beispiel belegt.

Beispiel:
Wir betrachten eine beliebige Funktion $t : \mathbb{N} \to \mathbb{N}$, die uniforme Rangfunktion rank_{uni} für binäre Zeichenketten und die Rangfunktion $\boldsymbol{\rho}_{\mathbf{Pali}}$ definiert mittels der Sprache L_{Pal} der binären Palindrome

$$\boldsymbol{\rho}_{\mathbf{Pali}}(\boldsymbol{x}) \;:=\; \begin{cases} 2 \cdot |{L_{\text{Pal}}}^{\leq x}| + 1 \;, & \text{falls } x \in L_{\text{Pal}} \;, \\ 2 \cdot |\overline{L_{\text{Pal}}}^{\,\leq x}| \;, & \text{sonst} \;. \end{cases}$$

Nun gilt für alle $x \in \{0,1\}^*$

$$\rho_{\text{Pali}}(x) \;\leq\; 2 \cdot \text{rank}_{\text{uni}}(x) + 5 \;.$$

Betrachtet man jetzt die Wahrscheinlichkeitsverteilungen hinter den Rangfunktionen, so bedeutet dies, daß die Wahrscheinlichkeit für eine Zeichenkette x bezüglich der uniformen Verteilung **dominiert** wird von der zu ρ_{Pali} korrespondierenden Wahrscheinlichkeit, da der Rang $\rho_{\text{Pali}}(x)$ kleiner ist als $2 \cdot \text{rank}_{\text{uni}}(x) + 5$.

Wüßten wir nur das über die Rangfunktionen und bliebe t weiter im Dunkeln, so ließen sich immerhin folgende Aussagen treffen (wie wir später in Lemma 21 und 22 allgemeiner beweisen werden)

$$\begin{aligned} (t, \rho_{\text{Pali}}) \in \text{Med}\Big(\mathcal{N}^2, \sqrt{\mathcal{N}}\Big) &\implies (t, \mu_{\text{uni}}) \in \text{Med}\Big(\mathcal{N}^2, \sqrt{2\mathcal{N}+5}\Big) \;, \\ (t, \rho_{\text{Pali}}) \in \text{Av}(\mathcal{N}^2) &\implies (t, \mu_{\text{uni}}) \in \text{Av}(\Theta(\mathcal{N}^6)) \;, \\ (t, \rho_{\text{Pali}}) \in \text{Eav}(\mathcal{N} \cdot \log \mathcal{N}) &\implies (t, \mu_{\text{uni}}) \in \text{Eav}(\Theta(\mathcal{N}^3 \cdot \log \mathcal{N})) \;. \end{aligned}$$

Scheinbar bleiben hier polynomielle Schranken erhalten. Betrachten wir nun die Dominanz in der anderen Richtung. Hier gilt

$$\text{rank}_{\text{uni}}(x) \;\leq\; \rho_{\text{Pali}}(x)^2 \;.$$

Der Schluß, daß diese Bedingung ähnliche Aussagen für beliebiges t zur Folge hat, wäre falsch.

Betrachten wir einmal $(t, \mu_{\text{uni}}) \in \text{Med}\Big(\mathcal{N}^2, \sqrt{\mathcal{N}}\Big)$ und rufen uns in Erinnerung, daß hier t „praktisch" nie quadratische Laufzeit überschreitet. Dies gilt zum Beispiel auch für die folgende Funktion t definiert als

$$\boldsymbol{t(x)} \;:=\; \begin{cases} \exp(|x|) \;, & \text{falls } x \in L_{\text{Pal}} \;, \\ |x| \;, & \text{sonst} \;. \end{cases}$$

Betrachtet man nun das Paar (t, ρ_{Pali}), so ergibt sich

$$(t, \rho_{\text{Pali}}) \notin \text{Med}(\text{POL}, \mathcal{N}/2 - O(1)) \ .$$

Der Anteil nicht polynomieller Laufzeiten hat sich auf die Hälfte aller Eingaben vergrößert.

Alle drei Durchschnittsmaße Av, Eav und Med besitzen allgemeine Formeln, um beliebige Dominanzeigenschaften in den Zeit- oder gegebenenfalls Häufigkeitsschranken widerzuspiegeln. Das Median-Maß sticht hierbei durch die einfachste und straffeste Beziehung hervor.

Lemma 21 *Für eine monoton zunehmende Funktion* $f : \mathbb{N} \to \mathbb{N}$ *und für Rangfunktionen* ρ_1 , ρ_2 *mit* $\rho_2(x) \leq f(\rho_1(x))$ *gilt*

$$(t, \rho_2) \in \text{Med}(T, a) \implies (t, \rho_1) \in \text{Med}(T, a \circ f) \ .$$

Beweis: Nehmen wir an, daß $(t, \rho_1) \notin \text{Med}(T, a \circ f)$. Dann existiert ein $l \in \mathbb{N}$ mit

$$|\{x \in X_{\rho_1}^{\leq l} \mid t(x) > T(|x|)\}| \ > \ a(f(l)) \ .$$

Nun ist

$$|\{x \mid \rho_2(x) \leq f(l) \text{ und } t(x) > T(|x|)\}| \ \geq \ |\{x \mid \rho_1(x) \leq l \text{ und } t(x) > T(|x|)\}| \ ,$$

da die Implikation $\rho_1(x) \leq l \ \Rightarrow \ \rho_2(x) \leq f(l)$ gilt.

Somit gilt für $\tilde{l} := f(l)$ die gewünschte Beziehung

$$|\{x \mid \rho_2(x) \leq \tilde{l} \text{ und } t(x) > T(|x|)\}| \ > \ a(\tilde{l}) \ .$$

∎

Die allgemeinen Dominanzsätze des Av- und des Eav-Maßes sind komplexer und außerdem eingeschränkt auf Rangfunktionen mit $\rho_1(x) \leq g(|x|)$ und $\rho_2(x) \leq \rho_1(x) \cdot f(|x|)$ für Funktionen $f, g : \mathbb{N} \to \mathbb{N}$. Hätten wir allein die Bedingung $\rho_1(x) \leq f(\rho_2(x))$, so könnten wir keine sinnvolle Beziehung folgern. Um die Relation möglichst genau zu formulieren, wird eine beliebige Funktion $q : \mathbb{N} \to \mathbb{N}$ eingeführt, deren Kehrwert-Summe konvergiert:

$$\sum_{i=0}^{\infty} \frac{1}{q(i)} \ \leq \ 1 \ .$$

Dabei verbessern sich die nun folgenden Aussagen, je langsamer q asymptotisch zunimmt. Eine mögliche Wahl ist $q \in \Theta(\mathcal{N}^2)$, besser ist dagegen $q \in \Theta(\mathcal{N} \cdot \log^2 \mathcal{N})$ oder gar $q \in \Theta(\mathcal{N} \cdot \log \mathcal{N} \cdot \text{llog}^2 \mathcal{N})$.

Lemma 22 *Für streng monoton zunehmende Funktionen* $f, g : \mathbb{N} \to \mathbb{N}$ *gelte*

$$\begin{aligned} \rho_2(x) &\leq \rho_1(x) \cdot f(|x|) \ , \\ \rho_2(x) &\leq g(|x|) \ , \quad \textit{falls } \rho_2(x) \neq \infty \ . \end{aligned}$$

Weiter sei $q : \mathbb{N} \to \mathbb{N}$ *eine Funktion, so daß* $q/\mathcal{N}$ *monoton zunehmend ist und* $\sum_{i=1}^{\infty} \frac{1}{q(i)} \leq 1$ *gilt. Hieraus folgt*

$$\begin{aligned} (t, \rho_2) \in \mathrm{Eav}(T) &\implies (t, \rho_1) \in \mathrm{Eav}\Big(T \cdot f \cdot \tfrac{q(g)}{g}\Big) \ , \\ (t, \rho_2) \in \mathrm{Av}(T) &\implies (t, \rho_1) \in \mathrm{Av}\Big(T \circ \Big(f \cdot \mathcal{N} \cdot \tfrac{q(g)}{g}\Big) \circ (2 \cdot \mathcal{N})\Big) \ . \end{aligned}$$

Beweis:
1.) $(t, \rho_2) \in \mathrm{Eav}\,(T) \implies (t, \rho_1) \in \mathrm{Eav}\,\Big(T \cdot f \cdot \frac{q(g)}{g}\Big)$

Sei $t : \Sigma^* \to \mathbb{N}$ eine Funktion und ρ_2 eine Rangfunktion, so daß $(t, \rho_2) \in \mathrm{Eav}\,(T)$. Nach der Definition von Eav gilt dann für alle monotonen Transformationen m von μ_2 mit $\mathrm{rank}_{\mu_2} = \rho_2$ die Ungleichung

$$\sum_x m(\mu_2(x)) \cdot \frac{t(x)}{T(|x|)} \ \leq \ 1 \ .$$

Insbesondere gibt es eine monotone Transformation m für μ_2, so daß $m(\mu_2(x)) = \frac{1}{q(\rho_2(x))}$, da q monoton zunehmend und $\sum_{i=1}^{\infty} \frac{1}{q(i)} \leq 1$. Also gilt

$$\sum_x \frac{t(x)}{q(\rho_2(x)) \cdot T(|x|)} \ \leq \ 1 \ .$$

Um zu beweisen, daß unter den oben angegeben Prämissen $(t, \rho_1) \in \mathrm{Eav}\,\Big(T \cdot f \cdot \frac{q(g)}{g}\Big)$ gilt, betrachten wir für alle l

$$\begin{aligned} \sum_{\rho_1(x) \leq l} \frac{t(x) \cdot g(|x|)}{T(|x|) \cdot f(|x|) \cdot q(g(|x|))} &\leq \sum_{\rho_2(x) \leq l \cdot f(|x|)} \frac{t(x) \cdot g(|x|)}{T(|x|) \cdot f(|x|) \cdot q(g(|x|))} \\ &\leq \sum_{\rho_2(x) \leq l \cdot f(|x|)} \frac{t(x) \cdot g(|x|)}{T(|x|) \cdot f(|x|) \cdot q(g(|x|))} \cdot \frac{l \cdot f(|x|)}{\rho_2(x)} \\ &\leq l \cdot \sum_{\rho_2(x) \leq l \cdot f(|x|)} \frac{t(x) \cdot g(|x|)}{T(|x|) \cdot q(g(|x|)) \cdot \rho_2(x)} \end{aligned}$$

Für $\rho_2(x) \leq l \cdot f(|x|)$ gilt nun $\frac{g(|x|)}{q(g(|x|))\rho_2(x)} \leq \frac{1}{q(\rho_2(x))}$

$$\leq \ l \cdot \sum_x \frac{t(x)}{q(\rho_2(x)) \cdot T(|x|)} \ \leq \ l \ .$$

2.) $(t,\rho_2) \in \mathrm{Av}(T) \implies (t,\rho_1) \in \mathrm{Av}\left(T \circ \left(f \cdot \mathcal{N} \cdot \frac{q(g)}{g}\right) \circ (2 \cdot \mathcal{N})\right)$

Sei $(t,\rho_2) \in \mathrm{Av}(T)$, dann gilt nach Definition von Av für alle monotonen Transformationen m von μ_2 mit $\mathrm{rank}_{\mu_2} = \rho_2$

$$\sum_x m(\mu_2(x)) \cdot \frac{T^{[-1]}(t(x))}{|x|} \;\leq\; 1 \; .$$

Auch hier gibt es eine monotone Transformation m für μ_2 mit $m(\mu_2(x)) = \frac{1}{q(\rho_2(x))}$. Also gilt

$$\sum_x \frac{T^{[-1]}(t(x))}{q(\rho_2(x)) \cdot |x|} \;\leq\; 1 \; .$$

Wir definieren $h(n) := n \cdot f(n) \cdot \frac{q(g(n))}{g(n)}$ und $g(n) := T(h(n))$. Mit Hilfe dieser Funktionen werde Σ^* unterteilt in die Mengen X_1 und X_2:

$$\begin{aligned} X_1 &:= \{x \mid t(x) \leq g(|x|)\} \; , \\ X_2 &:= \{x \mid t(x) > g(|x|)\} \; . \end{aligned}$$

Nach Korollar 1 genügt es zu zeigen, daß für alle l

$$\sum_{\rho_1(x) \leq l} \frac{g^{[-1]}(t(x))}{|x|} \leq 2 \cdot l \; .$$

Es gilt für $x \in X_1$ nach Definition der Umkehrfunktion $g^{[-1]}(t(x)) \leq |x|$. Also gilt für alle l

$$\sum_{x \in X_1 \cap X_{\rho_1}^{\leq l}} \frac{g^{[-1]}(t(x))}{|x|} \;\leq\; \sum_{x \in X_1 \cap X_{\rho_1}^{\leq l}} 1 \;\leq\; l \; .$$

Da h monoton zunehmend ist, gilt für $x \in X_2$ folgendes.

$$\begin{aligned} \sum_{\rho_1(x) \leq l} \frac{g^{[-1]}(t(x))}{|x|} &\leq \sum_{\rho_1(x) \leq l} \frac{h(g^{[-1]}(t(x)))}{h(|x|)} \leq \sum_{\rho_1(x) \leq l} \frac{T^{[-1]}(t(x)) \cdot g(n)}{|x| \cdot f(|x|) \cdot q(g(|x|))} \\ &\leq \sum_{\rho_1(x) \leq l} \frac{T^{[-1]}(t(x)) \cdot g(n)}{|x| \cdot f(|x|) \cdot q(g(|x|))} \cdot \frac{l \cdot f(|x|)}{\rho_2(x)} \\ &\leq l \cdot \sum_{\rho_2(x) \leq l \cdot f(|x|)} \frac{T^{[-1]}(t(x)) \cdot g(n)}{|x| \cdot q(g(|x|)) \cdot \rho_2(x)} \end{aligned}$$

Für $\rho_2(|x|) \leq l \cdot f(|x|)$ gilt $\frac{g(|x|)}{q(g(|x|))\rho_2(x)} \leq \frac{1}{q(\rho_2(x))}$

$$\leq \; l \cdot \sum_{\rho_2(x) \leq l \cdot f(|x|)} \frac{T^{[-1]}(t(x))}{|x| \cdot q(\rho_2(x))} \;\leq\; l \; . \qquad \blacksquare$$

Wählen wir für dieses Lemma $f \in \mathrm{POL}$ und $g \in \exp(\mathrm{POL})$ und für Lemma 21 $f \in \Theta(\mathcal{N} \cdot \mathrm{PLOG})$, so ergibt sich die Dominanz-Bedingung für polynomielle Schranken.

Korollar 19 *Seien ρ_1, ρ_2, ρ_3 und ρ_4 Rangfunktionen, die den Bedingungen*

$$\begin{aligned} \rho_1 &\in \text{POL–describable} , \\ \rho_2(x) &\leq \rho_1(x) \cdot \text{Pol}(|x|) \text{ und} \\ \rho_4(x) &\leq \rho_3(x) \cdot \text{PLog}(\rho_3(x)) \end{aligned}$$

genügen. Dann gelten folgende Implikationen.

$$\begin{aligned} (t, \rho_2) \in \text{Av(POL)} &\implies (t, \rho_1) \in \text{Av(POL)} , \\ (t, \rho_2) \in \text{Eav(POL)} &\implies (t, \rho_1) \in \text{Eav(POL)} , \\ (t, \rho_4) \in \text{Med}\Big(\text{POL}, \tfrac{\mathcal{N}}{\log^{\omega(1)}}\Big) &\implies (t, \rho_3) \in \text{Med}\Big(\text{POL}, \tfrac{\mathcal{N}}{\log^{\omega(1)}}\Big) . \end{aligned}$$

6.2 Reduktionen gewichteter Probleme

Eine polynomiell zeit-beschränkte Reduktion $R : \Sigma^* \to \Sigma^*$ von Sprache L_1 auf Sprache L_2 muß folgenden Bedingungen genügen

1. $x \in L_1 \iff R(x) \in L_2$,
2. $\text{time}_R(x) \leq \text{Pol}(|x|)$,

wobei time_R die Laufzeit einer Turing-Maschine bezeichnet, die R berechnet. Wir konzentrieren uns dabei auf den Spezialfall injektiver Reduktionen. Nun bestehen gewichtete Probleme aus einer Sprache und einer Wahrscheinlichkeitsverteilung. Wir wissen mittlerweile aber auch, daß die Wahl der Wahrscheinlichkeitsverteilung entscheidend die Komplexität des dazugehörigen Problems beeinflußt. Deswegen müssen wir noch eine Relation zwischen den beiden Verteilungen der zu reduzierenden Probleme finden.

Am einfachsten ist es wohl für die Probleme $P_1 := (L_1, \mu_1)$ und $P_2 := (L_2, \mu_2)$, die Wahrscheinlichkeit μ_2 als die Wahrscheinlichkeit von μ_1 an der reduzierten Stelle zu definieren:

3. $\mu_2(x) = \mu_1(R(x))$.

Überträgt man diese Überlegung auf die für uns viel interessanteren Rangfunktionen ρ_1 und ρ_2, könnte man eine Reduktion für gewichtete Probleme folgendermaßen definieren:

Ein gewichtetes Problem $P_1 := (L_1, \rho_1)$ kann auf ein gewichtetes Problem $P_2 := (L_2, \rho_2)$ in polynomieller Zeit reduziert werden, wenn gilt

1. $x \in L_1 \iff R(x) \in L_2$,
2. $\mathrm{time}_R(x) \leq \mathrm{Pol}(|x|)$,
3. $\rho_2(x) = \rho_1(R(x))$.

Machen diese Reduktionen bezüglich der Durchschnittsmaße Eav, Av und Med aber auch Sinn?

Für das Eav-Maß ist dies genau dann der Fall, wenn aus „ P_1 kann auf P_2 reduziert werden“ und aus „ $P_2 \in \mathrm{EavDis}\,\mathcal{P}$ “ folgt „ $P_1 \in \mathrm{EavDis}\,\mathcal{P}$ “.

Dieses ist aber nicht selbstverständlich. Schließlich hatten wir bereits für polynomiell zeit-beschränkte Simulationen im Eav-Maß Gegenbeispiele, die uns sogar veranlaßten, das Av-Maß als Alternative zu betrachten.

Bevor wir später auf gewichtete Probleme zurückkommen, betrachten wir zuerst wieder Paare (t, ρ) mit Funktion $t : \Sigma^* \to \Sigma^*$ und Rangfunktion ρ. Stehe hier t für die Zeitfunktion des Problems P_1.

Damit P_1 auf P_2 wie oben skizziert reduziert werden kann, muß die entsprechende Zeitfunktion $t(R) + \mathrm{time}_R$ des Problems P_2 betrachtet werden. time_R ist hier der Aufwand für die Reduktion. Diesen können wir nach dem, was wir über *average-* und Median-Maße mittlerweile wissen, getrost vernachlässigen, wenn die Reduktion den entsprechenden polynomiellen Zeitbeschränkungen genügt.

Die obige Frage kann also vereinfacht werden zu der Fragestellung: Gilt die Implikation $(t, \rho) \in \mathrm{Eav}\,(\mathrm{POL})$ und $|R(x)| \leq \mathrm{Pol}(|x|) \implies (t \circ R, \rho \circ R) \in \mathrm{Eav}\,(\mathrm{POL})$?

Dafür würde es genügen, die beiden Summen folgendermaßen gegeneinander abzuschätzen

$$\sum_{\rho(x) \leq l} \frac{t(x)}{T_1(|x|)} \geq \sum_{\rho(R(x)) \leq l} \frac{t(R(x))}{T_2(|x|)} .$$

Dies ist tatsächlich im allgemeinen möglich, wenn $|R(x)| \leq h(|x|)$ und $T_2 = T_1 \circ h$.

Man kann sogar zeigen, daß diese Reduktion für alle drei Durchschnittsmaße Sinn macht. Diese Eigenschaft läßt sich für das Median-Maß noch relativ einfach belegen. Die Eigenschaft $|R(x)| \leq h(|x|)$ kann hier durch die Median-Beschränkung $(|R|, \rho \circ R) \in \mathrm{Med}\,(h, a_1)$ verallgemeinert werden.

Lemma 23 *Sei* $R : \Sigma^* \to \Sigma^*$ *eine injektive Funktion mit*

$$(|R|, \rho \circ R) \in \mathrm{Med}\,(h, a_1) ,$$

wobei $h : \mathbb{N} \to \mathbb{N}$ *eine Komplexitätsschranke ist. Dann gilt die Implikation*

$$(t, \rho) \in \mathrm{Med}(T, a_2) \implies (t \circ R, \rho \circ R) \in \mathrm{Eav}(T \circ h, a_1 + a_2) .$$

Beweis: Wir betrachten für alle $l \in \mathbb{N}$

$$\begin{aligned}
|\{x \in X_{\rho \circ R}^{\leq l} \mid t(R(x)) > T(h(|x|))\}| &\leq |\{x \in X_{\rho \circ R}^{\leq l} \mid t(R(x)) > T(|R(x)|)\}| + a_1(l) \\
&\leq |\{x \in X_{\rho}^{\leq l} \mid t(x) > T(|x|)\}| + a_1(l) \\
&\leq a_1(l) + a_2(l)
\end{aligned}$$

∎

Überraschenderweise bleiben Reduktionen als wichtiges komplexitätstheoretisches Werkzeug im dem Erwartungswert nahen Eav-Maß erhalten.

Lemma 24 *Sei* $R : \Sigma^* \to \Sigma^*$ *eine injektive Funktion mit*

$$|R(x)| \leq h(|x|) ,$$

wobei $h : \mathbb{N} \to \mathbb{N}$ *eine Komplexitätsschranke ist. Dann gilt*

$$(t, \rho) \in \mathrm{Eav}(T) \implies (t \circ R, \rho \circ R) \in \mathrm{Eav}(T \circ h) .$$

Beweis: Wir betrachten für alle l

$$\sum_{\rho(R(x)) \leq l} \frac{t(R(x))}{T(h(|x|))} \leq \sum_{\rho(R(x)) \leq l} \frac{t(R(x))}{T(|R(x)|)} \leq \sum_{\rho(x) \leq l} \frac{t(x)}{T(|x|)} \leq 1 .$$

∎

Das Av-Maß hat auch die Möglichkeit solcher Reduktionen. Leider verkompliziert die Definition von Av mittels der Umkehrfunktion den entsprechenden Beweis.

Lemma 25 *Sei* $R : \Sigma^* \to \Sigma^*$ *eine injektive Funktion mit*

$$|R(x)| \leq h(|x|) ,$$

wobei $h : \mathbb{N} \to \mathbb{N}$ *eine Funktion mit der Eigenschaft* $\mathcal{N}/h^{[-1]}$ *monoton zunehmend ist. Dann folgt*

$$(t, \rho) \in \mathrm{Av}(T) \implies (t \circ R, \rho \circ R) \in \mathrm{Av}(T \circ h \circ (2 \cdot \mathcal{N} + 1)) .$$

Beweis: Wir unterteilen $X_{\rho \circ R}^{\leq l}$ in die Mengen X_1 und X_2.

$$\begin{aligned}
X_1 &:= \{x \in X_{\rho \circ R}^{\leq l} \mid t(R(x)) \leq T(h(|x|))\} , \\
X_2 &:= \{x \in X_{\rho \circ R}^{\leq l} \mid t(R(x)) > T(h(|x|))\} .
\end{aligned}$$

Nun betrachten wir für alle l

$$\begin{aligned}
\sum_{x \in X^{\leq l}_{\rho\circ R}} \frac{h^{[-1]}(T^{[-1]}(t(R(x))))}{2\cdot|x|} &\leq \sum_{x\in X_1\cup X_2} \frac{h^{[-1]}(T^{[-1]}(t(R(x))))}{2\cdot|x|} \\
&\leq \frac{|X_1|}{2} + \sum_{x\in X_2} \frac{h^{[-1]}(T^{[-1]}(t(R(x))))}{2\cdot|x|} \\
&\leq \frac{l}{2} + \sum_{x\in X_2} \frac{T^{[-1]}(t(R(x)))}{2\cdot|h(x)|} \\
&\leq \frac{l}{2} + \sum_{\rho(R(x))\leq l} \frac{T^{[-1]}(t(R(x)))}{2\cdot|R(x)|} \\
&\leq \frac{l}{2} + \frac{1}{2}\sum_{\rho(x)\leq l} \frac{T^{[-1]}(t(x))}{|x|} \;\leq\; l
\end{aligned}$$

Dabei gilt $\frac{h^{[-1]}(T^{[-1]}(t(R(x))))}{|x|} \leq \frac{T^{[-1]}(t(R(x)))}{|h(x)|}$, da $\mathcal{N}/h^{[-1]}$ monoton zunehmend ist und

$$\frac{|h(x)|}{|x|} \leq \frac{|h(x)|}{h^{[-1]}(h(|x|))} \leq \frac{T^{[-1]}(T(h(|x|)))}{h^{[-1]}(T^{[-1]}(T(h(|x|))))} .$$

Die oben benutzte Ungleichung $t(R(x)) \geq T(h(|x|))$ gilt, da $T^{[-1]}/(h^{[-1]}\circ T^{[-1]})$ monoton zunehmend ist und es gilt

$$\begin{aligned}
\frac{|h(x)|}{|x|} &\leq \frac{T^{[-1]}(t(R(x)))}{h^{[-1]}(T^{[-1]}(t(R(x))))} \\
\Longrightarrow \qquad \frac{h^{[-1]}(T^{[-1]}(t(R(x))))}{|x|} &\leq \frac{T^{[-1]}(t(R(x)))}{|h(x)|} .
\end{aligned}$$

∎

Die letzten beiden Lemmata hatten beide die Forderung

$$|R(x)| \leq h(|x|) .$$

Damit unterliegt die Reduktion einer *worst-case*-Bedingung. Die entsprechende Bedingung in Lemma 23 war wesentlich freizügiger gegeben durch

$$(|R|, \rho\circ R) \in \mathrm{Med}\,(h, a_1) .$$

Ziel sollte es aber sein, time_R und damit $|R|$ im *average* mit dieser Funktion h zu beschränken mit

$$(|R|, \rho\circ R) \in \mathrm{Eav}\,(h) \quad \text{oder} \quad \mathrm{Av}\,(h) .$$

Nun finden sich für das Eav-Maß Gegenbeispiele, die den Einsatz echter *average*-Reduktionen zwischen gewichteten Problemen verhindern. Jedoch gelingt es uns zu zeigen, daß im Av-Maß diese ihren Platz haben.

Lemma 26 *Sei* $R : \Sigma^* \to \Sigma^*$ *eine injektive Funktion mit*

$$(|R|, \rho \circ R) \ \in \ \mathrm{Av}\,(h) \ ,$$

wobei für die Funktion $h : \mathbb{N} \to \mathbb{N}$ *die Funktion* $\mathcal{N}/h^{[-1]}$ *monoton zunehmend ist. Zusätzlich definieren wir* $G := T \circ \frac{\mathcal{N}}{h^{[-1]}} \circ T^{[-1]}$ *und* $g := G^{[-1]}$*. Falls* g *monoton zunehmend ist, gilt*

$$\begin{aligned}(t, \rho) \in \mathrm{Av}(T) \text{ und } (|R|, \rho \circ R) \in \mathrm{Av}(h) \quad &\Longrightarrow \\ (t \circ R, \rho \circ R) \in \mathrm{Av}(g \circ T \circ h \circ (3 \cdot \mathcal{N} + 2)) \ .\end{aligned}$$

Beweis:
Wir unterteilen $X^{\leq l}_{\rho \circ R}$ in die vier Mengen X_1, X_2, X_3 und X_4:

$$\begin{aligned}
X_1 &:= \{x \in X^{\leq l}_{\rho \circ R} \mid t(R(x)) \leq g(T(|R(x)|))) \text{ und } |R(x)| \leq h(|x|)\} \ , \\
X_2 &:= \{x \in X^{\leq l}_{\rho \circ R} \mid t(R(x)) > g(T(|R(x)|))) \text{ und } |R(x)| \leq h(|x|)\} \ , \\
X_3 &:= \{x \in X^{\leq l}_{\rho \circ R} \mid t(R(x)) \leq g(T(|R(x)|))) \text{ und } |R(x)| > h(|x|)\} \ , \\
X_4 &:= \{x \in X^{\leq l}_{\rho \circ R} \mid t(R(x)) > g(T(|R(x)|))) \text{ und } |R(x)| > h(|x|)\} \ .
\end{aligned}$$

Wir beweisen jetzt mittels dieser Unterteilung, daß in der Summe

$$\sum_{\rho(R(x)) \leq l} \frac{(g \circ T \circ h)^{[-1]}(t(R(x)))}{|x|}$$

jeder Summenterm nach oben beschränkt ist durch entweder 1, $\dfrac{h^{[-1]}(|R(x)|)}{|x|}$ oder $\dfrac{T^{[-1]}(t(R(x)))}{|R(x)|}$. Damit ist die Summe höchstens $3 \cdot l$, und die gewünschte Implikation folgt sofort.

Für alle x gilt, da $G := T \circ \frac{\mathcal{N}}{h^{[-1]}} \circ T^{[-1]}$ und $g := G^{[-1]}$,

$$\frac{h^{[-1]}(T^{[-1]}(g^{[-1]}(t(R(x)))))}{|x|} \ \leq \ \frac{h^{[-1]}\left(\frac{T^{[-1]}(t(R(x)))}{h^{[-1]}(T^{[-1]}(t(R(x))))}\right)}{|x|} \ .$$

Also ergibt sich für alle Zeichenketten, da $g^{[-1]} \leq \mathcal{N}$.

1. für $x \in X_1$
$$\frac{h^{[-1]}(T^{[-1]}(g^{[-1]}(t(R(x)))))}{|x|} \ \leq \ 1 \ .$$

2. für $x \in X_2$

$$\begin{aligned}
\frac{h^{[-1]}(T^{[-1]}(g^{[-1]}(t(R(x)))))}{|x|} &\leq \frac{h^{[-1]}(T^{[-1]}(t(R(x))))}{|x|} \\
\leq \frac{T^{[-1]}(t(R(x)))}{h(|x|)} &\leq \frac{T^{[-1]}(t(R(x)))}{|R(x)|}
\end{aligned}$$

3. für $x \in X_3$

$$\frac{h^{[-1]}(T^{[-1]}(g^{[-1]}(t(R(x)))))}{|x|} \leq \frac{h^{[-1]}(|R(x)|)}{|x|}$$

4. für $x \in X_4$

$$\begin{aligned}
& & t(R(x)) &> g(T(|R(x)|)) \\
&\Longrightarrow & T^{[-1]}(g^{[-1]}(t(R(x)))) &\geq |R(x)| \\
&\Longrightarrow & T^{[-1]} \circ T \circ \frac{\mathcal{N}}{h^{[-1]}} \circ T^{[-1]} \circ t \circ R \circ x &\geq |R(x)| \\
&\Longrightarrow & \frac{T^{[-1]}(t(R(x)))}{h^{[-1]}(T^{[-1]}(t(R(x))))} &\geq |R(x)| \\
&\Longrightarrow & \frac{h^{[-1]}(T^{[-1]}(t(R(x))))}{|x|} &\leq \frac{T^{[-1]}(t(R(x)))}{|R(x)|} \\
&\Longrightarrow & \frac{h^{[-1]}(T^{[-1]}(g^{[-1]}(t(R(x)))))}{|x|} &\leq \frac{T^{[-1]}(t(R(x)))}{|R(x)|}
\end{aligned}$$

∎

6.3 Polynomial-Zeit-Reduktionen

Nachdem wir uns die Reduktionen der gewichteten Probleme für alle drei Durchschnittsmaße erarbeitet haben, kombinieren wir diese mit der Dominanz der Rangfunktionen und erhalten so eine große Bandbreite an zulässigen Polynomial-Zeit-Reduktionen.

Dabei orientiert sich die ***worst-case*-Reduktion** (***wc***) an der eingangs vorgestellten Reduktion. Jede *wc*-Reduktion ist auch eine ***average-case*-Reduktion** (***ac***). Hierbei wird die Zeit für die Reduktion polynomiell bezüglich des Av-Maßes beschränkt.

Andererseits kann eine *wc*-Reduktion auch zu einer ***median-case*-Reduktion** (***mc***) verallgemeinert werden. Hier wird nur noch verlangt, daß die Zeit für die Berechnung der Reduktion im Median beschränkt wird durch $\mathrm{Med}\left(\mathrm{POL}, \frac{\mathcal{N}}{\log^{\omega(1)}}\right)$.

Erlaubt man der Reduktion hier auch fehlerhafte Abbildungen, so erhält man die ***median-error*-Reduktion** (***me***).

Definition 37 *Eine injektive Funktion* $R : \Sigma^* \to \Sigma^*$ *ist eine gewichtete* ***h*** ***-Polynomial-Zeit-Reduktion****, wobei* $\boldsymbol{h \in \{wc, ac, mc, me\}}$ *,*

$$P_1 \quad \leq_h \quad P_2$$

des gewichteten Problems $P_1 := (L_1, \rho_1)$ *auf das gewichtete Problem* $P_2 := (L_2, \rho_2)$ *, falls folgende Bedingungen eingehalten werden.*

1. $x \in L_1 \iff R(x) \in L_2$.
2. *Es existiert eine Turing-Maschine* M *, die* R *berechnet, falls* $h \in \{\mathrm{wc}, \mathrm{ac}, \mathrm{mc}\}$ *. Zusätzlich genügt* M *(und* R *) der folgenden Zeit-Beschränkung.*

   wc: time_M *ist polynomiell beschränkt,*

   ac: $(\mathrm{time}_M, \rho_1) \in Av(\mathrm{POL})$,

   mc: $(\mathrm{time}_M, \rho_1) \in \mathrm{Med}\left(\mathrm{POL}, \frac{\mathcal{N}}{\log^{\omega(1)}}\right)$,

   me: $(\mathrm{time}_{M,R}, \rho_1) \in \mathrm{Med}\left(\mathrm{POL}, \frac{\mathcal{N}}{\log^{\omega(1)}}\right)$.
3. *Dominanz: Die Wahrscheinlichkeitsverteilung* ρ_2 *wird von* ρ_1 *dominiert.*

wc,ac: $\rho_2(R(x)) \leq \rho_1(x) \cdot \mathrm{Pol}(|x|)$,

mc,me: $\rho_2(R(x)) \leq \rho_1(x) \cdot \mathrm{plog}(\rho_1(x)) \cdot \mathrm{Pol}(|R(x)|)$.

Das folgende Theorem zeigt nun die Anwendungsmöglichkeiten der *wc*-, *ac*-, *mc*- und *me*-Reduktionen. Um für die *average*-Maße einen sinnvollen Reduktionsbegriff zu erhalten muß gemäß Lemma 22 zusätzlich die Eigenschaft

$$\rho_2, \rho_4 \ \in \ \mathrm{POL}\text{–describable}$$

gelten. Nun sind nach Proposition 5 alle gängigen polynomiell berechenbaren Klassen von Wahrscheinlichkeitsverteilungen POL–sampleable, POL–computable und POL–rankable Teilmengen von POL–describable. Somit handelt es sich hier um keine eigentliche Einschränkung.

Theorem 19 *Für* $P_2 := (L_2, \rho_2)$ *,* $P_4 := (L_4, \rho_4)$ *mit* $\rho_2, \rho_4 \in \mathrm{POL}\text{–describable}$ *gilt*

$$\begin{aligned}
P_1 \ \leq_{wc} \ P_2 \ \text{ und } \ P_2 \in \mathrm{EavDis}\mathcal{P} &\implies P_1 \ \in \ \mathrm{EavDis}\mathcal{P} \ , \\
P_3 \ \leq_{ac} \ P_4 \ \text{ und } \ P_4 \in \mathrm{AvDis}\mathcal{P} &\implies P_3 \ \in \ \mathrm{AvDis}\mathcal{P} \ , \\
P_5 \ \leq_{mc} \ P_6 \ \text{ und } \ P_6 \in \mathrm{MedDis}\mathcal{P} &\implies P_5 \ \in \ \mathrm{MedDis}\mathcal{P} \ , \\
P_7 \ \leq_{me} \ P_8 \ \text{ und } \ P_8 \in \mathrm{Med}^*\mathrm{Dis}\mathcal{P} &\implies P_7 \ \in \ \mathrm{Med}^*\mathrm{Dis}\mathcal{P} \ .
\end{aligned}$$

Beweis: Durch direkte Anwendung des Korollars 19, der Lemmata 23, 24 und 26 auf die Definition 37 unter Berücksichtigung der in Kapitel 2 beschriebenen Eigenschaften in den Lemmata 3, 8 und Korollar 5. ∎

Auf diese Weise erhält jede polynomiell beschränkte Komplexitätsklasse ihre eigene Reduktion. Dabei ist die *wc*-Reduktion eine Einschränkung der *ac*-Reduktion, wie auch die *mc*-Reduktion eine eingeschränkte *me*-Reduktion ist. Ist die Reduktion *honest*, das heißt $\mathrm{Pol}(|R(x)|) \geq |x|$, ist die restriktive *wc*-Reduktion ein Spezialfall aller vorgestellten Reduktionsarten.

Es liegt auf der Hand, daß wegen der Unvergleichbarkeit der polynomiellen *average*-Klassen mit den Median-Klassen auch die *ac*-Reduktion weder ein Spezialfall noch eine Verallgemeinerung einer *mc*- oder *me*-Reduktion ist.

Kapitel 7

Dis$\mathcal{NP}$ und Vollständigkeit

Wenn wir von Rangfunktionen sprechen, meinen wir in diesem Kapitel nur injektive Funktionen. Dies schränkt die zu erwartenden Ergebnisse kaum ein. Wie in Korollar 2 kann man zeigen, daß Probleme ihr *average*- und Median-Zeitverhalten nicht verändern, wenn man die Rangfunktion in eine injektive Funktion abwandelt. Somit umfaßt die Menge POL–rankable nun die Menge aller in polynomieller Zeit berechenbaren injektiven Rangfunktionen.

7.1 Dis$\mathcal{NP}$

Der Dualismus von $\mathcal{P}$ und $\mathcal{NP}$ ist nun schon seit fast einem Vierteljahrhundert das Diskussionsthema innerhalb der gesamten Informatik. Die vielen wichtigen Aspekte, die eine Reihe von angesehnster Wissenschaftler bis heute untersucht haben, können hier nicht abgehandelt werden.

Wir wollen uns nun fragen, wie die äquivalente Fragestellung in unseren Durchschnittsmaßen lautet. Die erste Beobachtung ist, daß wir $\mathcal{NP}$ in eine gewichtete Komplexitätsklasse übersetzen müssen. Es sollen aber nicht beliebige Wahrscheinlichkeitsverteilungen angegliedert werden, denn wir haben bereits gesehen, daß zu komplexe Verteilungen wieder nur zu einer klassischen *worst-case*-Betrachtung führen. Wir definieren deswegen **Dis$\mathcal{NP}$** (***distributional*** $\boldsymbol{\mathcal{NP}}$) als die Kombination von $\mathcal{NP}$ und POL–rankable.

Definition 38

$$\mathbf{Dis}\boldsymbol{\mathcal{NP}} \;:=\; \mathcal{NP} \times \text{POL–rankable} \;=\; \{(L, \mu) \mid L \in \mathcal{NP} \text{ und } \mu \in \text{POL–rankable}\}\,.$$

Das letzte Kapitel hat uns Reduktionen für gewichtete Problemklassen gezeigt. Dies veranlaßt uns jetzt, auch die Teilklasse von Dis $\mathcal{NP}$ aus vollständigen gewichteten Problemen zu definieren. Dabei ist zu beachten, daß wir vier verschiedene Reduktionstypen zur Verfügung stehen: ***wc, ac, mc*** und ***me***.

Definition 39 *Ein gewichtetes Problem P_1 ist in der Komplexitätsklasse* **Dis$\mathcal{NP}$C$_{\text{wc}}$** , **Dis$\mathcal{NP}$C$_{\text{ac}}$** , **Dis$\mathcal{NP}$C$_{\text{mc}}$** *oder* **Dis$\mathcal{NP}$C$_{\text{me}}$** *der* **Dis$\mathcal{NP}$-vollständigen** *Probleme bezüglich Reduktionen von Typ $h \in \{wc, ac, mc, me\}$ gdw.*

$$\begin{array}{rrcl} & P_1 & \in & \mathrm{Dis}\mathcal{NP} \ , \\ \forall P_2 \in \mathrm{Dis}\mathcal{NP} & P_2 & \leq_h & P_1 \ . \end{array}$$

In der klassischen *worst-case*-Komplexitätstheorie gelang es, tausende von $\mathcal{NP}$-Problemen als $\mathcal{NP}$-vollständig zu charakterisieren. Hätte ein einziges $\mathcal{P}$-Problem als $\mathcal{NP}$-vollständig dargestellt werden können, hätte dies weitreichende Folgen gehabt:

$$\mathcal{NP}\mathrm{C} \ \cap \ \mathcal{P} \ \neq \emptyset \Longrightarrow \mathcal{P} \ = \ \mathcal{NP} \ .$$

Für die gewichteten Komplexitätsklassen gelten analoge Implikationen.

Korollar 20

$$\begin{array}{rcl} \mathrm{EavDis}\mathcal{P} \ \cap \ \mathrm{Dis}\mathcal{NP}\mathrm{C}_{wc} \neq \emptyset & \Longrightarrow & \mathrm{Dis}\mathcal{NP} \subseteq \mathrm{EavDis}\mathcal{P} \subseteq \mathrm{AvDis}\mathcal{P} \ , \\ \mathrm{AvDis}\mathcal{P} \ \cap \ \mathrm{Dis}\mathcal{NP}\mathrm{C}_{ac} \neq \emptyset & \Longrightarrow & \mathrm{Dis}\mathcal{NP} \subseteq \mathrm{AvDis}\mathcal{P} \ , \\ \mathrm{MedDis}\mathcal{P} \ \cap \ \mathrm{Dis}\mathcal{NP}\mathrm{C}_{mc} \neq \emptyset & \Longrightarrow & \mathrm{Dis}\mathcal{NP} \subseteq \mathrm{MedDis}\mathcal{P} \subseteq \mathrm{Med}^*\mathrm{Dis}\mathcal{P} \ , \\ \mathrm{EavDis}\mathcal{P} \ \cap \ \mathrm{Dis}\mathcal{NP}\mathrm{C}_{me} \neq \emptyset & \Longrightarrow & \mathrm{Dis}\mathcal{NP} \subseteq \mathrm{Med}^*\mathrm{Dis}\mathcal{P} \ . \end{array}$$

Beweis: folgt direkt aus der Definition und Theorem 19 im vorherigen Kapitel. ∎

Wie sehen Dis $\mathcal{NP}$-vollständige Probleme aus? Die naheliegende Vermutung, eine Kombination eines $\mathcal{NP}$-vollständigen Problems mit einer möglichst einfachen Wahrscheinlichkeitsverteilung würde das gewünschte Ergebnis liefern, erweist sich als irreführend, wie folgendes Beispiel zeigt.

Beispiel:
Wir betrachten das **Erfüllbarkeitsproblem Boolescher Funktionen** (`SAT`– ***satisfiability problem***).

Die Boolesche Formel wird mit dem Zeichensatz $\mathbf{\Sigma}_{\mathtt{Sat}} := \{0, 1, \neg, \vee, \wedge\}$ kodiert. Alle Boolesche Formeln sind in konjunktiver Normalform gegeben:

$$(x_0 \vee \overline{x_{110}} \vee x_1) \wedge (x_1 \vee x_{110}) \wedge (x_\lambda \vee x_0)$$

Die Umsetzung der Klammern ist nicht notwendig. Auch werden für die Variablen nur die Indizes als binäre Zeichenketten gesetzt. Wir schreiben also für obige Formel einfach

$$0 \vee \neg 110 \vee 1 \wedge 1 \vee 110 \wedge \vee 0 \,.$$

Wir wollen SAT im Zusammenhang mit der uniformen Wahrscheinlichkeitsverteilung betrachten. Kommt dann beispielsweise $0\neg 1$ oder $0\neg 1\neg\neg$ vor, so interpretieren wir es als die negierte Variable $\overline{x_{01}}$.

Betrachtet man *worst-case*-Laufzeiten, sind solche Fragen der Kodierung weniger wichtig, solange es sich nur die Eingabelänge höchstens polynomiell verändert. Hier sind sie aber essentiell.

Wird dieses Problem $(\mathtt{SAT}, \mu_{\mathrm{uni}})$ nun einer *average-case*-Analyse unterzogen, stellt man fest, daß die Wahrscheinlichkeit für $x \notin \mathtt{SAT}$ sehr hoch ist. Schließlich impliziert das Vorkommen der Zeichenfolge $\wedge\neg 0 \wedge 0 \wedge$, die als $\ldots \wedge \ \overline{x_0} \ \wedge \ x_0 \ \wedge \ldots$ interpretiert wird, daß die zugehörige Formel nicht mehr erfüllbar ist. Die Wahrscheinlichkeit dieser Zeichenkette ist lokal $\frac{1}{5^6}$. Damit ist die Wahrscheinlichkeit für mindestens ein Vorkommen in einer Zeichenkette der Länge n größer als $1 - \left(\frac{5^6-1}{5^6}\right)^{n/6}$. Zeichenketten solcher Form sind andererseits in linearer Zeit erkennbar.

Für eine Boolesche Formel der Kodierungslänge n mit m Variablen kann aber eine deterministische Turing-Maschine in $O(n^2 \cdot \exp m)$ Schritten entscheiden, ob sie erfüllbar ist. Dies kann dadurch geschehen, daß jede mögliche Belegung der Formel eingesetzt wird. Da $m \leq \frac{n}{\log n}$ ergibt sich folgende Analyse für eine Konstante $c < 1$ (unter Benutzung des Median-Maßes und Theorem 3).

$$\begin{array}{rcl} \mathtt{SAT} & \in & \mathrm{DTime}\left(\mathcal{N}^2 \cdot \exp\left(\dfrac{\mathcal{N}}{\log \mathcal{N}}\right)\right) \ \text{ und} \\ (\mathtt{SAT}, \mu_{\mathrm{uni}}) & \in & \mathrm{MedDisTime}(\mathcal{N}, \mathcal{N}^c) \\ & \Longrightarrow & (\mathtt{SAT}, \mu_{\mathrm{uni}}) \ \in \ \mathrm{EavDis}\mathcal{P} \,. \end{array}$$

Also ist $(\mathtt{SAT}, \mu_{\mathrm{uni}})$ im Erwartungswert effizient berechenbar. Eine Charakterisierung als vollständiges gewichtetes Problem ist dagegen nicht in Sicht.

7.2 Ein vollständiges Problem

Als Sprache wählen wir hierzu das **beschränkte Halteproblem nicht-deterministischer Turing-Maschinen** NBH.

Definition 40 *Das* **beschränkte Halteproblem** NBH *für nicht-deterministische Turing-Maschinen* M_i *beschreibt die Sprache* $\mathtt{NBH} \subseteq \{0,1\}^*$

$$\mathtt{NBH} \ := \ \{(x01^t0^i) \mid time_{M_i}(x) \leq t\} \ .$$

Als geeignete Verteilung wird sich die folgende herausstellen

Definition 41

$$\boldsymbol{\rho_{\mathtt{NBH}}(w)} \quad := \quad \begin{cases} \mathrm{ord}(x \ \# \ \mathrm{bin}(t) \ \# \ \mathrm{bin}(i)) & \textit{falls } w = x01^t0^i, \\ \infty & \textit{sonst.} \end{cases}$$

Die Ordnungsrelation $\mathrm{ord}(a\#b\#c)$ für $a,b,c \in \Sigma^*$ sollte dabei in linearer Zeit berechenbar sein, und für eine Konstante k und eine beliebige totale Funktion f sollte gelten

$$\mathrm{ord}(a\#b\#c) \ \leq \ \mathrm{ord}(a) \cdot \mathrm{ord}(b)^k \cdot f(\mathrm{ord}(c)) \ .$$

Eine mögliche Realisation wäre zum Beispiel

$$\mathrm{ord}(\mathrm{cod}_{\mathrm{length}}(c, (\mathrm{cod}_{\mathrm{length}}(b,a)))) \ ,$$

wobei $\mathrm{cod}_{\mathrm{length}}$ bereits auf Seite 110 beschrieben wurde. Damit ist die Rangfunktion in linearer Zeit berechenbar. Das gewichtete Problem $(\mathtt{NBH}, \rho_{\mathtt{NBH}})$ ist Dis $\mathcal{NP}$ vollständig bezüglich Reduktionen vom Typ wc. Die folgende Reduktion benutzt eine von Levins Techniken bezüglich anderer Berechenbarkeitsbeschränkungen von Verteilungen (siehe [Levin 86, Gure 91b]).

Theorem 20

$$(\mathtt{NBH}, \rho_{\mathtt{NBH}}) \ \in \ \mathrm{Dis}\mathcal{NP}\mathrm{C}_{wc} \ \cap \ \mathrm{Dis}\mathcal{NP}\mathrm{C}_{mc}.$$

Beweis: Sei $(L,\rho) \in$ Dis $\mathcal{NP}$, wobei ρ eine Rangfunktion ist. Sei M eine nicht-deterministische Turing-Maschine, die Elemente in L in polynomieller Zeit $q(|x|)$ akzeptiert. Falls $x \notin L$, gibt es keine haltende Berechnung von M.

Wir betrachten nun eine nicht-deterministische Maschine M_j, die sich aus der Kenntnis von ρ folgendermaßen ergibt.

M_j berechnet auf Eingabe bx, für $b \in \{0,1\}$ und $x \in \Sigma^*$:

$$M_j(bx) \ := \ \begin{cases} x \ , & \text{falls } b = 1 \ , \\ y \ , & \text{sonst, wobei } \rho(y) = \mathrm{ord}(x) \ . \end{cases}$$

Es ist nicht schwer einzusehen, daß M_j polynomiell zeit-beschränkt ist. Sei p ein entsprechendes zeit-beschränkendes Polynom.

Nun definieren wir die Reduktion R durch

$$R(x) := \begin{cases} 1\ x\ 0\ 1^{p(|x|)}\ 0^j\ , & \text{falls } \rho(x) \geq \mathrm{ord}(x)\ , \\ 0\ \mathrm{ord}^{[-1]}(\rho(x))\ 0\ 1^{p(|x|)}\ 0^j\ , & \text{sonst}\ . \end{cases}$$

R ist eine *wc*-Reduktion von (L, ρ) auf $(\mathtt{NBH}, \rho_{\mathtt{NBH}})$, denn es gilt

1. $x \in L \iff R(x) \in \mathtt{NBH}$,
2. R ist in polynomieller Zeit berechenbar ,
3. die Dominanzbedingung:
   Zu beweisen ist hier
   $$\rho_{\mathtt{NBH}}(R(x)) \ \leq\ \mathrm{Pol}(|x|) \cdot \rho(x)\ .$$
   Betrachten wir die beiden Fälle
   (a) $\rho(x) \geq \mathrm{ord}(x)$:
   $$\begin{aligned} \rho_{\mathtt{NBH}}(R(x)) &\leq \rho_{\mathtt{NBH}}(1\ x\ 0\ 1^{p(|x|)}\ 0^j) \\ &= \mathrm{ord}(x \# \mathrm{bin}(p(|x|)) \# \mathrm{bin}(j)) \\ &\leq \mathrm{ord}(x) \cdot p(|x|)^k \cdot f(j) \end{aligned}$$
   (b) $\rho(x) \leq \mathrm{ord}(x)$:
   $$\begin{aligned} \rho_{\mathtt{NBH}}(R(x)) &\leq \rho_{\mathtt{NBH}}(0\ \mathrm{ord}^{[-1]}(\rho(x))\ 0\ 1^{p(|x|)}\ 0^j) \\ &= \mathrm{ord}(\mathrm{ord}^{[-1]}(\rho(x)) \# \mathrm{bin}(p(|x|)) \# \mathrm{bin}(j)) \\ &\leq \rho(x) \cdot p(|x|)^k \cdot f(j)\ . \end{aligned}$$
   Offensichtlich erfüllt R die Eigenschaft einer *wc*-Reduktion, und somit auch die einer *ac*-Reduktion.

   Da $|R(x)| \geq p(|x|)$, handelt es sich bei R auch um eine *mc*-Reduktion, also auch *me*-Reduktion.

∎

Die Sprache NBH ist mit dieser Eigenschaft kein Unikum. Praktisch für jedes bekannte $\mathcal{NP}$-vollständige Problem kann eine Rangfunktion gefunden werden, so daß das entstehende Paar ein Dis $\mathcal{NP}$ -vollständiges Problem entsteht, wie nachstehendes Beispiel zeigt.

Beispiel:
Wir betrachten wieder die Sprache SAT und benutzen eine Standard-Reduktion von NBH auf SAT, welche in Zeit $O(\mathcal{N}^2 \cdot \log)$ berechnet werden kann. Für eine Eingabe $x = x_1 x_2 \ldots x_n \in \{0,1\}^n$ erzeugt diese Reduktion eine Ausgabe der Länge $\Theta(\mathcal{N}^2 \cdot \log)$.

Wir modifizieren diese Reduktion leicht, um schnelle Umkehrbarkeit zu erreichen, die resultierende Reduktion heiße $\tilde{\boldsymbol{R}}$: Das Ergebnis der Standard-Reduktion, die Boolesche Formel $F(y_1, y_2 \ldots y_m)$, ersetzen wir durch

$$\boldsymbol{F'} \quad := \quad (1 \vee x_1) \wedge \cdots (1 \vee x_n) \wedge F(y_1, y_2 \ldots y_m) \; .$$

Auf diese Weise kann die inverse Funktion in linearer Zeit berechnet werden. Zuerst wird x am Anfang von F' gelesen (wobei gleichzeitig getestet wird, ob x zu lang ist). Dann simulieren wir die Reduktion R auf Eingabe-Zeichenkette x. Falls die Reduktion ein Ergebnis ausgibt, das sich syntaktisch von der Eingabe F' unterscheidet, verwirft sie, andernfalls gibt sie x aus.

Die Rangfunktion ρ_{Sat} berechnet sich hierbei aus

$$\boldsymbol{\rho_{\mathrm{Sat}}(x)} \quad := \quad \rho_{\mathtt{NBH}}(\tilde{R}^{[-1]}(x)) \; .$$

Für Eingaben x, die nicht im Bildbereich von $\tilde{R}$ liegen, setzen wir $\rho_{\mathrm{Sat}}(x) := \infty$. Somit gilt

$$\begin{array}{rlcl} & (\mathtt{NBH}, \rho_{\mathtt{NBH}}) & \leq_{wc,mc} & (\mathtt{SAT}, \rho_{\mathrm{Sat}}) \\ \Longrightarrow & (\mathtt{SAT}, \rho_{\mathrm{Sat}}) & \in & \mathrm{Dis}\mathcal{NP}\mathrm{C}_{wc,mc} \; . \end{array}$$

7.3 Eav$\mathcal{P}$, Av$\mathcal{P}$ und Med$\mathcal{P}$

In den Kapiteln 3 und 4 untersuchten wir ausführlich die *average*-Sprachen-Klassen EavTime(T, V–rankable), AvTime(T, V–rankable), sowie die Median-Sprachen-Klassen MedTime(T, V–rankable), Med* Time(T, V–rankable). Die entsprechenden effizienten *average*- und Median-Klassen nennen wir **Eav $\mathcal{P}$**, **Av $\mathcal{P}$**, **Med $\mathcal{P}$** und **Med $^*\mathcal{P}$**, und definieren sie folgendermaßen:

Definition 42

$$\begin{array}{rcl} \mathbf{Av}\boldsymbol{\mathcal{P}} & := & \mathrm{AvTime(POL, POL–rankable)} \; , \\ \mathbf{Eav}\boldsymbol{\mathcal{P}} & := & \mathrm{EavTime(POL, POL–rankable)} \; , \\ \mathbf{Med}\boldsymbol{\mathcal{P}} & := & \mathrm{MedTime}\left(\mathrm{POL}, \dfrac{\mathcal{N}}{\log^{\omega(1)}}, \mathrm{POL–rankable}\right) \; , \\ \mathbf{Med}^*\boldsymbol{\mathcal{P}} & := & \mathrm{Med}^*\mathrm{Time}\left(\mathrm{POL}, \dfrac{\mathcal{N}}{\log^{\omega(1)}}, \mathrm{POL–rankable}\right) \; . \end{array}$$

Wenn wir zum Beispiel die Klasse Av$\mathcal{P}$ betrachten, so wird für eine Sprache L in dieser Klasse verlangt, daß für jede Wahrscheinlichkeitsverteilung mit effizient berechenbarer Rangfunktion, ein effizienter Algorithmus für L existiert.

Hierbei handelt es sich um harte Bedingungen für gutes *average*- oder Median-Verhalten einer Sprache. Beeinflußt dies aber das qualitative Verhältnis zu den nicht-deterministischen polynomiell zeit-beschränkten Zeitklassen?

Ziel dieses Abschnittes soll es daher sein, die Aussage Av$\mathcal{P}$ versus $\mathcal{NP}$ in Relation zu setzen mit der bereits untersuchten Frage AvDis$\mathcal{P}$ versus Dis$\mathcal{NP}$.

Hierzu führen wir einen Reduktionsbegriff für Sprachen ein, der sich auf die 4 Reduktionsarten (*wc*, *ac*, *mc* und *me*) gewichteter Probleme bezieht.

Definition 43 *Eine Sprache L ist* ***$\mathcal{NP}$-vollständig bezüglich einer Reduktion vom Typ $h \in \{wc, ac, mc, me\}$*** *gdw.*

$$\begin{array}{rrcl} & L & \in & \mathcal{NP}\ , \\ \forall P' \in \mathrm{Dis}\mathcal{NP} \quad \exists \rho \in \mathrm{POL\text{–}rankable} & P' & \leq_h & (L,\rho)\ . \end{array}$$

Die Klassen aller $\mathcal{NP}$-vollständigen Sprachen bezüglich h heißen $\mathcal{NP}\mathrm{C}_h$.

Die Beziehungen zwischen $\mathcal{NP}\mathrm{C}_h$ und Dis$\mathcal{NP}\mathrm{C}_h$ ergeben sich so durch

Lemma 27

$$\begin{array}{rcl} \exists \rho_1 \in \mathrm{POL\text{–}rankable} \quad (L_1,\rho_1) \in \mathrm{Dis}\mathcal{NP}\mathrm{C}_{wc} & \Longleftrightarrow & L_1 \in \mathcal{NP}\mathrm{C}_{wc}\ , \\ \exists \rho_2 \in \mathrm{POL\text{–}rankable} \quad (L_2,\rho_2) \in \mathrm{Dis}\mathcal{NP}\mathrm{C}_{ac} & \Longleftrightarrow & L_2 \in \mathcal{NP}\mathrm{C}_{ac}\ , \\ \exists \rho_3 \in \mathrm{POL\text{–}rankable} \quad (L_3,\rho_3) \in \mathrm{Dis}\mathcal{NP}\mathrm{C}_{mc} & \Longleftrightarrow & L_3 \in \mathcal{NP}\mathrm{C}_{mc}\ , \\ \exists \rho_4 \in \mathrm{POL\text{–}rankable} \quad (L_4,\rho_4) \in \mathrm{Dis}\mathcal{NP}\mathrm{C}_{me} & \Longleftrightarrow & L_4 \in \mathcal{NP}\mathrm{C}_{me}\ . \end{array}$$

Beweis: durch direktes Einsetzen der entsprechenden Definition. ∎

Damit gelingt es uns, den zu Korollar 20 analogen Satz zu zeigen.

Theorem 21

$$\begin{array}{rcl} \mathrm{Eav}\mathcal{P} \cap \mathcal{NP}\mathrm{C}_{wc} \neq \emptyset & \Longrightarrow & \mathcal{NP} \subseteq \mathrm{Eav}\mathcal{P} \subseteq \mathrm{Av}\mathcal{P}\ , \\ \mathrm{Av}\mathcal{P} \cap \mathcal{NP}\mathrm{C}_{ac} \neq \emptyset & \Longrightarrow & \mathcal{NP} \subseteq \mathrm{Av}\mathcal{P}\ , \\ \mathrm{Med}\mathcal{P} \cap \mathcal{NP}\mathrm{C}_{mc} \neq \emptyset & \Longrightarrow & \mathcal{NP} \subseteq \mathrm{Med}\mathcal{P} \subseteq \mathrm{Med}^*\mathcal{P}\ , \\ \mathrm{Eav}\mathcal{P} \cap \mathcal{NP}\mathrm{C}_{me} \neq \emptyset & \Longrightarrow & \mathcal{NP} \subseteq \mathrm{Med}^*\mathcal{P}\ . \end{array}$$

Beweis: beispielsweise für *ac* (analog für alle anderen Typen).

Aus $\mathrm{Av}\mathcal{P} \cap \mathcal{NP}\mathrm{C}_{ac} \neq \emptyset$ folgt die Existenz eines gewichteten Problems $(L, \rho) \in \mathrm{AvDis}\mathcal{P} \cap \mathrm{Dis}\mathcal{NP}\mathrm{C}_{ac}$. Nach Lemma 27 gilt jetzt für alle $(L', \rho') \in \mathrm{Dis}\,\mathcal{NP}$: $(L', \rho') \leq_{ac} (L, \rho)$.

Damit ist $(L', \rho') \in \mathrm{AvDis}\mathcal{P}$, für alle $\rho' \in$ POL–rankable und $L' \in \mathcal{NP}$. Hieraus folgt $\mathrm{Dis}\,\mathcal{NP} \subseteq \mathrm{Av}\mathcal{P}$. ∎

Dieser abschließende Satz zeigt, daß es letztlich äquivalent ist, *average*-Sprachen-Klassen mit $\mathcal{NP}$ zu vergleichen oder entsprechend gewichtete Problem-Klassen mit $\mathrm{Dis}\,\mathcal{NP}$.

Theorem 22

$$\begin{aligned}
\mathcal{NP} \subseteq \mathrm{Eav}\mathcal{P} &\iff \mathrm{Dis}\mathcal{NP} \subseteq \mathrm{EavDis}\mathcal{P}\,,\\
\mathcal{NP} \subseteq \mathrm{Av}\mathcal{P} &\iff \mathrm{Dis}\mathcal{NP} \subseteq \mathrm{AvDis}\mathcal{P}\,,\\
\mathcal{NP} \subseteq \mathrm{Med}\mathcal{P} &\iff \mathrm{Dis}\mathcal{NP} \subseteq \mathrm{MedDis}\mathcal{P}\,,\\
\mathcal{NP} \subseteq \mathrm{Med}^*\mathcal{P} &\iff \mathrm{Dis}\mathcal{NP} \subseteq \mathrm{Med}^*\mathrm{Dis}\mathcal{P}\,.
\end{aligned}$$

Beweis: „$\Rightarrow$“: (beispielsweise für Av)

Aus $\mathcal{NP} \subseteq \mathrm{Av}\mathcal{P}$ folgt

$$\begin{aligned}
\mathtt{NBH} \in \mathrm{Av}\mathcal{P} &\Rightarrow (\mathtt{NBH}, \mathrm{rank}_{\mathtt{NBH}}) \in \mathrm{AvDis}\mathcal{P}\\
&\Rightarrow \mathrm{Dis}\mathcal{NP} \subseteq \mathrm{AvDis}\mathcal{P}\,.
\end{aligned}$$

„$\Leftarrow$“: (beispielsweise für Med)

Aus $\mathrm{Dis}\,\mathcal{NP} \subseteq \mathrm{MedDis}\mathcal{P}$ folgt

$$\begin{aligned}
(\mathtt{NBH}, \mathrm{rank}_{\mathtt{NBH}}) \in \mathrm{MedDis}\mathcal{P} &\Rightarrow \forall L \in \mathcal{NP} \quad \forall \rho \in \text{POL–rankable} \quad (L, \rho) \in \mathrm{MedDis}\mathcal{P}\\
&\Rightarrow \mathcal{NP} \subseteq \mathrm{Med}\mathcal{P}\,.
\end{aligned}$$

∎

Kapitel 8

Diskussion und Ausblicke

8.1 Entwicklung der Durchschnittsmaße

Mit dem Begriff *average-case*-Analyse verbindet man in der Regel die Untersuchung der erwarteten Laufzeit eines Algorithmus. Dabei wählt man als Wahrscheinlichkeitsverteilung möglichst einfache, gleichgewichtete Verteilungen.

So wurde bereits eine ganze Reihe von Algorithmen analysiert, wobei meistens die erforderlichen mathematischen Techniken weitaus komplizierter und aufwendiger waren als in entsprechenden *worst-case*-Analysen. Einen Überblick solcher Techniken erhält man in [ViFl 90, Kemp 84].

Unter komplexitätstheoretischen Gesichtspunkten stößt man jedoch auf eine fundamentale Schwäche des Erwartungswertes: Polynomiell zeit-beschränkte Simulationen erhalten keine polynomiellen Zeit-Schranken.

Leonid Levin erkannte dieses Problem und kam zu folgender Lösung. Er definierte das Maß *polynomial on the average* [Levin 86, John 84b] als ein Durchschnittsmaß, das zwar schwächer als der Erwartungswert ist, aber abgeschlossen gegenüber Polynomial-Zeit-Simulationen verschiedener Maschinen-Modelle. Dieses Maß ermöglichte den entscheidenden Durchbruch in der *average*-Komplexitätstheorie [Gure 91b].

Er beschränkte die Menge der Wahrscheinlichkeitsverteilungen mittels der Klasse POL-computable (die diesen Namen erst später erhielt) und fand ein gewichtetes $\mathcal{NP}$-vollständiges Problem.

Das Äquivalent zu DTime(T) sollte die Klasse AverDTime(T) [BCGL 89] (entsprechend der Namenskonvention in dieser Arbeit LavDisTime) darstellen, die aus allen gewichteten Problemen mit Lav(T)-zeit-beschränkten Algorithmen besteht.

Die Klasse AverDTime($\mathcal{N}^2$) beinhaltet aber auch Probleme, die Algorithmen für alle Eingaben gerade mal in Zeit $\mathcal{N}^3$ lösen können. Grund hierfür ist die fehlende Präzision des Levinschen Maßes. Genau dieses Problem wurde in [Schi 91] besprochen. Das resultierende Maß Av(T) ermöglicht die präzise Klassifikation gewichteter Probleme zumindest für die polynomiellen Zeitklassen. Die Abgeschlosssenheitseigenschaft für Polynomial-Zeit-Simulationen übernahm dieses Maß von seinem Ursprung, dem Levinschen Maß. In [ReSc 93a] wird gezeigt, daß dieses Maß an komplexitätstheoretischer Bedeutung dem Levinschen Maß mindestens ebenbürtig ist. Für nicht zu stark abnehmende polynomiell beschreibbare Wahrscheinlichkeitsverteilungen zeigen wir hier, daß Lav(POL) und Av(POL) sogar äquivalent sind.

Dem Maß Av wird wie im Levinschen Maß Lav eine globale Wahrscheinlichkeitsverteilung zugrunde gelegt. Betrachtet man den Erwartungswert bezüglich solcher, haben wir in dieser Arbeit gesehen, daß auch hier die Präzision verloren geht. Es gelingt uns hier den Hauptgedanken zur Wiederherstellug umzusetzen: die Betrachtung der Äquivalenzklasse aller Wahrscheinlichkeitsvereilungen mit gleichrangig geordneten Wahrscheinlichkeiten.

Dies ergab das Durchschnittsmaß Eav, welches allein aufgrund der großen Verwandtheit zum lokal gewichteten Erwartungswert keine Abgeschlossenheit gegenüber polynomiellen Simulationen bieten kann: Für viele Wahrscheinlichkeitsverteilungen definieren Eav und lokal gewichteter Erwartungswert gleiche Komplexitätsklassen.

Für beide Maße untersuchten wir das Verhalten gegenüber elementaren Veränderungen der Zeitfunktionen und der Wahrscheinlichkeitsverteilungen. Diese verhielten sich ähnlich wie ihre nicht präzisen Verwandten Lav [Gure 91b] und der global gewichtete Erwartungswert. Insbesondere zeigen wir, daß für überlineare Schranken Av schwächer beschränkt als Eav.

Tiefere Einblicke in die Unterschiede für eine feste Wahrscheinlichkeitsverteilung ermöglicht das Median-Maß. Diese Übertragung des Medians in die Zeit-Analyse von Algorithmen ist neuartig. Wir zeigen unter anderen, wie die Kenntnis guter Median-Klassifikationen eines Problems obere und untere Schranken in den *average*-Maßen implizieren.

Betrachtet man Eigenschaften bezüglich Simulationen verschiedener Maschinen-Modelle sowie elementare Abbildungen der Laufzeiten und der Wahrscheinlichkeit, ergeben sich im Median-Maß wesentlich bessere Resultate als für die *average*-Maße.

8.2 Separationen

Präzise Maße ermöglichen erst enge Separationsergebnisse, und so finden wir für AvTime und EavTime zwei Arten von Hierarchien [Schi 91, ReSc 93a, ReSc 93b]. Zum einen entdeckt man die klassischen Zeithierarchien, wobei wir für $\mathcal{N} \leq T_1 \leq T_2$ für alle $V > \mathcal{N}$ erhalten

$$\begin{aligned} \mathrm{AvTime}(T_1, V\text{–rankable}) &\subset \mathrm{AvTime}(T_2, V\text{–rankable}) \ , \\ \mathrm{EavTime}(T_1, V\text{–rankable}) &\subset \mathrm{EavTime}(T_2, V\text{–rankable}) \ . \end{aligned}$$

Dieses Ergebnis wurde in schwächerer Form für ein anderes *average*-Maß in [GGH 93] bestätigt.

V –rankable kann hier durch jede andere Klasse ersetzt werden, die die uniforme Wahrscheinlichkeitsverteilung beinhaltet. Also gilt zum Beispiel auch

$$\begin{aligned} \mathrm{AvTime}(T_1, \mathrm{POL\text{-}computable}) &\subset \mathrm{AvTime}(T_2, \mathrm{POL\text{-}computable}) \ , \\ \mathrm{EavTime}(T_1, \mathrm{POL\text{-}computable}) &\subset \mathrm{EavTime}(T_2, \mathrm{POL\text{-}computable}) \ . \end{aligned}$$

Dieses wird impliziert durch Theorem 12

$$\begin{aligned} \mathrm{DTime}(T_2) \ \setminus \ \mathrm{AvTime}(T_1, \{\mu_{\mathrm{uni}}\}) &\neq \emptyset \ , \\ \mathrm{DTime}(T_2) \ \setminus \ \mathrm{EavTime}(T_1, \{\mu_{\mathrm{uni}}\}) &\neq \emptyset \end{aligned}$$

und $\mathrm{DTime}(T_2) \subseteq \mathrm{EavTime}(T_2, C)$, für alle Mengen C von Verteilungen.

Wir betrachten daraufhin in dieser Arbeit Hierarchien vollkommen anderen Typs. Untersucht wird, ob die Steigerung des Rechenaufwands zur Erzeugung der Wahrscheinlichkeitsverteilung schlechteres *average*-Verhalten impliziert und bewiesen

$$\begin{aligned} \mathrm{AvTime}(T, V_2\text{–rankable}) &\subset \mathrm{AvTime}(T, V_1\text{–rankable}) \ , \\ \mathrm{EavTime}(T, V_2\text{–rankable}) &\subset \mathrm{EavTime}(T, V_1\text{–rankable}) \ , \end{aligned}$$

für $V_1 \leq o(V_2) \leq T \leq \mathrm{POL}$ (Für Eav entfällt die Beschränkung durch POL).

Die Menge V –rankable ist dabei die Klasse von Verteilungen, deren Rangfunktionen in Zeit V berechenbar ist. Andere Ansätze zur Berechnung der Verteilung wie *computable* [Levin 86] und *sampleable* [BCGL 89] scheinen kaum geeignet für die präzisen Durchschnittsmaße Av und Eav zu sein.

Diese Verteilungs-Hierarchien enden für $T = V$. Für Aussagen $T \geq V$ bleibt dieses Problem offen.

8.3 Bösartige Wahrscheinlichkeitsverteilungen

Erhöht man V weiter (größer als exponentiell in T), entspricht der *worst-case* dem *average-case*. Für polynomielles T gilt

$$\begin{aligned} \mathrm{AvTime}(T, T \cdot \mathrm{EXL\text{–}rankable}) &= \mathrm{DTime}(T) , \\ \mathrm{EavTime}(T, T \cdot \mathrm{EXL\text{–}rankable}) &= \mathrm{DTime}(T) . \end{aligned}$$

Somit existiert in $T \cdot \mathrm{EXL}$–rankable für jede Sprache in $\mathrm{DTime}(T)$ eine bösartige Wahrscheinlichkeitsverteilung, die keine Verbesserung im *average* gegenüber dem *worst-case* zuläßt.

Solche bösartigen Wahrscheinlichkeitsverteilungen sind schon andernorts untersucht worden. Li und Vitanyi [LiVi 92] bewiesen sogar die Existenz einer universell bösartigen Verteilung mit Hilfe der Kolmogoroff-Komplexität. In [Koba 93] gibt es einen etwas modifizierten Ansatz für eine solche Verteilung. Universell bösartig heißt dabei, daß für jede Sprache diese Verteilung bösartig ist.

Über die Komplexität einer solchen Verteilung wurden keine Aussagen getroffen. Milterson [Milt 91] wies nach, daß für erwartet polynominell zeit-beschränkte Zeitklassen ein Σ_2-Orakel genügt. Wir zeigen, daß für Av und Eav ein $\mathcal{NP}$-Orakel für die Berechnung der Rangfunktion einer bösartigen Verteilung für eine Sprache aus $\mathcal{P}$ genügt.

Zwischen den Bereich der Verteilungs-Hierarchien und der Komplexität dieser bösartigen Rangfunktionen besteht also eine Lücke, für diese vermutet der Autor folgendes:

Für bösartige Rangfunktionen in den polynomiellen *average*-Maßen ist mindestens ein $\mathcal{NP}$-Orakel notwendig.

8.4 $\mathcal{NP}$-Probleme im Average

Betrachten wir nun am Beispiel des $\mathcal{NP}$-vollständigen Erfüllungsproblems Boolescher Funktionen `SAT` die Beziehungen, die zwischen $\mathcal{NP}$ und *Average*--Klassen untersucht wurden.

Davis und Putman [DaPu 60] stellten den bekanntesten Algorithmus zur Berechnung von `SAT` vor (Sie lösten damit sogar ein allgemeineres Problem). Für einige ganz einfache Wahrscheinlichkeitsverteilungen ist dieser Algorithmus schon erwartet polynomiell zeitbeschränkt [PuBr 85]. Für das Levinsche Maß, sei auf [MaSh 92] verwiesen.

Leider sind von einem algorithmischen Standpunkt aus diese Ergebnisse relativ unbefriedigend: Es wurden keine neuen algorithmischen Methoden speziell für den *average-case* erarbeitet, wie etwa für parallele oder für probabilistische Algorithmen. Vielmehr wurde ein sehr einfacher Algorithmus mittels zuweilen hohem technischen Aufwand analysiert, ob er im *average* effizient ist (siehe auch Beispiel auf Seite 132).

Mittlerweile liegen eine ganze Reihe solcher Untersuchungen auch für andere vermutlich schwierige Probleme wie zum Beispiel Graph-Isomorphie [BuKB 92], Graph-Färbung [BeWi 85], Hamiltonsche Kreise [AnVa 77, BFF 85, Frie 88] und Clique [DTH 86] vor. Für weitere sei auf [John 84b] und [DyFr 89] verwiesen.

Das bemerkenswerte an diesen Arbeiten ist das hohe Niveau der *average-case*-Analyse und nicht etwa die Konstruktion der Algorithmen. Abgesehen davon wissen wir jetzt, daß im allgemeinen der Beweis, daß ein gewichtetes $\mathcal{NP}$-Problem im *average* effizient ist, keine komplexitätstheoretischen Implikationen hat, solange das Problem nicht Dis $\mathcal{NP}$ -vollständig ist.

Die Bedeutung der Verteilung ist also hier mindestens genau so groß wie die der Sprache. So gelingt es für den Davis-Putnam-Algorithmus, für bestimmte Verteilungen untere Schranken zu zeigen [BPP 89]. In [MSL 92] wird vermutet, daß für gewisse gleichgewichtete Verteilungen das Verhältnis der Anzahl der Klauseln zu der Anzahl der Variablen für eine Konstante ungefähr bei 4,3 die maximale *average*-Komplexität impliziert. Ausschlaggebend soll dabei die Wahrscheinlichkeit für die Erfüllbarkeit einer Formel sein. Ist diese 50%, zeigen empirische Untersuchungen, daß bisherige Algorithmen besonders lange benötigen.

Für das Suchproblem in erfüllbaren Formeln schlagen die Autoren daher in [SLM 92] ein *simulated-annealing*-Verfahren vor. Bisher gelang es jedoch nicht, für diese das prognostizierte gute *average*-Verhalten zu beweisen.

Andererseits haben wir nicht zuletzt in dieser Arbeit auf Seite 136 eine einfache berechenbare Verteilung kennengelernt, die `SAT` genauso schwer macht wie jedes andere Problem in Dis $\mathcal{NP}$ (mittels einer Technik, die in [ReSc 93a] schon vorgestellt wurde). Diese Technik läßt sich auf alle bekannten $\mathcal{NP}$-vollständigen Sprachen L übertragen. Voraussetzung ist die Existenz einer effizient invertierbaren Reduktion dieser Sprache L auf `NBH`.

Diese so konstruierten Verteilungen kommen aber im allgemeinen in der Praxis nicht vor. Aus diesem Grunde wurde bisher versucht, gewichtete Probleme in Dis $\mathcal{NP}$, als vollständig zu charakterisieren, deren Verteilung möglichst einfach ist (bestenfalls uniform).

Dies gelang bisher für das 2-dimensionale Domino-Problem [Levin 86], das Postsche Korrespondenz-Problem [WaBe 92a], Graph-Färbung [VeLe 88], die Lösung Diophantischer Gleichungen [VeRa 92], Matrix Dekomposition [Gure 91a] und einige andere [Gure 90, Gure 91b, BlGu 91, WaBe 92a].

Abweichend von der vorliegenden Arbeit benutzt man hier probabilistische Maschinenmodelle, was die Menge vollständiger Probleme vergrößert. Es ist ein offenes Problem, ob die bezüglich probabilistischer Reduktionen und probabilistischer *average*-Klassen vollständigen Probleme auch vollständig sind bezüglich der hier verwendeten deterministischen Komplexitätsklassen[1].

Betrachtet man Verteilungen in POL-sampleable, ergeben sich zusätzliche Möglichkeiten. Hier gibt es eine universelle Verteilung μ_0 [BCGL 89, ImLe 90, Gure 90], die unter der Annahme der Existenz spezieller *one-way*-Funktionen [Levin 85], jede andere Verteilung aus POL-sampleable dominiert. Das impliziert, daß für jede beliebige $\mathcal{NP}$-vollständige Sprache L aus $(L, \mu_0) \in \text{AvDis}\,\mathcal{P}$ folgt

$$\mathcal{NP} \times \text{POL-sampleable} \subseteq \text{AvDis}\,\mathcal{P} \ .$$

Genauso kollabiert auch

$$\text{Dis}\,\mathcal{NP} \subseteq \text{AvDis}\,\mathcal{P} \ ,$$

wenn eines der Dis $\mathcal{NP}$-vollständigen Probleme wie etwa $(\mathtt{NBH}, \rho_{\mathtt{NBH}})$ in AvDis $\mathcal{P}$ enthalten ist. Eine neue Erkenntnis dieser Arbeit ist: Wenn ein Dis $\mathcal{NP}$-vollständiges Problem P in EavDis $\mathcal{P}$ enthalten ist, folgt nicht nur

$$\text{Dis}\,\mathcal{NP} \subseteq \text{AvDis}\,\mathcal{P} \ ,$$

sondern auch

$$\text{Dis}\,\mathcal{NP} \subseteq \text{EavDis}\,\mathcal{P} \ .$$

Der Erwartungswert wurde bisher nicht in dieses Konzept eingebunden. Er ist anscheinend nur für polynomiell zeitbeschränkte Simulationen ungeeignet und doch überraschenderweise geeignet für Reduktionen.

Wie erkennt man in der Praxis, ob ein Algorithmus ein $\mathcal{NP}$-Problem schon effizient löst? Buro und Kleine Büning waren die Veranstalter des `SAT`-*competition* 1992 [BuKB 92]. Teilnehmer konnten Algorithmen einschicken, die möglichst effizient `SAT` lösen sollten. Der Wettbewerb sah folgendermaßen aus: Gemäß einer Wahrscheinlichkeitsverteilung wurde eine Reihe von Eingaben berechnet, die dann alle eingesandten Programme zu lösen hatten. Diese Testanordnung durch Vorgabe von polynomiell vielen Eingaben ähnlicher Länge entspricht unseren Überlegungen beim Median-Maß.

Da der Nachweis der Korrektheit nicht verlangt wurde, hätte mit hoher Wahrscheinlichkeit ein Med^*-effizienter Algorithmus gut abschneiden können.

[1] Zur Klarstellung: *Average* ist nicht Probabilismus. Eine *average*-Betrachtung mittelt über eine Menge von Eingaben einer Maschinen. Probabilistische Turing-Maschinen sind Turing-Maschinen, die zur Berechnung Zufallsexperimente benutzen können. Somit können beispielsweise probabilistische Turing-Maschinen im *worst-case*, genauso wie deterministische Maschinen im *average* betrachtet werden.

Hätte man die Wahrscheinlichkeitsverteilung ρ_{Sat} (siehe Seite 136) zugrundegelegt und wäre ein Algorithmus effizient und korrekt auf den Test-Eingaben gewesen, so hätte man einen empirischen Nachweis für

$$\mathrm{Dis}\,\mathcal{NP} \quad \subseteq \; \mathrm{Med}^*\,\mathrm{Dis}\,\mathcal{P}$$

führen könen. Durch die hier vorgestellten Reduktionen bekäme man dann auch für alle anderen Probleme in Dis $\mathcal{NP}$ Med*-effiziente Algorithmen.

Wir schlagen deswegen folgenden empirischen Test für die Effizienz eines Programms vor, von dem behauptet wird, es löse ein $\mathcal{NP}$-vollständiges Problem effizient.

A behauptet einen effizienten Algorithmus M für die $\mathcal{NP}$-vollständige Sprache L zu haben.

B bestimmt eine Verteilung ρ_L (mit derselben Technik wie schon für `SAT`), so daß $(L, \rho_L) \in \mathrm{Dis}\mathcal{NP}\mathrm{C}_{me}$.

Dann erzeugt A für ein genügend großes l $m := \mathrm{PLog}(l)$ Eingaben $X = \{x_1, \ldots, x_m\}$ gleichverteilt aus der Menge $X^{\leq l}_{\rho_L}$ und sendet diese an A.

A berechnet $M(x_1), \ldots, M(x_m)$ und liefert die Resultate.

Wenn A nicht in der Lage ist, das Entscheidungsproblem von L effizient zu lösen, und falls Dis $\mathcal{NP} \not\subseteq \mathrm{Med}^*\,\mathrm{Dis}\,\mathcal{P}$, wird A mit hoher Wahrscheinlichkeit im letzten Schritt fehlerhafte Ergebnisse liefern oder sehr lang rechnen.

Damit könnte man in Zukunft viel Zeit und Mühe bei der Überprüfung von angeblich effizienten Algorithmen für $\mathcal{NP}$-vollständige Probleme sparen, indem man für einen Test Eingaben auf diese Art generiert.

8.5 Jenseits von AvTime

In [Grap 90] wurde eine *average-logspace*-Reduktion beschrieben, um gewichtete $\mathcal{P}$- und NL-vollständige Probleme nachzuweisen. Außer diesen gibt es bis jetzt kaum Untersuchungen zum Platzverbrauch von Algorithmen im Durchschnitt.

Der Grund ist klar. Wichtig für einen Algorithmus ist in der Regel nicht der durchschnittliche Platzverbrauch, sondern der maximale Platzbedarf. Vielleicht hilft hier das Median-Maß, das dann die Wahrscheinlichkeit für eine bestimmte Platz-Überschreitung klassifizieren könnte.

Eine anderer interessanter Aspekt des Median-Maßes, der in dieser Arbeit bereits angesprochen wurde, ist die Fähigkeit, mittels der Med^*-Klassen nicht-berechenbare Probleme als effizient approximierbar zu klassifizieren.

Wirft man einen Blick in die strukturelle Komplexitätstheorie, sieht man, daß dies kein Einzelfall ist. In [KOSW 91] wurde der Begriff *instance complexity* vorgestellt. Hier ist es einer Turing-Maschine nicht erlaubt Fehler zu machen, um konsistent zu einer Sprache zu sein. Zumindestens darf sie ein Sonderzeichen ausgeben, wenn sie nicht Ja oder Nein ausgibt. Solche Maschinen sind auch mit nicht-berechenbaren Problemen konsistent.

Reduktionsmengen auf *sparse* oder *tally* Mengen beinhalten auch immer nicht-berechenbare Sprachen. Wir haben in Abschnitt 5.3 gesehen, daß ein Zusammenhang mit solchen Mengen besteht. Es ist zu erwarten, daß weitere Ergebnisse gefunden werden können, die Bindeglieder zwischen diesen Gebieten darstellen. Als Übersichtsartikel zum Thema Reduktionsmengen sei auf [HOW 92, AHHK 92] verwiesen.

Nicht zu vergessen, beinhaltet auch $\mathcal{P}/\mathrm{Poly}$, die Menge aller Booleschen Funktionen mit polynomiell großen Schaltkreisen, nicht-berechenbare Probleme.

Wir untersuchen hier zwei prominente nicht-berechenbare Probleme, das Halteproblem und das Kolmogoroff-Komplexitätsproblem, auf Med^*-Effizienz. Es gelingt uns für die Anzahl fehlerhafter Berechnungen sehr starke untere Schranken zu zeigen. So bleibt die Frage nach der Existenz praktisch relevanter nicht-berechenbare Probleme in $\mathrm{Med}^*\,\mathrm{Dis}\,\mathcal{P}$ offen.

Kapitel 9

Zusammenfassung

In seiner Diplomarbeit [Schi 91] stellte der Autor dieser Arbeit ein präzises Maß **Av**, das den besonderen Anforderungen in der Komplexitätstheorie genügt. In der vorliegenden Arbeit gesellen sich diesem Maß das **Eav**-Maß und das **Med**-Maß hinzu.

Die Definition des **Av-Maßes** in dieser Arbeit orientiert sich an Levins Maß *polynomial on the average* (in dieser Arbeit notiert als **Lav(POL)**). Das **Eav-Maß** hingegen leitet sich vom Erwartungswert ab. Dies führt dazu, daß die Komplexitätsklasse der Zeit **EavDisTime(T)** definiert durch Eav, sich von klassischen *average*-Klassen definiert durch den Erwartungswert nicht unterscheidet, solange man Wahrscheinlichkeitsverteilungen betrachtet, die uniform auf *fülligen* Mengen gewichten.

Das **Med-Maß** ergibt sich aus einem prinzipiell anderen Ansatz. Die Menge $\mathrm{Med}\,(T, a)$ beschreibt für eine Schranke T und eine Häufigkeitsfunktion a, die Menge aller Paare $f : \Sigma^* \to \mathbb{N}$ und Wahrscheinlichkeitsverteilung μ, so daß der Fall $f(x) > T(|x|)$ höchstens die Häufigkeit a bezüglich μ besitzt.

Prinzipiell kann im Med-Maß die Funktion f die Schranke T beliebig stark überschreiten, solange es nicht zu häufig passiert, ganz im Gegensatz zu Av und Eav, denn hier darf eine Laufzeit der Größenordnung $\omega(T \circ \exp)$ bzw. $\omega(T \cdot \exp)$ kein einziges Mal vorkommen.

Diese drei Durchschnittsmaße sind den besonderen Erfordernissen der Klassifikation der Zeit time_M eines Algorithmus besser angepaßt als das Levinsche Maß und der Erwartungswert bezüglich lokal gewichteter Verteilungen. Gegenüber elementarer Operatoren verhalten sich Eav, Av und Med wie in Tabelle 9.1 dargestellt.

Beim Vergleich der drei Durchschnittsmaße ergab sich folgendes. Aus den Definitionen von Eav und Av folgt direkt $\mathrm{Eav}\,(\mathcal{N}) = \mathrm{Av}\,(\mathcal{N})$. Für Schranken $T \leq \mathcal{N}$ gilt

$(f,\mu) \in \mathrm{Eav}(T_f)$ $(g,\mu) \in \mathrm{Eav}(T_g)$	$(f,\mu) \in \mathrm{Av}(T_f)$ $(g,\mu) \in \mathrm{Av}(T_g)$	$(f,\mu) \in \mathrm{Med}(T_f, a_f)$ $(g,\mu) \in \mathrm{Med}(T_g, a_g)$
$(c \cdot f, \mu)$ $\in \mathrm{Eav}(c \cdot T_f + 1)$	$(c \cdot f, \mu)$ $\in \mathrm{Av}(c \cdot T_f)$	$(c \cdot f, \mu)$ $\in \mathrm{Med}(c \cdot T_f, a_f)$
$(f+g, \mu)$ $\in \mathrm{Eav}(2T_f + 2T_g)$	$(f+g, \mu)$ $\in \mathrm{Av}(\max[T_f(2\mathcal{N}), T_g(2\mathcal{N})])$	$(f+g, \mu)$ $\in \mathrm{Med}(T_f + T_g, a_f + a_g)$
—	$(f \cdot g, \mu)$ $\in \mathrm{Av}(\max[T_f(2\mathcal{N})^2, T_g(2\mathcal{N})^2])$	$(f \cdot g, \mu)$ $\in \mathrm{Med}(T_f \cdot T_g, a_f + a_g)$
—	$(f^c, \mu) \in \mathrm{Av}\left(T_f^c\right)$	$(f^c, \mu) \in \mathrm{Med}\left(T_f^c, a_f\right)$

Tabelle 9.1: Das Verhalten der Durchschnittsmaße Eav, Av und Med gegenüber elementarer Operationen.

$\mathrm{Av}\,(T) \subseteq \mathrm{Eav}\,(2 \cdot T)$. Der für Zeitkomplexitätsklassen weitaus interessantere Fall $T \geq \mathcal{N}$ impliziert

$$\mathrm{Eav}\,(T) \subseteq \mathrm{Av}\,(T(2 \cdot \mathcal{N}))\ .$$

Vergleicht man die *Average*-Durchschnittsmaße mit dem Median-Maß, so folgt aus der Beschränkung $(f,\mu) \in \mathrm{Eav}\,(T_1)$ für beliebige T_2 die Median-Beschränkung $(f,\mu) \in \mathrm{Med}\left(T_2, a_\mu^{(T_2+1)/T_1}\right)$ (für a werden hier drei alternative Funktionen vorgestellt). Analog folgt aus $(f,\mu) \in \mathrm{Av}\,(T_1)$ für beliebige T_2 die Median-Beschränkung $(f,\mu) \in \mathrm{Med}\left(T_2, a_\mu^{T_1^{[-1]}(T_2+1)}\right)$. Andererseits kann man aus einer Menge von Median-Beschränkungen und einer *worst-case*-Beschränkung Av- und Eav-Schranken berechnen.

Average-Komplexitätsklassen bestehen aus **gewichteten Problemen**. Das sind Paare bestehend aus einem Problem und einer Wahrscheinlichkeitsverteilung. Für die hier vorgestellten Maße Eav und Av bilden diese die Komplexitätsklassen **AvDisTime** und **EavDisTime**.

Median-Komplexitätsklassen kann man mittels der Zeitfunktion time_M der Turing-Maschine M , die die Sprache L akzeptiert, oder mittels der Zeitfunktion $\mathrm{time}_{M,L}$ definieren. Letztere ist definiert als ∞ , falls M auf der Eingabe nicht hält oder bezüglich L fehlerhaft rechnet und entspricht ansonsten time_M . Die daraus resultierenden fehlerfreien Median-Komplexitätsklassen heißen **MedDisTime**, während die fehlerbehafteten Median-Komplexitätsklassen **Med*DisTime** genannt werden.

Diese vier Klassen definieren für polynomielle Zeitschranken $\mathbf{AvDis}\boldsymbol{\mathcal{P}}$, $\mathbf{EavDis}\boldsymbol{\mathcal{P}}$, $\mathbf{MedDis}\boldsymbol{\mathcal{P}}$ und $\mathbf{Med^*Dis}\boldsymbol{\mathcal{P}}$. Für diese Klassen werden die Inklusionen $\mathrm{EavDis}\,\mathcal{P} \subseteq \mathrm{AvDis}\,\mathcal{P}$ und $\mathrm{MedDis}\,\mathcal{P} \subseteq \mathrm{Med}^*\,\mathrm{Dis}\,\mathcal{P}$ gezeigt. Zwischen polynomiellen *Average*-Klassen $\mathrm{EavDis}\,\mathcal{P}$ oder $\mathrm{AvDis}\,\mathcal{P}$ und Median-Klassen $\mathrm{MedDis}\,\mathcal{P}$ und $\mathrm{Med}^*\,\mathrm{Dis}\,\mathcal{P}$ hingegen gibt es beweisbar keinerlei Teilmengenrelation.

Betrachtet man nur polynomiell beschreibbare Wahrscheinlichkeitsfunktionen (**POL-describable**), die für große Eingaben moderat (**modest**) abnehmen, so entspricht AvDis$\mathcal{P}$ der von Levin definierten Komplexitätsklasse **LavDis$\mathcal{P}$**, basierend auf der Zeitbeschränkung *polynomial on the average*.

Analog der Definition von AvTime in der Diplomarbeit des Autors, definieren wir hier EavTime, MedTime und Med*Time. Diese Klassen beinhalten nur Sprachen, was einen Vergleich mit den üblichen *worst-case*-Komplexitätsklassen ermöglicht. Insbesondere zeigen wir für uniforme Wahrscheinlichkeitsverteilungen μ_{uni}

$$\begin{aligned}
\text{EavTime}(T, \{\mu_{\text{uni}}\}) &\subseteq \text{DTime}(T \cdot \text{EXL}) \ , \\
\text{AvTime}(T, \{\mu_{\text{uni}}\}) &\subseteq \text{DTime}(T \circ \text{EXL}) \ , \\
\text{EavTime}(T, \{\mu_{\text{uni}}\}) &\setminus \text{DTime}(o(T \cdot \text{EXL})) \neq \emptyset \ , \\
\text{AvTime}(T, \{\mu_{\text{uni}}\}) &\setminus \text{DTime}(o(T \circ \exp \tfrac{\mathcal{N}}{2})) \neq \emptyset \ .
\end{aligned}$$

Als Maß für die Komplexität einer Wahrscheinlichkeitsverteilung benutzen wir **rankable** [Schi 91]. Damit eine Wahrscheinlichkeitsverteilung μ in V–rankable enthalten ist, muß eine deterministische Turing-Maschine den Rang der Eingabe x bezüglich der Verteilung μ in Zeit $V(|x|)$ berechnen.

Für Mengen von sehr komplexen Wahrscheinlichkeitsverteilungen konnte in [Schi 91] bewiesen werden

$$\text{AvTime}(T, (T \cdot \text{EXL})\text{–rankable}) = \text{DTime}(T) \ ,$$

für polynomielle Schranken T. In dieser Arbeit kann dieses Ergebnis für allgemeine Schranken T auf EavTime übertragen werden:

$$\text{EavTime}(T, (T \cdot \text{EXL})\text{–rankable}) = \text{DTime}(T) \ .$$

Für AvTime ist eine solche Aussage für alle T nicht möglich, denn für $T \geq \text{EEXP}$ gilt

$$\text{DTime}(T) \subset \text{AvTime}(T, U) \ ,$$

für die Menge U aller Wahrscheinlichkeitsverteilungen.

In [Schi 91] wurde bewiesen, daß

$$\text{AvTime}(o(T), V\text{–rankable}) \subset \text{AvTime}(T, V\text{–rankable}) \ .$$

Wir geben einen ausführlicheren Beweis und übertragen dieses Ergebnis auf EavTime:

$$\text{EavTime}(o(T), V\text{–rankable}) \subset \text{EavTime}(T, V\text{–rankable}) \ .$$

Schließlich gelingt es in dieser Arbeit, ein Hierarchie-Ergebnis bezüglich der Komplexität der Verteilungen zu erreichen. Insbesondere zeigen wir, daß gilt

$$\begin{aligned} \mathrm{EavTime}(T, V\text{–rankable}) &\subset \mathrm{EavTime}(T, o(V)\text{–rankable}) \ , \\ \mathrm{AvTime}(T', V\text{–rankable}) &\subset \mathrm{AvTime}(T', o(V)\text{–rankable}) \ , \end{aligned}$$

für $V \leq T$ und $V(\delta\mathcal{N}) \leq T'$ ($\delta > 1$).

Die Median-Klasse Med* DisTime beinhaltet auch nichtberechenbare Probleme. In diesem Zusammenhang untersuchen wir die Median-Zeit-Komplexität des Halteproblems `HP` und des Kolmogoroff-Komplexitätsproblems `NBH`. Hier beweisen wir enge untere Schranken, denn es gilt für alle berechenbaren Zeitschranken T

$$\begin{aligned} (\mathtt{HP}, \rho_{\mathrm{diag}}) &\notin \mathrm{Med}^*\mathrm{DisTime}(T, o(\mathcal{N})) \ , \\ (\mathtt{NBH}, \mu_{\mathrm{uni}}) &\notin \mathrm{Med}^*\mathrm{DisTime}(T, o(\mathcal{N})) \ . \end{aligned}$$

In dieser Arbeit wird der Spezialfall deterministischer Reduktionen zwischen gewichteten Problemen untersucht, der bisher kaum Beachtung fand. Es gelingt geeignete Einschränkungen für polynomiell zeitbeschränkte Reduktionen zu formulieren, so daß Aussagen für alle vier polynomieller *Average*- und Median-Klassen möglich werden.

Als Äquivalent zu $\mathcal{NP}$ betrachten wir **Dis**$\boldsymbol{\mathcal{NP}}$ die Menge $\mathcal{NP} \times$ POL–rankable. Es gelang uns zu zeigen, daß das beschränkte Halteproblem nichtdeterministischer Turingmaschinen `NBH` mit geeigneter Wahrscheinlichkeitsverteilung $\rho_{\mathtt{NBH}}$ vollständig ist für Dis $\mathcal{NP}$ bezüglich aller hier vorgestellten deterministischen Reduktionsarten.

Literaturverzeichnis

[AHHK 92] V. Arvind, Y. Han, L. Hemachandra, J. Köbler, A. Lozaro, M. Mundhenk, M. Ogiwaram, U. Schöning, R. Silvestri, T. Thierauf, *Reductions to Sets of Low Information Content*, Technischer Bericht, University of Rechester, 1992.

[AnVa 77] Dana Angluin, Leslie G. Valiant, *Fast Probabilistic Algorithms for Hamiltonian Circuits and Matchings*, Proc. 17. SToC, 1977, 30-41.

[BaKu 79] Lászlò Babai, Ludik Kučera, *Canonical Labelling of Graphs in Linear Average Time*, Proc. 20. FoCS, 1979, 39-46.

[BCGL 89] S. Ben-David, B. Chor, O. Goldreich, M. Luby, *On the Theory of Average Case Complexity*, Journal CSS, 1992, 44; see also Proc. 29. SToC, 1989, 204-261.

[BeWi 85] Edward A. Bender, Herbert S. Wilf, *A Theoretical Analysis of Backtracking in the Graph Coloring Problem*, Journal of Algorithms 6, 1985, 275-282.

[BFF 85] B. Bollobás, T.I. Fenner, A.M. Frieze, *An Algorithm for finding Hamilton Cycles in a Random Graph*, Proc. 25. SToC, 1985, 430-439.

[BlGu 91] Andreas Blass, Yuri Gurevich, *Randomizing Reductions of Search Problems*, 11. Conference on Foundations of Software Technology and Theoretical Computer Science, New Delhi, India, Springer Lecture Notes in Computer Science 560, 10-24.

[BPP 89] Khaled M. Bugrara, Youfang Pan, Paul Walton Purdom, *Exponential Average Time for the Pure Literal Rule*, SIAM Journal Computing, 1989, 409-418.

[BuKB 92] Michael Buro, Hans Kleine Büning, *Report on a Sat Competition*, Technischer Bericht, Gesamthochschule Paderborn, 1992.

[DaPu 60] Martin Davis, Hillary Putnam, *A Computing Procedure for Quantification Theory*, Journal of the ACM, 1960, 201-215.

[DTH 86] Phan Dinh Dieu, Le Cong Thanh, Le Tuan Hoa, *Average Polynomial Time Complexity of Some $\mathcal{NP}$-Complete Problems*, Theoretical Computer Science 46, 1986, 219-327.

[DyFr 89] M.E. Dyer, A.M. Frieze, *The Solution of Some Random $\mathcal{NP}$-Hard Problems in Polynomial Expected Time*, Journal of Algorithms, 1989, 451-489.

[Frie 88] A.M. Fireze, *An Algorithm for Finding Hamilton Cycles in Random Directed Graphs*, Journal of Algorithms, 1988, 181-204.

[GGH 93] Mikael Goldmann, Per Grape, Johan Håstad, *On Average Time Hierarchies*, 1993, Technischer Bericht.

[Grap 90] Per Grape, *Complete Problems with L-Sampleable Distributions*, Proc. 2. SWAT, 1990, 360-367.

[Gure 87] Yuri Gurevich, *Complete and Incomplete Randomized $\mathcal{NP}$ Problems*, Proc. 28. FoCS, 1987, 111-117.

[Gure 90] Yuri Gurevich, *Matrix Decomposition Problem Is Complete for the Average Case*, Proc. 31. FoCS, 1990, 802-811.

[Gure 91a] Yuri Gurevich, *Average Case Completeness*, Journal CSS 42, 1991, 346-398.

[Gure 91b] Yuri Gurevich, *Average Case Complexity*, Proc. 18. ICALP, 1991, 613-628

[HOW 92] L. Hemachandra, M. Ogiwara, O. Watanabe, *How hard are Sparse Sets?*, 7. Struc., 1992, 222-238.

[ImLe 90] R. Impagliazzo, L. Levin, *No Better Ways to Generate Hard $\mathcal{NP}$ Instances than Picking Uniformly at Random*, Proc. 31. FoCS, 1990, 812-821.

[John 84a] D. Johnson, *The $\mathcal{NP}$ –Completeness Column*, Journal of Algorithms 5, 1984, 284-299.

[John 84b] D. Johnson, *The $\mathcal{NP}$ –Completeness Column*, Journal of Algorithms 5, 1984, 433-447.

[JRS 94] Andreas Jakoby, Rüdiger Reischuk, Christian Schindelhauer, *Circuit Complexity: From the Worst Case to the Average Case*, Proc. 26. SToC, 1994, 58-67.

[JRSW 94] Andreas Jakoby, Rüdiger Reischuk, Christian Schindelhauer, Stephan Weis, *The Average Case Complexity of the Parallel Prefix Problem* 21. ICALP, 1994, 593-604.

[Karp 76] Richard M. Karp, *The Probabilistic Analysis of Some Combinatorial Search Algorithms*, Symposium on New Directions and Recent Results in Algorithms and Complexity, 1976, 1-20.

[Kemp 84] Rainer Kemp, *Fundamentals of the Average Case Analysis of Particular Algorithms*, Monographie, Teubner, 1984.

[Koba 93] Kojiro Kobayashi, *On Malign Input Distributions for Algorithms*, IEICE Trans. Inf. & Syst. Vol. E76-D, No. 6, 1993, 634-640

[KOSW 91] K. Ko, P. Orponen, U. Schöning, O. Watanabe, *Instance Complexity*, Ulmer Informatik-Berichte, 1991.

[Levin 86] Leonid Levin, *Average Case Complete Problems*, SIAM Journal on Computing 15, 1986, 285-286.

[Levin 85] Leonid Levin, *One-Way Functions and Pseudo-Random Generator*, Proc. 25. SToC, 1985, 363-375.

[LiVi 92] Ming Li, Paul M.B. Viányi, *Average Case Complexity under the Universal Distribution Equals Worst-Case Complexity*, Information Processing Letters 42, 1992, 145-149.

[LiVi 93] Ming Li, Paul M.B. Viányi, *An Introduction to Kolmogorov Complexity and its Applications*, Monographie, Springer, 1993.

[MaSh 92] J. Makowsky, A. Sharell, *On Average Case Complexity of SAT for Symmetric Distributions*, Technical Report, 1992.

[Milt 91] Peter Bro Miltersen, *The Complexity of Malign Ensembles,* Proc. 6. StruC, 1991, 164-171, siehe auch SIAM Journal on Computing 22, 1993, 147-156.

[MSL 92] David Mitchell, Bart Selman, Hector Levesque, *Hard and Easy Distributions of SAT Problems*, Proc. 10. Nat. Conf. on Artificial Intelligence, 1992, 459-465.

[PuBr 85] Paul Walton Purdom, Cynthia A. Brown, *The Pure Literal Rule and Polynomial Average Time*, SIAM J. Comput., 1985, 943-953.

[Rabin 76] Michael O. Rabin, *Probabilistic Algorithms*, Symposium on New Directions and Recent Results in Algorithms and Complexity, 1976, 21-40.

[Reis 90] Rüdiger Reischuk, *Einführung in die Komplexitätstheorie,* Monographie, Teubner, 1990.

[ReSc 93a] Rüdiger Reischuk, Christian Schindelhauer, *Precise Average Case Complexity,* Proc. 10. STACS, 1993, 650-661.

[ReSc 93b] Rüdiger Reischuk, Christian Schindelhauer, *Precise Average Case Complexity*, Technical Report, Technische Hochschule Darmstadt, 1993.

[SLM 92] Bart Selman, Hector Levesque, David Mitchell, *Hard and Easy Distributions of SAT Problems*, Proc. 10. Nat. Conf. on Artificial Intelligence, 1992, 440-446.

[Schi 91] Christian Schindelhauer, *Neue Average Case Komplexitätsklassen,* Diplomarbeit, Technische Hochschule Darmstadt, 1991.

[VeLe 88] Ramarathnam Venkatesan, Leonid Levin, *Random Instances of Graph Coloring Problems are Hard*, Proc. 20. SToC, 1988, 217-222.

[VeRa 92] Ramarathnam Venkatesan, Sivaramakrishnan Rajagopalan, *Average Case Intractability of Matrix and Diophantine Problems*, Proc. 22. SToC, 1992, 632-642.

[ViFl 90] Jeffrey Scott Vitter, Philipp Flajolet, *Average-Case-Analysis of Algorithms and Data Structures*, Handbook of Computer Science, Elsevier, The MIT Press, 1990, 431-524.

[WaBe 92a] Jie Wang, Jay Belanger, *Average Case Intractability of Some Classical Problems*, Technical Report, 1992.

[WaBe 92b] Jie Wang, Jay Belanger, *On Average P versus Average NP*, Proc. of 7. IEEE Conference on Structure in Complexity Theory, 1992, 318-326.

Index

Lebenslauf

Name	Christian Schindelhauer	
Geburtstag	17.10.1967	
Geburtsort	Miltenberg	
Familienstand	verheiratet seit 6.6.1995	
Anschrift	Am Schellbruch 17 23568 Lübeck	
Ehefrau	Barbara Goedecke, Key-Account Manager Vertrieb, Diplom-Kauffrau, Master of Science	
Eltern	Wilfriede Schindelhauer, geb. Krippner, Immobilienmakler Wilfried Schindelhauer, Geschäftsführer, Diplom-Chemiker	
Geschwister	Antje Schindelhauer, Studentin des Wirtschaftsingenieurwesens Joachim Schindelhauer, Diplom-Bauingenieur	
Werdegang	1973-1977	Wolfram-von-Eschenbach-Grundschule
	1977-1986	Johannes-Butzbach-Gymnasium (math.-naturw. Zweig)
	1986-1991	Studium der Informatik an der Technischen Hochschule Darmstadt
	1991-1994	Wissenschaftlicher Mitarbeiter am Institut für Theoretische Informatik in der Arbeitsgruppe von Prof. Dr. R. Reischuk an der Technischen Hochschule Darmstadt
	1994-	Wissenschaftlicher Mitarbeiter am Institut für Theoretische Informatik an der Medizinischen Universität zu Lübeck
Abschlüsse	1986	Abitur
	1991	Diplom in Informatik

Publikationen

Chr. Schindelhauer, *Neue Average-Case-Komplexitätsklassen*, Diplomarbeit, Technische Hochschule Darmstadt, 1991

R. Reischuk, Chr. Schindelhauer, *Precise Average Case Complexity*, Proceedings of the 10. Symposium on Theoretical Aspects of Computer Science (STACS), 1993, 650-661.

R. Reischuk, Chr. Schindelhauer, *Precise Average Case Complexity*, Technischer Bericht, Technische Hochschule Darmstadt, 1993.

A. Jakoby, R. Reischuk, Chr. Schindelhauer, *The Complexity of Broadcasting in Planar and Decomposable Graphs*, 20. International Workshop on Graph-Theoretic Concepts in Computer Science (WG' 94), 1994, 219-231.

A. Jakoby, R. Reischuk, Chr. Schindelhauer, *Circuit Complexity: From the Worst Case to the Average Case*, Proceedings of the 26. Symposium on the Theory of Computer Science (SToC), 1994, 58-67.

A. Jakoby, R. Reischuk, Chr. Schindelhauer, St. Weis, *The Average Case Complexity of the Parallel Prefix Problem*, 21. International Conference of Algorithms, Languages and Programming (ICALP), 1994, 593-604.

A. Jakoby, R. Reischuk, Chr. Schindelhauer, *Malign Distributions for Circuit Complexity*, Proceedings of the 12th Symposium on Theoretical Aspects in Computer Science (STACS), 1995, 629 - 639.