Seminar Mathematik & Informatik
A Glimpse beyond (Exp und mehr)

von Torsten Langer

January 26, 2002

1 Exponentialzeit

1.1 Definition:

- Exp $:= \bigcup_{k \in N} Time\left(2^{n^k}\right)$,
- NExp $:= \bigcup_{k \in N} NTime\left(2^{n^k}\right)$.

1.2 Satz: Falls $P = NP$, so gilt auch $Exp = NExp$.

Beweis: Sei $L \in NExp$. Nach Definition gibt es also ein k und eine nichtdeterministische Turingmaschine N, die L in Zeit 2^{n^k} entscheidet. Betrachte nun die „wattierte" Fassung L' von L. Dazu nehme ich ein Quasiblanko-Zeichen $\tilde{\sqcup}$ neu in das Alphabet auf und definiere: $L' = \left\{ x \frown \tilde{\sqcup}^{2^{|x|^k} - |x|} | x \in L \right\}$.

Nun gilt: $L' \in NP$. Die zugehörige Maschine N' betrachtet Quasiblankos als Blankos und verhält sich ansonsten wie N, zählt allerdings zu Anfang, ob genügend Quasiblankos vorhanden sind, die Eingabe also die Form $x \frown \tilde{\sqcup}^{2^{|x|^k} - |x|}$ hat. Damit entscheidet N' offensichtlich L' in Laufzeit $O\left(2^{|x|^k}\right)$, da dies die Länge der Eingabe ist, ist N' polynomiell, sogar linear. Aus der Voraussetzung $P = NP$ ergibt sich die Existenz einer deterministischen Maschine M', die L' in polynomieller Laufzeit entscheidet, die Laufzeit sei n^l.

Nun definiere M, die wiederum L deterministisch in exponentieller Laufzeit entscheiden soll, wie folgt: Zunächst füllt M die Eingabe x zu $x \frown \tilde{\sqcup}^{2^{|x|^k} - |x|}$ auf, danach simuliert sie M'. Die Laufzeit ist in $O\left(\left(2^{|x|^k}\right)^l\right) = O\left(2^{|x|^{k+1}}\right)$. QED

1.3 Bemerkung: (Dekhtyar, 1976) Es gibt ein Orakel A, so dass gilt: $P^A = NP^A$ und $Exp^A \neq NExp^A$.

1.4 Bemerkung:

- $2SAT \in P$.
- $SAT \in NPC$.
- $QSAT_i$ ist $\Sigma_i P - vollständig$.
- $AGPL \notin R$, das heisst das Allgemeingültigkeitsproblem der Prädikatenlogik, welches wie folgt definiert wird, $\{(R_i)_{i \leq l}, (f_j)_{j \leq m}, (c_k)_{k \leq n}, \varphi \,|\, \varphi \, ist \, Formel \, in \, (R_i)_{i \leq l}, (f_j)_{j \leq m}, (c_k)_{k \leq n} \, und \vdash \varphi\}$ ist nicht entscheidbar.

1.5 Definition: Eine *Schönfinkel-Bernays-Formel* ist eine Aussage φ, in deren Alphabet nur Relationssymbole und Konstanten, aber keine Funktionssymbole, vorkommen, in der Form $\varphi \equiv \exists x_1 \dots \exists x_k \forall y_1 \dots \forall y_l \psi$, ψ quantorenfrei.
Schönfinkel-Bernays SAT ist die Frage nach der Erfüllbarkeit eines solchen Ausdrucks.

Bemerkungen:

1. Üblicherweise wird hier ausserdem noch verlangt, dass das Alphabet kein Gleichheitszeichen enthalten darf. Da ich allerdings keinen Grund für diese Voraussetzung finden konnte, diese aber den Beweis erschwert, verzichte ich hier auf diese Forderung.
2. Diese Definition ist natürlich rein syntaktischer Natur.

1.6 Definition:

- $E := \bigcup_{k \in N} Time(k^n)$,
- $NE := \bigcup_{k \in N} NTime(k^n)$.

1.7 Lemma: Für jede Sprache $L \in NExp$ gibt es eine Sprache $L' \in NE$, so dass L auf L' reduziert werden kann: $L \leq L'$.
Beweis: Sei $L \in NTime(2^{n^k})$. Dann definiere $L' := \left\{ x \frown \tilde{\sqcup}^{|x|^k} | x \in L \right\}$. $L' \in NTime(2^n) \subset NE$ und L wird durch Anhängen von Quasiblankos auf L' reduziert. QED

1.8 Korollar: L ist $NExp-vollständig$ genau dann, wenn $L \in NExp$ und für alle $L' \in NE$ gilt: $L' \leq L$.

1.9 Lemma: Eine Schönfinkel-Bernays-Formel $\varphi \equiv \exists x_1 \dots \exists x_k \forall y_1 \dots \forall y_l \psi$, die m Konstantensymbole enthält, ist genau dann erfüllbar, wenn sie ein Modell mit höchstens $k+m$ Elementen besitzt.
Beweis: Sei $\mathcal{A} = (A, \beta)$ ein Modell für φ mit Variablenbelegung β. Dann gibt es (aus der Definition der Modellbeziehung) $u_1, \dots u_k \in A$, so dass $\left(A, \beta \frac{u_1, \dots u_k}{x_1, \dots x_k} \right) \models \forall y_1 \dots \forall y_l \psi$. Seien weiterhin $v_1 \dots v_m$ die Belegungen der Konstantensymbole. Definiere dann $A' := \{u_1 \dots u_k, v_1 \dots v_m\} \subset A$. Nun ist auch $\left(A', \beta \frac{u_1, \dots u_k}{x_1, \dots x_k} \Big|_{A'} \right) \models \forall y_1 \dots \forall y_l \psi$ und damit auch $(A', \beta|_{A'}) \models \varphi$. QED

1.10 Satz: Schönfinkel-Bernays SAT ist $NExp-vollständig$; insbesondere ist also $Schönfinkel-Bernays\, SAT \notin P$.
Beweis: Sei $\varphi \equiv \exists x_1 \dots \exists x_k \forall y_1 \dots \forall y_l \psi$ ein Schönfinkel-Bernays-Ausdruck mit m Konstantensymbolen.
1) $Schönfinkel - Bernays\, SAT \in NExp$.
Sei n die Länge der Formel. Es gilt $k + l \leq n$, $k + m \leq n$. Ich benutze jetzt Lemma 1.9, um ein Modell mit höchstens $k + m$ Element zu „raten".

- Höchstens n Konstanten müssen mit den n Elementen belegt werden. Dies kann nichtdeterministisch für alle möglichen Belegungen „gleichzeitig" in $\log n$ Schritten geschehen, zum Beispiel, indem jedes Element durch Binärzahlen kodiert wird. Zeitaufwand: $O(n \log n)$.
- Bis zu n Relationen müssen belegt werden, indem jeweils für n^2 Paare entschieden wird, ob sie in Relation zueinander stehen oder nicht. Nichtdeterministisch können beide Möglichkeiten gleichzeitig ausprobiert werden. Zeitaufwand: $O(n^3)$.

Jetzt muss getestet werden, ob das erstellte Modell erfüllbar ist:

- n Variablen mit Existenzquantoren werden belegt. Dies kann genauso wie bei den Konstantensymbolen in Zeit $O(n \log n)$ geschehen.

- Für n Allquantoren müssen jeweils n Belegungen ausprobiert werden. Der Zeitaufwand ist $O\left(2^{n^2}\right)$, da $n^n = \left(2^{\log n}\right)^n = 2^{n \log n} \leq 2^{n^2}$.

Notwendige Bewegungen auf den Bändern, um Konstanten und Relationen abzufragen während ein Modell getestet wird, sind sicherlich polynomiell beschränkt. qed(1)
2) $Schönfinkel - Bernays\,SAT$ ist $NExp - vollständig$.
Nach Korollar 1.8 genügt es zu zeigen, dass jede Sprache $L \in NE$ auf $Schönfinkel - Bernays\,SAT$ reduziert werden kann.
Sei L solch eine Sprache und M eine nichtdeterministische Turingmaschine, die L in Zeit k^n entscheidet. Es gilt:

$$\begin{aligned} x \in L \quad &\Leftrightarrow \quad M\,haelt\,bei\,Eingabe\,x\,im\,Zustand\,ja \\ &\Leftrightarrow \quad in\,jedem\,Schritt\,und\,an\,jeder\,Stelle\,des\,Bandes\,verhalten\,wir\,uns\,wie\,M \Rightarrow Endzustand\,ja \end{aligned}$$

Ähnlich wie in Cook's Theorem soll die letzte Aussage jetzt als Schönfinkel-Bernays-Formel, die nur Allquantoren enthält formalisiert werden. Da M Laufzeit k^n hat, wird M durch ein $k^n \times k^n$ -Gitter vollständig beschrieben. In einem k-adischen System kann jeder Gitterpunkt durch zwei n-stellige Zahlen $x_1 \ldots x_n$ und $y_1 \ldots y_n$ kodiert werden. An jedem Gitterpunkt sollen wir uns „wie M verhalten". Die Formel bekommt somit die Form: $\forall x_1 \ldots \forall x_n \forall y_1 \ldots \forall y_n \psi$, ψ quantorenfrei.
Im folgenden werde ich ψ beschreiben.

- Um über bestimmte Stellen auf dem Gitter sprechen zu können, führe ich Konstanten $0, \ldots k-1$ ein und verlange, dass diese paarweise verschieden sind und jedes x_i und jedes y_j gleich einem dieser Konstantensymbole ist.
- Da sich M zeilenweise durch das Gitter vorarbeitet, ist es nützlich, eine Nachfolgerrelation $S_n\left(x_1, \ldots x_n, y_1, \ldots y_n\right)$ zu haben. Wir definieren

  $$S_1\left(x_1, y_1\right) \Leftrightarrow \left(\left(x_1 = 0 \wedge y_1 = 1\right) \vee \ldots \left(x_1 = k-2 \wedge y_1 = k-1\right)\right)$$

  für 1-stellige Zahlen. Rekursiv kann dann S_2 bis S_n definiert werden.
- Für jeden Zustand s aus der Zustandsmenge von M führen wir eine Relation $Z_s\left(x_1, \ldots x_n\right)$ mit der Bedeutung „M nimmt zum Zeitpunkt x Zustand s an" ein und verlangen, dass M an jedem Zeitpunkt genau einen Zustand annimmt, sowie dass M im Startzustand beginnt und im Zustand JA endet.
- Für jeden Buchstaben σ aus dem Alphabet von M führen wir Relationen $T_\sigma\left(x_1, \ldots x_n, y_1, \ldots y_n\right)$ und $\widetilde{T_\sigma}\left(x_1, \ldots x_n, y_1, \ldots y_n\right)$ ein. T_σ hat die Bedeutung „zum Zeitpunkt x steht an der Stelle y ein σ auf dem Band". $\widetilde{T_\sigma}$ verwenden wir stattdessen, „wenn sich der Schreib-/Lesekopf von M dort befindet". Wir verlangen wieder, dass M zu jedem Zeitpunkt und an jeder Stelle genau ein Symbol auf dem Band stehen hat und dass in der ersten Zeile die Eingabe gefolgt von Leerzeichen steht mit dem Schreib-/Lesekopf ganz links, sowie dass in der linken Spalte immer das Startsymbol $\triangleright$ steht.
- Schliesslich muss noch sichergestellt werden, dass wir uns entsprechend der Zustandsübergangsfunktion von M verhalten.
  Wir verlangen also für jeden Zeitpunkt und für jede Stelle des Bandes, dass sich der Inhalt des Bandes nur an den 3 Stellen um den Schreib-/Lesekopf herum verändern kann und sich dort gemäss einer der nichtdeterministisch möglichen Übergänge verhält; entsprechend muss sich auch der Zustand von M anpassen.

Wie man leicht sieht, ist die so entstehende Formel genau dann erfüllbar, wenn die Eingabe von M akzeptiert wird. QED

1.11 Bemerkung: Weiterhin ist für Sprachen ohne Funktionssymbole und ohne Gleichheitszeichen bekannt:

- Die Ackermann-Klasse: „$\exists^*\forall\exists^*\psi$” ist $Exp - vollständig$.
- Die Gödel-Klasse „$\exists^*\forall\forall\exists^*\psi$” ist $NExp - vollständig$.

$()^*$ steht dabei für eine beliebig lange, endliche Folge, in der Mengenlehre üblicherweise mit $()^{<\omega}$ bezeichnet.

1.12 Definition: Weitere Komplexitätsklassen:

- $ExpSpace := \bigcup_{k\in N} Space\left(2^{n^k}\right)$.
- $2 - Exp := \bigcup_{k\in N} Time\left(2^{2^{n^k}}\right)$.
- $l - Exp := \bigcup_{k\in N} Time\left(2^{\cdot^{\cdot^{2^{n^k}}}}\right)$, l-fache Exponentation.
- $Elementary := \bigcup_{l\in N} l - Exp$.

1.13 Übersicht: $P \subset NP \subset \ldots PH \subset Exp \subset NExp \subset 2 - Exp \subset \ldots Elementary \subset R$.

2 Ein einheitlicher Ansatz für Komplexitätsklassen

(vgl. [2])

Wir betrachten jetzt nicht-deterministische, polynomiell-beschränkte Turingmaschinen. Diese sollen ohne Einschränkung so normiert sein, dass sie

1. in jedem Schritt in genau zwei Möglichkeiten verzweigen (vgl. „Cook's Theorem” im Referat von Martina Eckner),
2. nach genau $p(|x|)$ Schritten hält, wobei p polynomiell beschränkt sei (vgl. Begriff der „präzisen Maschine”).

Damit produziert eine solche (festgehaltene) Maschine M bei (variabler) Eingabe x genau $2^{p(|x|)}$ mögliche Antworten JA oder NEIN als Blätter (, diese werden in der Mengenlehre „terminale Knoten” genannt,) des Berechnungsbaumes. Werden diese als 0 bzw. 1 interpretiert, können wir jedem x ein $M(x) \in (0,1)^{2^{p(|x|)}} \subset (0,1)^*$ zuordnen.

2.1 Beispiel: Sei $L \in NP$, M die zugehörige Turingmaschine wie oben beschrieben.
Dann ist

$$\begin{aligned} x \in L &\Leftrightarrow M\, akzeptiert\, x \\ &\Leftrightarrow ein\, Entscheidungspfad\, endet\, mit\, ja \\ &\Leftrightarrow M(x) \in (0,1)^* 1 (0,1)^*. \end{aligned}$$

2.2 Definition: Seien $A, B \subset (0,1)^*$. Dann definiert C(A,B) := {L Sprache | ∃ NTM M: $(x \in L \Leftrightarrow M(x) \in A) \wedge (x \notin L \Leftrightarrow M(x) \in B)$} eine Komplexitätsklasse.

2.3 Satz: Wir können alle bisher kennengelernten Klassen so darstellen. In der folgenden Tabelle habe ich einige Beispiele aufgelistet. Dabei habe ich folgende Notation verwendet:

1. „#” bezeichnet Berechnungsäste, die in einem „weiss nicht” Zustand enden. Dieser regulär nicht vorhandene Zustand kann simuliert werden, indem man die Zustände „1” durch „11”, „0” durch „00” und „#” durch „01” ersetzt.

2. „$\diamond$" ist das Verketten von Wörtern, wobei allerdings erkennbar bleibt, welche Buchstaben zu welchem Wort gehörten. Die genaue Definition für $x, y \in \Sigma^*$ist $x \diamond y := (x)_1(y)_0$, dabei ist $(x)_a := x_1 a x_2 a \ldots x_{|x|} a$ für beliebige $x \in \Sigma^*$, $a \in \Sigma$.

C(A,B)	A	B
P	1^*	0^*
NP	$\{0,1\}^*1\{0,1\}^*$	0^*
$coNP$	1^*	$\{0,1\}^*0\{0,1\}^*$
$NP \cap coNP$	$\{1,\#\}^*1\{1,\#\}^*$	$\{0,\#\}^*0\{0,\#\}^*$
$\Sigma_2 P$	$A_2 = \{z_1 \ldots z_n \,\vert \forall i : \vert z_i\vert = n \wedge \exists i : z_i = 1^n\}$	$B_2 = \{z_1 \ldots z_n \,\vert \forall i : \vert z_i\vert = n \wedge \forall i : z_i \neq 1^n\}$
$\Sigma_{k+1} P$	$A_{k+1} = \left\{z_1 \ldots z_n \,\vert \forall i : \vert z_i\vert = n^k \wedge \exists i : z_i \in B_k\right\}$	$B_{k+1} = \left\{z_1 \ldots z_n \,\vert \forall i : \vert z_i\vert = n^k \wedge \forall i : z_i \in A_k\right\}$
$PSpace$	$\{z \,\vert z = k \diamond x \wedge x \in A_k\}$	$\{z \,\vert z = k \diamond x \wedge x \in B_k\}$

References

[1] Christos H. Papadimitriou: Computational Complexity, Addison-Wesley, 1995.

[2] D. P. Bovet, P. Crescenzi, R. Silvestri: A uniform approach to define complexity classes, Theor. Comp. Sci. **104** (1992), pp. 263-283.