Komplexitätstheorie

WiSe 2008/09 in Trier

Henning Fernau
Universität Trier
fernau@uni-trier.de

Komplexitätstheorie

- Diskussion verschiedener Komplexitätsklassen:
  parameterisierte (Zeit-)Komplexität
- zugehörige Reduktionsbegriffe
- vollständige Probleme
- Anpassung von Klassenbegriffen und Reduktionen

Was macht gute Komplexitätstheorie aus ?

- Eine schöne Klasse $\mathcal{K}$ “guter Probleme”:
  Bsp.: $\mathbf{FPT}$.
- Eine entsprechend passende Reduktion, unter der $\mathcal{K}$ abgeschlossen ist:
  Bsp.: $\mathbf{FPT}$-Reduktion $f = (f_1, f_2)$; ist (P, k) “gut”, so auch $(f_1(P), f_2(P))$.
- Reduktionsbegriff ist kompositionsstabil
- Eine größere, böse Klasse $\mathcal{K}'$, z.B. $\mathbf{NP}$.
- Reduktionsbegriff ist schwächer als $\mathcal{K}'$, $\mathcal{K}'$ ist aber abgeschlossen gegenüber den betrachteten Reduktionen..
- Die Echtheit der (bekannten) Inklusion $\mathcal{K} \subseteq \mathcal{K}'$ ist unbekannt.

Parameterisierte Algorithmik

Die Klasse FPT (fixed parameter tractable) enthält Sprachen $L \subseteq \Sigma^* \times \mathbb{N}$, für die es einen Algorithmus gibt, der die Frage "$(x, k) \in L$?" in Zeit

$$f(k) \cdot p(|x|)$$

entscheidet (f bel., p Polynom).

Satz. Gleichwertig hiermit ist: Es gibt einen Problemkern (x', k') zu Instanz (x, k) mit $k', |x'| \in \mathcal{O}(g(k))$ für g beliebig.
Die zugehörige Problemkernreduktion ist in Polynomzeit berechenbar.

Mehr ?! $\rightsquigarrow$ Spezialvorlesung Parameterisierte Algorithmen

Reduktionen

Es seien (P, k) und (P', k') parameterisierte Probleme (formal gegeben als Sprachen über den Alphabeten Σ bzw. Σ').
Eine **FPT**-*(many-one)-Reduktion* von (P, k) auf (P', k') ist eine Abbildung $R : (\Sigma^* \times \mathbb{N}) \to ((\Sigma')^* \times \mathbb{N})$ mit:
(1) $\forall (x, k) \in \Sigma^* \times \mathbb{N} : ((x, k) \in L(P) \iff R(x, k) \in L(P'))$
(2) R ist in Zeit $f(k) \cdot p(|x|)$ berechenbar für bel. f und Polynom p.
(3) Es gibt Funktion $g : \mathbb{N} \to \mathbb{N}$, sodass $\forall k \in \mathbb{N} : k' \leq g(k)$ für alle x, sodass $R(x, k) = (x', k')$.
Schreibweise: $(P, k) \leq_{\mathbf{FPT}} (P', k')$.

Das Halteproblem auf Turing-Maschinen $\leadsto$ Klasse W[1]
ist einer der natürlichen Kandidaten, um "harte Probleme" zu definieren.
In der "klassischen Berechenbarkeit" ist das allgemeine Halteproblem das typische Beispiel für ein unentscheidbares Problem.
Schränkt man die Schrittzahl, die eine Turing-Maschine machen darf, geeignet ein, so gelangt man in der "klassischen Komplexitätstheorie" zu typischen NP-harten Problemen.
Man kann hierbei sowohl Ein- als auch Mehrband-Turing-Maschinen betrachten.
Die einfachste Variante ist vielleicht:
KURZE NTM-AKZEPTANZ

Eingabe: Einband-NTM M, Eingabe x
Parameter: $k \in \mathbb{N}$
Frage: Gibt es Berechnung von M auf x, die in $\leq k$ Schritten akzeptiert?

Die A-Hierarchie: Das parameterisierte Analogon zur Polynomzeithierarchie

Es sei Φ eine Menge von Formeln.

MODEL-CHECKING(Φ)

Eingabe: Struktur $\mathcal{S}$, Formel $\phi \in \Phi$
Parameter: Größe von ϕ
Frage: Gilt $\phi(\mathcal{S}) \neq \emptyset$?

Sei $A[t]$ die Klasse aller Probleme, die sich auf MODEL-CHECKING(Σ_t) **FPT**-reduzieren lassen.

Beispiel für Σ_1-Formel über der Struktur "Graph" $G = (V, E)$:
$is_k = \exists x_1 \ldots \exists x_k \left(\bigwedge_{1 \leq i < j \leq k} (x_i \neq x_j \wedge \neg E x_i x_j) \right)$.

Die Abbildung $(G, k) \mapsto (G, is_k)$ ist eine **FPT**-Reduktion vom natürlich parameterisierten Cliquenproblem auf MODEL-CHECKING(Σ_1).

Σ_1^+ bezeichnet Σ_1-Logik ohne Negation.

Lemma 1 MODEL-CHECKING$(\Sigma_1) \leq_{\mathbf{FPT}}$ MODEL-CHECKING(Σ_1^+).

Sei $(\mathcal{S}, \phi)$ eine Instanz von MODEL-CHECKING(Σ_1).
O.E.: ϕ ist in Negationsnormalform (Negationen stehen nur vor Atomen).
Sei ρ das relationale Alphabet von $\mathcal{S}$ und U das Universum.
Wir konstruieren negationsfreie Instanz $(\mathcal{S}', \phi')$ mit $\mathcal{S}' = (U, \rho')$.
$\rho' \supset \rho$ enthält eine neue binäre Relation $>$ sowie, für alle $R \in \rho$, drei Relationen R_f, R_ℓ und R_s mit $\alpha(R_f) = \alpha(R_\ell) = \alpha(R)$ und $\alpha(R_s) = 2\alpha(R)$.
R_f enthält das $<$-lexikographisch kleinste Element aus R.
R_ℓ enthält das $<$-lexikographisch letzte Element aus R.
$R_f = R_\ell = \emptyset$ gdw. $R = \emptyset$.
$R_s(a, b)$ gilt gdw. $R(a)$ und $R(b)$ gelten, $a < b$, und es gibt kein c mit $a < c$ und $c < b$ sodass $R(c)$ gilt.
Eine Negation in $(\mathcal{S}, \phi)$ kann wie folgt übersetzt werden:
$a \notin R$ gdw. entweder $a < x$ mit $R_f(x)$ oder ($x < a$ und $a < y$ und $R_s(x, y)$) oder $x < a$ mit $R_\ell(x)$.
Details ? $\rightsquigarrow$ Übungsaufgabe !

$\Sigma_1^+[2]$ bezeichnet Σ_1-Logik ohne Negation, mit lauter zweistelligen Relationen

Lemma 2 MODEL-CHECKING$(\Sigma_1^+) \leq_{\mathbf{FPT}}$ MODEL-CHECKING$(\Sigma_1^+[2])$.

Sei $(\mathcal{S}, \phi)$ eine Instanz von MODEL-CHECKING(Σ_1^+).
Sei $\mathcal{S} = (U, \rho)$.
Konstruiere $\mathcal{S}' = (U', \rho')$ mit $U = \bigcup_r U^r \cup \rho$.
Vereinigt wird über alle U^r, sodass $r = 1$ oder $r = \alpha(R)$ für ein $R \in \rho$.
ρ' enthält für $R \in \rho$ Relation P_R sowie Projektionen π_i für solche r mit $i \leq \alpha(R)$.
Ersetze $Rx_1 \dots x_r$ (mit $r = \alpha(R)$) in $(\mathcal{S}', \phi')$ durch:
$\exists y(P_R y \wedge \pi_1 x_1 y \wedge \dots \wedge \pi_r x_r y)$.

$(\mathcal{S}', \phi')$ heißt auch *bipartite Struktur* oder *Inzidenzstruktur* zu $(\mathcal{S}, \phi)$. Warum ?

Satz 3 *Das parameterisierte Clique-Problem ist A[1]-vollständig.*

Erinnerung: Beispiel für Σ_1-Formel über der Struktur "Graph" $G = (V, E)$:

$is_k = \exists x_1 \ldots \exists x_k \left(\bigwedge_{1 \leq i < j \leq k} (x_i \neq x_j \wedge \neg E x_i x_j) \right)$.

Entsprechend:

$cl_k = \exists x_1 \ldots \exists x_k \left(\bigwedge_{1 \leq i < j \leq k} (x_i \neq x_j \wedge E x_i x_j) \right)$.

$\rightsquigarrow$ Das parameterisierte Clique-Problem liegt in A[1].

Cliquen und Logik

Sei zunächst ϕ eine Formel der Form $\exists x_1 \exists x_2 \ldots \exists x_k \bigwedge_{i \in I} A_i$, mit A_i atomar und zweistellig. Sei U das Universum.
Definiere Graphen $G_\phi = (V, E)$ mit $V = U \times \{1, \ldots, k\}$.
E enthält Kante zwischen (a, r) und (b, s) gdw. für alle $i \in I$, sodass x_r und x_s die Variablen von A_i sind, d.h., $A_i = R_i(x_r, x_s)$, gilt: $\phi(R_i(a, b))$.
Wir können $x_i = a_i$ (mit $a_i \in U$) als gültige Belegung von ϕ wählen, gdw. G_ϕ enthät Clique $(a_1, 1), \ldots, (a_k, k)$.

Überführe eine allgemeine $\Sigma_1^+[2]$-Formel ϕ in disjunktive Normalform, d.h. o.E.: $\phi = \exists x_1 \exists x_2 \ldots \exists x_k \bigvee_{j \in J} \bigwedge_{i \in I} A_{i,j}$, mit $A_{i,j}$ atomar und zweistellig.
Konstruiere zu $\phi_j = \exists x_1 \exists x_2 \ldots \exists x_k \bigwedge_{i \in I} A_{i,j}$ Graphinstanz G_{ϕ_j}.
Die disjunkte Vereinigung $G = \bigcup G_{\phi_j}$ enthält Clique der Größe k gdw. ϕ wahr in der vorgelegten Struktur.

vorige Lemmata $\rightsquigarrow$ Das parameterisierte Clique-Problem ist A[1]-hart.

Folgerungen

Aus den bereits gezeigten parameterisierten Reduktionen ergibt sich:

Folgerung 4 *Das parameterisierte Problem, eine unabhängige Menge der Größe* k *in einem Graphen zu finden, ist A[1]-vollständig.*

Folgerung 5 KURZE NTM-AKZEPTANZ *ist A[1]-hart.*

Wir werden im Folgenden sogar die A[1]-Vollständigkeit dieses Problems zeigen.

$\rightsquigarrow$

Satz 6 *A[1]=W[1].*

Nochmals ein Rechenteppich

Ziel: Beschreibe Berechnung einer NTM durch Formel über geeigneter Struktur.

Hier nur grobe Skizze (siehe Flum / Grohe S. 115f.).
Das Universum beinhaltet u.a. das Zustands- und Eingabealphabet der NTM sowie ein Zähleralphabet bis $k+1$.
Das Turingband wird mit genügend vielen Variablen dargestellt.
So könnte die Anfangskonfiguration beschrieben werden durch:
$\text{init}(z, t, x_1, \ldots, x_{k+1}) = \text{INIT}z \wedge P_1 t \wedge \bigwedge_{1 \leq i \leq k+1} \text{BLANK}x_i$.
z ist hierbei die Zustandsangabe, t der Schrittzähler (Zeit),
x_i der Bandinhalt des i-ten Felds.

Die Länge der Formel ist schließlich quadratisch in k.

Schwierigere Probleme ?

Die Klasse W[2] kann durch das k-Schritt-Halteproblem auf nichtdeterministischen Mehrband-TM charakterisiert werden.
Wichtig ist der “Einschrittzugriff” auf mehrere Bänder parallel.

Lemma 7 DOMINATING SET *liegt in W[2].*

Betrachte zu gegebenem Graph $G = (V, E)$ und Parameter k eine Turingmaschine mit $|V| + 1$ Bändern und einem Bandalphabet, das u.a. V umfasst.

Eingangs schreibt die TM ein #-Symbol auf alle Bänder und geht einen Schritt nach rechts.
Dann rät es k Zeichen aus V auf sein nulltes Band.
Schließlich überprüft die TM, ob die geratene Knotenmenge eine dominierende Menge ist, indem (durch Linksbewegung auf dem nullten Band) beim Einlesen des Zeichens v auf allen Bändern w mit $w \in N[v]$ eine Linksbewegung vollzogen wird (sofern möglich).
Die TM akzeptiert, wenn alle Köpfe wieder am linken Rand angekommen sind.

Hinweis: Das Problem ist sogar W[2]-vollständig.

Alternierung im Feinen

Das Problem, das leere Wort in höchstens k Schritten zu akzeptieren, kann man auch für alternierende Turingmaschinen formulieren.
Genauer bekommen wir eine ganze Familie von Problemen: KURZE ℓ-ATM-AKZEPTANZ, wobei ℓ die Anzahl der Alternierungen (auf akzeptierenden Konfigurationspfaden) beschränkt.

Satz 8 *Für jedes* $\ell \geq 1$ *ist* KURZE ℓ-ATM-AKZEPTANZ $\mathrm{A}[\ell]$*-vollständig.*

Dies sieht man wieder durch eine geeignete Analyse des Rechenteppichs.

Alternierung im Allgemeinen

Ist die Zahl der Alternierungen unbeschränkt, gelingt eine Kennzeichnung der Klasse $AW[*]$, die über MODEL-CHECKING(FO) beschrieben werden kann.

Satz 9 KURZE ATM-AKZEPTANZ *ist* $AW[*]$*-vollständig.*

Fragt man bei GEOGRAPHY, ob es einen Gewinn des Anziehenden in höchstens k Schritten gibt, hat man ein weiteres $AW[*]$-vollständiges Problem.

Abschließend

Ist alles unbekannt, was Komplexitätsklassen angeht?
Muss man sich stets mit Aussagen abspeisen, die bestenfalls “Vollständigkeit” zeigen?

NEIN!

Durch Diagonalisierung kann man z.B. zeigen, dass $\mathbf{L} \neq \mathbf{PSPACE}$ gilt.
Ebenso kann man z.B. $\mathbf{P}$ von Exponentialzeit trennen.

Das Polynomialzeit Halte-Problem

$\text{HALT}_{\text{P}} = \{\langle M \rangle \,\$\, \langle w \rangle \mid \text{TM } M \text{ hält auf } w \text{ und liefert 1 in höchstens } 2^{|w|} \text{ Schritten } \}$

Satz 10 *$HALT_{\text{P}}$ ist in Exponentialzeit entscheidbar, liegt aber nicht in* $\mathbf{P}$.

Die Entscheidbarkeit in Exponentialzeit ist trivial.
Z.z.: $\text{HALT}_{\text{P}} \notin \mathbf{P}$. Andernfalls gölte:

$$H_0 = \{\langle M \rangle \mid \langle M \rangle \,\$\, \langle M \rangle \in \text{ HALT}_{\text{P}}\} \in \mathbf{P}$$

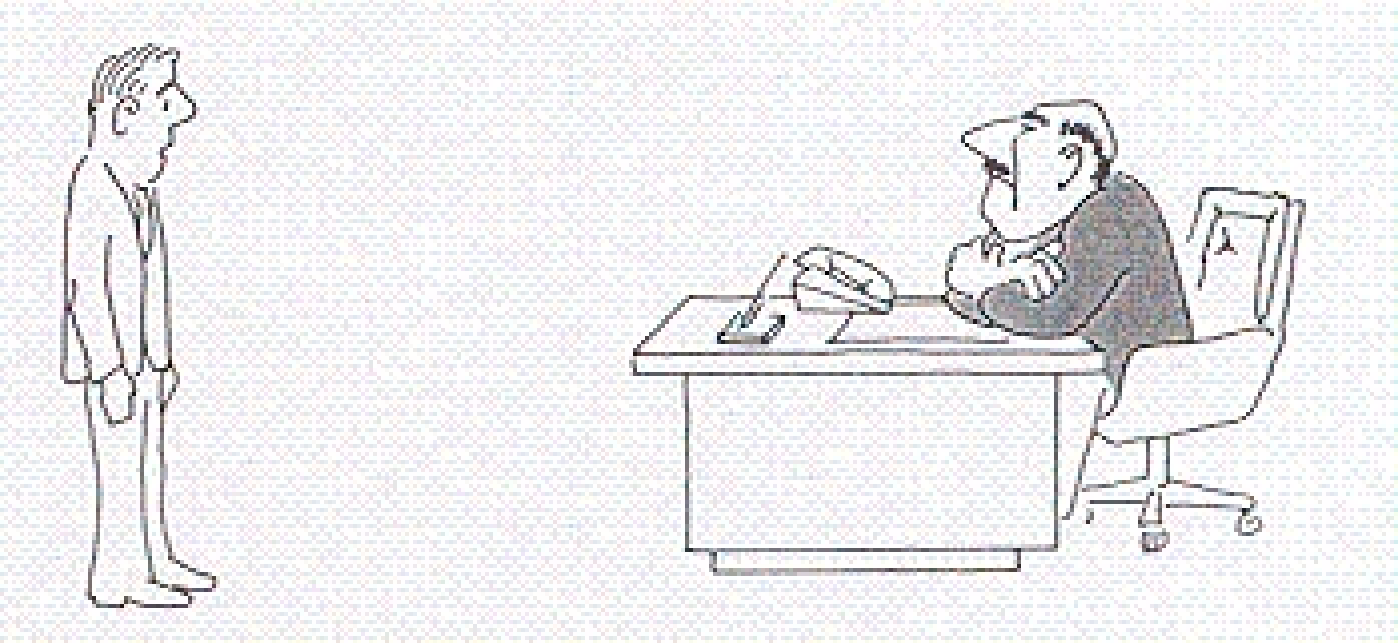
Also läge das Komplement $\overline{H_0}$ ebenfalls in $\mathbf{P}$.
Sei M_0 eine TM, die $\overline{H_0}$ in pol. Zeit p entscheidet, mit $p(x) \leq 2^x$ für alle $x \geq |\langle M_0 \rangle|$.

Wenn M_0 das Wort $\langle M_0 \rangle$ akzeptiert, dann $\langle M_0 \rangle \in L(M_0) = \overline{H_0}$, was der Annahme widerspricht,
H_O enthalte alle TMs, die höchstens $2^{|M_0|}$ Schritte machen, wenn sie ihre eigene Beschreibung als Eingabe bekommen.
$\rightsquigarrow M_0$ akzeptiert $\langle M_0 \rangle$ nicht. $\rightsquigarrow \langle M_0 \rangle \notin L(M_0) = \overline{H_0}$, d.h., $\langle M_0 \rangle \in H_0$.
Nach Definition von H_0 bedeutet dies: M_0 macht $2^{|M_0|}$ Schritte auf $\langle M_0 \rangle$, Widerspruch!

Daher liegen H_0 (und folglich ebensowenig HALT_{P}) nicht in $\mathbf{P}$.

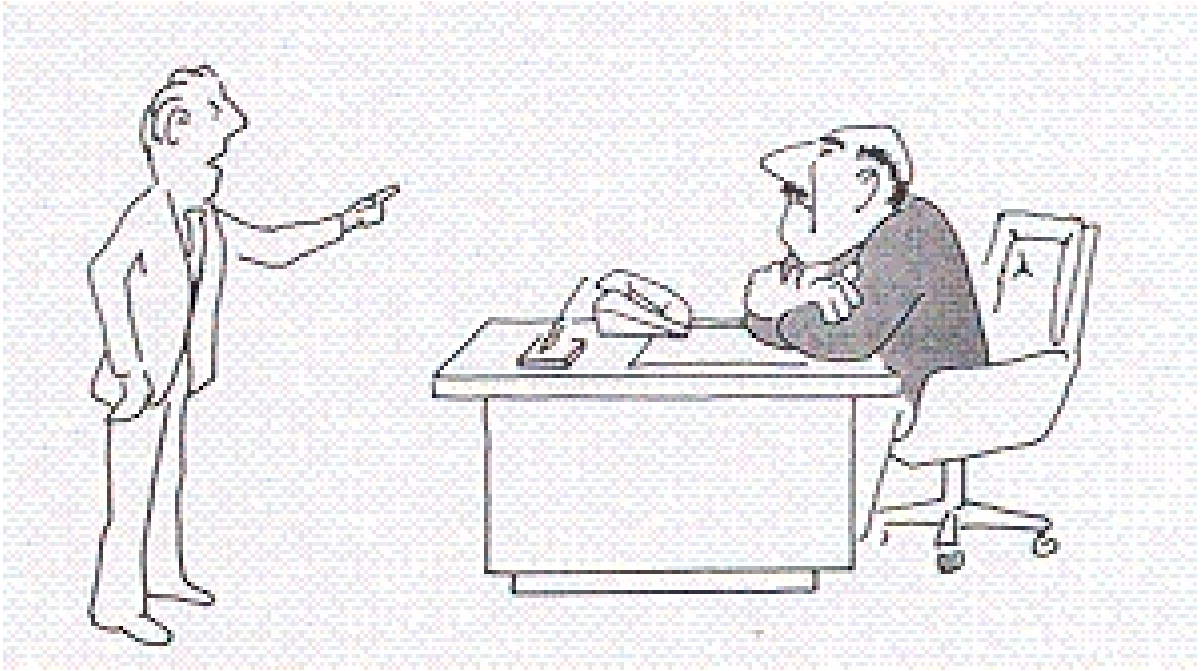
Motivation

siehe `http://max.cs.kzoo.edu/~kschultz/CS510/ClassPresentations/NPCartoons.html` wiederum aus Garey / Johnson



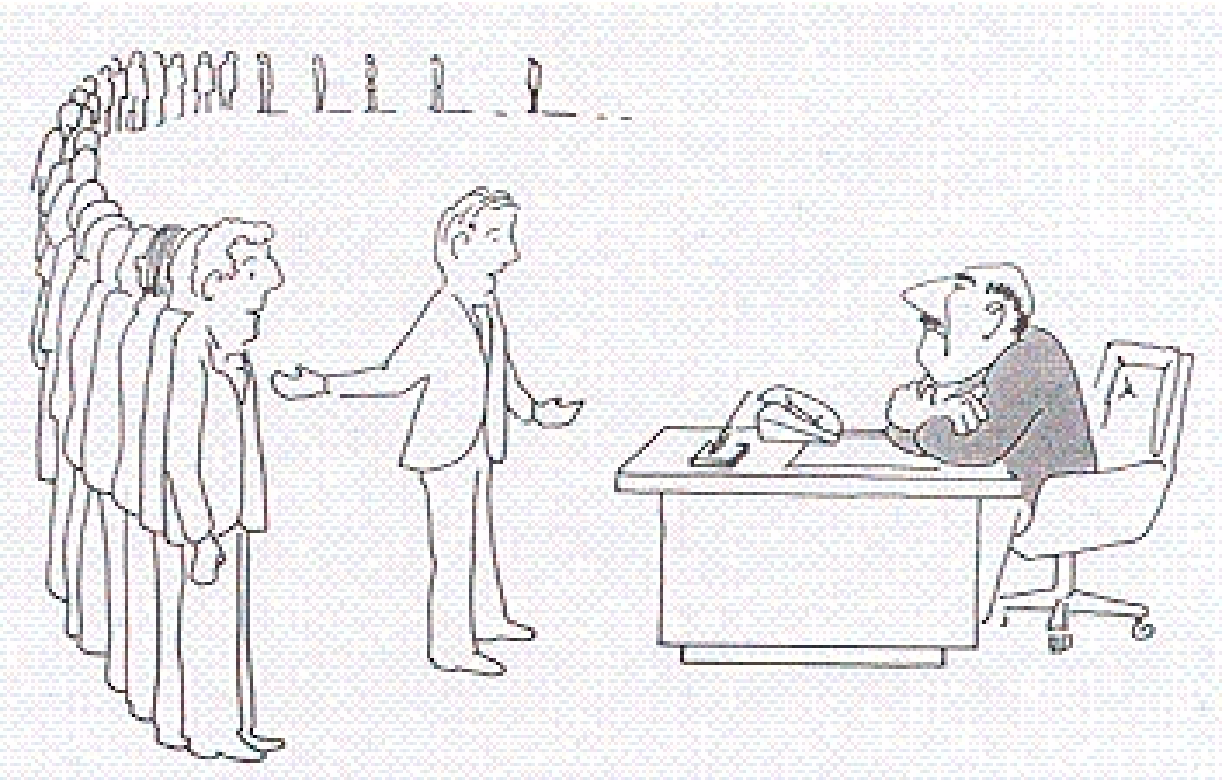
Sorry Chef, aber ich kann für das Problem keinen guten Algorithmus finden. . .

Die beste Antwort wäre hier aber. . .



. . . Ich kann aber beweisen, dass es für das Problem keinen guten Algorithmus geben kann !

Was die Komplexitätstheorie statt dessen liefert. . .



. . . Ich kann aber beweisen, dass das alle anderen auch nicht können !

Das Credo: $\mathbf{P} \subsetneq \mathbf{NP}$

Folgerung: Für NP-harte Probleme “glaubt man” nicht an Polynomzeitalgorithmen zu ihrer Lösung.

“Ergo” (?) Exponentialzeitalgorithmen sind unvermeidlich für exakte Lösungen NP-harter Probleme.

Ähnliche Zusammenhänge und Folgerungen sind in anderen Bereichen der KT möglich und üblich.

Nochmals Hinweise:
Vorlesungen zu Approximationsalgorithmen / Parameterisierten Algorithmen.
Klassischer Theorie-Kanon ebenfalls möglich: Vorlesung zu Formalen Sprachen.