das große „O" (1)

- Aufwandsklasse $O(g)$ für $g: \mathbb{N} \to \mathbb{N}$ enthält alle Funktionen $f: \mathbb{N} \to \mathbb{N}$ mit

  $$f(n) \leq \underset{\text{konstant}}{c} \cdot g(n) \quad \text{für alle} \quad n \geq \underset{\text{konstant}}{n_0}$$

- $O(g)$ beschreibt alle Datenverarbeitungsprobleme, die eine algorithmische Lösung op besitzen mit $T^{op} \in O(g)$

- obere Schranke, bei der konstante Faktoren und kleine Eingaben nicht zählen

das große „O" (2)

- Idee: wähle g einfach & einprägsam

  z.B. $\log n$, n, $n \cdot \log n$, n^2, n^3, …, 2^n, …

  logarithmisch, linear, quadratisch, kubisch, exponentiell

- $T^{mergesort} \in O(n \cdot \mathrm{ld}\, n)$ $\quad T^{quicksort} \in O(n^2)$

- $T^{xmpl}(n) \leq 1/3\, n\, \mathrm{ld}\, n + 22n + 1/2\, n^2 + 7$

  d.h. $T^{xmpl} \in O(n^2)$

  bzw. xmpl hat quadratischen Aufwand

Beob.: $O(\log n) \subsetneq O(n) \subsetneq O(n^2) \subsetneq O(n^3) \subsetneq \dots \subsetneq O(2^n)$

(vgl. Übung)

Anmerkungen

statt schlechtestem Fall macht auch Durchschnitt viel Sinn
(ist aber meist viel schwerer zu ermitteln)

Standardmaß für Zeitaufwand ist
die Zahl der Rechenschritte einer Turing-Maschine

zur Ermittlung wird allerdings meist eine Aktivität gezählt,
die konstant beschränkt viele Rechenschritte braucht und
sich selbst proportional zu den Gesamtaktivitäten verhält

Gleichungsanwendung ist nicht immer durch
konstant beschränkt viele Rechenschritte realisierbar

CE-S-Berechnungen sind i.a. nichtdeterministisch

P = NP - Problem

- besitzen (Entscheidungs-)Probleme mit einer Nichtdeterministischen Polynomiellen Lösung immer auch eine deterministische Polynomielle Lösung?
- P enthält die Probleme, die mit herkömmlichen Computern zeitgerecht lösbar sind
- NP enthält eine Vielzahl praktisch relevanter Probleme (wie Tourenplanung, Maschinenbelegung, Stundenplanung, Lagerhaltung, ...)
- eines der bekanntesten offenen Probleme der Inf.
- Nobelpreis-verdächtig (1 Mill. US $ ausgesetzt)

Anmerkungen zum Nichtdeterminismus

- manches ist so
  - Suchen in Bereichen, unsicheres & unvollständiges Wissen, Expertensysteme, Spiele, ...
- Gleichwertigkeit ist so
  - Auswerten & Beweisen im selben Rahmen
- manchmal bequem (und schadet nicht)
  - Anforderungsdefinitionen
    z.B. $|\, sqr(x)^2 - x \,| \leq \varepsilon$
- einschränken, wenn gewünscht o. nötig

Definition von NP und P

- Entscheidungsprobleme: Abbildungen der Form $dp : A^* \to BOOL$
- Lösung von dp: CE-S-Operation $sol : A^* \to BOOL$, so dass für alle w A^* gilt: $dp(w) = T$ g.d.w. $sol(w) = T$

  gleich　　　gleichwertig

  (aber: $dp(w) = F$ g.d.w. $sol(w) \neq T$)
- $dp \in NP$: Lösung sol und $k \in \mathbb{N}$ existieren mit $T^{sol} \in O(n^k)$
- $dp \in P$: zusätzlich gilt für alle $w \in A^*$:

  $dp(w) = T$ & $sol(w) = t$ impl. $t = T$

offensichtlich: $P \subseteq NP$　　　offen: $NP \subseteq P$

Ordnung auf NP durch Reduktion

- Reduktion von $dp_1 : A_1^* \to BOOL$ auf $dp_2 : A_2^* \to BOOL$:
  CE-S-Operation $red: A_1^* \to A_2^*$, so dass $T^{red} \in O(n^\ell)$ für ein $\ell \in \mathbb{N}$
  und für alle $w \in A_1^*$ gilt: $dp_1(w) = T$ g.d.w. $dp_2(red(w)) = T$

- in diesem Fall: $dp_1 \leq dp_2$

Theorem: $dp_1 \leq dp_2$ und $dp_2 \in P$ impl. $dp_1 \in P$

Beweis: solution$_1$

opns: $sol_1 : A_1^* \to BOOL$
vars: $w \in A_1^*$
eqns: $sol_1(w) = sol_2(red(w))$

$$
\begin{aligned}
T^{sol_1}(n) &= 1 + T^{red}(n) + T^{sol_2}(\overbrace{m}^{\text{Länge von } red(w)}) \\
&\leq 1 + c \cdot n^\ell + c_2 \cdot m^{k_2} \\
&\leq 1 + c \cdot n^\ell + c_2 \cdot (n^\ell)^{k_2} \quad \text{(Ausgabelänge von red in } O(n^\ell)\text{)} \\
&\leq c_1 \cdot n^{\ell \cdot k_2}
\end{aligned}
$$

(unvollständig)

NP-Vollständigkeit

▷ $dp_0 \in NP$ NP-vollständig, falls $dp \leq dp_0$ für alle $dp \in NP$

Theorem: dp_0 NP-vollständig und $dp_0 \in P$ impl. $NP \subseteq P$

Beweis: $dp \in NP$ impl. $dp \leq dp_0$ (nach Vor.) impl. $dp \in P$
(nach obigem Theorem)

Theorem (Cook 1971)

Erfüllbarkeitsproblem der Aussagenlogik NP-vollständig

P = NP - Problem*

- Läßt sich das Erfüllbarkeitsproblem der Aussagenlogik deterministisch in polynomieller Zeit lösen?
- entsprechende Frage für viele praktisch relevanten Probleme

  (Maschinenbelegung, Tourenplanung, Stundenplan, Handelsreisende; Färbung, Lagerhaltung, ...)
- richtige Antwort mit Beweis wäre Nobel-Preis-verdächtig

* P : alle polynomiell lösbaren (Entscheidungs-)Probleme

NP : alle nichtdeterministisch polynomiell lösbaren P.

Erfüllbarkeitsproblem

- aussagenlogische Formel $f = c_1 \wedge c_2 \wedge \dots \wedge c_n$
- Klausel $c_i = l_{i1} \vee l_{i2} \vee l_{i3}$ für $i = 1, \dots, n$
- Literale $l_{ij} = x$ oder $l_{ij} = \neg y$ für $j = 1, 2, 3$
- Grundaussagen $x, y, \dots \in X$ (k Elem.)

Gesucht: Belegung der Grundaussagen mit TRUE oder FALSE, so daß f gilt.

Lösung 1: Probiere alle 2^k Belegungen $\in O(2^k)$

Lösung 2: Rate richtige Belegung $\in O(k)$
(nichtdeterministisch)

Ein Gerät mit 4 0/1-Schaltern befindet sich in einem betriebssicheren Zustand, wenn folgende Schalterstellungen beachtet werden:

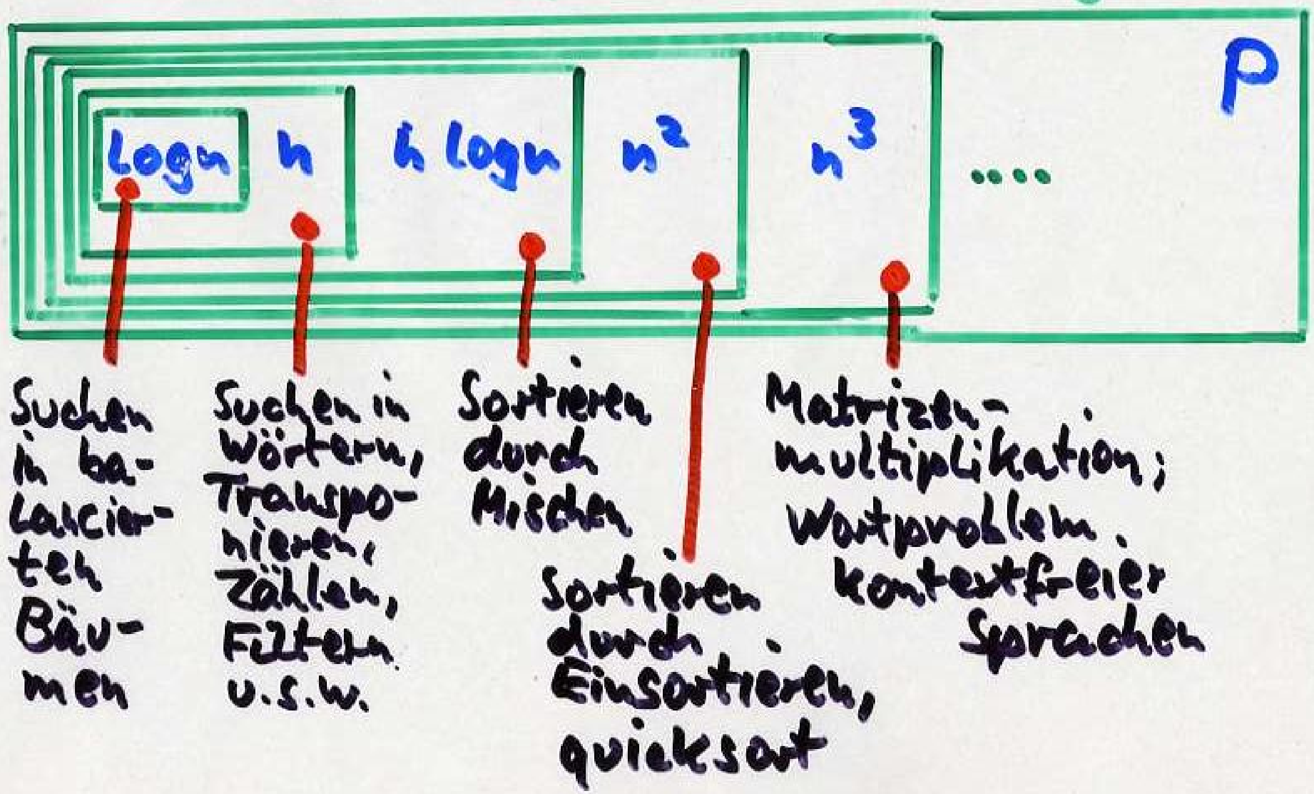
(1) wenn A und B 0, dann C 1

(2) A oder C 1

(3) A,B,C 0 oder B,C,D 1 oder A,D 1

polynomieller Aufwand: die Klasse P

- Probleme in P können mit herkömmlicher Rechnertechnik in verfügbarer Zeit gelöst werden (wenn der Exponent nicht zu groß wird)



polynomieller Platzbedarf

NPSPACE: Probleme mit nichtdeterministischen Lösungsalgorithmen, die polynomiellen Speicherplatz brauchen (im Verhältnis zur Größe der Eingabe)

PSPACE: analog für deterministische Lösungen

Beobachtung: PSPACE $\overset{(1)}{=}$ NPSPACE $\overset{(2)}{\supseteq}$ NP $\supseteq$ P

(1) Beweis durch Backtracking

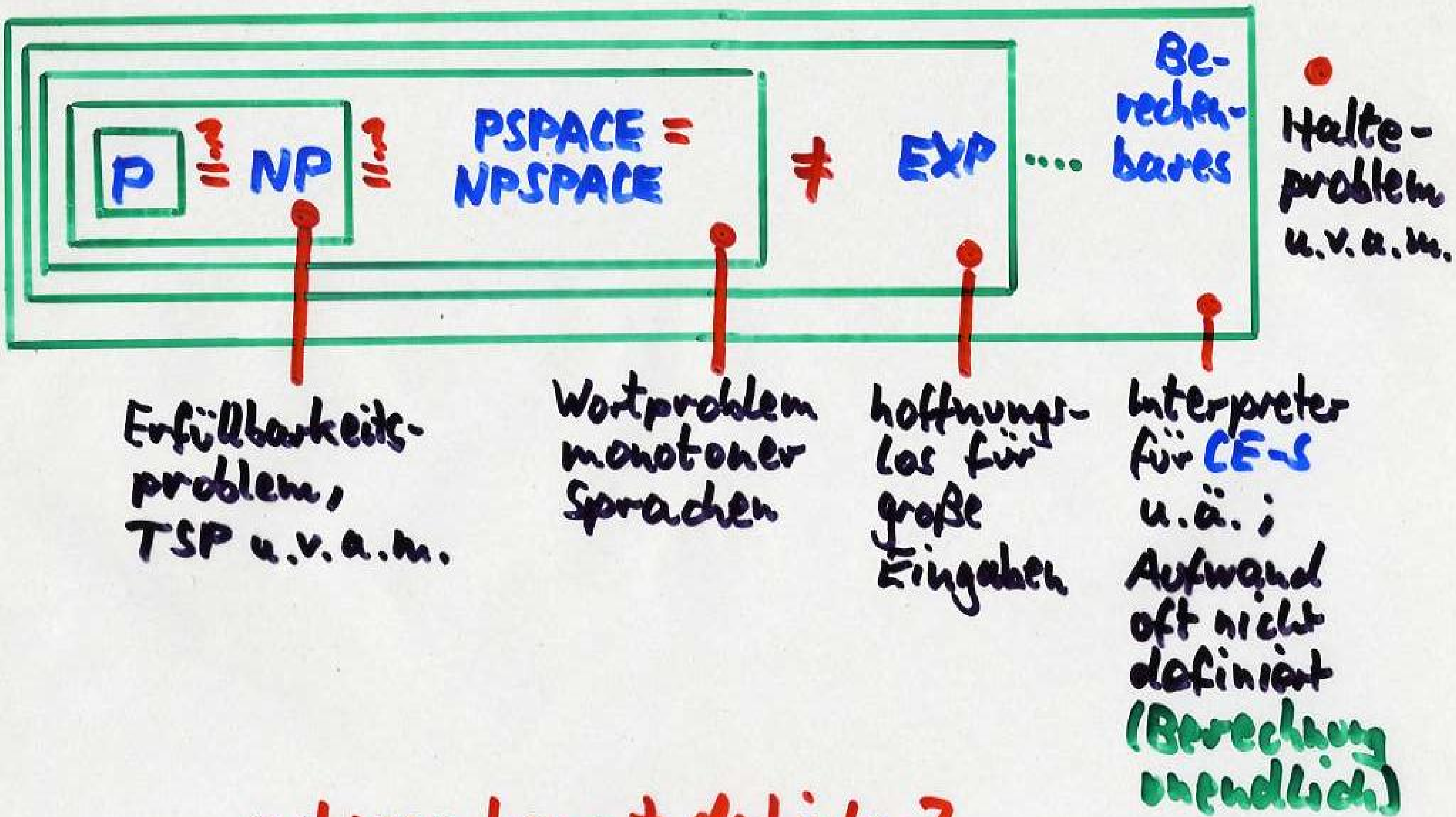
offenes Problem: P $\overset{?}{=}$ PSPACE

typisches Beispiel: Wortproblem monotoner Sprachen

---

(2) konstanter Speicherplatzbedarf pro Rechenschrift

wie weit reicht P ?



... und was kommt dahinter?