SUBSET-SUM (Variante vom Rucksackproblem)

<u>Gegeben:</u> $a_1, \dots, a_k \in \mathbb{N}$ und $b \in \mathbb{N}$.

<u>Gefragt:</u> Gibt es $I \subseteq \{1, \dots, k\}$ mit $\underline{\sum_{i \in I} a_i = b}$?

<u>Satz</u> SUBSET-SUM ist NP-vollständig.

<u>Bew:</u>

SUBSET-SUM $\in$ NP: Rate I, verifiziere $\sum_{i \in I} a_i = b$. ✓

NP-Härte: Reduktion von 3KNF-SAT.

Sei $F = (z_{11} \vee z_{12} \vee z_{13}) \wedge \dots \wedge (z_{m1} \vee z_{m2} \vee z_{m3})$

mit $z_{ij} \in \{x_1, \dots, x_n\} \cup \{\neg x_1, \dots, \neg x_n\}$.

D.h. n Variablen, m Klauseln.

Konstruiere SUBSET-SUM Instanz:

$$b = \underbrace{44 \ldots 4}_{m}\,\underbrace{11 \ldots 1}_{n}$$

m: Anz. der Klauseln

n: Anz. der Variablen

$a_1 \ldots a_k$:

Vier verschiedene Gruppen: v_i, v_i', c_j, d_j $\quad i \in \{1, \ldots, n\}$

$j \in \{1, \ldots, m\}$

Für ein positives Vorkommen von x_i in Klauseln $j_1, j_2, \ldots j_l$: $j_1$ten $j_2$ten $j_l$ten

ite Stelle

$$v_i = \underbrace{0 \ldots 1 \ldots 1 \ldots 1 \ldots 0}_{m}\ \boxed{\underbrace{0 \ldots 010 \ldots 0}_{n}}$$

Für negative Vorkommen ~~v~~on x_i entsprechend

$$v_i' = \underbrace{0 \ldots 1 \ldots 1 \ldots 1 \ldots 0}_{m}\ \underbrace{0 \ldots 010 \ldots 0}_{n}$$

$$c_j = 0 \ldots\ldots 1 \ldots\ldots 0 \quad 0 \ldots\ldots 0$$

$$d_j = 0 \ldots\ldots 2 \ldots\ldots 0 \quad 0 \ldots\ldots 0$$

jte Stelle

Bsp

$$F = (\boxed{x_1} \vee \neg x_3 \vee x_5) \wedge (\neg x_1 \vee x_5 \vee x_4) \wedge (\boxed{\neg x_2} \vee \neg x_2 \vee \neg x_5)$$

$n = 5 \quad m = 3$

$b = 444\ 11111$

$\to v_1 = 100\ 10000$ | $v_1' = 010\ 10000$
$v_2 = 000\ 01000$ | $v_2' = 001\ 01000$
$v_3 = 000\ 00100$ | $v_3' = 100\ 00100$
$v_4 = 010\ 00010$ | $v_4' = 000\ 00010$
$v_5 = 110\ 00001$ | $v_5' = 001\ 00001$

$x_1 \mapsto 1$
$x_2 \mapsto 0$
$x_3 \mapsto 0$
$x_4 \mapsto 1$
$x_5 \mapsto 0$

$\Sigma = 212\ 11111$

$c_1 = 100\ 00000$ | $d_1 = 200\ 00000$
$c_2 = 010\ 00000$ | $d_2 = 020\ 00000$
$c_3 = 001\ 00000$ | $d_3 = 002\ 00000$

$\Sigma\ 444\ 11111$

$$\sum_{i \in I} a_i = 444\ 11111$$

mit Hilfe der v_i und v_i's immer mindestens 1!

entweder v_i oder v_i' ausgewählt!

Wenn es eine erfüllende Belegung für F gibt, dann können wir die Indexmenge I finden mit $\sum_{i \in I} a_i = b$.

Wenn es eine Indexmenge I mit $\sum_{i \in I} a_i = b$, dann gibt es eine erfüllende Belegung.

Da Konstruktion in polynomieller Zeit möglich

$3KNF\text{-}SAT \leq_p SUBSET\text{-}SUM$. D.h. SUBSET-SUM ist NP-hart. $\square$

Dynamic-Programming-Alg.

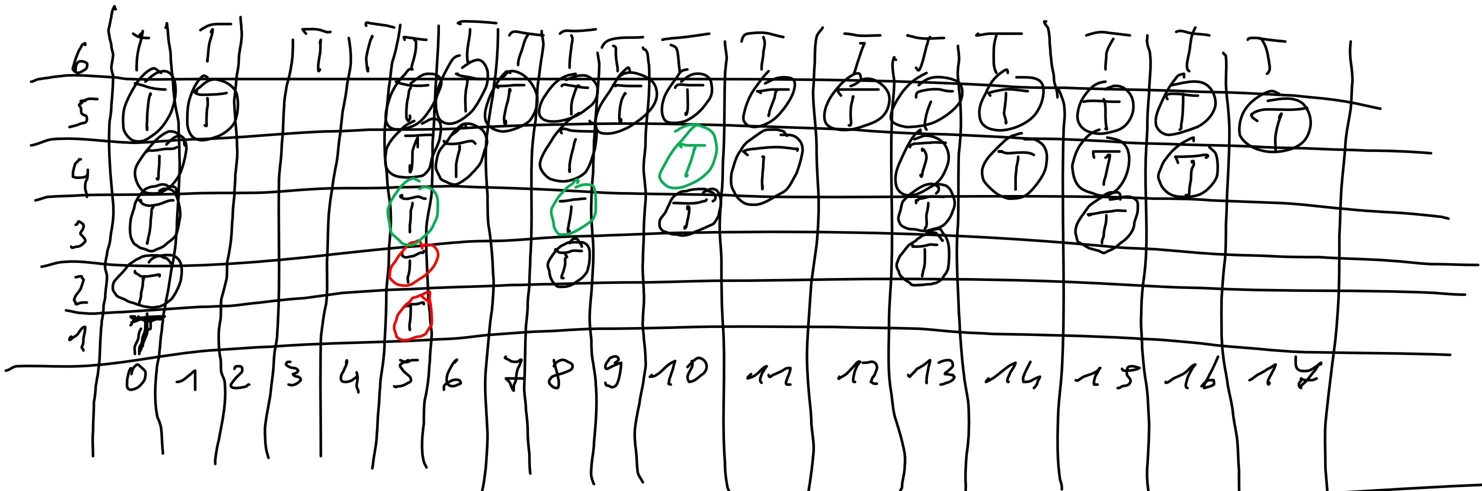
Baue Tabelle $T[\cdot,\cdot]$ auf:

$T[i,j]$ = wahr gdw. es ex. Teilmenge $I \subseteq \{1,\dots,i\}$ mit $\sum_{i\in I} a_i = j$

Lösung ist "ja" gdw. $T[k,b]$ = wahr ist.

$T[i,j] = (a_i = j) \vee T[i-1,j] = \text{wahr} \vee T[i-1, j-a_i] = \text{wahr}$

Bsp $a_1 = 5 \quad a_2 = 8 \quad a_3 = 10 \quad a_4 = 6 \quad a_5 = 1 \quad a_6 = 3 \qquad b = 17$



Laufzeit $O(\underline{b} \cdot k)$ $\quad \log b \rightarrow \boxed{\text{pseudo-polynomiell}}$

Jenseits von NP-vollständig

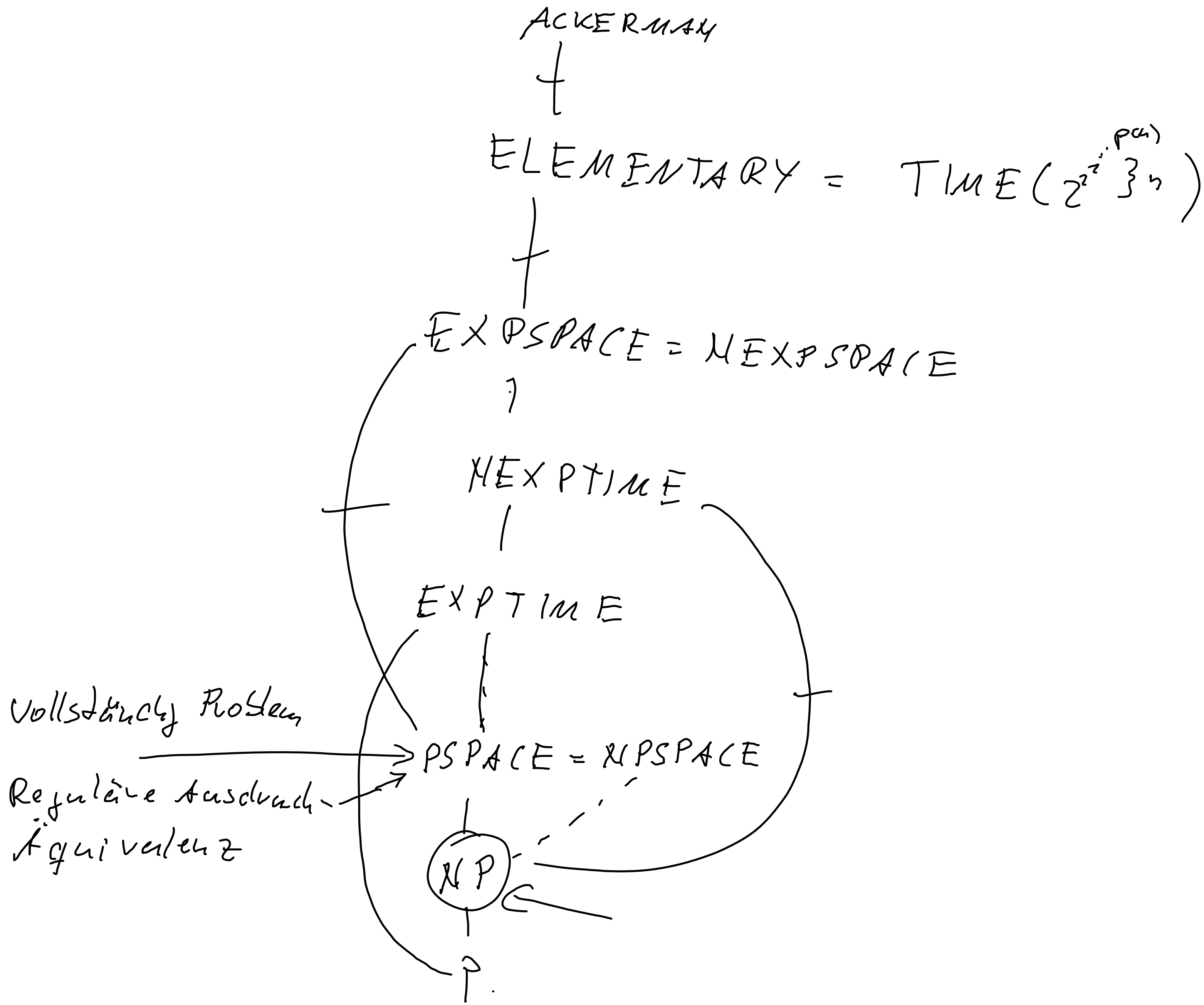
$$PSPACE = \bigcup_{p \text{ Polynom}} SPACE(p(n))$$

$$NPSPACE = \qquad NSPACE(p(n))$$

$$EXPTIME = \bigcup_{p \text{ Polynom}} TIME(2^{p(n)})$$

$$NEXPTIME = \dots \qquad NTIME$$

$$EXPSPACE = \dots \qquad SPACE(2^{p(n)})$$


ACKERMAN
ELEMENTARY = TIME($2^{2^{2^{\cdot^{\cdot^{p(n)}}}}}\}n$)
EXPSPACE = NEXPSPACE
NEXPTIME
EXPTIME
Vollständig Problem
PSPACE = NPSPACE
Reguläre Ausdruck-Äquivalenz
NP
P

NP-vollst. Problem Lösetechniken:

- Backtracking suche
- Branching - Heuristiken
- Lokale Suche
- Approximationsalg.