Technische Universität Braunschweig

Algorithmen und Datenstrukturen 2 – Übung #6

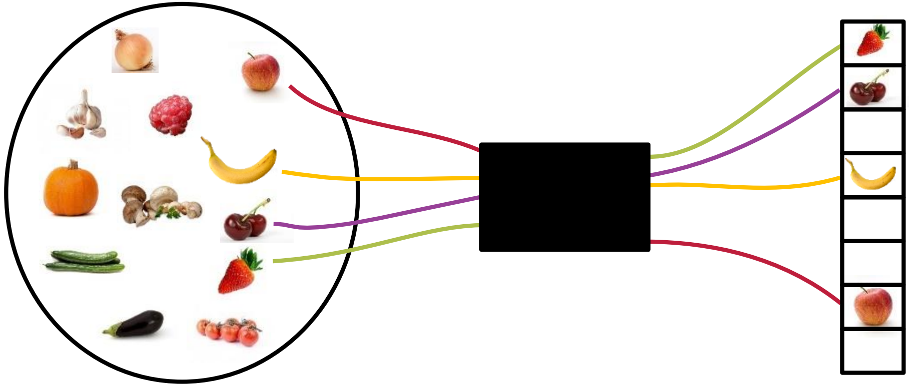
Hashing, Modulo, Wiederholung

Arne Schmidt

11.07.2019

Heute

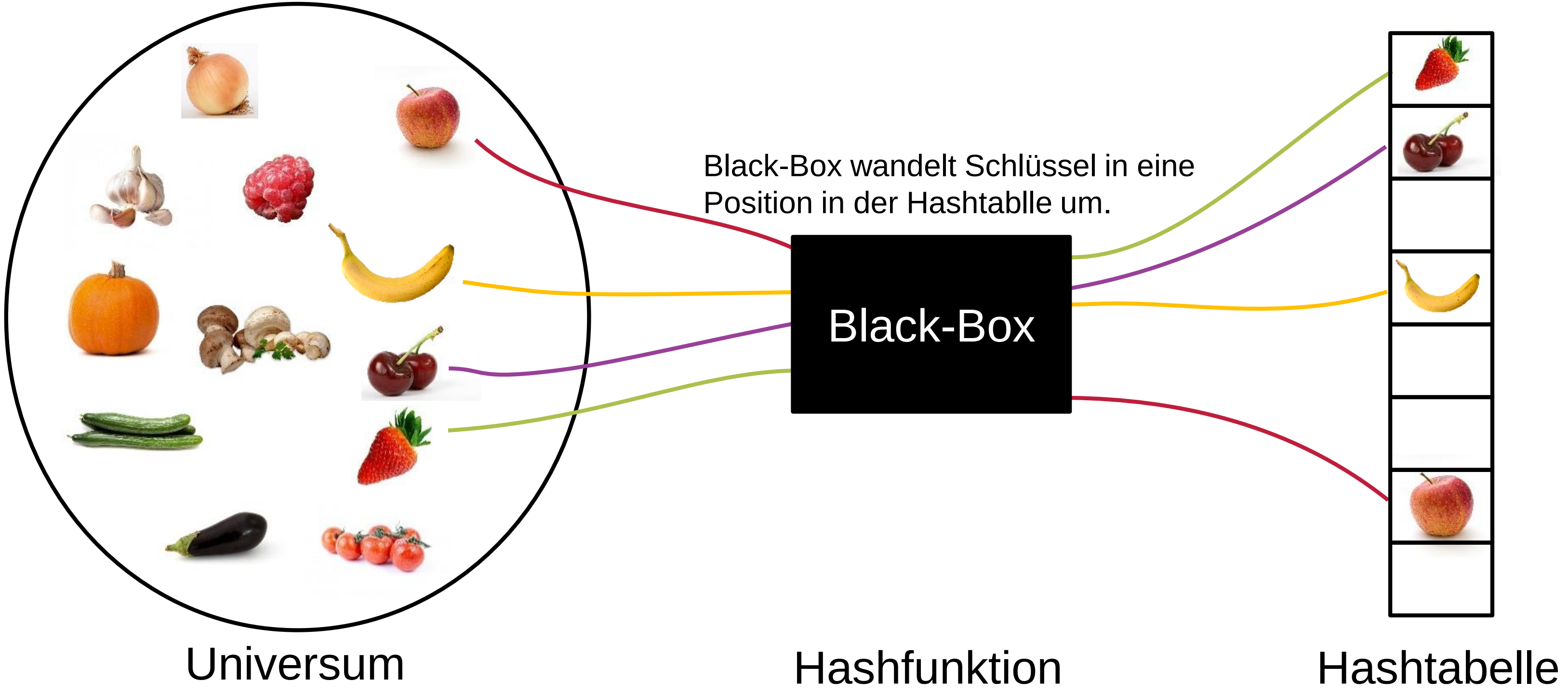
- Hashing
- Modulo
- Fragen/Wiederholung
  - Dynamische Programmierung
  - Approximation
  - Reduktionen



Hashing

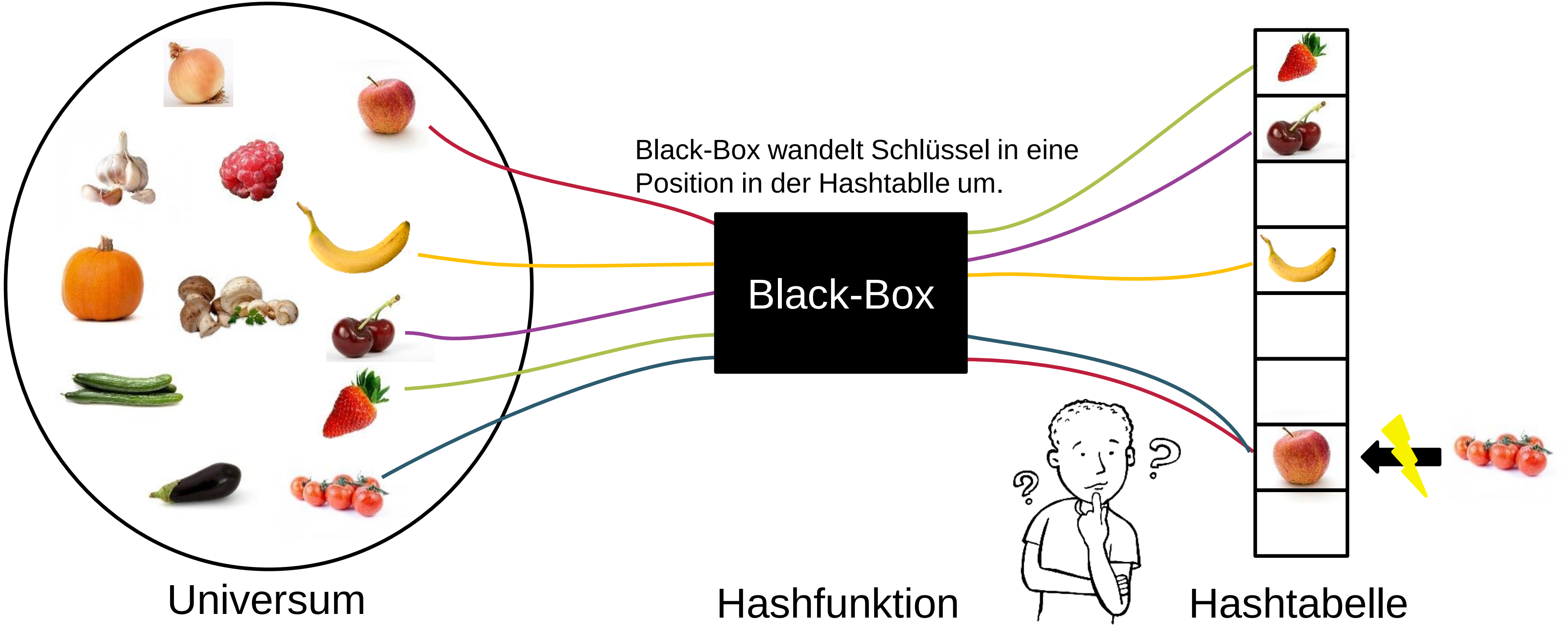
Hashing

Jedes Objekt besitzt einen Schlüssel



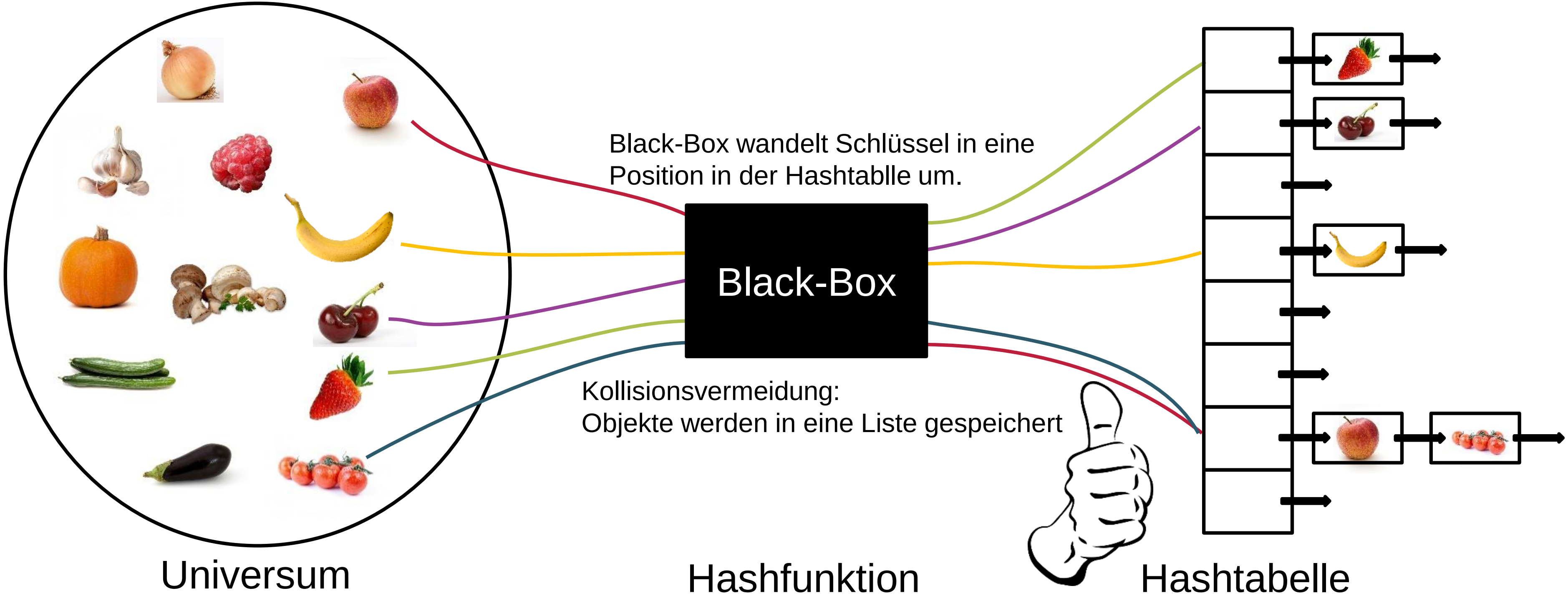
Hashing - Kollision

Jedes Objekt besitzt einen Schlüssel



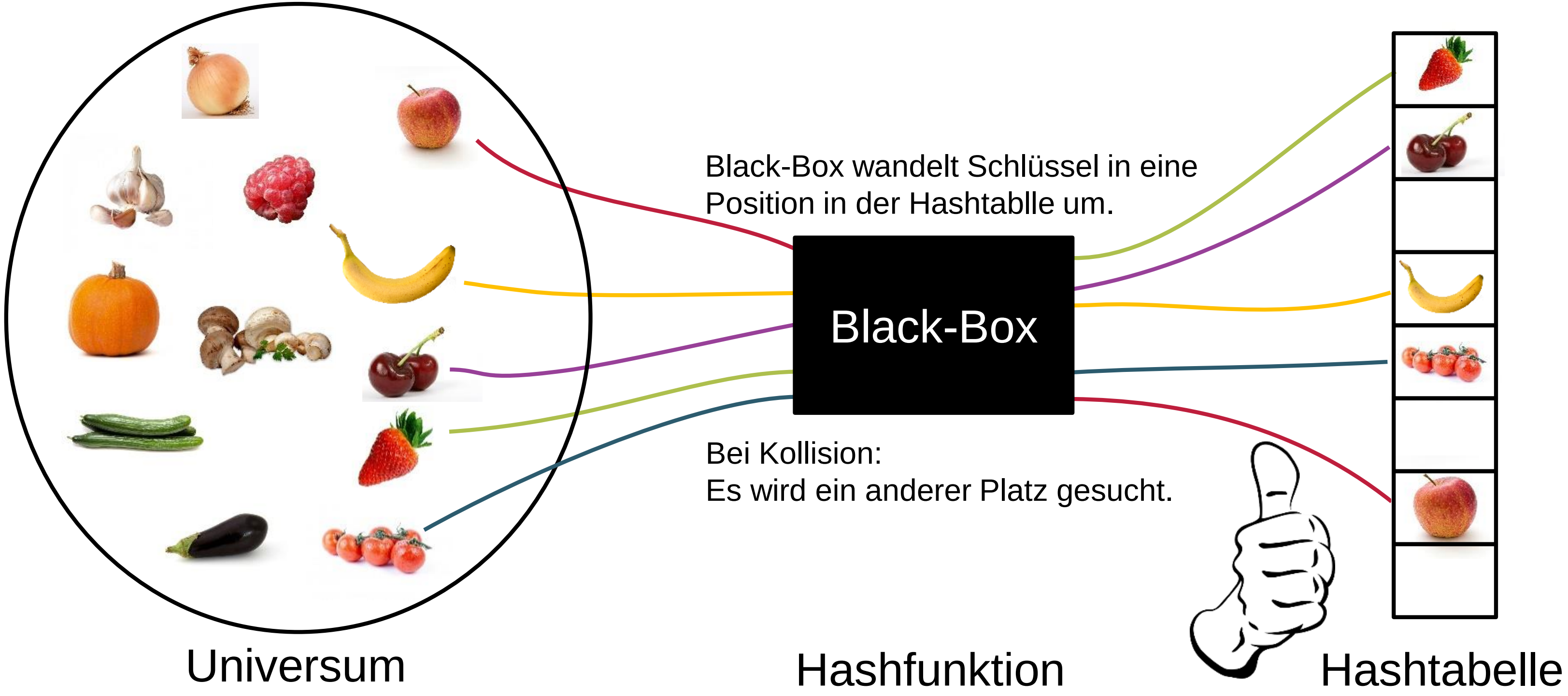
Hashing - Listen

Jedes Objekt besitzt einen Schlüssel



Hashing – Offenes Hashverfahren

Jedes Objekt besitzt einen Schlüssel



Hashfunktion – Offenes Hashverfahren

Es sind

$$m \in \mathbb{N},$$
$$h_1 : \mathbb{N} \to \{0, \dots, m-1\},$$
$$h_2 : \mathbb{N} \to \{1, \dots, m-1\}$$

und

$$f : \mathbb{N} \to \{0, \dots, m-1\}$$

Offenes Hashing benutzt eine Hashfunktion der Form

$$t(i, x) \coloneqq \left(h_1(x) + f(i) \cdot h_2(x)\right) \bmod m$$

Beispiel für $f(i) \coloneqq (a \cdot i + b) \bmod m$ in rekursiver Schreibweise:

$$t(i, x) \coloneqq \begin{cases} \left(h_1(x) + b h_2(x)\right) \bmod m & \text{, falls } i = 0 \\ \left(t(i-1, x) + a \cdot h_2(x)\right) \bmod m & \text{, falls } i > 0 \end{cases}$$

Hashfunktionen – Offenes Hashverfahren

$$t(i,x) \coloneqq \big(h_1(x) + f(i) \cdot h_2(x)\big) \bmod m$$

$t(i,x)$ nutzt

- **Lineares Sondieren**, falls $f(i) \in \Theta(i)$ und $h_2(x) \in \Theta(1)$
- **Quadratisches Sondieren**, falls $f(i) \in \Theta(i^2)$ und $h_2(x) \in \Theta(1)$
- **Doppeltes Hashing**, andernfalls

Hashing Ohne Kollisionen

Sei $P[h(x) = j] = \frac{1}{m}$.

Wahrscheinlichkeit, dass k Elemente ohne Kollisionen eingefügt werden können:

$$P[k \text{ Elemente kollisionsfrei einfügen}] = \prod_{i=1}^{k} \frac{\#\text{freie Felder}}{\text{Größe Tabelle}} = \prod_{i=1}^{k} \frac{m - i + 1}{m}$$

Wahrscheinlichkeit, dass das nächste Element kollisionsfrei eingefügt wird, wenn n Elemente eingefügt wurden:

$$P[\text{nächstes kollisionsfrei}] = \frac{m - n}{m} = 1 - \frac{n}{m} = 1 - \beta$$

β bezeichnet den *Belegungsfaktor*.

Modulo

Modulo – Restdivision

Für $m \in \mathbb{N}, x \in \mathbb{Z}$ und $r \in \{0, \dots, m-1\}$ ist

$$x \bmod m = r,$$

falls eine Zahl $q \in \mathbb{Z}$ existiert, sodass gilt

$$x = q \cdot m + r$$

Damit ist auch für beliebiges $i \in \mathbb{Z}$

$$(x + i \cdot m) \bmod m = x \bmod m$$

Beispiel:

$$27 \bmod 7 = 6$$
$$31 \bmod 9 = 4$$

Man kann zeigen:

$$(a + b) \bmod m = \big((a \bmod m) + (b \bmod m)\big) \bmod m$$
$$(a \cdot b) \bmod m = \big((a \bmod m) \cdot (b \bmod m)\big) \bmod m$$

Wiederholung I
Dynamische Programmierung

Würfelsumme

Gegeben: n Würfel mit je m Seiten (und Werten $1, \dots, m$) und eine Zahl K.

Gesucht: Anzahl Möglichkeiten den Wert K mit den Würfeln zu erzeugen.

Ein 6-seitiger Würfel:

K	0	1	2	3	4	5	6	7
#	0	1	1	1	1	1	1	0

Würfelsumme

Zwei 6-seitige Würfel

K	0	1	2	3	4	5	6	7	8	9	10	11	12	13
#	0	0	1	2	3	4	5	6	5	4	3	2	1	0

Summe!

Drei 6-seitige Würfel

K	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
#	0	0	0	1	3	6	10	15	21	25	27	27	25	21	15	10	6	3	1	0

Würfelsumme

Sei $\mathcal{A}(i, j)$ die Anzahl der Möglichkeiten den Wert i mit j m-seitigen Würfeln zu erzeugen. Dann ist

$$\mathcal{A}(i,j) := \begin{cases} 0 & \text{, falls } j = 0, i > 0 \\ 1 & \text{, falls } j = i = 0 \\ 0 & \text{, falls } i < j \\ 0 & \text{, falls } i > mj \\ \sum_{\ell=1}^{\min(i,m)} \mathcal{A}(i-\ell, j-1) & \text{, sonst} \end{cases}$$

K	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19
$\mathcal{A}(K,3)$	0	0	0	1	3	6	10	15	21	25	27	27	25	21	15	10	6	3	1	0

Welchen Wert besitzt $\mathcal{A}(14,4)$? Antwort: $21 + 25 + 27 + 27 + 25 + 21 = 146$

Würfelsumme

Function $A(K,n)$

$A[0..nm][0..n]$ //2D Array
$A[0][0] \coloneqq 1$
for $i = 1$ **to** K **do**
  $A[i][0] \coloneqq 0$

for $j = 1$ **to** n **do**
  for $i = 0$ **to** $j-1$ **do**
    $A[i][j] \coloneqq 0$
  for $i = j$ **to** jm **do**
    $k \coloneqq \min(i,m)$
    $A[i][j] \coloneqq \sum_{\ell=1}^{k} A[i-\ell][j-1]$
  for $i = jm+1$ **to** nm **do**
    $A[i][j] \coloneqq 0$
return $A[K][n]$

$$\mathcal{A}(i,j) \coloneqq \begin{cases} 0 & \text{, falls } j = 0, i > 0 \\ 1 & \text{, falls } j = i = 0 \\ 0 & \text{, falls } i < j \\ 0 & \text{, falls } i > mj \\ \sum_{\ell=1}^{\min(i,m)} \mathcal{A}(i-\ell, j-1) & \text{, sonst} \end{cases}$$

Wiederholung II
Approximation

Approximation – Probleme mit Parameter

Manchmal besitzt man Informationen über die Instanzen.

Beispiele:

MAXIMUM KNAPSACK mit $\boldsymbol{\alpha} \in \mathbb{N}$

Für alle $i \in \{1, \dots, n\}$ gilt $z_i \leq \frac{Z}{\alpha}$

➔ GREEDY_0 ist eine $\frac{\alpha-1}{\alpha}$ - Approx.

BIN PACKING mit $\boldsymbol{\alpha} \in \mathbb{N}$

Für alle $i \in \{1, \dots, n\}$ gilt $z_i \leq \frac{Z}{\alpha}$

➔ FIRST FIT ist eine $\frac{\alpha}{\alpha-1}$ - Approx.

SET COVER mit $\boldsymbol{k} \in \mathbb{N}$

Jedes Element $u \in U$ kommt in maximal k Teilmengen in $\mathcal{F}$ vor.

➔ Es gibt eine k - Approx.

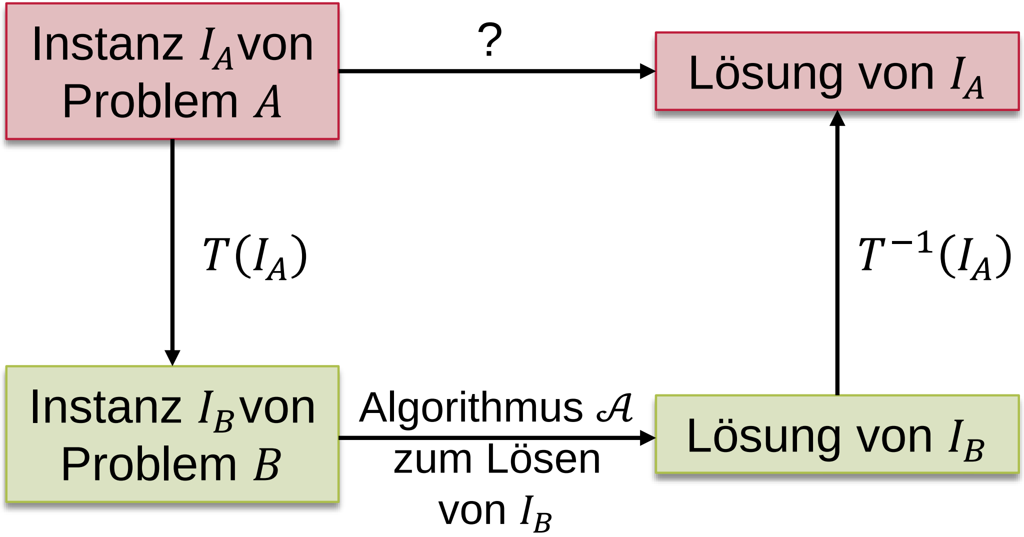
Für $k = 2$ ist das VERTEX COVER.

Wiederholung III
Reduktion

Reduktionen

$T(I_A)$ und $T^{-1}(I_A)$ besitzen polynomielle Laufzeit.

Daher: Besitzt $\mathcal{A}$ polynomielle Laufzeit, dann können wir die Lösung von I_A in polynomieller Zeit bestimmen.



3SAT-3

Gegeben: Formel wie bei 3SAT, aber jede Variable kommt in maximal drei Klauseln vor.
Frage: Lässt sich die Formel erfüllen?

Dieses Problem ist NP-schwer!

Wir zeigen: 3SAT $\leq_p$ 3SAT-3

Problematisch nur Variablen, die öfter als drei Mal vorkommen.
Wie können wir das auflösen?

3SAT-3

$$(x_i \vee x_j \vee x_k) \qquad (\bar{x}_i \vee \bar{x}_o \vee x_q) \qquad (x_i \vee \bar{x}_p \vee x_j) \qquad (x_i \vee x_p \vee \bar{x}_o)$$

Für jedes der n_i Literale von Variable x_i:

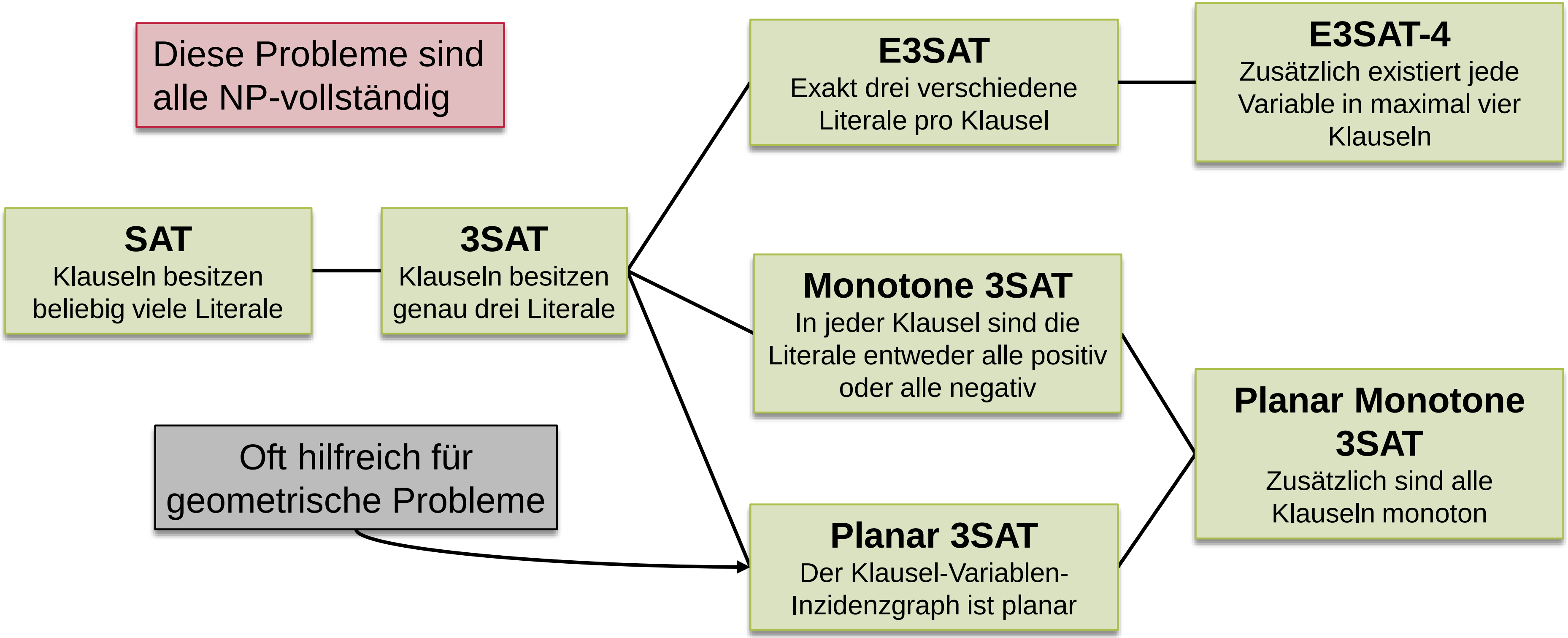
- Erzeuge Variablen $x_{1,i}, \dots, x_{n_{i},i}$.
- Ersetze das c-te Literale von x_i mit einem Literal der Variablen $x_{c,i}$.
- Füge Klauseln der folgenden Form hinzu.

$$(x_{1,i}, \bar{x}_{2,i}) \wedge (x_{2,i}, \bar{x}_{3,i}) \wedge \cdots \wedge (x_{n_i,i}, \bar{x}_{1,i})$$

Beobachtungen:

- Die neuen Variablen tauchen genau drei Mal auf.
- Ist eine Variable auf *true* gesetzt, sind alle *true*. (Repräsentieren die gleiche Variable!)
- Die alte Variable taucht nicht mehr auf.

Weitere 3SAT-Varianten



Noch mehr 3SAT Varianten

3SAT
Klauseln besitzen genau drei Literale

1-in-3SAT
In jeder Klausel darf maximal ein Literal *true* sein

Positive 1-in-3SAT
Zusätzlich sind alle Literale positive

Positive Planar 1-in-3SAT
Zusätzlich ist der Klausel-Variablen-Graph planar

NAE 3SAT
In jeder Klausel dürfen nicht alle Literale den gleichen Wert besitzen

NAE = Not-All-Equal

Positive NAE 3SAT
Zusätzlich sind alle Literale positive

Diese Probleme sind alle NP-vollständig

Klausur

Klausurinfos

- **Klausur:** Die Klausur findet am 09.08.2019 von 8:30 Uhr bis 10:30 Uhr statt. Bitte seid 15 Minuten vorher anwesend. Raumaufteilung wird einen Tag vor der Klausur bekanntgegeben. Mitzubringen sind:
  - Ein gültiger Studentenausweis (plus gültiger Lichtbildausweis, falls kein Foto auf dem Studentenausweis vorhanden ist)
  - Ein dokumentenechter Stift (kein Rot!)
  - Ein Wörterbuch, falls nötig (auf Papier, ohne Notizen)

  Andere Hilfsmittel sind nicht erlaubt; eigenes Papier wird nicht benötigt.

Was man wissen sollte:

- Alles aus der Vorlesung
- Übersicht über die Übungen/Hausaufgaben