

Übung 13: Optimierung

Einführung in die Rechnerarchitektur

Niklas Ladurner

School of Computation, Information and Technology
Technische Universität München

23. Januar 2026



TUM Uhrenturm

Keine Garantie für die Richtigkeit der Tutorfolien.
Bei Unklarheiten/Unstimmigkeiten haben VL/ZÜ-Folien recht!

- Logiksynthese: Realisierung boolesche Funktion in Hardware
- naive Synthese (direkte Übertragung der Wahrheitstabelle) nicht skalierbar
- verschiedene Verfahren zur Optimierung und Reduktion von Funktionen auf ihr Minimalpolynom \rightarrow optimale Schaltung

Minimalpolynom

Ein Polynom p ist Minimalpolynom einer booleschen Funktion f , falls $\psi(p) \equiv f$ (d.h. p eine Formel für f ist) und es keine weitere Vereinfachungen gibt.

Karnaugh-Veitch-Diagramme¹

- rechteckiges Schema, in dem alle Literalkombinationen (positiv und negativ) vorkommen
- nebeneinander liegende Zeilen/Spalten dürfen sich immer nur in 1 Bit unterscheiden (Gray-Code)!
- Zusammenfassen von Einsen in 2^n -Blöcken, Don't Care können als 0 oder 1 gewählt werden.
- jedes maximal große Päckchen steht für einen Primimplikanten der Funktion → minimale Mengen von Päckchen die alle 1en abdeckt ergibt ein Minimalpolynom

cd \ ab	00	01	11	10
00	-	-	0	1
01	1	1	0	0
11	0	1	1	1
10	1	0	1	1

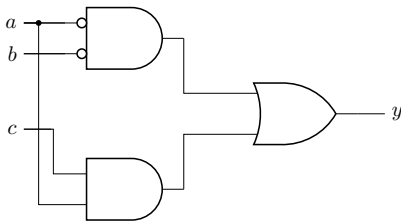
$$f = \overline{a}\overline{c} + \overline{b}\overline{d} + ac + b\overline{c}d$$

¹ Oft auch als K-Maps bezeichnet

Logik-Hazards

- Glitches (kurzzeitige Signaländerung) am Ausgang, die durch Gatterlaufzeit auftritt
- kann zu Problemen führen (z.B. Oszillation)
- in K-Map: Zwei nebeneinanderliegende Einsen, die nicht durch ein gemeinsames Päckchen abgedeckt sind
- Lösung: Hinzufügen von zusätzlichem Päckchen → kein Minimalpolynom!

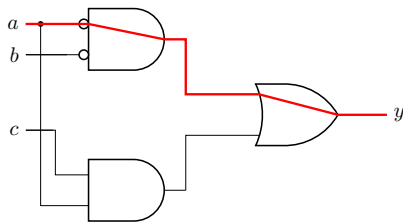
$\backslash b_c$	00	01	11	10
0	1	1	0	0
1	0	1	1	0



Logik-Hazards

- Glitches (kurzzeitige Signaländerung) am Ausgang, die durch Gatterlaufzeit auftritt
- kann zu Problemen führen (z.B. Oszillation)
- in K-Map: Zwei nebeneinanderliegende Einsen, die nicht durch ein gemeinsames Päckchen abgedeckt sind
- Lösung: Hinzufügen von zusätzlichem Päckchen → kein Minimalpolynom!

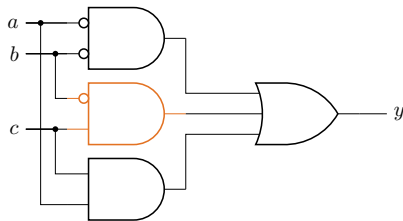
$\backslash bc$	00	01	11	10
0	1	1	0	0
1	0	1	1	0



Logik-Hazards

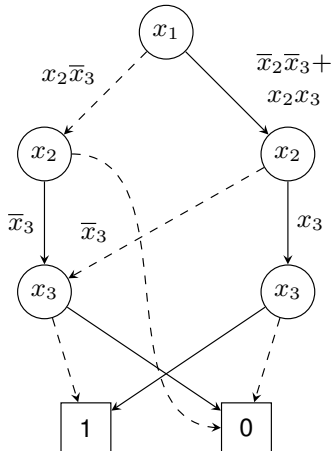
- Glitches (kurzzeitige Signaländerung) am Ausgang, die durch Gatterlaufzeit auftritt
- kann zu Problemen führen (z.B. Oszillation)
- in K-Map: Zwei nebeneinanderliegende Einsen, die nicht durch ein gemeinsames Päckchen abgedeckt sind
- Lösung: Hinzufügen von zusätzlichem Päckchen → kein Minimalpolynom!

$\backslash b_c$	00	01	11	10
0	1	1	0	0
1	0	1	1	0



- Darstellung einer booleschen Funktion als gerichteter azyklischer Graph (DAG)
- Knoten repräsentieren Teilfunktionen, 2 ausgehende Kanten: 0 (low), 1 (high)
- Aufbau bspw. mittels Shannon-Zerlegung:
 $f(x_0, x_1) \rightarrow f_{x_0=0}(x_1), f_{x_0=1}(x_1)$
- **ROBDDs sind kanonisch (eindeutig)**

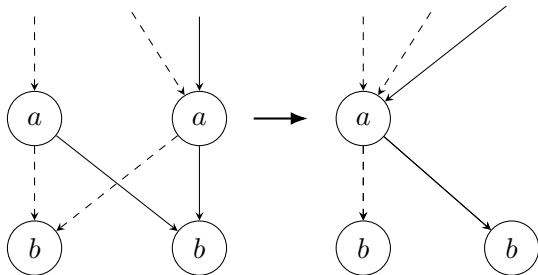
$$\bar{x}_1 x_2 \bar{x}_3 + x_1 \bar{x}_2 \bar{x}_3 + x_1 x_2 x_3$$



Variablenordnung: $x_1 \prec x_2 \prec x_3$

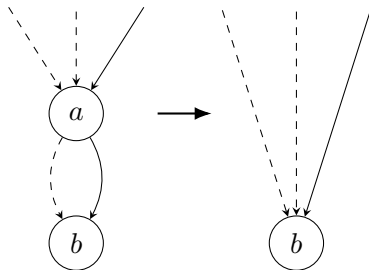
I-Reduktion

Zusammenführung isomorpher Knoten

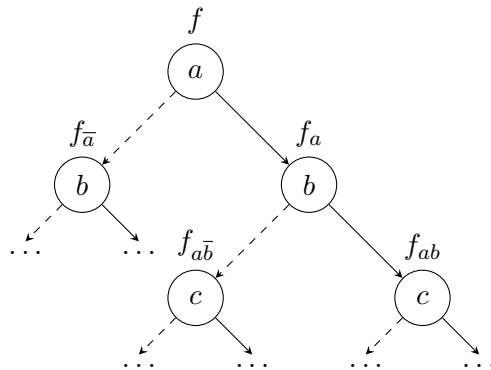


S-Reduktion

Entfernen von überflüssigen Knoten



- Ternärer Operator $\text{ITE}(a, b, c) \equiv ab + \bar{a}c$;
falls a , dann b , sonst c
- alle booleschen Operatoren können als
ITE dargestellt werden, z.B.
 1. $\text{NOT}(f) \equiv \text{ITE}(f, 0, 1)$
 2. $\text{OR}(f, g) \equiv \text{ITE}(f, 1, g)$
 3. $\text{AND}(f, g) \equiv \text{ITE}(f, g, 0)$
- Zusammenhang mit BDDs: High-Kind im
True-Fall, Low-Kind im False-Fall

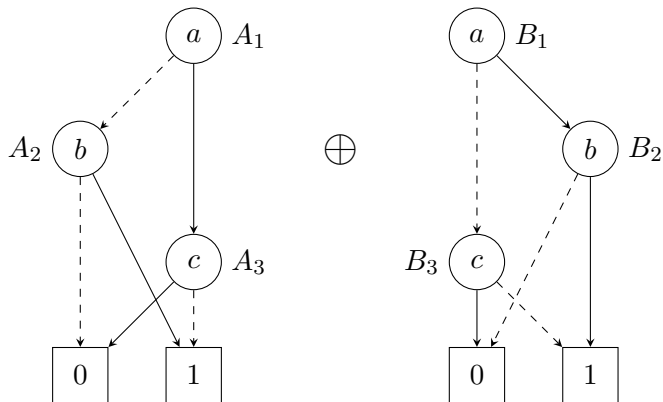


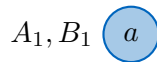
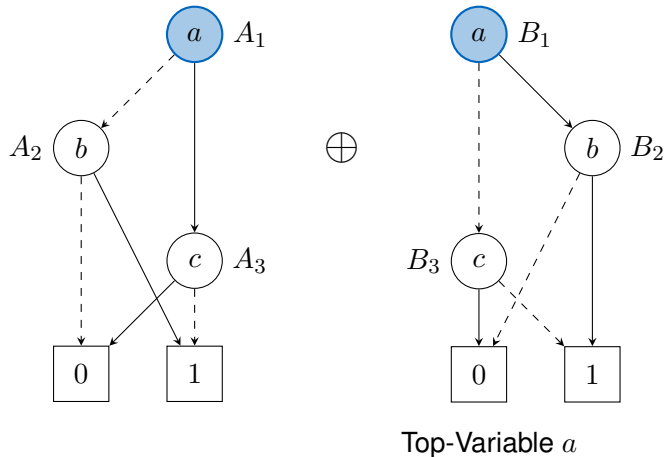
$$\begin{aligned}
 f &\equiv \text{ITE}(a, f_a, f_{\bar{a}}) \\
 &\equiv \text{ITE}(a, \text{ITE}(b, f_{ab}, f_{a\bar{b}}), f_{\bar{a}})
 \end{aligned}$$

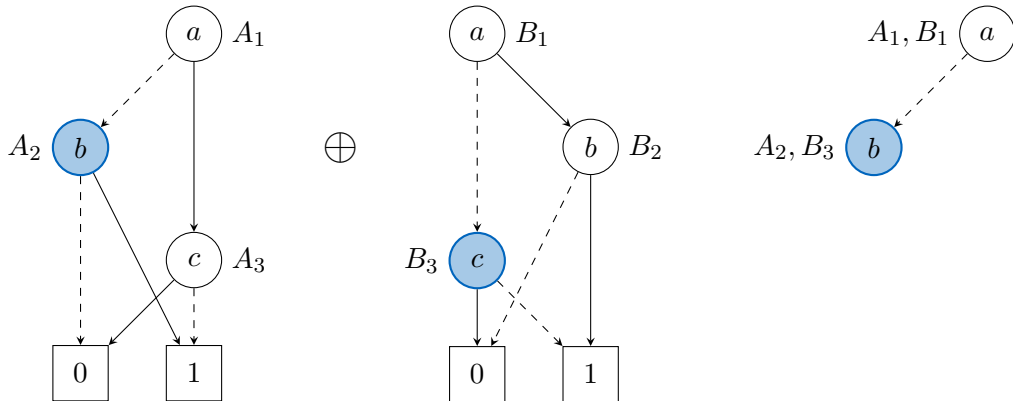
BDDs: ITE-Algorithmus

Am Beispiel der Verknüpfung von zwei BDDs A, B mittels \oplus :

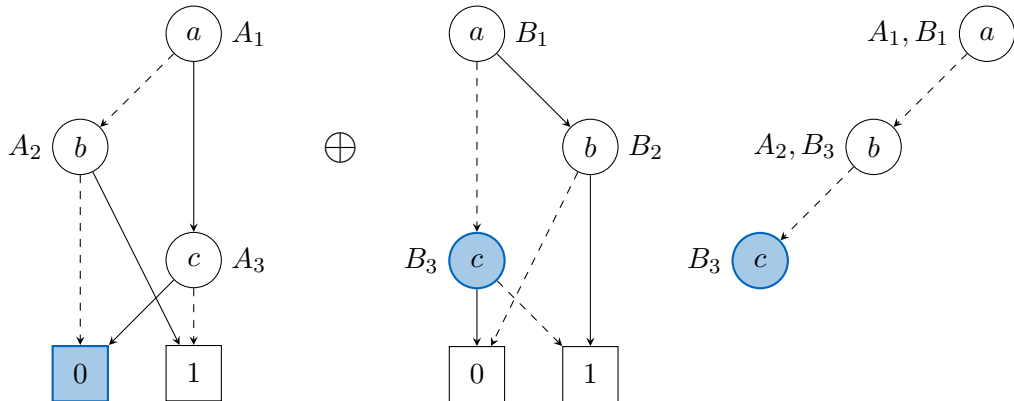
1. Bezeichne alle Knoten mit einem eindeutigen Bezeichner
2. Erzeuge im Ergebnis-BDD einen Knoten $A_0 \oplus B_0$
3. Solange noch nicht alle Knoten besucht wurden:
 - (a) Falls Terminalfall (d.h. A_i und B_i sind Terminalknoten), Rückgabewert berechnen und im Ergebnis-BDD an der aktuellen Stelle einen neuen Terminalknoten $A_i \oplus B_i$ erzeugen
 - (b) Top-Variable t anhand der Variablenordnung bestimmen
 - (c) Für alle mit t bezeichneten Knoten: Abstieg ins Low-Kind, erzeuge im Ergebnis-BDD an der aktuellen Stelle den Kindknoten $A_{i,t=0} \oplus B_{i,t=0}$
 - (d) Für alle mit t bezeichneten Knoten: Abstieg ins High-Kind, erzeuge im Ergebnis-BDD an der aktuellen Stelle den Kindknoten $A_{i,t=1} \oplus B_{i,t=1}$
4. Eventuell I- und S-Reduktionen anwenden



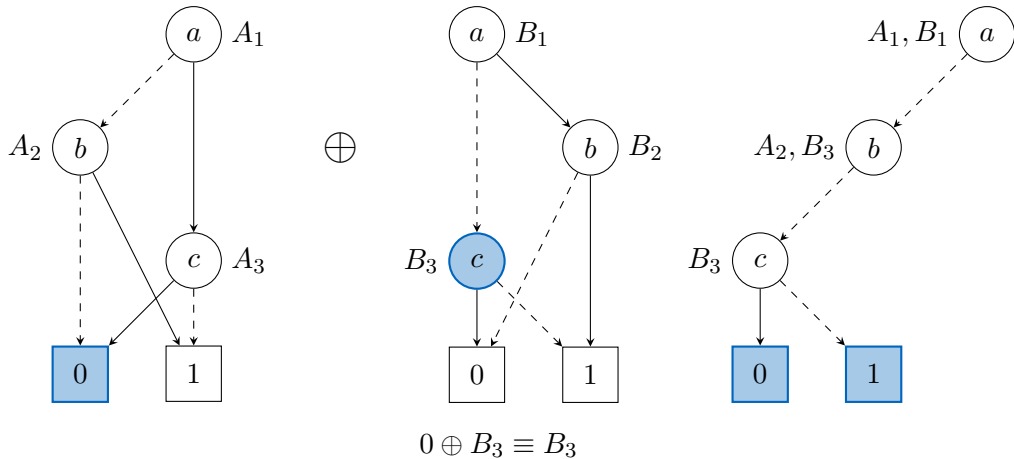


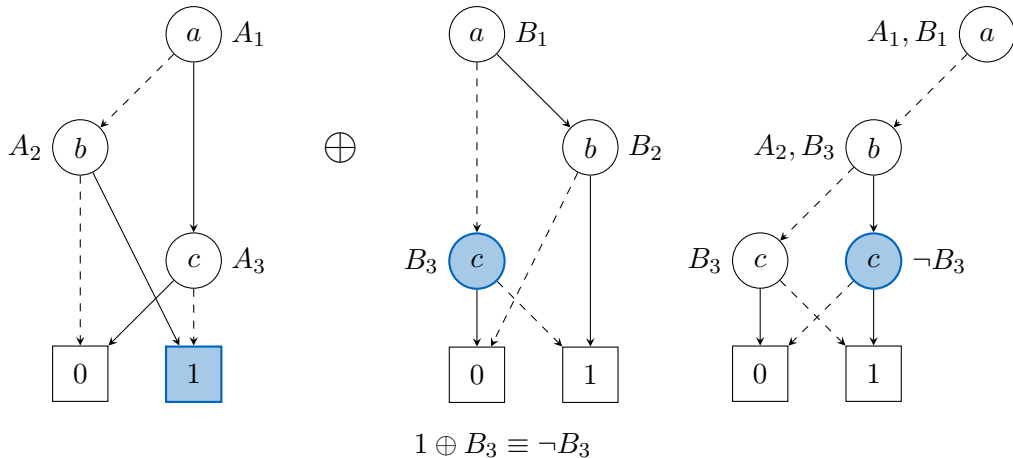


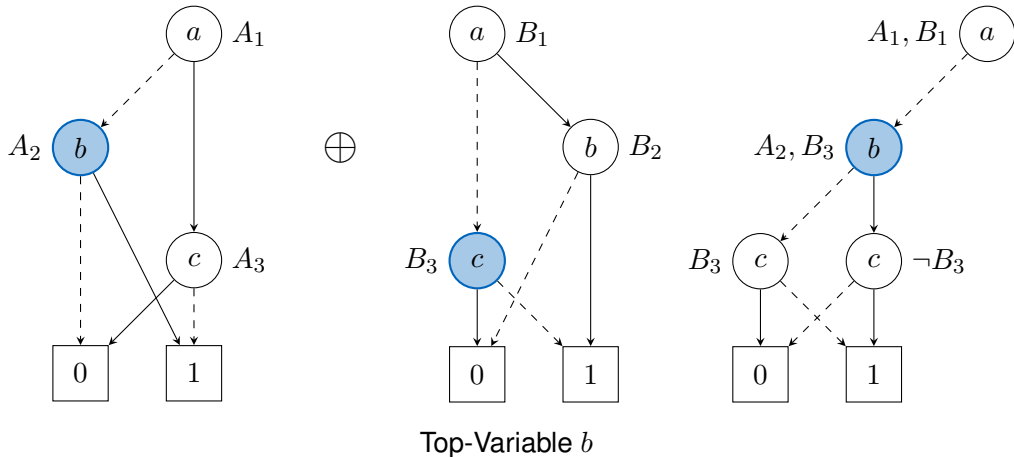
Top-Variable b , Abstieg ins Low-Kind

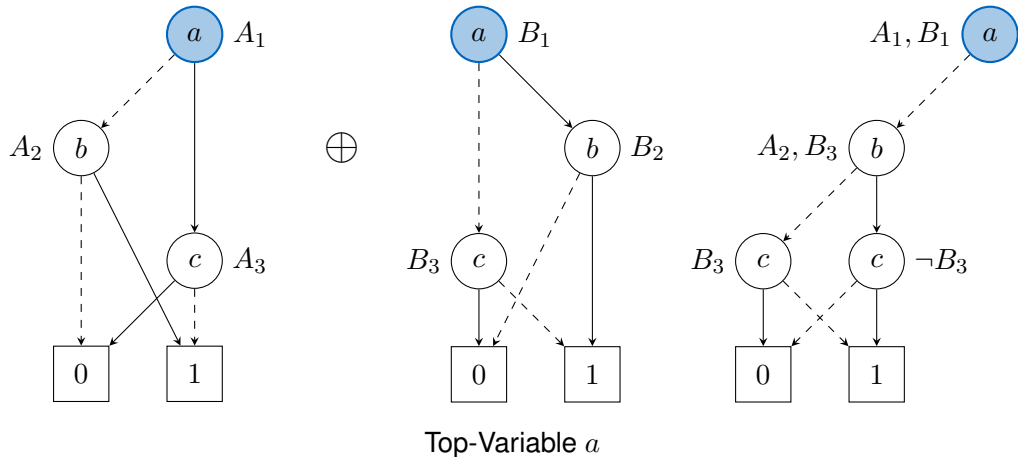


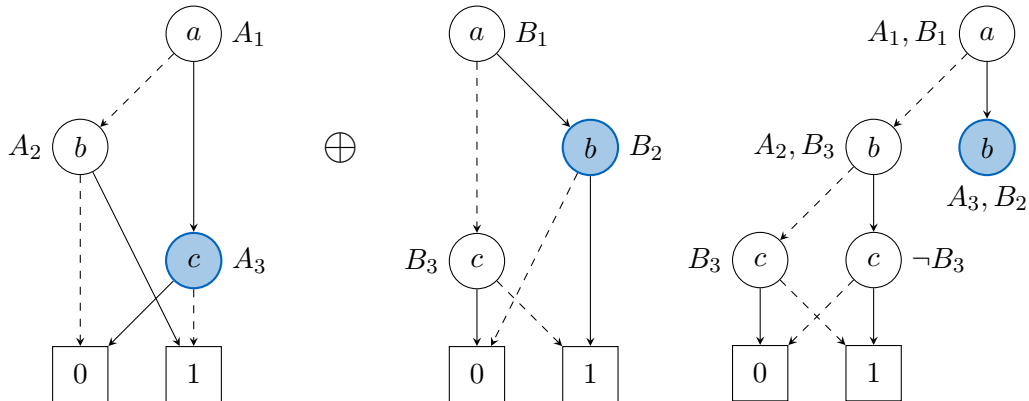
Top-Variable c , Abstieg ins Low-Kind



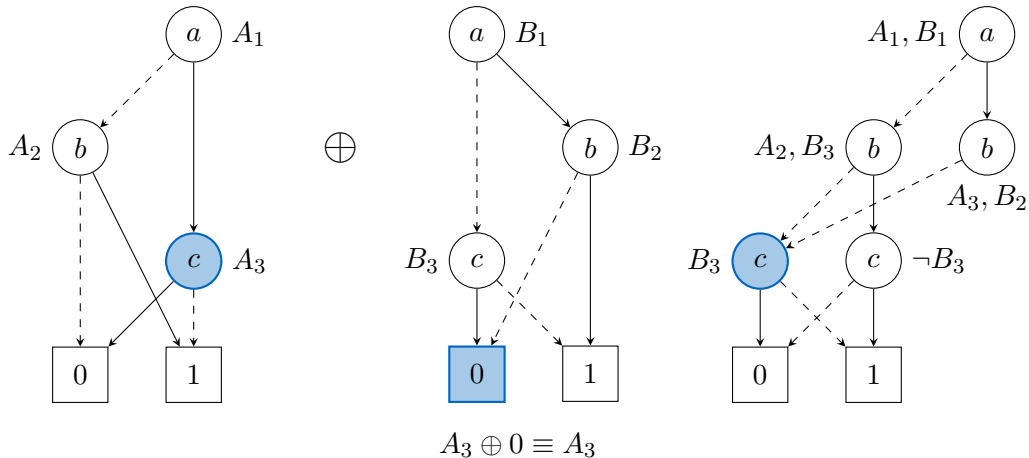


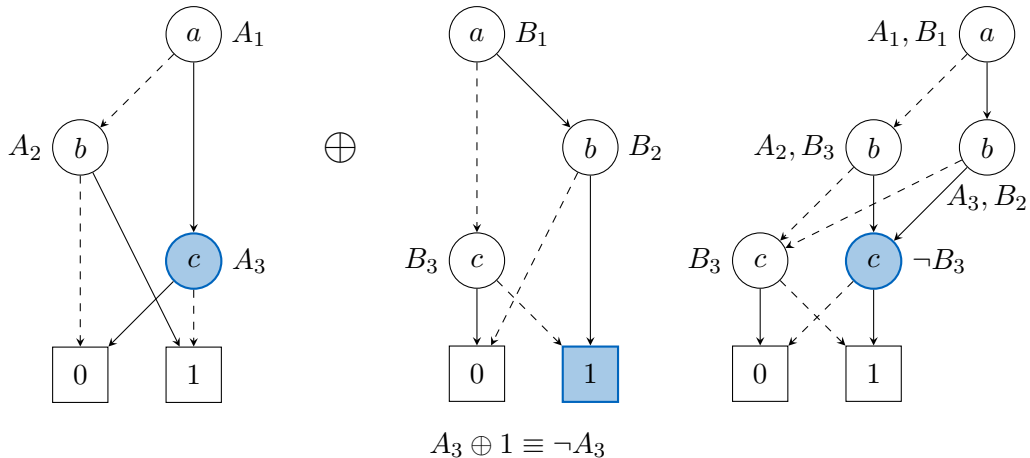


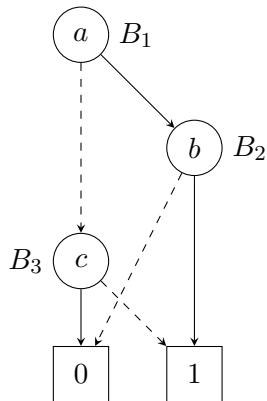
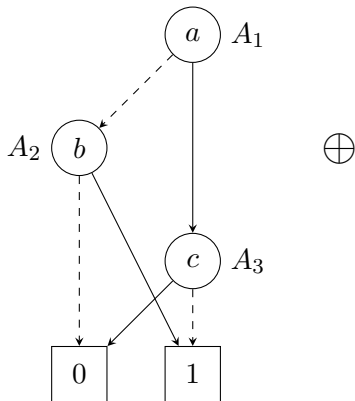




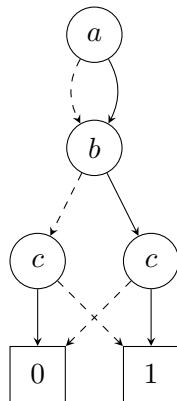
Top-Variable b , Abstieg ins High-Kind

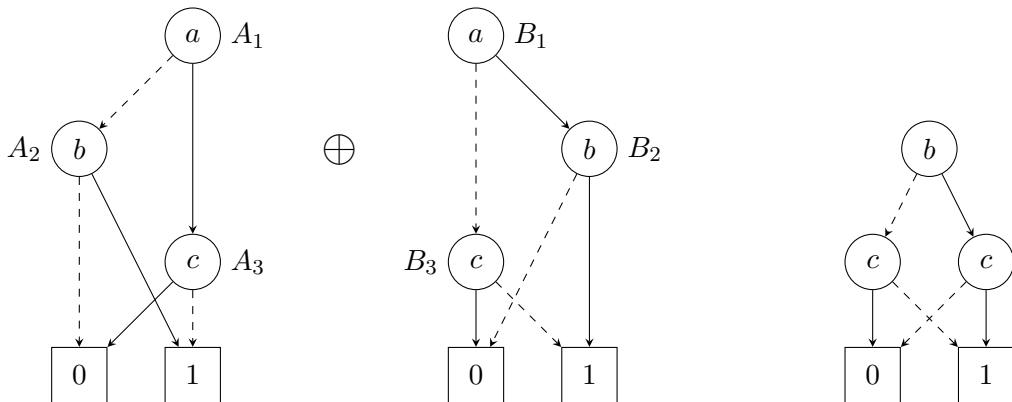






I-Reduktion





S-Reduktion

Fragen?

- Zulip: „ERA Tutorium – Mi-1600-3“ bzw. „ERA Tutorium – Fr-1500-1“
- ERA-Moodle-Kurs
- ERA-Artemis-Kurs
- Wikipedia zu BDDs

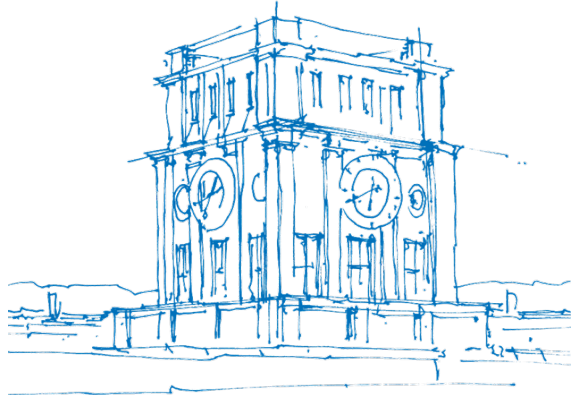
Übung 13: Optimierung

Einführung in die Rechnerarchitektur

Niklas Ladurner

School of Computation, Information and Technology
Technische Universität München

23. Januar 2026



TUM Uhrenturm