Automatisches Beweisen
Reduced Ordered BDDs

Christoph Zengler

Universität Tübingen

07.06.2010

Übersicht — 1

BDDs — Binary Decision Diagrams

- Gerichteter azyklischer Graph $G = (V, E)$ mit Wurzel $r \in V$
- Augezeichnete Endknoten 0 und 1 mit Ausgangsgrad 0
- Variablenknoten $U = V \setminus \{0, 1\}$ mit Ausgangsgrad 2
- Beschriftung jedes Variablenknotens $u \in U$ mit var(u)
- Ausgangsgrad 2 für alle Variablenknoten
  - low(u) — Variable var(u) wird auf 0 gesetzt
  - high(u) — Variable var(u) wird auf 1 gesetzt

OBDD — Ordered BDDs

- Auf allen Pfaden durch den Graph G gilt eine gegebene lineare Ordnung $x_1 < x_2 < \cdots < x_n$ auf den Variablen var(u) der Variablenknoten $u \in U$.

Übersicht — 2

ROBDDs — Reduced OBDDs

Ein OBDD ist ein ROBDD, wenn folgende beiden Eigenschaften gelten:

- **Eindeutigkeit:** keine zwei Variablenknoten $u \neq v$ haben den selben Variablennamen und gleiche low und high Kanten, d.h.

$$\text{var}(u) = \text{var}(v) \wedge \text{low}(u) = \text{low}(v) \wedge \text{high}(u) = \text{high}(v) \longrightarrow u = v$$

- **Nicht-Redundanz:** kein Variablenknoten u hat identische low und high Kanten, d.h.

$$\text{low}(u) \neq \text{high}(u)$$

Lemma (Canonicity Lemma)

Für jede Boolesche Funktion $f : \mathbb{B}^n \to \mathbb{B}$ gibt es genau ein ROBDD u mit Variablenordnung $x_1 < x_2 < \cdots < x_n$, so dass $f^u = f(x_1, \ldots, x_n)$.

Konstrukton eines ROBDDs

Zwei Möglichkeiten:

- Generieren eines BDDs und anschließende Vereinfachung
  - **Vorteil:** Einfache Algorithmen (BDD Algorithmen bereits bekannt)
  - **Nachteil:** Knoten werden weiterhin doppelt generiert und erst später wieder gelöscht $\rightarrow$ erhöhter Speicheraufwand
- Vereinfachen des BDDs während des Generierens
  - **Vorteil:** Platzverbrauch wird schon während des Generierens minimal gehalten
  - **Nachteil:** Kompliziertere Datenstrukturen

Grundidee

Aus vielen anderen Bereichen der Informatik bekannt: Doppelte Arbeit vermeiden! Mittel: Caching!

Datenstrukturen zur ROBDD Generation

- Knoten werden als Zahlen $0, 1, 2, \ldots$ repräsentiert. 0 und 1 sind reserviert für die Terminalknoten 0 und 1.
- Die Variablen in der Ordnung $x_1, x_2, \ldots, x_n$ werden durch ihre Indizes $1, 2, \ldots, n$ repräsentiert.
- Zwei Tabellen: T zur Speicherung des ROBDDs, H als „Inverses“ zu T wird während der Generation benötigt.

Tabelle T

$T : u \to (i, l, h)$. Zu einem Knoten u werden seine Variablenbeschriftung $\text{var}(u)$, und die Zielknoten von $\text{low}(u)$ und $\text{high}(u)$ gespeichert.

Tabelle H

$H : (i, l, h) \to u$. Zu einem Tripel aus Variablenname $\text{var}(u)$, Zielknoten von $\text{low}(u)$ und Zielknoten von $\text{high}(u)$ wird die zugehörige Variable u gespeichert.

$$T(u) = (i, l, h) \qquad \longleftrightarrow \qquad H(i, l, h) = u$$

Beispiel für Tabelle T

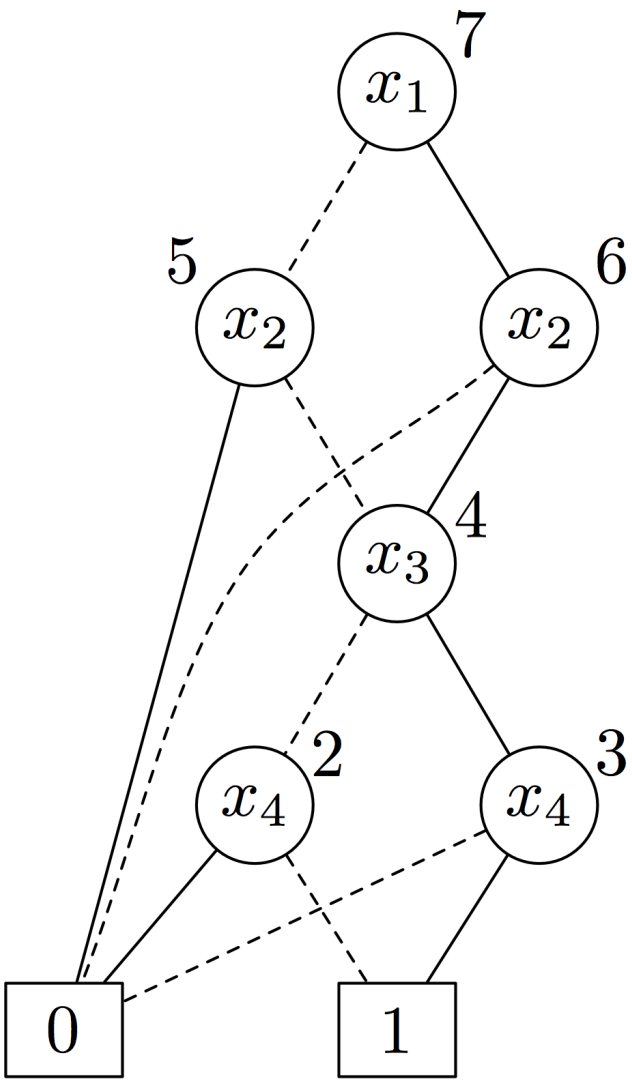


Tabelle $T : u \to (i, l, h)$:

u	var(u)	low(u)	high(u)
0	5		
1	5		
2	4	1	0
3	4	0	1
4	3	2	3
5	2	4	0
6	2	0	4
7	1	5	6

Operationen auf den Tabellen

Operationen auf T

- init(T), initialisiere T mit Terminalknoten 0 und 1
- $u \leftarrow$ add(T, i, l, h), erzeuge neuen Knoten mit Attributen (i, l, h) in T
- var(u), low(u), high(u), Zugriff auf die Attribute von u in T

Operationen auf H

- init(H), initialisiere leere Tabelle H
- $b \leftarrow$ member(H, i, l, h), überprüfe, ob Eintrag (i, l, h) in H vorhanden ist
- $u \leftarrow$ lookup(H, i, l, h), finde den passenden Knoten u zum Eintrag (i, l, h) in H
- insert(H, i, l, h, u), füge neuen Eintrag $(i, l, h) \rightarrow u$ in H hinzu

Operationen können effizient in $O(1)$ ausgeführt werden

Erzeugen von neuen Knoten

Algorithmus Mk(T, H, i, l, h) erzeugt neuen Knoten

Neuer Knoten wird nur erzeugt, wenn er noch nicht in H ist!

Algorithmus Mk

Input: Tabellen T und H, Variablenindex i, Zielknoten von high und low l und h
Output: Variablenknoten u

if $l = h$ **then**
  return l

if member(H, i, l, h) **then**
  return lookup(H, i, l, h)

$u \leftarrow$ add(T, i, l, h)
insert(H, i, l, h, u)
return u

Erzeugen eines ROBDD — 1

Für eine Booleschen Ausdruck t generieren wir das zugehörige ROBDD mit Aufruf des Algorithmus BUILD(T, H, t).

Idee von BUILD

- Fixiere Variablenordnung $x_1, x_2, \ldots, x_n$
- Shannon Expansion:

$$t = (x \to t|_{x=1}) \wedge (\neg x \to t|_{x=0})$$

- D.h. Erzeuge Knoten für t mit MK, nachdem die Knoten für $t|_{x=0}$ und $t|_{x=1}$ zuvor rekursiv erzeugt wurden
- Hilfsfunktion BUILD'(T, H, t, i) konstruiert ein ROBDD für t mit Variablen in $\{x_i, x_{i+1}, \ldots, x_n\}$
- BUILD(T, H, t) ruft dann einfach BUILD'$(T, H, t, 1)$ auf

Erzeugen eines ROBDD — 2

Algorithmus BUILD'

Input: Tabellen T und H, Boolescher Ausdruck t, Variablenbeschränkung i
Output: Variablenknoten u

if $i > |\text{vars}(t)|$ **then**
  if $\beta(t) = 0$ **then**
    return *0*
  else
    return *1*
else
  $v_0 \leftarrow$ BUILD'$(T, H, t|_{x_i=0}, i+1)$
  $v_1 \leftarrow$ BUILD'$(T, H, t|_{x_i=1}, i+1)$
  return MK(T, H, i, v_0, v_1)

Beispiel zur ROBDD Erzeugung

Funktion $t = (x_1 \longleftrightarrow x_2) \vee x_3$

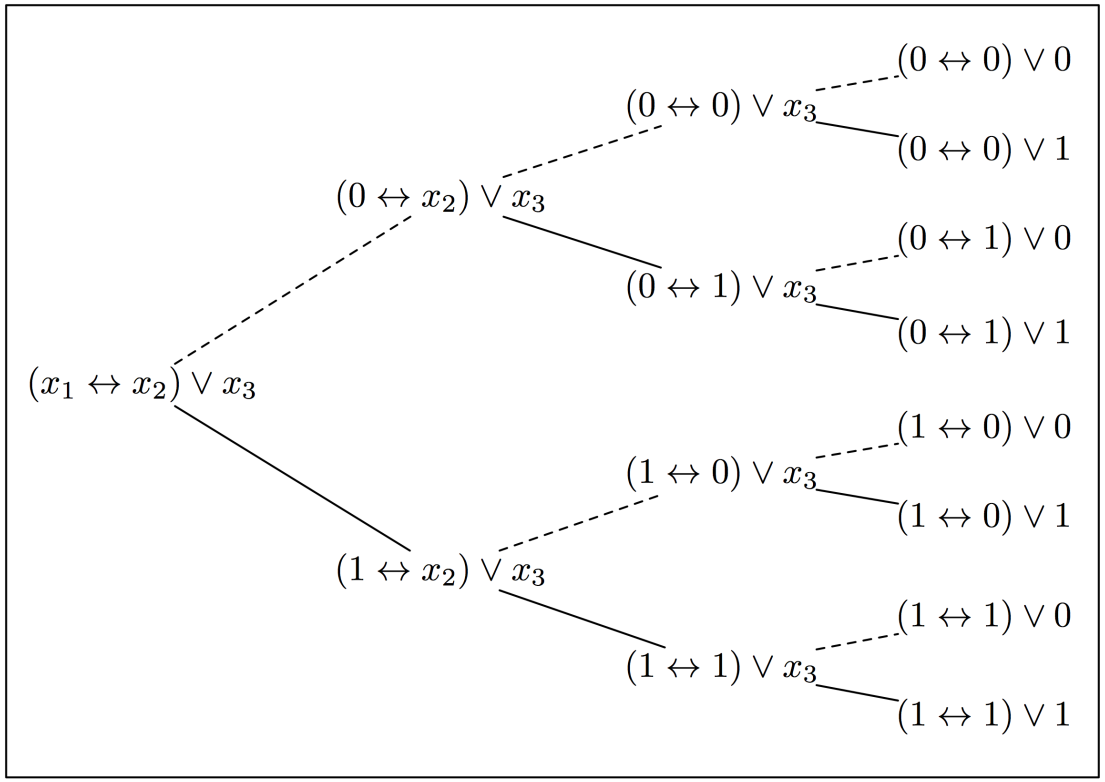


Tabelle T, H

u	var(u)	low(u)	high(u)
0	4		
1	4		

Beispiel zur ROBDD Erzeugung

Funktion $t = (x_1 \longleftrightarrow x_2) \vee x_3$

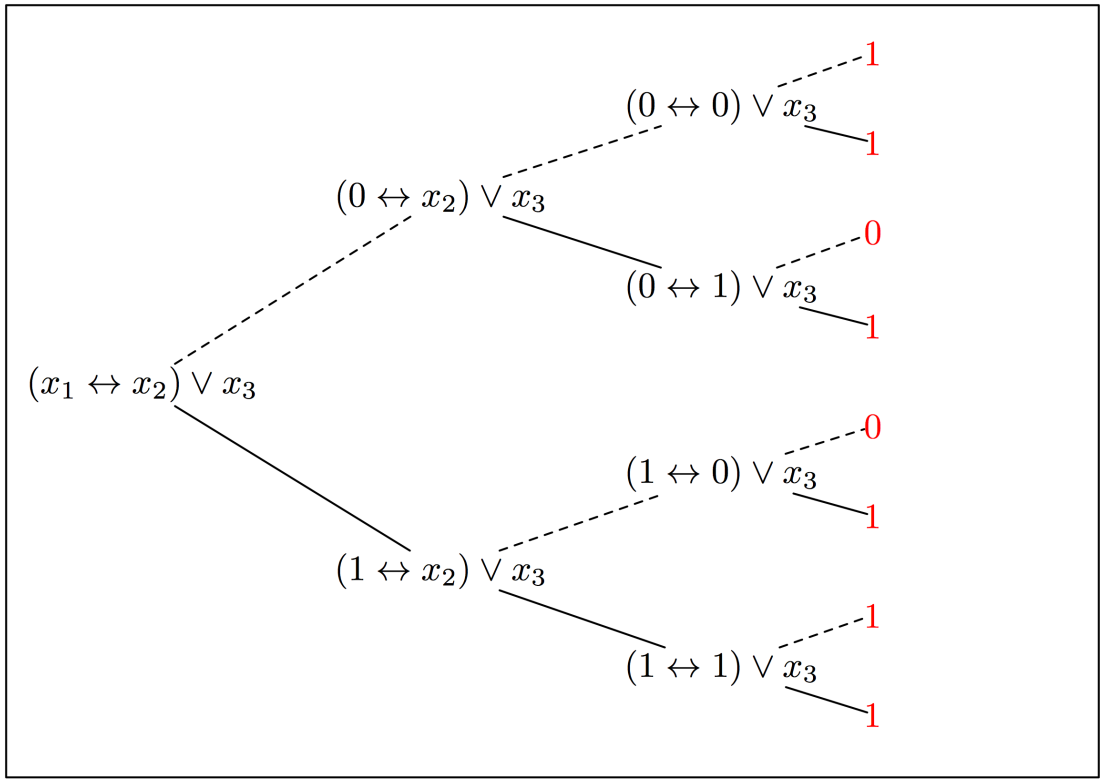


Tabelle T, H

u	var(u)	low(u)	high(u)
0	4		
1	4		

Beispiel zur ROBDD Erzeugung

Funktion $t = (x_1 \longleftrightarrow x_2) \vee x_3$

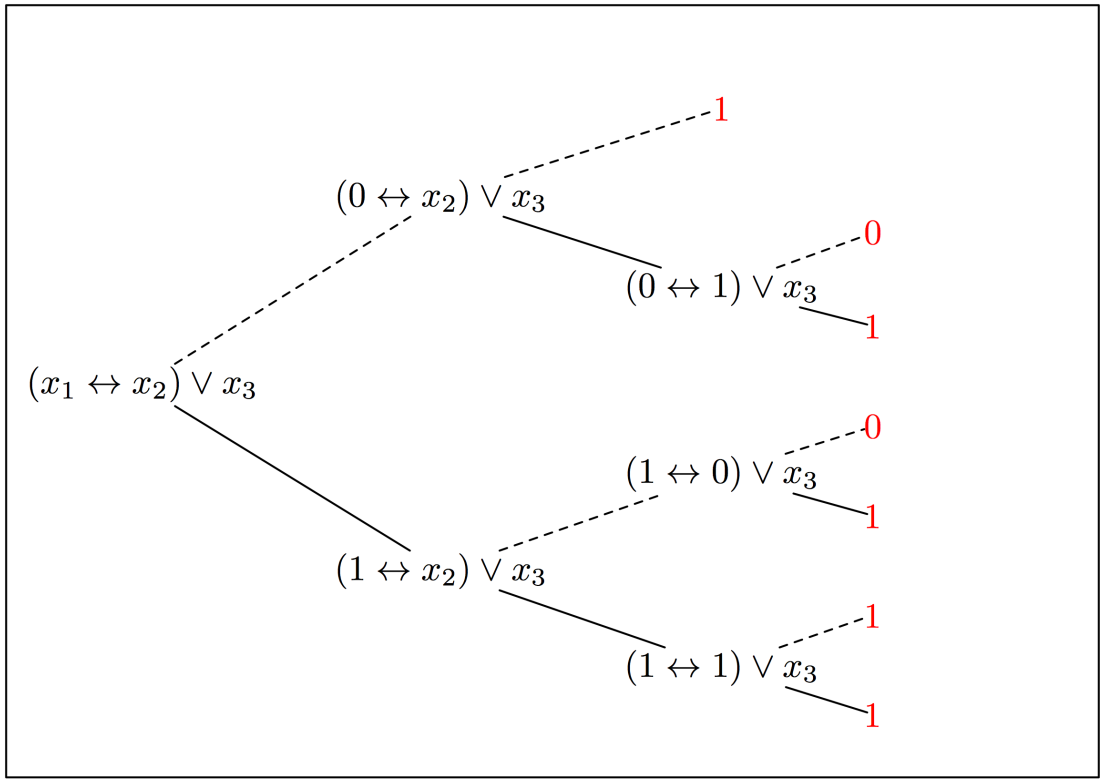


Tabelle T, H

u	var(u)	low(u)	high(u)
0	4		
1	4		

Beispiel zur ROBDD Erzeugung

Funktion $t = (x_1 \longleftrightarrow x_2) \vee x_3$

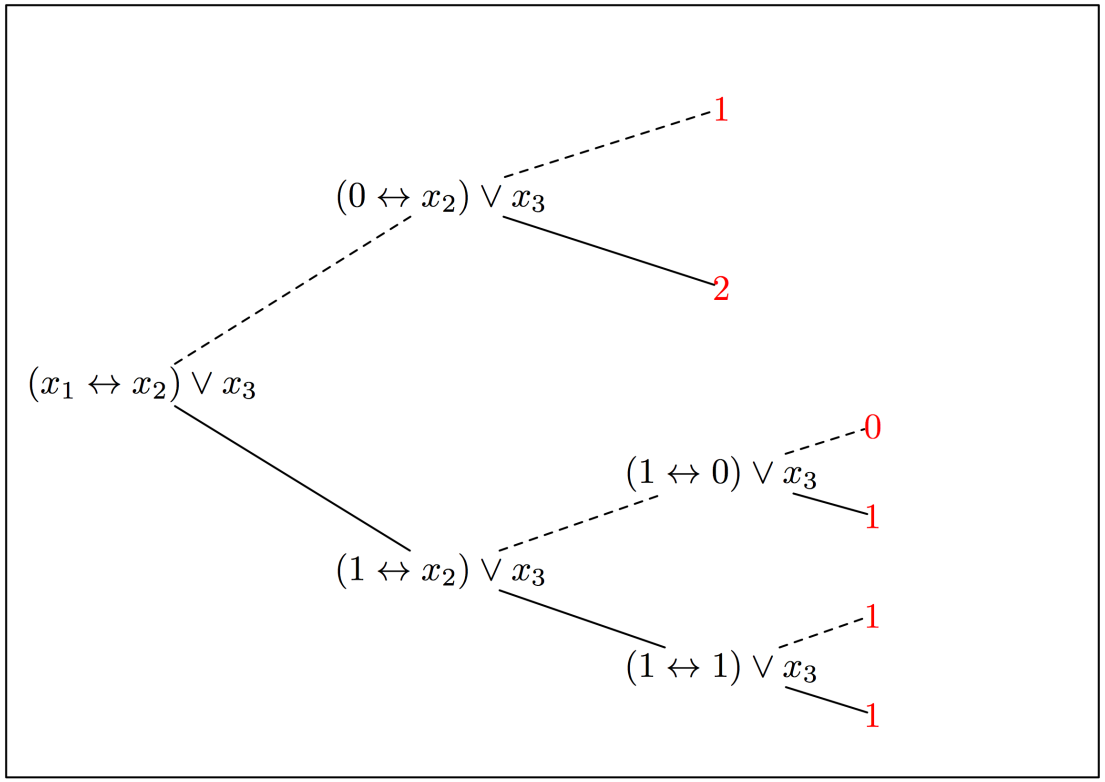
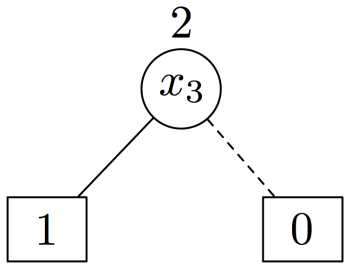


Tabelle T, H

u	var(u)	low(u)	high(u)
0	4		
1	4		
2	3	0	1



Beispiel zur ROBDD Erzeugung

Funktion $t = (x_1 \longleftrightarrow x_2) \vee x_3$

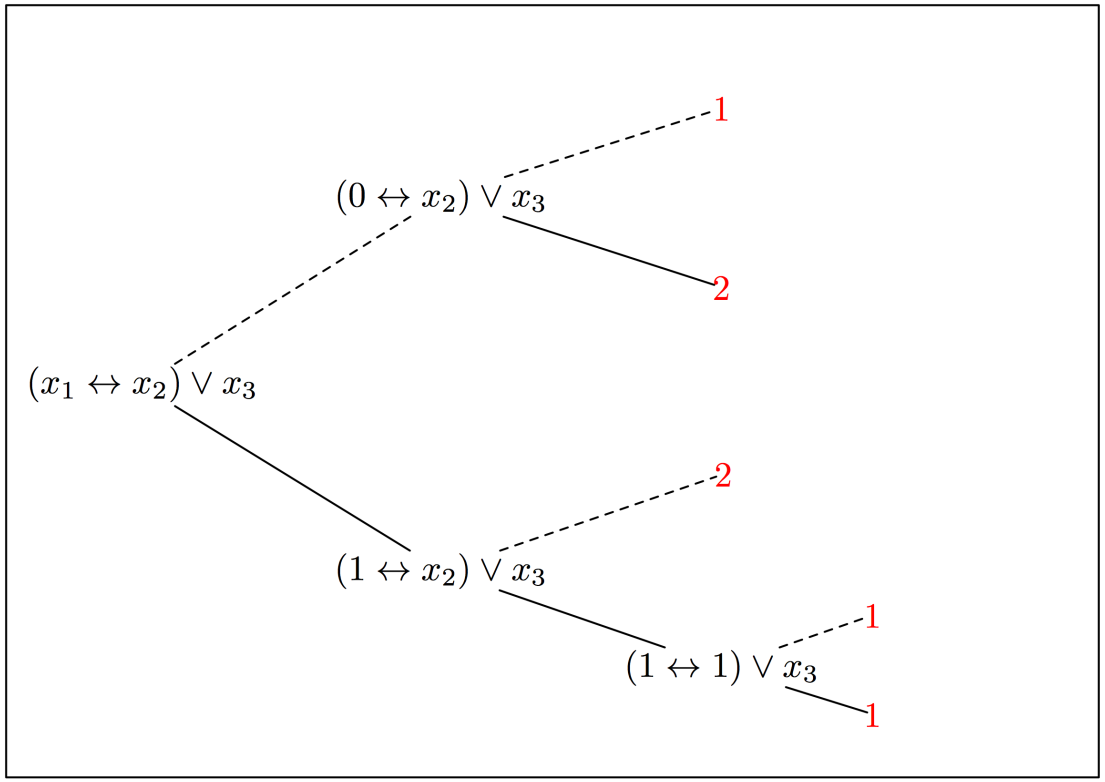
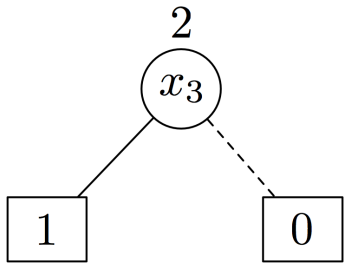


Tabelle T, H

u	var(u)	low(u)	high(u)
0	4		
1	4		
2	3	0	1



Beispiel zur ROBDD Erzeugung

Funktion $t = (x_1 \longleftrightarrow x_2) \vee x_3$

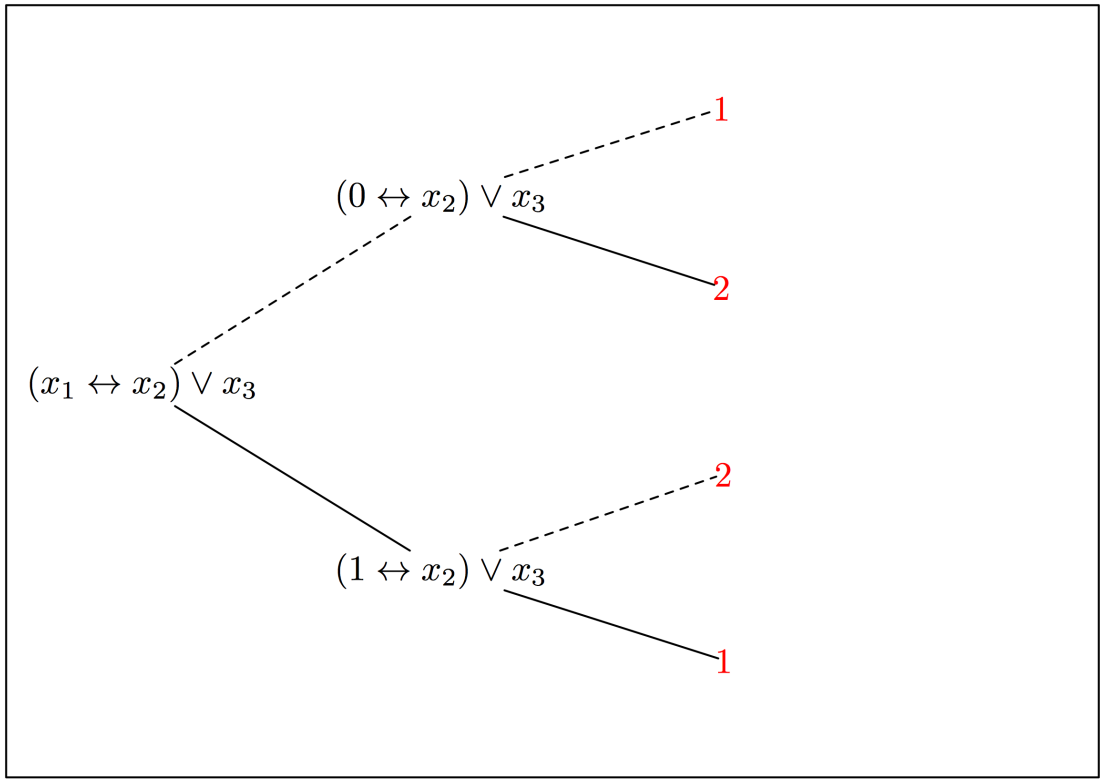
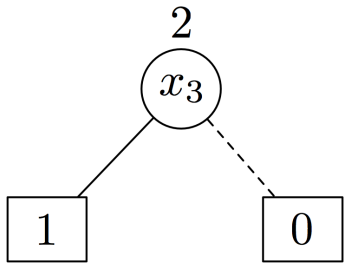


Tabelle T, H

u	var(u)	low(u)	high(u)
0	4		
1	4		
2	3	0	1



Beispiel zur ROBDD Erzeugung

Funktion $t = (x_1 \longleftrightarrow x_2) \vee x_3$

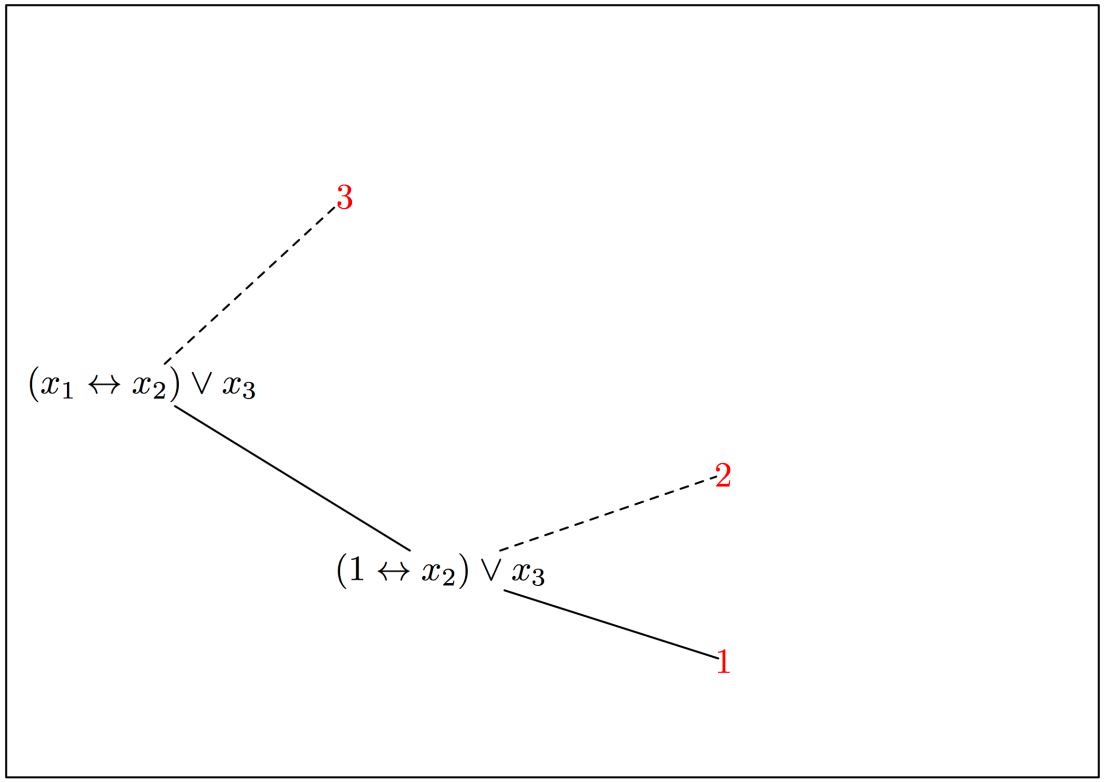
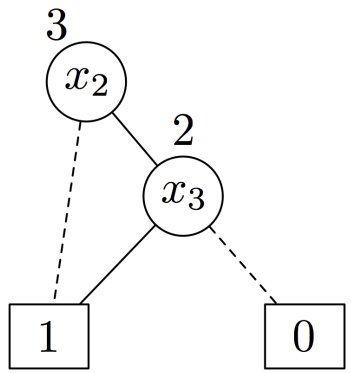


Tabelle T, H

u	var(u)	low(u)	high(u)
0	4		
1	4		
2	3	0	1
3	2	1	2



Beispiel zur ROBDD Erzeugung

Funktion $t = (x_1 \longleftrightarrow x_2) \vee x_3$

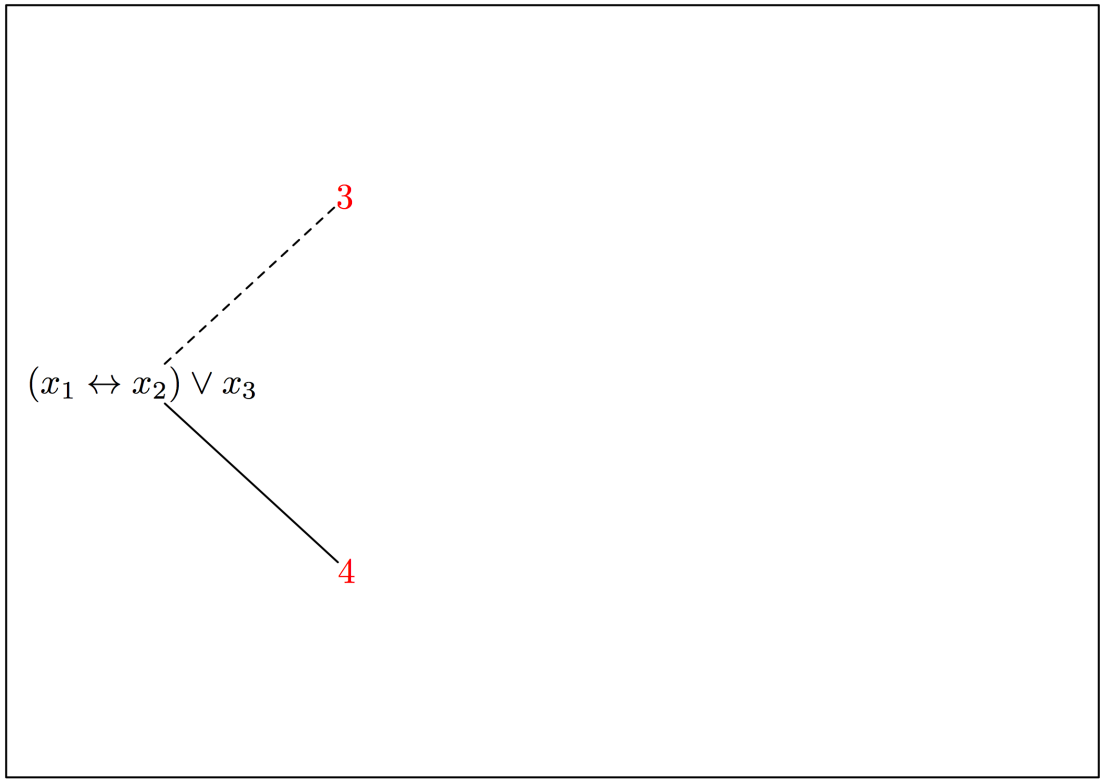
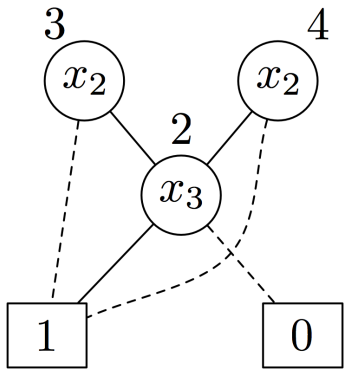


Tabelle T, H

u	var(u)	low(u)	high(u)
0	4		
1	4		
2	3	0	1
3	2	1	2
4	2	2	1



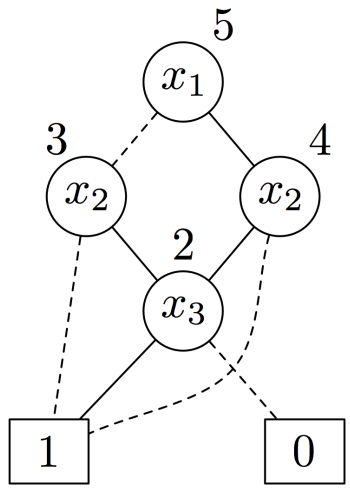
Beispiel zur ROBDD Erzeugung

Funktion $t = (x_1 \longleftrightarrow x_2) \vee x_3$

5

Tabelle T, H

u	var(u)	low(u)	high(u)
0	4		
1	4		
2	3	0	1
3	2	1	2
4	2	2	1
5	1	3	4



Der Apply Algorithmus

Verknüpfen von ROBDDs mit Booleschen Operatoren wie $\wedge, \vee, \rightarrow, \leftrightarrow, \oplus$ mit generischem APPLY Algorithmus möglich.

- APPLY(T, H, G, op, u_1, u_2) bekommt den Booleschen Operator op und zwei ROBDDs u_1 und u_2 übergeben
- APPLY funktioniert wieder über Shannon Expansion
- Beobachtung:

$$[(x \rightarrow t|_{x=1}) \wedge (\neg x \rightarrow t|_{x=0})] \quad op \quad [(x \rightarrow t'|_{x=1}) \wedge (\neg x \rightarrow t'|_{x=0})]$$

ist äquivalent zu

$$(x \rightarrow [t|_{x=1} \ op \ t'|_{x=1}]) \wedge (\neg x \rightarrow [t|_{x=0} \ op \ t'|_{x=0}])$$

- Wir gehen wieder rekursiv duch die ROBDDs

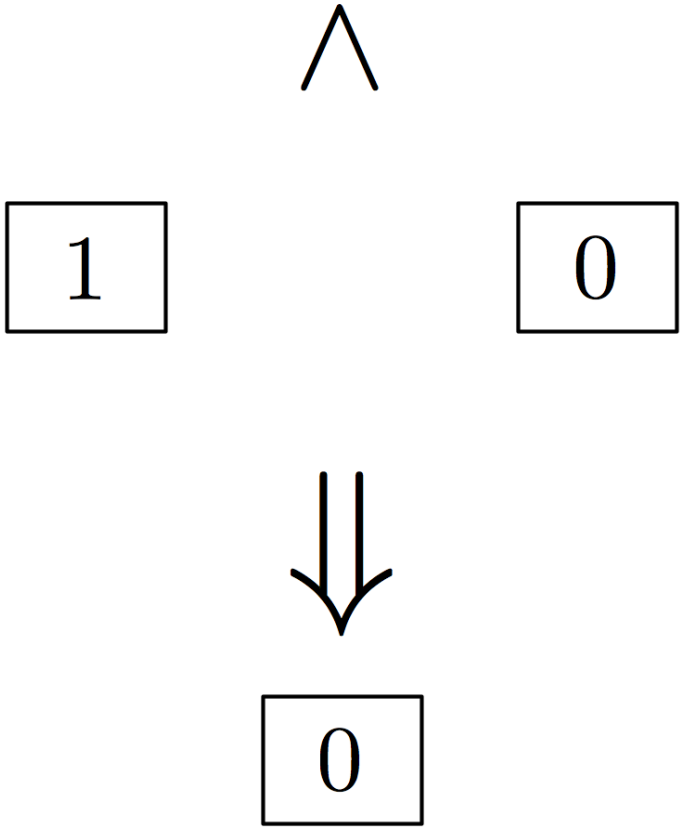
Vorgehen von APPLY

- Dynamisches Programmieren, d.h. Zwischenergebnisse werden in Tabelle G zwischengespeichert
  - Ein Eintrag (i, j) kann entweder leer sein, oder er speichert das Ergebnis der früheren Berechnung APPLY(T, H, G, op, i, j)
- Vier verschiedene Fälle:
  1. Beide Argumente u_1 und u_2 sind Terminalknoten
  2. Beide Argumente u_1 und u_2 sind Variablenknoten und $\text{var}(u_1) = \text{var}(u_2)$
  3. Mindestens ein Knoten u_1 oder u_2 ist ein Variablenknoten und $\text{var}(u_1) < \text{var}(u_2)$
  4. Mindestens ein Knoten u_1 oder u_2 ist ein Variablenknoten und $\text{var}(u_2) < \text{var}(u_1)$

Apply — Fall 1

Beide Argumente u_1 und u_2 sind Terminalknoten.

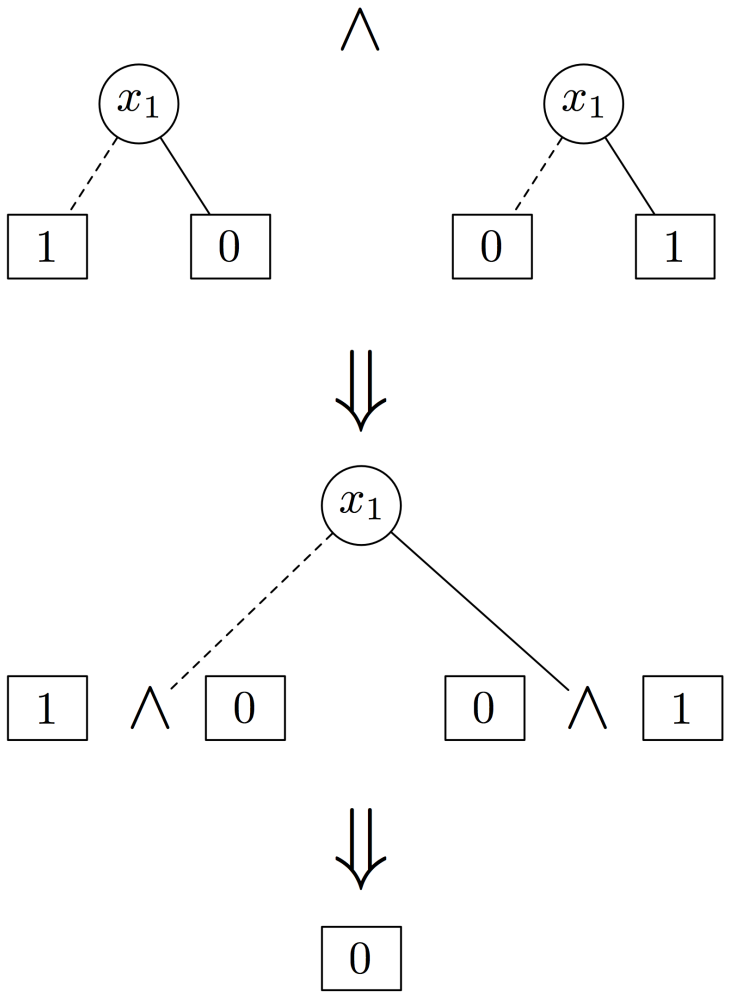
- Erzeuge neuen Terminalknoten mit u_1 *op* u_2



APPLY — Fall 2

Beide Argumente u_1 und u_2 sind Variablenknoten und $\text{var}(u_1) = \text{var}(u_2)$.

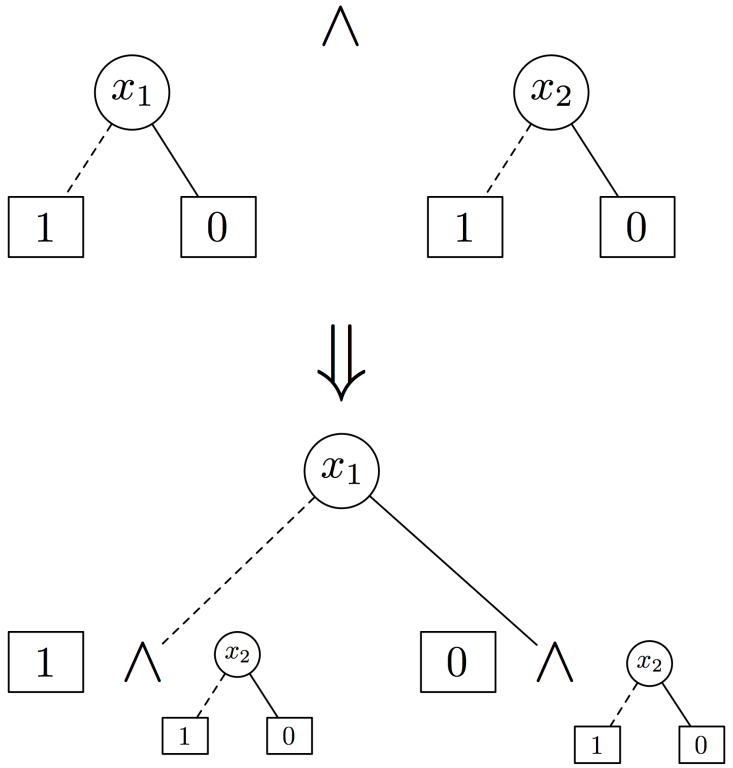
- Kombiniere rekursiv $\text{low}(u_1)$ und $\text{low}(u_2)$ sowie $\text{high}(u_1)$ und $\text{high}(u_2)$



APPLY — Fall 3/4

Mindestens ein Argument ist ein Variablenknoten

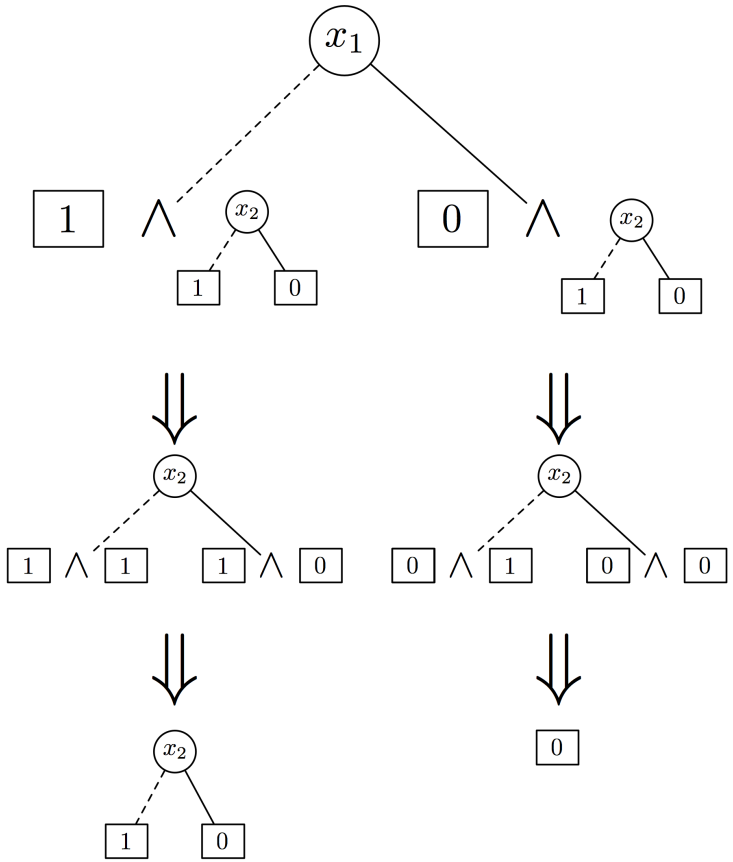
- Kombiniere den low und high Zweig des Knoten mit dem kleineren Variablenindex mit dem anderen Knoten



APPLY — Fall 3/4

Mindestens ein Argument ist ein Variablenknoten

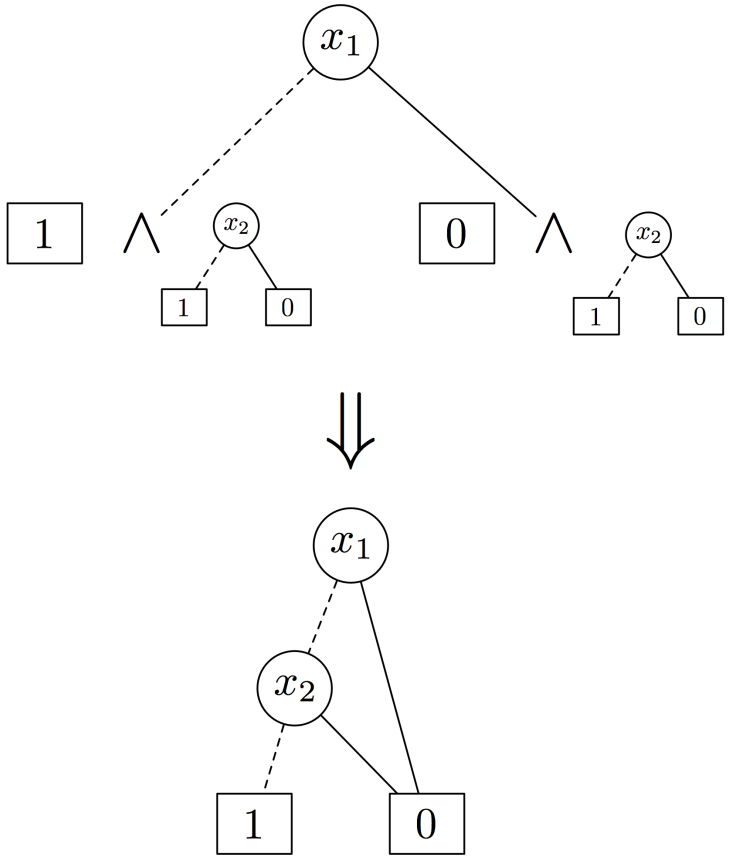
- Kombiniere den low und high Zweig des Knoten mit dem kleineren Variablenindex mit dem anderen Knoten



APPLY — Fall 3/4

Mindestens ein Argument ist ein Variablenknoten

- Kombiniere den low und high Zweig des Knoten mit dem kleineren Variablenindex mit dem anderen Knoten



APPLY(T, H, G, op, u_1, u_2) — Der Algorithmus

Input: Tabellen T, H und G, Boolescher Operand op, ROBDDs u_1 und u_2
Output: ROBDD für u_1 op u_2

if $(G(u_1, u_2) \neq empty)$ **then**
  return $G(u_1, u_2)$

if $u_1 \in \{0, 1\}$ *and* $u_2 \in \{0, 1\}$ **then**
  $u \leftarrow u_1$ op u_2
else
  if $\text{var}(u_1) = \text{var}(u_2)$ **then**
    $u \leftarrow \text{MK}(T, H, \text{var}(u_1)$, APPLY$(T, H, G, op, \text{low}(u_1), \text{low}(u_2))$,
    APPLY$(T, H, G, op, \text{high}(u_1), \text{high}(u_2))$
  else
    if $\text{var}(u_1) < \text{var}(u_2)$ **then**
      $u \leftarrow \text{MK}(T, H, \text{var}(u_1)$, APPLY$(T, H, G, op, \text{low}(u_1), u_2)$,
      APPLY$(T, H, G, op, \text{high}(u_1), u_2)$
    else
      $u \leftarrow \text{MK}(T, H, \text{var}(u_2)$, APPLY$T, H, G, op, (u_1, \text{low}(u_2))$,
      APPLY$(T, H, G, op, u_1, \text{low}(u_2))$
$G(u_1, u_2) \leftarrow u$
return u

APPLY — Beispiel