



Free Binary Decision Diagrams

*Eine Datenstruktur zur Repräsentation
Boolescher Funktionen*

Dirk Stegemann

`dirk.stegemann@students.uni-mannheim.de`

Problemstellung

Gesucht: Datenstruktur G für Boolesche Funktionen

$$f \in B_n = \{f \mid f : \{0, 1\}^n \rightarrow \{0, 1\}\}$$

Anforderungen:

- Basisoperationen sind effizient (in **polynomieller Laufzeit**) ausführbar.
- Möglichst viele $f \in B_n$ besitzen eine kompakte (**polynomiell große**) Repräsentation G_f .

Problemstellung

Problem:

Je kompakter die Datenstruktur, desto schwieriger die Operationen.

→ *Finde den richtigen Kompromiss*

Operationen

- Analyse

Operation	Eingabe	Ausgabe
EVAL	G_f $a \in \{0, 1\}^n$	$f(a)$
SAT	G_f	<i>true</i> gdw. $\exists a \in \{0, 1\}^n : f(a) = 1$
SAT-COUNT	G_f	$ \{a \in \{0, 1\}^n : f(a) = 1\} $
EQUIV	G_f, G_g	<i>true</i> gdw. $f \equiv g$
REDUNDANT	G_f $x_i \in X_n$	<i>true</i> gdw. $f _{x_i=0} \equiv f _{x_i=1}$

Operationen

- Manipulation

Operation	Eingabe	Ausgabe
MIN	G_f	G_f^* vom selben Typ wie G_f mit minimaler Größe für f
BIN-SYNTH	G_{f_1}, G_{f_2} $\otimes : \{0, 1\}^2 \rightarrow \{0, 1\}$	$G_{f_1 \otimes f_2}$
REPLACE-CONST	G_f $c \in \{0, 1\}, x_i \in X_n$	$G_{f _{x_i=c}}$
REPLACE-FUNC	G_f, G_g $x_i \in X_n$	$G_{f _{x_i=g}}$
QUANTIFY	G_f $Q \in \{\forall, \exists\}$ $x_i \in X_n$	G_g mit $g := \begin{cases} (\exists x_i) f = f _{x_i=0} + f _{x_i=1} & \text{für } Q = \exists \\ (\forall x_i) f = f _{x_i=0} \cdot f _{x_i=1} & \text{für } Q = \forall \end{cases}$

klassische Datenstrukturen

Beispiele: $(a \in \{0, 1\}^n, f \in B_n)$

- Wahrheitstabellen

$$|\text{Tabelle}| = O(2^n)$$

a_1	...	a_n	$f(a)$
0	...	0	0
0	...	1	1
\vdots	\vdots	\vdots	\vdots
1	1	1	0

- zweistufige Normalformen

3-SAT ist NP-schwer

$$f(a) = a_0 \cdot a_1 \cdot a_3 + a_2 \cdot a_3 \cdot a_5$$

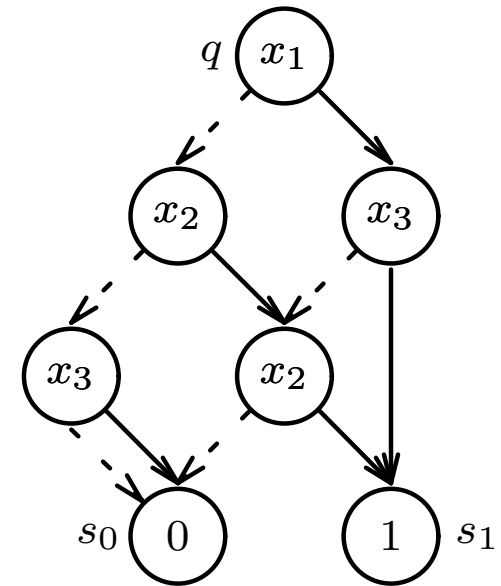
„innovative“ Datenstrukturen

- Binary Decision Diagrams
 - Ordered Binary Decision Diagrams
 - Free Binary Decision Diagrams

Binary Decision Diagrams

Definition 1. Ein *Binary Decision Diagram* (BDD) über der Variablenmenge $X_n = \{x_1, \dots, x_n\}$ ist ein gerichteter, azyklischer Graph $G = (V, E)$ mit

- $\exists_1 v \in V : \text{indegree}(v) = 0$ (Quelle q)
- $\exists_2 v \in V : \text{outdegree}(v) = 0$ (Senken s_1, s_2)
- $\forall v \in \underbrace{V \setminus \{s_0, s_1\}}_{\text{innere Knoten}} : \text{outdegree}(v) = 2$
- **Knotenbeschriftungen**
 - **Senken:** $s_0.\text{label} = 0, s_1.\text{label} = 1$
 - **innere Knoten:** $v.\text{label} \in X_n$
- **Kantenbeschriftungen**
 - - - - $\text{label}=0$
 - ——— $\text{label}=1$



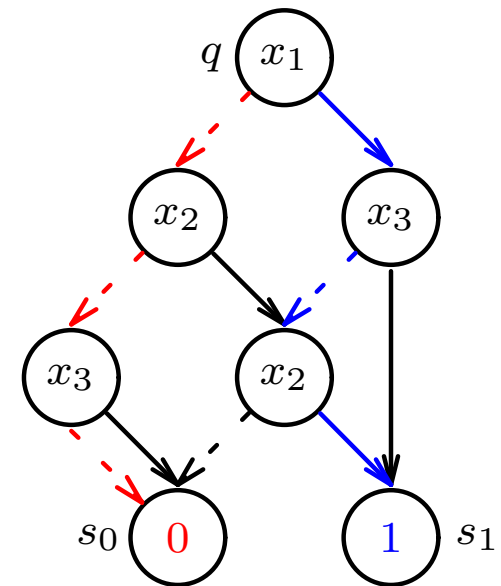
Binary Decision Diagrams

Jedes $v \in V$ repräsentiert $f_v \in B_n$:

$\forall x \in \{0, 1\}^n : f_v(x) = \text{Label der Senke, zu der der durch } x \text{ definierte, in } v \text{ beginnende Pfad führt.}$

Beispiele:

- $f_q(0, 0, 0) = 0$
- $f_q(1, 1, 0) = 1$



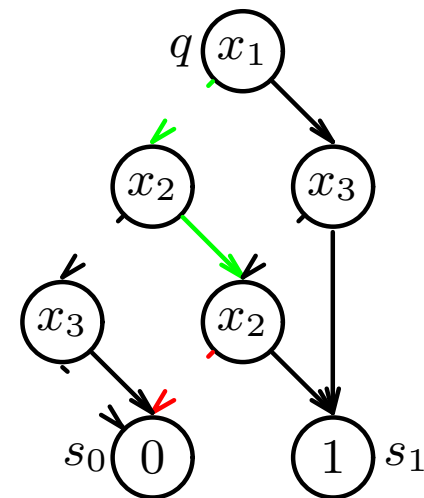
Binary Decision Diagrams

Operation	Laufzeit und Speicherplatz
	BDD
EVAL	$O(n)$
SAT	<i>NP-vollständig</i>
SAT-COUNT	<i>NP-schwer</i>
EQUIV	<i>coNP-vollständig</i>
REDUNDANT	<i>coNP-vollständig</i>
MIN	<i>NP-schwer</i>
BIN-SYNTH	$O(G_f + G_g)$
REPLACE-CONST	$O(G_f)$
REPLACE-FUNC	$O(G_f + G_g)$
QUANTIFY	$O(G_f + G_g)$

Binary Decision Diagrams

Weitere Eigenschaften:

- **Inkonsistente** Pfade sind zugelassen.



- Der minimale BDD für eine Funktion $f \in B_n$ ist **nicht** eindeutig bestimmt.

Ordered Binary Decision Diagrams

Idee: Variablen in bestimmter Reihenfolge einlesen

Definition 2. Eine *Variablenordnung* π über der Variablenmenge X_n ist eine Permutation der Indexmenge $I = \{1, \dots, n\}$.

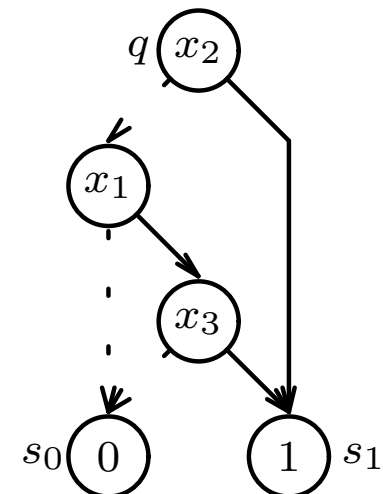
$$\pi(2) = 1$$

$$\pi(1) = 2$$

$$\pi(3) = 3$$

Definition 3 (Bryant86). Ein π -*Ordered Binary Decision Diagram* (π -OBDD) über X_n ist ein BDD, in dem auf jedem Pfad von der Quelle zu einer Senke die Variablenordnung π respektiert wird.

Beobachtung 4 (read-once Eigenschaft). Auf allen Pfaden wird jede Variable $x_i \in X_n$ höchstens einmal eingelesen.



Ordered Binary Decision Diagrams

Operation	Laufzeit und Speicherplatz	
	BDD	OBDD
EVAL	$O(n)$	$O(n)$
SAT	<i>NP-vollständig</i>	$O(G_f)$
SAT-COUNT	<i>NP-schwer</i>	$O(G_f)$
EQUIV	<i>coNP-vollständig</i>	$O(G_f + G_g)$
REDUNDANT	<i>coNP-vollständig</i>	$O(G_f)$
MIN	<i>NP-schwer</i>	$O(G_f)$
BIN-SYNTH	$O(G_f + G_g)$	$O(G_f \cdot G_g)$
REPLACE-CONST	$O(G_f)$	$O(G_f)$
REPLACE-FUNC	$O(G_f + G_g)$	$O(G_f + G_g)$
QUANTIFY	$O(G_f + G_g)$	$O(G_f ^2)$

Ordered Binary Decision Diagrams

Weitere Eigenschaften:

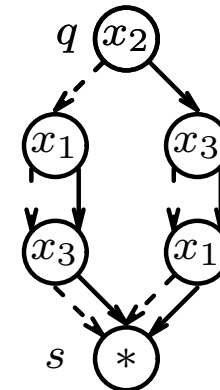
- Für fest gewähltes π ist der **minimale π -OBDD** für eine Funktion $f \in B_n$ **eindeutig bestimmt**.
→ Minimaler π -OBDD ist kanonische Darstellung für $f \in B_n$.
- \exists „einfache“ $f \in B_n$: Größe des minimalen π -OBDDs für f ist **exponentiell** in n für **jede** Variablenordnung π .
Beispiel: $f_n(s, X) = s \cdot ROW_n(X) + \bar{s} \cdot COLUMN_n(X)$

Free Binary Decision Diagrams

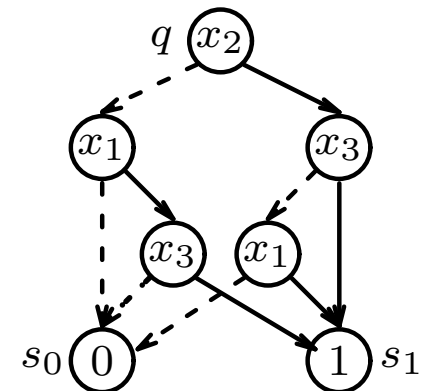
Idee: globale Variablenordnung \rightarrow „lokale“ Variablenordnung

Definition 5. Ein *Steuerungsgraph* $G_0 = (V, E)$ über X_n ist ein BDD mit

- nur einer Senke s , $s.label = *$
- genau einem mit x_i bezeichneten Knoten auf jedem Pfad von q zu s für alle $x_i \in X_n$



Definition 6 (Masek76). Ein G_0 -Free Binary Decision Diagram (G_0 -FBDD) über X_n ist ein BDD, der für jedes $a \in \{0, 1\}^n$ die durch G_0 vorgeschriebene Einlesereihenfolge respektiert.



Free Binary Decision Diagrams

Folgerungen:

- Jeder G_0 -FBDD besitzt die *read-once* Eigenschaft.
→ keine inkonsistenten Pfade
- OBDDs sind spezielle FBDDs.

Operationen auf FBDDs

Vorüberlegungen

Operation	Eingabe	Ausgabe
QUANTIFY	G_f $Q \in \{\forall, \exists\}$ $x_i \in X_n$	G_g mit $g = \begin{cases} f _{x_i=0} + f _{x_i=1} & \text{für } Q = \exists \\ f _{x_i=0} \cdot f _{x_i=1} & \text{für } Q = \forall \end{cases}$
REPLACE-FUNC	G_f, G_g $x_i \in X_n$	$G_{f _{x_i=g}} = ite(g, f _{x_i=1}, f _{x_i=0})$ mit $ite : \{0, 1\}^3 \rightarrow \{0, 1\}$ $(x, y, z) \mapsto xy + \bar{x}z$
REDUNDANT	G_f $x_i \in X_n$	$true$ gdw. $f _{x_i=0} \equiv f _{x_i=1}$

- QUANTIFY = BIN-SYNTH \circ REPLACE-CONST

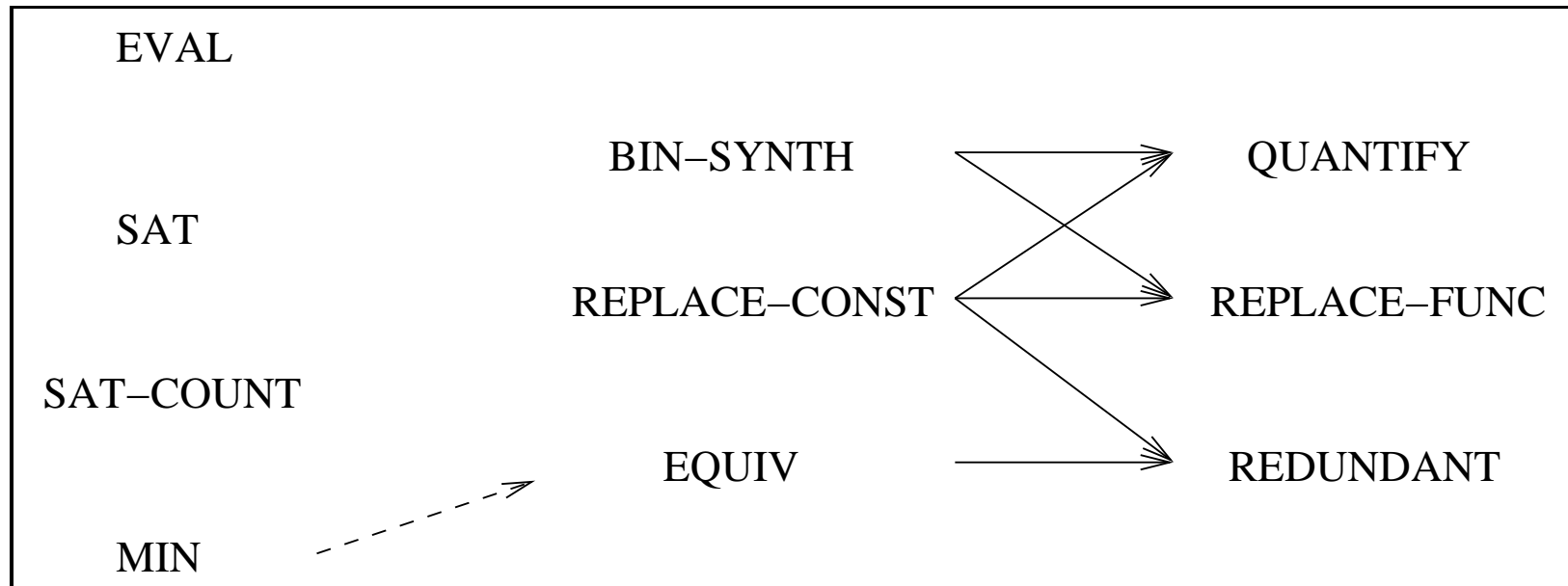
→ Substitutionen:

- REPLACE-FUNC = TENARY-SYNTH \circ REPLACE-CONST

- REDUNDANT = EQUIV \circ REPLACE-CONST

Operationen auf FBDDs

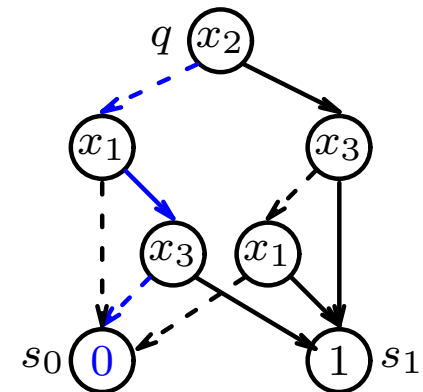
Abhängigkeitsgraph der Operationen



Evaluation

Eingabe G_0 -FBDD $G_f, a \in \{0, 1\}^n$

Problem Berechne $f(a)$



*Verfolge den durch a
definierten Pfad bis zu
einer Senke.*

Laufzeit $O(n)$

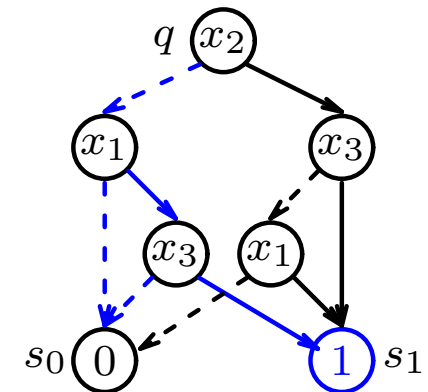
SAT

Eingabe G_0 -FBDD G_f
Problem Entscheide, ob f erfüllbar

Read-once Eigenschaft

→ keine inkonsistenten Pfade

Jeder Pfad in G_f kann durch ein $a \in \{0, 1\}^n$ aktiviert werden.



Tiefensuche nach s_1 in
 G_f

Laufzeit $O(|G_f|)$

SAT-COUNT

Eingabe: G_0 -FBDD G_f

Problem: Bestimme Anzahl der erfüllenden Belegungen für f

Es gilt für jeden inneren Knoten v , $x_i := v.label$:

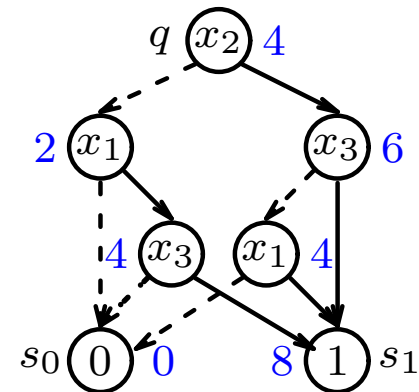
$$f_v(a) = 1$$

$$\Leftrightarrow (a_i = 0 \wedge f_{v_0}(a) = 1) \vee (a_i = 1 \wedge f_{v_1}(a) = 1)$$

Read-once Eigenschaft

→ Anzahl $c(v)$ der f_v erfüllenden Belegungen

$$c(v) = \frac{1}{2}(c(v_0) + c(v_1))$$



*Bestimme $c(v)$
bottom-up*

*Laufzeit und
Speicherplatz $O(|G_f|)$*

Min

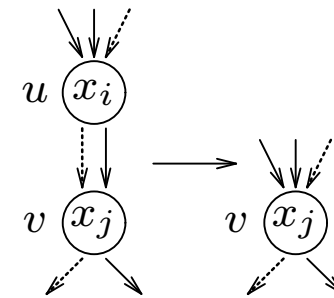
Eingabe: G_0 -FBDD G_f

Problem: Bestimme minimalen
 G_0 -FBDD für f

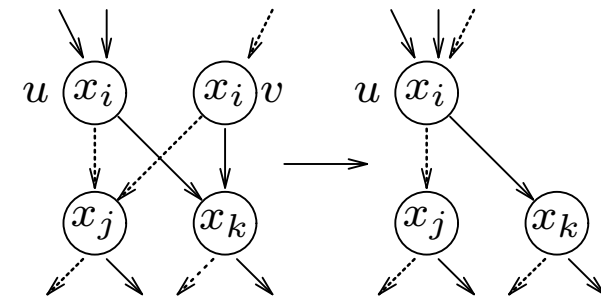
Theorem 7. G_f ist genau dann minimal,
wenn weder die merging rule noch die
elimination rule anwendbar ist.

Wende die Regeln bottom-up an

Laufzeit und Speicherplatz $O(|G_f|)$



elimination rule



merging rule

Min

Zusätzliche Eigenschaft:

Für fest gewähltes G_0 ist der **minimale G_0 -FBDD** für eine Funktion $f \in B_n$ **eindeutig bestimmt**.

→ Minimaler G_0 -FBDD ist kanonische Darstellung für $f \in B_n$.

REPLACE-CONST

Eingabe: G_0 -FBDD G_f , $x_i \in X_n$,
 $c \in \{0, 1\}$

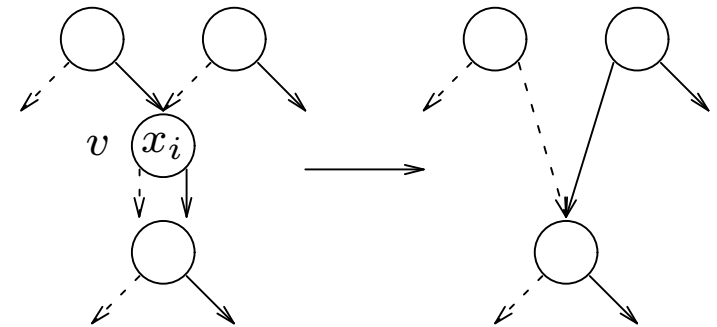
Problem: Ersetze x_i durch Konstante c

Ersetzung durch Konstanten kann im Allgemeinen zu **exponentieller** FBDD-Größe führen.

Beispiel: $f_n(s, X)|_{s=0}$

Definition 8. Ein Steuerungsgraph G_0 über X_n ist **x_i -vernachlässigend**, wenn für alle $v \in V$ mit $v.\text{label} = x_i$ gilt $v_0 = v_1$.

Theorem 9. Falls G_0 x_i -vernachlässigend, kann REPLACE-CONST für x_i in G_f **effizient** durchgeführt werden.



Ersetze die Endpunkte aller eingehenden Kanten von v durch v_c .

Laufzeit und Speicherplatz
 $O(|G_f|)$

BIN-SYNTH

Eingabe: G_0 -FBDDs G_f, G_g
 $\otimes : \{0, 1\}^2 \rightarrow \{0, 1\}$
 Problem: Bestimme $G_{f \otimes g}$

$$f \otimes g : \{0, 1\}^n \rightarrow \{0, 1\}$$

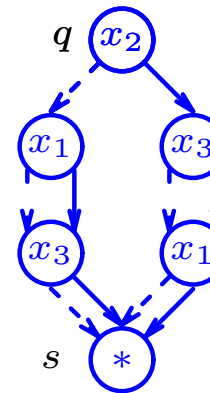
$$(f \otimes g)(a) \mapsto f(a) \otimes g(a)$$

Traversiere G_0, G_f und G_g
 simultan mittels DFS.

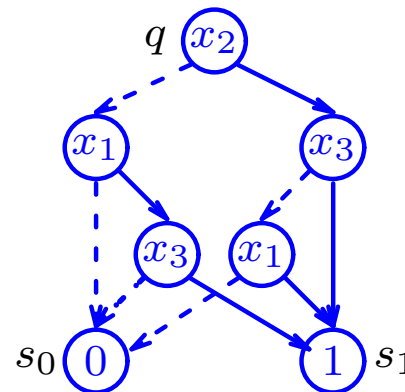
Wenn Senken erreicht, berechne
 Funktionswert als

$$(f \otimes g)(a) = f(a) \otimes g(a)$$

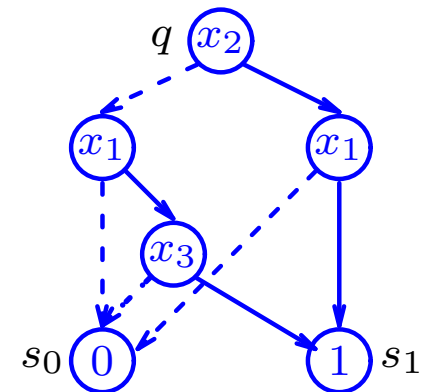
Steuerungsgraph G_0



G_0 -FBDD G_f



G_0 -FBDD G_g



BIN-SYNTH

Implementation

- *computed table* speichert Berechnungsergebnisse
- *unique table* speichert einen Repräsentanten für jedes Tripel $(v, v_1, v_2) \in V \times V \times V$
→ Synthese mit integrierter Reduktion

insgesamt: Laufzeit und Speicherplatz $O(|G_0| |G_f| |G_g|)$

Verallgemeinerung: Laufzeit und Speicherplatz für n -äre Synthese $O(|G_0| \prod_{i=1}^n |G_{f_i}|)$

EQUIV

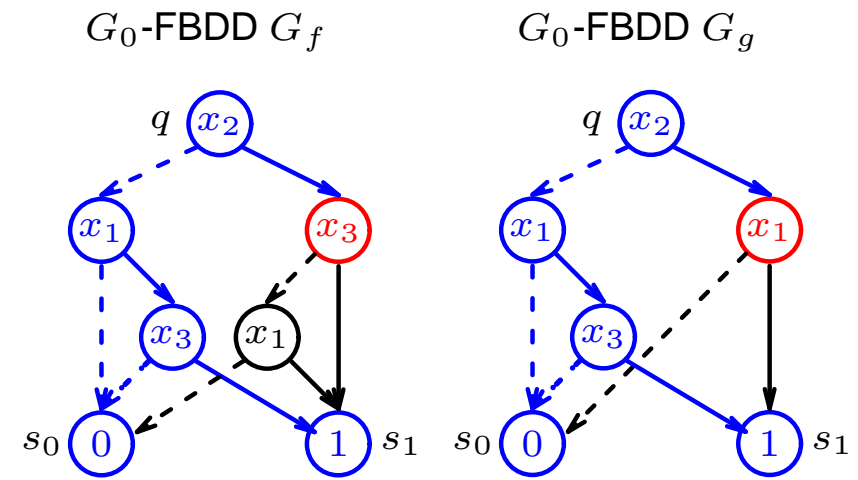
Eingabe: G_0 -FBDDs G_f, G_g

Problem: Entscheide, ob $f \equiv g$

Minimaler G_0 -FBDD für $h \in B_n$ ist kanonische Darstellung für h .

→ Strategie:

- Minimiere G_f und G_g .
- Führe Isomorphietest durch simultane DFS-Traversierung in G_f und G_g durch.



Simultane DFS-Traversierung

Laufzeit und Speicherplatz

$$O(|G_f| + |G_g|)$$

Operationen auf FBDDs

Operation	Laufzeit und Speicherplatz		
	BDD	π -OBDD	G_0 -FBDD
EVAL	$O(n)$	$O(n)$	$O(n)$
SAT	<i>NP-vollständig</i>	$O(G_f)$	$O(G_f)$
SAT-COUNT	<i>NP-schwer</i>	$O(G_f)$	$O(G_f)$
EQUIV	<i>coNP-vollständig</i>	$O(G_f + G_g)$	$O(G_f + G_g)$
REDUNDANT	<i>coNP-vollständig</i>	$O(G_f)$	$O(G_f)^*$
MIN	<i>NP-schwer</i>	$O(G_f)$	$O(G_f)$
BIN-SYNTH	$O(G_f + G_g)$	$O(G_f \cdot G_g)$	$O(G_0 \cdot G_f \cdot G_g)$
REPLACE-CONST	$O(G_f)$	$O(G_f)$	$O(G_f)^*$
REPLACE-FUNC	$O(G_f + G_g)$	$O(G_f + G_g)$	$O(G_0 G_g G_f ^2)^*$
QUANTIFY	$O(G_f + G_g)$	$O(G_f ^2)$	$O(G_0 \cdot G_f ^2)^*$

*= falls G_0 x_i -vernachlässigend

Zusammenfassung

Eigenschaften von FBDDs

- keine inkonsistenten Pfade
- (fast) alle Operationen effizient ausführbar
- minimaler G_0 -FBDD ist kanonische Repräsentation für jedes $f \in B_n$
- Menge der mit polynomielltem Speicherplatz darstellbaren Funktionen größer als bei OBDDs

Zusammenfassung

Einschränkungen

- wenige Heuristiken zur Bestimmung des optimalen Ordnungsgraphen G_0 für eine gegebene Funktion $f \in B_n$ bekannt
- Polynomiell große Repräsentation für Funktionen mit exponentieller FBDD-Größe?

Beispiel: $f_n(X) = ROW_n(X) + COLUMN_n(X)$

→ weitere BDD-Derivate

Vielen Dank!

Min

Verwalte folgende Partition von V :

- $bottom = \{v \in V \mid \textit{merging- und elimination rule für } v \textit{ nicht anwendbar}\}$
- $middle = \{v \in V \mid v \notin bottom \wedge v_1, v_2 \in bottom\}$
- $top = \{v \in V \mid v \notin bottom \cup middle\}$

Algorithmus $Min(G_f)$

$bottom := \{s_0, s_1\}$

while $bottom \neq V$ **do**

$middle := \{v \in V \mid v \notin bottom \wedge v_1, v_2 \in bottom\}$

 Wende *elimination rule* auf *middle* an

 Wende *merging rule* auf *middle* an

$bottom := bottom \cup middle$

Laufzeit und Speicherplatzbedarf

- *elimination rule*: $O(|middle|)$
- *merging rule*
 - vergleichsbasiertes Sortieren: $O(|middle| \cdot \log(|middle|))$
 - bucket sort: $O(|middle|)$

Gesamt: $O(|G_f|)$