8.3 Algorithmus zur Berechnung eines Minimalpolynoms

- Verfahren von Quine/McCluskey zur Berechnung der Primimplikanten
- Verfahren zur Lösung des Überdeckungsproblems

Verfahren von Quine: der Algorithmus

polynom function Quine (f: $\mathbf{B}^n$ --> $\mathbf{B}$)
begin
  L_0 := $\mathcal{Minterm}$(f);
  i=0;
  // L_i enthält alle Implikanten von f der Länge n-i
  $\mathcal{Prim}$(f) := ∅;
  while (L_i ≠∅) and (i<n)
  loop L_{i+1} := {m | mx und mx' sind in L_i für ein x};
    $\mathcal{Prim}$(f) := $\mathcal{Prim}$(f) È
        {m | m∈ L_i und m wird von keinem q ∈ L_{i+1} überdeckt};
    i:=i+1;
  pool;
  return $\mathcal{Prim}$(f) È L_i ;
end;

Verbesserung durch McCluskey

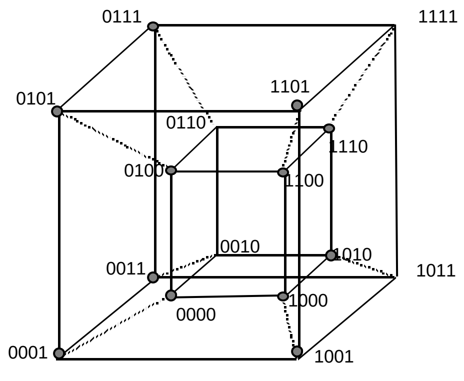
Vergleiche nur Monome untereinander

- die die gleichen Variablen enthalten und
- bei denen sich die Anzahl der positiven Literale um 1 unterscheidet.

Kann erreicht werden durch

- Partitioniere L_i in Klassen L_i^M, mit $M \subseteq \{x_1,...x_n\}$ und $|M|=n-i$. L_i^M enthalte die Implikanten aus L_i, deren Literale alle aus M sind.
- Ordne die Monome in L_i^M gemäß der Anzahl der positiven Literale.

Das Verfahren von Quine-McCluskey: Beispiel



$L_0^{\{x1,x2,x3,x4\}}$:

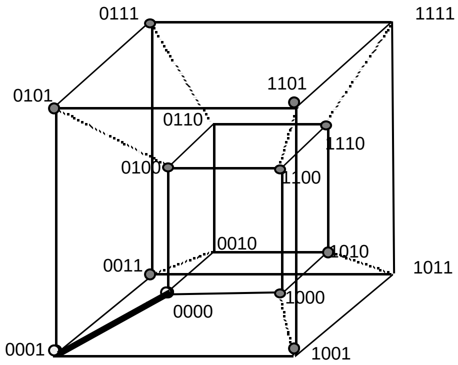
0 0 0 0
0 0 0 1
0 1 0 0
1 0 0 0
0 0 1 1
0 1 0 1
1 0 0 1
1 0 1 0
1 1 0 0
0 1 1 1
1 1 0 1
1 1 1 0

steht für $x_1'x_2'x_3'x_4'$

steht für $x_1x_2'x_3'x_4$

Vergleiche im folgenden nur Monome aus benachbarten Blöcken!

Beispiel Quine-McCluskey ff



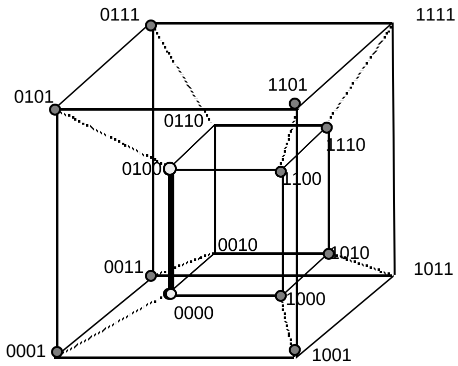
$L_0^{\{x1,x2,x3,x4\}}$:

0 0 0 0
0 0 0 1
0 1 0 0
1 0 0 0
0 0 1 1
0 1 0 1
1 0 0 1
1 0 1 0
1 1 0 0
0 1 1 1
1 1 0 1
1 1 1 0

$L_1^{\{x1,x2,x3\}}$:

0 0 0 -

Beispiel Quine-McCluskey ff



$L_0^{\{x1,x2,x3,x4\}}$:

0 0 0 0
0 0 0 1
0 1 0 0
1 0 0 0
0 0 1 1
0 1 0 1
1 0 0 1
1 0 1 0
1 1 0 0
0 1 1 1
1 1 0 1
1 1 1 0

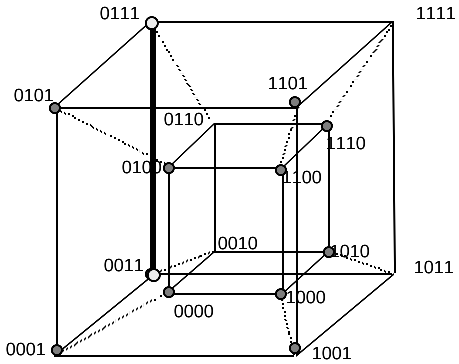
$L_1^{\{x1,x2,x3\}}$:

0 0 0 -

$L_1^{\{x1,x3,x4\}}$:

0 - 0 0

Beispiel Quine-McCluskey ff



$L_0^{\{x1,x2,x3,x4\}}$:

0 0 0 0
0 0 0 1
0 1 0 0
1 0 0 0
0 0 1 1
0 1 0 1
1 0 0 1
1 0 1 0
1 1 0 0
0 1 1 1
1 1 0 1
1 1 1 0

$L_1^{\{x1,x2,x3\}}$:

0 0 0 -

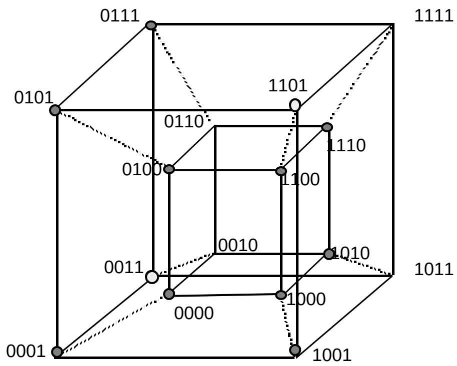
$L_1^{\{x1,x3,x4\}}$:

0 - 0 0

....

0 - 1 1

Beispiel Quine-McCluskey ff



$L_0^{\{x1,x2,x3,x4\}}$:

0 0 0 0
0 0 0 1
0 1 0 0
1 0 0 0
0 0 1 1
0 1 0 1
1 0 0 1
1 0 1 0
1 1 0 0
0 1 1 1
1 1 0 1
1 1 1 0

$L_1^{\{x1,x2,x3\}}$:

0 0 0 -

$L_1^{\{x1,x3,x4\}}$:

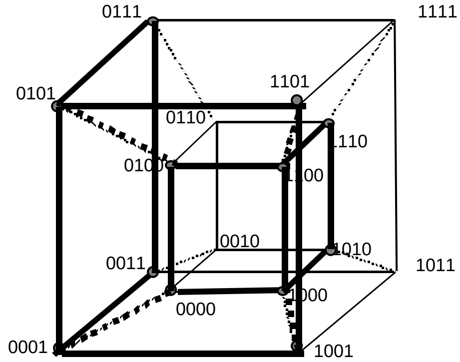
0 - 0 0

....

0 - 1 1

Nicht kürzbar, da nicht Ecken der gleichen Kante (Consensus existiert nicht!)

Beispiel Quine-McCluskey ff



$L_1^{\{x1,x2,x4\}}$:

0 0 - 1
1 0 - 0
0 1 - 1
1 1 - 0

$L_1^{\{x1,x2,x3\}}$:

0 0 0 -
0 1 0 -
1 0 0 -
1 1 0 -

$L_1^{\{x2,x3,x4\}}$:

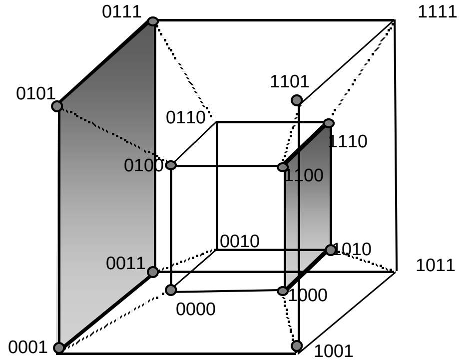
- 0 0 0
- 0 0 1
- 1 0 0
- 1 0 1

$L_1^{\{x1,x3,x4\}}$:

0 - 0 0
0 - 0 1
1 - 0 0
0 - 1 1
1 - 0 1
1 - 1 0

Alle Minterme von *f* sind Eckpunkte von Kanten, die Implikanten sind $\Rightarrow$ $\mathcal{Prim}(f) = \emptyset$.

Beispiel Quine-McCluskey ff



$L_1^{\{x1,x2,x4\}}$:

0 0 - 1
1 0 - 0
0 1 - 1
1 1 - 0

$L_1^{\{x1,x2,x3\}}$:

0 0 0 -
0 1 0 -
1 0 0 -
1 1 0 -

$L_1^{\{x2,x3,x4\}}$:

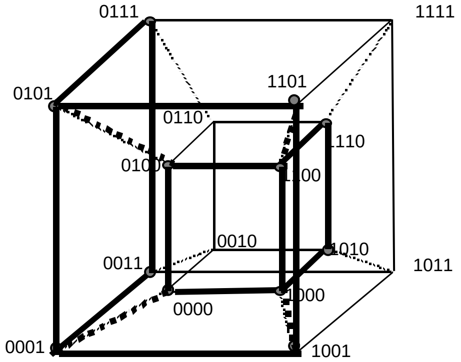
- 0 0 0
- 0 0 1
- 1 0 0
- 1 0 1

$L_1^{\{x1,x3,x4\}}$:

0 - 0 0
0 - 0 1
1 - 0 0
0 - 1 1
1 - 0 1
1 - 1 0

Alle Implikanten aus $L_1^{\{x1,x2,x4\}}$ sind Kanten von Flächen, die Implikanten sind $\Rightarrow$ $\mathcal{Prim}(f) = \emptyset$.

Beispiel Quine-McCluskey ff



$L_1^{\{x1,x2,x4\}}$:
0 0 - 1
1 0 - 0
0 1 - 1
1 1 - 0

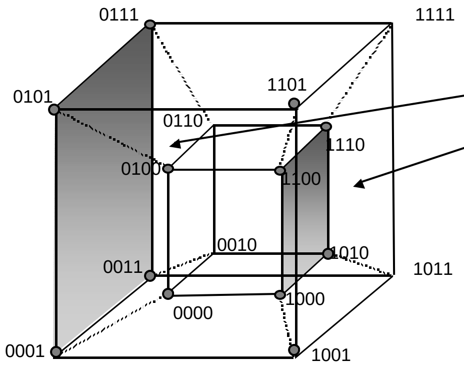
$L_1^{\{x1,x2,x3\}}$:
0 0 0 -
0 1 0 -
1 0 0 -
1 1 0 -

$L_1^{\{x2,x3,x4\}}$:
- 0 0 0
- 0 0 1
- 1 0 0
- 1 0 1

$L_1^{\{x1,x3,x4\}}$:
0 - 0 0
0 - 0 1
1 - 0 0
0 - 1 1
1 - 0 1
1 - 1 0

Alle Implikanten aus L_1^M sind Kanten von Flächen, die Implikanten sind ⇒ $\mathcal{Prim}(f)=\emptyset$.

Beispiel Quine-McCluskey ff



$L_2^{\{x1,x2\}}$:

$L_2^{\{x1,x3\}}$:
0 - 0 -
1 - 0 -

$L_2^{\{x1,x4\}}$:
0 - - 1
1 - - 0

$L_2^{\{x2,x3\}}$:
- 0 0 -
- 1 0 -

$L_2^{\{x2,x4\}}$:

$L_2^{\{x3,x4\}}$:
- - 0 0
- - 0 1

Die markierten Implikanten-Flächen sind nicht Rand eines 3-dim. Implikanten. Sie sind also prim! ⇒ $\mathcal{Prim}(f) = \{x_1' x_4, x_1 x_4'\}$

Beispiel Quine-McCluskey ff

$L_2^{\{x1,x2\}}$:

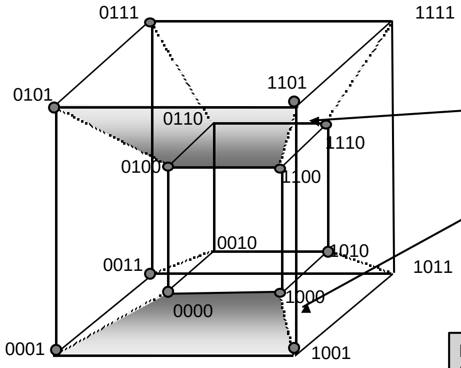
$L_2^{\{x1,x3\}}$:
0 - 0 -
1 - 0 -

$L_2^{\{x1,x4\}}$:
0 - - 1
1 - - 0

$L_2^{\{x2,x3\}}$:
- 0 0 -
- 1 0 -

$L_2^{\{x2,x4\}}$:

$L_2^{\{x3,x4\}}$:
- - 0 0
- - 0 1



Die markierten Implikanten-Flächen sind Rand eines 3-dimensionalen Implikanten. Sie sind also nicht prim! $\Rightarrow$ $Prim(f) = \{x_1' x_4, x_1 x_4'\}$

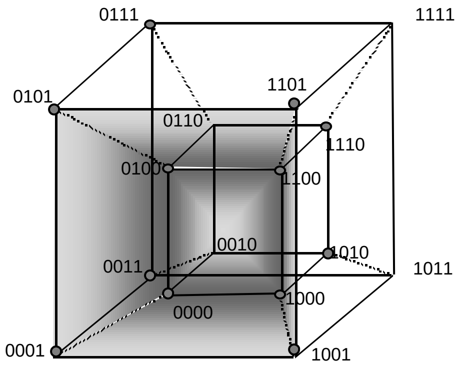
Beispiel Quine-McCluskey ff

$L_3^{\{x1\}}$:

$L_3^{\{x2\}}$:

$L_3^{\{x3\}}$:
- - 0 -

$L_3^{\{x4\}}$:



$Prim(f) = \{x_1' x_4, x_1 x_4'\}$

$\Rightarrow$ $Prim(f) = \{ x_1' x_4, x_1 x_4', x_3' \}$

$\Rightarrow$ $p_{complete}(f) = x_1' x_4 + x_1 x_4' + x_3'$

Korrektheit von Quine-McCluskey

polynom function Quine (f: $\mathbf{B}^n$ --> **B**)

begin

L_0 := $\mathcal{Minterm}$(f);

i=0;

// L_i enthält alle Implikanten von f der Länge n-i

$\mathcal{Prim}$(f) := $\emptyset$;

while ($L_i \neq \emptyset$) and (i<n)

loop L_{i+1} := {m | mx und mx' sind in L_i für ein x};

$\mathcal{Prim}$(f) := $\mathcal{Prim}$(f) $\cup$ { m | m$\in L_i$ und m wird von keinem q $\in L_{i+1}$ überdeckt};

i:=i+1;

pool;

return $\mathcal{Prim}$(f) $\cup$ L_i ;

end**;**

Korrektheit von Quine-McCluskey

Theorem

Für alle i=0,1,...,n gilt

:

- L_i enthält nur Monome mit n-i Literalen
- L_i enthält genau die Implikanten von f mit n-i Literalen
- Nach Iteration i enthält $\mathcal{Prim}$(f) genau die Primimplikanten von f mit n-i Literalen

Beweis

Induktion über i

Zur Abbruchbedingung: $L_i = \emptyset$ bedeutet, dass keine Implikanten bei der „Partnersuche" entstanden sind, d.h. L_{i-1} ist ganz in $\mathcal{Prim}$(f) aufgegangen

i=n bedeutet, dass L_n berechnet wurde, es gilt dann $L_n = \emptyset$ oder $L_n = \{1\}$, letzteres bedeutet f ist die Eins-Funktion und $\mathcal{Prim}$(f) = $\{1\}$

Kosten des Verfahrens

Lemma

Es gibt 3^n verschiedene Monome in n Variablen

Beweis

Für jedes Monom m und jede der n Variablen x liegt genau eine der drei folgenden Situationen vor :

- m enthält weder das positive noch das negative Literal von x
- m enthält das positive Literal x
- m enthält das negative Literal x'

Jedes Monom ist durch diese Beschreibung auch eindeutig bestimmt.

Kosten des Verfahrens ff

Lemma
Das Verfahren von Quine-McCluskey macht höchstens

$$2\ n!\ 3^n \text{ Vergleiche}$$

zwischen je 2 Monomen.

Beweis
Jeder Block besteht aus $\leq n!$ vielen Monomen
(die nach Definition eines Blockes die gleiche Anzahl von positiven Literalen haben).

Ein Monom wird mit den Implikanten der beiden benachbarten Blöcke verglichen.
Mithilfe des letzten Lemma folgt damit die Behauptung.

Korollar
Das Verfahren von Quine-McCluskey benötigt höchstens

$$2\ n!\ 3^n n \quad \text{Bitvergleiche}$$

Beweis
Jeder Vergleich zwischen 2 Monomen kostet $\leq n$ Bitvergleiche.

Skizze: Implementierung und Beschleunigung der Berechnung

- Reserviere im Hauptspeicher ein Bitvektor IMPLIKANT der Länge 3^n.
- Initialisiere den Bitvektor auf (0,...,0)
- Interpretiere im folgenden jedes Monom m als Folge m der Länge n :
  - Enthält m das Literal x_i` , so setze m[i]=0
  - Enthält m das Literal x_i , so setze m[i]=1
  - Enthält m weder x_i noch x_i` , so setze m[i]=2

  Identifiziere m mit der von ihm dargestellten ternären Zahl $\Sigma m[i] \cdot 3^i$.
- Setze IMPLIKANT[m]=1, falls Monom m als Implikant gefunden ist.
- **Beobachtung**

  Ein Monom m kann nur zu einem Monom benachbart sein, welches durch Kippen eines (gültigen) Bits entsteht. Dies sind höchstens n Stück.

  Jeden dieser Nachbarn erhält man durch „entsprechende" Multiplikationen und Additionen.

BB TI I 8.3/19

Skizze: Implementierung und Beschleunigung der Berechnung ff

Satz -- Komplexität des Verfahrens von Quine-McCluskey

Die Laufzeit des Verfahrens liegt in $O(n^2\, 3^n \log(n))$ beziehungsweise in $O(\log^2 N\; N^{\log 3} \log \log N)$, wobei $N=2^n$ die Größe der Funktionstabelle ist.

Beweis

Jedes der 3^n Monome wird mit höchstens n anderen Monomen verglichen.

Jeder Vergleich, vielmehr die Berechnung des "Nachbarn", benötigt Laufzeit O(n log n), verwendet man schnelle Multiplikations- und Additionsschaltungen (siehe entsprechendes Kapitel der Vorlesung). Hieraus folgt die erste Teilbehauptung.

Die zweite Teilbehauptung folgt durch Nachrechnen.

BB TI I 8.3/20

Das Matrix-Überdeckungsproblem

Sei im folgenden die Menge $\mathcal{prim}$(f) der Primimplikanten von f gegeben.

Gesucht

ist eine kostenminimale Teilmenge M von $\mathcal{prim}$(f) , mit deren Monomen die Funktion f beschreibbar ist.

Das Matrix-Überdeckungsproblem ff

Definiere eine boolesche Matrix PIT(f) , die **Primimplikantentafel von** f :

- die **Zeilen** entsprechen eineindeutig den **Primimplikanten von** f
- die **Spalten** entsprechen eineindeutig den **Mintermen von** f
- Sei min($\boldsymbol{a}$) ein beliebiger Minterm von f . Dann gilt für PI m
  PIT(f) [m,min($\boldsymbol{a}$)]=1 $\Leftrightarrow y$(m)($\boldsymbol{a}$)=1,

  Der Eintrag an der Stelle [m, min($\boldsymbol{a}$)] ist also genau dann 1, wenn min($\boldsymbol{a}$) eine Ecke des Würfels m beschreibt.

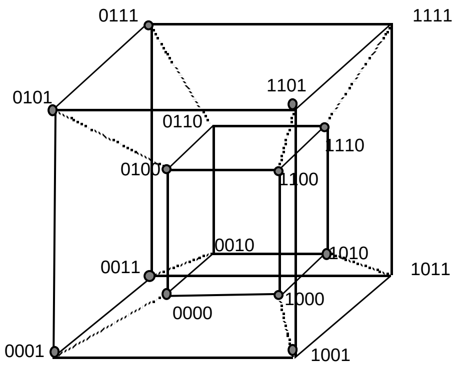
Gesucht

eine kostenminimale Teilmenge M von $\mathcal{prim}$(f), so dass jede Spalte von PIT(f) überdeckt ist,

d.h. $\forall\ \boldsymbol{a} \in$ ON(f) $\exists$m $\in$ M mit PIT(f) [m, min($\boldsymbol{a}$)]=1.

Das Matrix-Überdeckungsproblem ff

Ein erstes Beispiel



$\mathcal{P}rim(f) = \{ x_1'x_4 , x_1x_4' , x_3' \}$

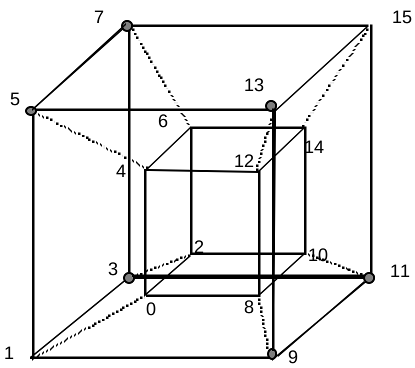
Primimplikantentafel PIT *(f)*:

	0	1	3	4	5	7	8	9	10	12	13	14
$x_1'x_4$		1	1		1	1						
x_1x_4'							1		1	1		1
x_3'	1	1		1	1		1	1		1	1	

⇒ Alle Primimplikanten sind **wesentlich**!

Das Matrix-Überdeckungsproblem ff

Ein anderes Beispiel



$\mathcal{P}rim(f) = \{\{7,5\},\{5,13\},\{13,9\}\{9,11\},\{11,3\},\{3,7\}\}$

Primimplikantentafel PIT *(f)*:

	3	5	7	9	11	13
{7,5}		1	1			
{5,13}		1				1
{13,9}				1		1
{9,11}				1	1	
{11,3}	1				1	
{3,7}	1		1			

Kein Primimplikant ist wesentlich!

Erste Reduktionsregel

Definition

Ein Primimplikant m von f heißt **wesentlich**, wenn es einen Minterm min(a) von f gibt, der nur von diesem Primimplikanten überdeckt wird, also:

- PIT (f) [m,min(a)]= 1
- PIT (f) [m',min(a)]= 0 für jeden anderen PI m' von f

Lemma

Jedes Minimalpolynom von f enthält alle wesentlichen Primimplikanten von f.

1. Reduktionsregel

Entferne aus der Primimplikantentafel PIT (f) alle wesentlichen Primimplikanten und alle Minterme, die von diesen überdeckt werden.

Erste Reduktionsregel ff

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
1	1				1												
2		1				1											
3			1				1										
4				1				1									
5					1				1								1
6						1				1							1
7							1				1						
8								1				1					
9									1				1				
10										1				1			1
11											1				1		
12												1				1	
13													1	1	1	1	

wesentlich

Erste Reduktionsregel ff

Überdeckungsproblem nach Anwendung der 1-ten Reduktionsregel:

	9	10	11	12	13	14	15	16	17
5	1								1
6		1							1
7			1						
8				1					
9	1				1				
10		1				1			1
11			1				1		
12				1				1	
13					1	1	1	1	

Die Matrix enthält keine wesentlichen Zeilen mehr !

Zweite Reduktionsregel

Definition

Es sei A eine Boolesche Matrix.
Spalte j der Matrix A **dominiert** Spalte i der Matrix A, wenn $A[k,i] \leq A[k,j]$ für jede Zeile k gilt.

Nutzen für unser Problem

Dominiert ein Minterm w' von f einen anderen Minterm w von f, so braucht man w' nicht mehr weiter zu betrachten, da w auf jeden Fall überdeckt werden muss und hierdurch auch Minterm w' überdeckt wird.

Jeder noch in PIT (f) vorhandene Primimplikant p, der w überdeckt, überdeckt auch w' .

2. Reduktionsregel

Entferne aus der Primimplikantentafel PIT (f) alle Minterme, die einen anderen Minterm in PIT (f) dominieren.

Zweite Reduktionsregel ff

	9	10	11	12	13	14	15	16	17
5	1								1
6		1							1
7			1						
8				1					
9	1				1				
10		1				1			1
11			1				1		
12				1				1	
13					1	1	1	1	

Spalte 17 dominiert Spalte 10
=> Spalte 17 kann gelöscht werden!

Dritte Reduktionsregel

Definition

Sei A eine Boolesche Matrix.
Zeile i der Matrix A **dominiert** Zeile j der Matrix A,
wenn $A[i,k] \geq A[j,k]$ für jede Spalte k gilt.

Nutzen für unser Problem

Dominiert ein Primimplikant m einen Primimplikanten m' , so braucht man m' nicht mehr weiter zu betrachten, wenn $cost(m') \geq cost(m)$ gilt.

Der Primimplikant m überdeckt jeden noch nicht überdeckten Minterm von f, der von m' überdeckt wird, obwohl er nicht teurer ist.

3. Reduktionsregel

Entferne aus PIT (f) alle Primimplikanten, die durch einen anderen nicht teureren Primimplikanten dominiert werden.

Dritte Reduktionsregel ff

Nehme an, dass die Zeilen 5 bis 12 gleiche Kosten haben.

werden dominiert

	9	10	11	12	13	14	15	16
5	1							
6		1						
7			1					
8				1				
9	1				1			
10		1				1		
11			1				1	
12				1				1
13					1	1	1	1

Dritte Reduktionsregel ff

Überdeckungsproblem nach Anwendung der 3-ten Reduktionsregel:

	9	10	11	12	13	14	15	16
9	1				1			
10		1				1		
11			1				1	
12				1				1
13					1	1	1	1

Offensichtlich kann nun wieder die erste Reduktionsregel angewendet werden, da die Zeilen 9, 10, 11, 12 wesentlich sind.

⇒ Die resultierende Matrix ist leer

⇒ Das gefundene Minimalpolynom ist 1+2+3+4+9+10+11+12

... enthält nicht wie erwartet die Zeile mit der maximalen Anzahl Einser !

Zyklische Überdeckungsprobleme

Definition

Eine Primimplikantentafel heißt **reduziert** , wenn keine der drei Reduktionsregeln anwendbar ist.

Ist eine reduzierte Tafel nicht-leer, spricht man von einem **zyklischen Überdeckungsproblem**

Primimplikantentafel PIT *(f)*:

	3	5	7	9	11	13
{7,5}		1	1			
{5,13}		1				1
{13,9}				1		1
{9,11}				1	1	
{11,3}	1				1	
{3,7}	1		1			

Ansätze zum Lösen des zyklischen Überdeckungsproblems

- heuristische Verfahren (→ ESPRESSO)
- Verfahren von Petrick
- Branch-and-Bound Verfahren

Petrick's Methode

Verfahren

- Übersetze die PIT in eine Produktsumme, d.h. in ein (OR,AND)-Polynom, das alle Möglichkeiten der Überdeckung enthält.
- Multipliziere die Produktsumme aus, so daß ein (AND-OR) Polynom entsteht.
- Die gesuchte minimale Überdeckung ist gegeben durch das **kürzeste Monom**

	1	2	3	4
1	1	1		
2			1	1
3	1		1	
4		1		1
5	1			1
6		1	1	

notwendig, um Spalte 4 zu überdecken

wird übersetzt in

(1+3+5)(1+4+6)(2+3+6)(2+4+5)

= (1+14+16+13+34+36+15+45+56)*
(2+24+25+23+34+35+26+46+56)

= **12**+124+125+123+134+...+**34**+...+**56**