

# Minimierung

- **Schritt 1:**  
Berechnung aller Primimplikanten (Primimplikate) der gegebenen Funktion  
→ **Gatter mit möglichst wenig Eingängen**
- **Schritt 2:**  
Auswahl einer Menge von Primimplikanten (Primimplikate) zur Bildung der Minimalform: Kernprimimplika(n)te(n) und möglichst wenigen Primimplika(n)ten zur Überdeckung der Funktion  
→ **Minimale Anzahl an Gattern**

## Wdh. Quine-McCluskey-Verfahren

- Arbeitet nach dem gleichen Prinzip wie KV-Diagramme:
  - Terme, die sich in nur einer Variablen unterscheiden, werden zusammengefasst.
  - Ausgangspunkt: Funktionstabelle einer Funktion.
  - Disjunktive wie konjunktive Minimalformen können erzeugt werden.
- Bei **disjunktiven Minimalformen** DMF (Primimplikanten) arbeitet man mit den Mintermen der Funktion. Bei **konjunktiven Minimalformen** KMF (Primimplikate) mit den Maxtermen.
- Das Verfahren wird im folgenden für DMF erläutert  
→ Ausgangspunkt sind die Minterme der Funktion
- Für KMF geht man analog mit den Maxtermen vor.

## Wdh. Quine-McCluskey-Verfahren

### 3. Schritt:

Schritt 2 wird solange wiederholt, bis keine neuen Spalten mehr in der Tabelle entstehen.

**Alle nicht abgehakten Ausdrücke in der Tabelle sind die Primblöcke (→ Primimplikanten).**

### 4. Schritt:

Umsetzen der entstehenden Primblöcke (Würfel) in Primimplikanten

## Bearbeitung der Überdeckungstabelle (1)

1. Zunächst können die Zeilen der Kernprimimplikanten und die Minterme, die durch sie überdeckt werden, gestrichen werden.
2. **Spaltendominanz:** Minterme, die andere Minterme dominieren, d.h. Spalten, deren "x" die "x" einer anderen Spalte überdecken, können ebenfalls gestrichen werden.
3. **Zeilendominanz:** Dominierte Primimplikanten (deren "x" durch die "x" eines anderen Primimplikanten überdeckt werden, nachdem ein Teil der Minterme gestrichen wurde) sind ebenfalls mögliche „Streichkandidaten“. Zusätzlich Kosten beachten.

# Minimierungsverfahren

- **Algebraische Verfahren:**
  - Vereinfachung mit Hilfe der Gesetze der Booleschen Algebra
  - Nelson-Verfahren (später)
- **Graphische Verfahren**
  - KV-Diagramme
- **Tabellarische Verfahren:**
  - Quine-McCluskey-Verfahren
  - Consensus-Verfahren

**Diese Verfahren bestimmen alle Primterme einer Booleschen Funktion (Schritt 1 der Minimierung)**

## Wdh. Quine-McCluskey-Verfahren

### 1. Schritt:

Die Minterme werden nach der Anzahl der in ihnen vorkommenden nicht negierten Variablen geordnet  
→ **1. Quineschen Tabelle**

### 2. Schritt:

Zwei Ausdrücke, **die sich nur in einer Variablen unterscheiden** werden durch Streichen der unterschiedlichen Variablen zusammengefasst.

Zwei Ausdrücke, aus denen ein neuer entstanden ist, werden **abgehakt** und sind somit **Keine Primimplikanten; sie nehmen jedoch weiter an den Vergleichen teil.**

## Wdh. Beispiel

Primimp.	2	4	5	6	10	12	13	14	18	22	26	30	Kosten
(A)		x	x			x	x						3
B			x		x			x					3
(C)	x			x	x			x	x	x	x	x	2

➔ Primimplikanten A und C überdecken alle Minterme

➔ **Minimalform:**  $f = \bar{b} \bar{c} \bar{e} \vee \bar{a} b$

## Spalten- und Zeilendominanz

### Spaltendominanz:

Spalten, die andere überdecken, können gestrichen werden.

	i	j	k	l	m	n
A	x	x				
B	x	x	x	x		
C			x	x		
D				x	x	x
E					x	x
F	x					x

### Zeilendominanz:

Zeilen mit nur Einträgen, die andere Zeilen haben, können gestrichen werden (falls sie nicht „billiger“ sind)

	i	j	k	l	m	n
A	x	x				
B	x	x	x	x		
C			x	x		
D				x	x	x
E					x	x
F	x					x

# Überdeckungsfunktion (1)

## Systematischer Ansatz:

Es wird eine "Überdeckungsfunktion"  $\ddot{u}_f$  aufgestellt, die für jeden Primimplikanten "x" eine Boolesche Variable  $w_x$  enthält.

Die Variable kennzeichnet, ob der betreffende Primimplikant in die vereinfachte Lösung für f aufgenommen wird oder nicht.

# Überdeckungsfunktion (3)

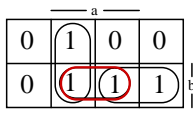
Primimp.	2	4	5	6	10	12	13	14	18	22	26	30	Kosten
A $w_A$		x	x			x	x						3
B $w_B$			x		x			x					3
C $w_C$	x			x	x			x	x	x	x	x	2

Handwritten notes:  $w_A (w_A \vee w_B)$  with arrows pointing to cells (4,5) and (12,13);  $w_B$  with arrow pointing to (13,14);  $(w_B \vee w_C) \dots$  with arrows pointing to (13,14) and (14,18).

# Consensus-Verfahren

## Motivation:

Beispiel (KV-Diagramm):



$$a\bar{c} \vee bc = a\bar{c} \vee bc \vee ab$$

$a\bar{c}, bc, ab$  sind Würfel

$ab$  nennt man Consensussterm oder Consensus-Würfel

**Beachte:** die Variable  $c$  kommt in  $a\bar{c}$  und  $bc$  negiert und nicht negiert vor.

## Beweis

$$xu \vee \bar{x}w = xu \vee \bar{x}w \vee uw$$

Absorptionsgesetz:  $x = x(x \vee w)$   
 $\bar{x} = \bar{x}(\bar{x} \vee u)$

$$x(x \vee w)u \vee \bar{x}(\bar{x} \vee u)w =$$

$$xu \vee xwu \vee \bar{x}w \vee \bar{x}uw =$$

$$xu \vee \bar{x}w \vee uw(x \vee \bar{x}) =$$

$$xu \vee \bar{x}w \vee uw$$

# Überdeckungsfunktion (2)

## Vorgehen:

- Für jeden Minterm verknüpft man zunächst die diesen Minterm überdeckenden Primimplikanten disjunktiv.
- Anschließend bildet man die Konjunktion all dieser Disjunktionen.
- Den erhaltenen konjunktiven Ausdruck formt man in einen disjunktiven um.
- Man sucht im Hinblick auf die Minimalität den konjunktiven Term mit den geringsten Kosten.

# Überdeckungsfunktion (4)

Überdeckungsfunktion für die 2. Quinesche Tabelle:

$$\begin{aligned} \ddot{u}_f &= w_C (w_A \vee w_B) w_A (w_B \vee w_C) w_C (w_A \vee w_B) \\ &= w_A (w_B \vee w_C) w_C w_C w_C w_C \\ &= w_C (w_A \vee w_B) w_A (w_B \vee w_C) \\ &= (w_A w_C \vee w_B w_C) (w_A w_B \vee w_A w_C) \\ &= w_A w_B w_C \vee w_A w_C \\ &= w_A w_C \end{aligned}$$

In der Überdeckungsfunktion für 2. Quinesche Tabelle ist der Term mit der geringsten Variablenanzahl  $w_A w_C$  ( $3 + 2 = 5$ )

➔ **Minimalform:**  $f = \bar{b} c \bar{e} \vee \bar{a} b$

# Consensus-Verfahren

Das Consensus-Verfahren kann als Erweiterung des Quine- McCluskey Verfahrens gesehen werden.

## Consensus-Regel:

In der Booleschen Algebra gelten die beiden folgenden, hier für die Schaltalgebra beschriebenen Identitäten:

$$x u \vee \bar{x} w = x u \vee \bar{x} w \vee u w$$

$$(x \vee W) \cdot (\bar{x} \vee U) = (x \vee W) \cdot (\bar{x} \vee U) \cdot (U \vee W)$$

$x$  ist eine beliebige Variable,

$u$  und  $w$  bzw.  $U$  und  $W$  sind beliebige Schaltfunktionen.

( $u$  und  $w$  sind meist Konjunktionen,  $U$  und  $W$  Disjunktionen in den unabhängigen Variablen)

# Consensus-Verfahren

$v_e = u \cdot w$  **Consensussterm** zu  $x \cdot u$  und  $\bar{x} \cdot w$

$V_e = U \vee W$  **Consensussterm** zu  $x \vee W$  und  $\bar{x} \vee U$

Man schreibt auch:

$v_e = u \cdot w = x \cdot u \ \& \ \bar{x} \cdot w$  bzw.

$V_e = U \vee W = (x \vee W) \ \& \ (\bar{x} \vee U)$

## Beispiel:

$a\bar{c} \vee bc = a\bar{c} \vee bc \vee ab$

$v_e = a b$

# Consensus-Verfahren

## Consensus-Bildung im Würfelmalkül:

Schaltfunktion und Consensussterme werden als Würfel dargestellt.

### Definition 2.13:

Es seien zwei Würfel  $C_1$  und  $C_2$  gegeben, und es gelte  $C_1 \not\subseteq C_2 \neq \emptyset$ . Dann heißt  $C_c = C_1 \not\subseteq C_2$  **Consensus-Würfel**.

$$a\bar{c} \vee bc = a\bar{c} \vee bc \vee ab$$

Im Beispiel:

$$\begin{array}{lcl} a\bar{c} & \Rightarrow & 1 - 0 \quad C_1 \\ b\bar{c} & \Rightarrow & - 1 1 \quad C_2 \\ \vee_c = \underline{ab} & \Rightarrow & 1 1 - \quad C_c \end{array}$$

## Consensus-Würfel (2)

### Verfahren zur Konstruktion :

Es seien  $C_1 = \{c_{11}, \dots, c_{1n}\}$  und  $C_2 = \{c_{21}, \dots, c_{2n}\}$  zwei Würfel mit einem Paar komplementär belegter Komponenten  $c_{1i}$  und  $c_{2i}$  ( $i \in \{1, \dots, n\}$ ), d.h.

$$(c_{1i} = 0) \wedge (c_{2i} = 1) \quad \text{oder} \quad (c_{1i} = 1) \wedge (c_{2i} = 0)$$

Der Consensus-Würfel  $C_c = \{c_{c1}, \dots, c_{cn}\}$  wird dann nach folgender Vorschrift gebildet:

- Die Komponente  $c_{1i}$  im Würfel  $C_1$  und die Komponente  $c_{2i}$  im Würfel  $C_2$  wird "don't care" gesetzt, d.h.  $c_{1i} = -$  und  $c_{2i} = -$
- Dann bildet man die Schnittmenge aus den entstandenen Würfeln:  
 $C_c = C_1 \cap C_2$

$$\begin{array}{l} C_1: \quad 1 - 0 - 1 \\ C_2: \quad - - 0 - 0 \\ \hline C_c = C_1 \cap C_2 \quad \begin{array}{l} 1 - 0 - - \\ - - 0 - - \\ \hline 1 - 0 - - \end{array} \end{array}$$

## Sonderfälle bei der Consensus-Bildung (2)

$$x \cdot u \vee \bar{x} \cdot w = x \cdot u \vee \bar{x} \cdot w \vee u \cdot w$$

### Sonderfall 2:

$u = w$  für alle Belegungen der Variablen von  $u$  und  $w$

$$\text{Dann ist} \quad x u \vee \bar{x} w = x u \vee \bar{x} u = u$$

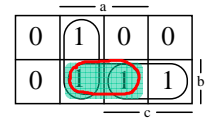
Hier verschmelzen beide erzeugenden Terme zu einem kürzeren.

Es handelt sich um die Zusammenfassungsregel des Quine-McCluskey-Verfahrens

# Consensus-Würfel (1)

Der Consensus-Würfel  $C_c$  ist der **größte Würfel**, der weder in  $C_1$  noch in  $C_2$ , wohl aber in  $C_1 \cup C_2$  enthalten ist, d.h. es gilt  $(C_c \not\subseteq C_1) \wedge (C_c \not\subseteq C_2) \wedge (C_c \subseteq \{C_1 \cup C_2\})$ , und es gibt keinen Würfel  $C \supset C_c$  mit den gleichen Eigenschaften.

Beispiel (KV-Diagramm):



Es gilt:

Der Consensus-Würfel existiert genau dann, wenn es in  $C_1$  und  $C_2$  genau eine komplementär belegte Komponente gibt!

## Beispiel

$$\begin{array}{lcl} a\bar{c} & 1 - 0 & C_1 \\ b\bar{c} & - 1 1 & C_2 \end{array}$$

- setzen

$$\Rightarrow \begin{array}{l} 1 - - \\ - 1 - \end{array}$$

Schnittmenge bilden

$$\Rightarrow \begin{array}{l} \underline{1 1 -} \\ \underline{- 1 -} \\ \vee_c = a b \end{array} \quad C_c$$

## Sonderfälle bei der Consensus-Bildung (1)

$$x \cdot u \vee \bar{x} \cdot w = x \cdot u \vee \bar{x} \cdot w \vee u \cdot w$$

### Sonderfall 1:

$(u \rightarrow w)$  ist wahr für alle Belegungen der Variablen von  $u$  und  $w$ . ( $u$  ist Implikant von  $w$ )  $\Rightarrow u w = u$

$$\text{Dann ist:} \quad x u \vee \bar{x} w = x u \vee \bar{x} w \vee u = u \vee \bar{x} w$$

Der Consensussterm absorbiert einen erzeugenden Term.

Aus einem Implikanten entsteht ein kürzerer Implikant.

## Anwendung der Consensusbildung auf den Schaltzentwurf

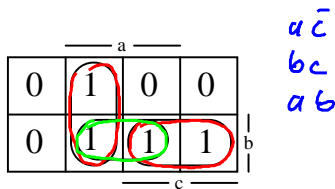
Durch erschöpfende Bildung von Consensus-Würfeln und Streichung aller Würfel, die in einem anderen enthalten sind, können für eine durch beliebige Menge von Implikanten (Einsblöcke) gegebene Schaltfunktion alle Primimplikanten gebildet werden.

➔ Consensus-Verfahren

# Consensus-Verfahren

Unser Beispiel:

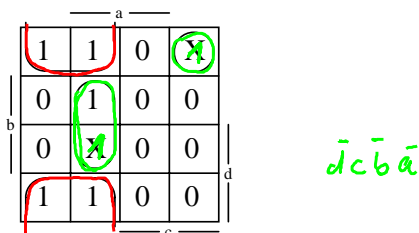
$$\begin{array}{r} 1 - 0 \\ - 1 1 \\ \hline 1 1 - \end{array}$$



Die Primimplikanten  $(1, -, 0)$  und  $(-, 1, 1)$  erzeugen den Consensus-Würfel  $(1, 1, -)$ , der hier ebenfalls ein Primimplikant ist.

## Beispiel

Eine Funktion  $y$  sei durch ihr KV-Diagramm gegeben:



Die "don't care"-Belegungen X werden zu 1 verfügt.

$\Rightarrow C = \{(-, 0, 0, -), (-, 0, 1, 1), (0, 1, 0, 0)\}$ , wobei  $C = (d, c, b, a)$ .

$\bar{c} \bar{b}$     $\bar{c} b a$

# Consensus-Verfahren

Das Verfahren endet, wenn keine Consensus-Würfel oder nur noch in anderen Würfeln enthaltene Consensus-Würfel gebildet werden können.

Nr.	Gebildet aus	Würfel	Gestrichen wegen
1		$-, 0, 0, -$	$\leftarrow$
<del>2</del>		<del><math>-, 0, 1, 1</math></del>	$\subset 4$
<del>3</del>		<del><math>0, 1, 0, 0</math></del>	$\subset 5$
4	2, 1	$-, 0, -, 1$	$\leftarrow$
5	3, 1	$0, -, 0, 0$	$\leftarrow$
	5, 4	<del><math>0, 0, 0, -</math></del>	$\subset 1$

Ergebnis  $C = \{(-, 0, 0, -), (-, 0, -, 1), (0, -, 0, 0)\}$

# Consensus-Verfahren

- > Das Consensusverfahren dient nur der Bestimmung aller Primimplikanten.
- > Das Überdeckungsproblem muss anders, z. B. wie bisher mit der 2. Quineschen Tabelle, gelöst werden.

# Consensus-Verfahren zur Bestimmung aller Primimplikanten

Gegeben: Eine Schaltfunktion durch eine beliebige Menge  $C = \{C_1, \dots, C_n\}$  von Würfeln.

(Bei unvollständig definierten Funktionen werden "don't care" Belegungen zu 1 verfügt)

- Zu je zwei Würfeln  $C_i, C_j \in C$  bildet man, falls möglich, den Consensus-Würfel  $C_e = C_i \cap C_j$ .
- Wenn  $C_e$  in keinem anderen Würfel aus  $C$  enthalten ist, wird  $C_e$  zu  $C$  hinzugefügt, und alle Würfel aus  $C$ , die in  $C_e$  enthalten sind, werden aus  $C$  entfernt.
- Das Verfahren bricht ab, sobald nur noch Consensus-Würfel gebildet werden können, die in anderen Würfeln enthalten sind.

## Consensus-Verfahren

Man stellt die Würfel in einer Liste dar:

Nr.	Gebildet aus	Würfel	Gestrichen wegen
1		$-, 0, 0, -$	
2		$-, 0, 1, 1$	
3		$0, 1, 0, 0$	

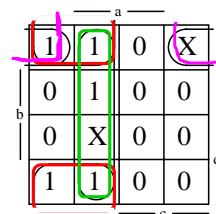
Mit dem zweiten beginnend wird jeder Würfel mit jedem noch nicht gestrichenen, über ihm liegenden Würfel verglichen und der Consensus-Würfel gebildet, falls er existiert.

Dieser Consensus-Würfel wird ans Ende der Liste geschrieben.

Alle im neuen Würfel enthaltenen Würfel werden gestrichen.

## Consensus-Verfahren

Ergebnis  $C = \{(-, 0, 0, -), (-, 0, -, 1), (0, -, 0, 0)\}$  im KV-Diagramm dargestellt:



Für eine Minimalform der Funktion  $y$  kann der Würfel  $(0, -, 0, 0)$  weggelassen werden, da die Einstellen der Funktion auch durch die übrigen Würfel überdeckt werden.

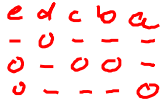
## Weitere Anwendung der Consensus-Bildung

Prüfung, ob eine gegebene Überdeckung sämtliche Primimplikanten enthält:

- > Man bildet zu jedem Würfelpaar den Consensus-Würfel.
- > Falls jeder Consensus-Würfel schon von einem vorhandenen Würfel erfasst wird, dann enthält die Überdeckung alle Primimplikanten der Funktion.

Beispiel:

$$y = \bar{d} \vee \bar{c} \cdot \bar{b} \cdot \bar{e} \vee \bar{e} \cdot \bar{a}$$



Da nur negierte Variablen vorkommen, kann man überhaupt keine Consensus-Würfel bilden.

➔ Der Ausdruck ist die Disjunktion aller Primimplikanten.

## Nelson-Verfahren

Die so entstehenden Ausdrücke werden unter ausschließlicher Verwendung der elementaren booleschen Regeln vereinfacht:

$$\begin{aligned} x \vee x \cdot y &= x & x \cdot (x \vee y) &= x & \text{(Absorptionsgesetz)} \\ x \cdot x &= x & x \vee x &= x & \text{(Idempotenzgesetz)} \\ x \cdot \bar{x} &= 0 & x \vee \bar{x} &= 1 & \text{(Komplementregeln)} \\ x \vee 0 &= x & x \cdot 1 &= x & \end{aligned}$$

Es entsteht die Disjunktion aller Primimplikanten.

## Nelson-Verfahren

Das Verfahren liefert alle Primimplikanten

Die Auswahl einer minimalen Überdeckung muss wie bisher vorgenommen werden, z.B.

- mit der 2. Quineschen Tabelle oder
- mit Hilfe von KV-Diagrammen

## Nelson-Verfahren

Das duale Verfahren erzeugt aus einer beliebigen Menge an Implikanten  $\{K_0, \dots, K_r\}$ , deren Disjunktion die Funktion vollständig beschreibt, d.h.

$$y = K_0 \vee \dots \vee K_r$$

die Konjunktion aller Primimplikate.

Eventuelle "don't care"-Belegungen werden dann zu 0 verfügt.

Nach Ausdistribution und Vereinfachung entsteht die Konjunktion aller Primimplikate.

# Nelson-Verfahren

Das Verfahren von Nelson unterscheidet sich von den anderen dadurch, dass es die **Primimplikanten aus den Nullstellen** und die **Primimplikate aus den Einstellen** der Funktion berechnet.

Will man die Primimplikanten einer Funktion  $y$  berechnen, so benötigt man eine beliebige Menge an Implikanten  $\{D_0, \dots, D_r\}$ , deren Konjunktion die Funktion vollständig beschreibt, d.h.

$$y = D_0 \wedge \dots \wedge D_r$$

Eventuelle "don't care"-Belegungen werden zu 1 verfügt.

Die Implikate werden nach und nach ausdistribuiert.

## Beispiel

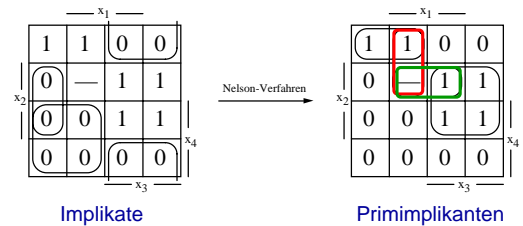
$$\begin{aligned} f &= (\bar{x}_3 \vee x_2) \cdot (\bar{x}_4 \vee x_3) \cdot (x_3 \vee \bar{x}_2 \vee x_1) \\ &= (\bar{x}_3 \bar{x}_4 \vee \bar{x}_3 x_3 \vee x_2 \bar{x}_4 \vee x_2 x_3) \cdot (x_3 \vee \bar{x}_2 \vee x_1) \\ &= \bar{x}_3 \bar{x}_4 x_3 \vee \bar{x}_3 \bar{x}_4 x_1 \vee x_2 \bar{x}_4 x_3 \vee x_2 x_3 x_1 \\ &= \bar{x}_3 \bar{x}_4 \bar{x}_2 \vee \bar{x}_3 \bar{x}_4 x_2 \vee x_2 \bar{x}_4 x_3 \vee x_2 x_3 x_1 \\ &= \bar{x}_3 \bar{x}_4 x_1 \vee x_2 \bar{x}_4 x_1 \vee x_2 x_3 x_1 \\ &= \bar{x}_3 \bar{x}_4 \bar{x}_2 \vee \bar{x}_3 \bar{x}_4 x_1 \vee x_2 \bar{x}_4 x_1 \vee x_2 x_3 \end{aligned}$$

## Beispiel

Aus dem KV-Diagramm erkennt man, dass die Primimplikanten

$$\bar{x}_3 \bar{x}_4 x_1 \text{ und } x_2 \bar{x}_4 x_1$$

für eine vollständige Überdeckung der Einstellen nicht benötigt werden.



Implikate

Primimplikanten

## Zusammenfassung der Verfahren

Aufgabenstellung	1. Schritt Primterme	2. Schritt Auswahl
Variablenanzahl $\leq 6$	KV-Diagramm	KV-Diagramm Überdeckungstabelle
Geg. DF(KF) Ges. DMF(KMF)	Consensus Quine-McCluskey	Überdeckungstabelle
Geg. DF(KF) Ges. KMF(DMF)	Nelson	Überdeckungstabelle

# Vergleich der Verfahren

- Das **KV-Diagramm** ist bei **kleiner Variablenzahl** für den Menschen, jedoch kaum für den Rechner geeignet.
- Das **Consensus-Verfahren** ist sehr rechnergeeignet.
- Das **Consensus-Verfahren** passt seine Rechenzeit der Anfangsüberdeckung weitgehend an. An einer Liste von Einzelbelegungen arbeitet das Verfahren länger als an einer minimalen Liste aus Primimplikanten.  
Anfängliche Würfelmenge aus möglichst wenigen, großen Würfeln zusammenstellen.

## Erweiterter Minimierungsansatz

Es wird versucht, mehrere Boolesche Funktionen gemeinsam zu minimieren.

- Implikanten werden mehrfach ausgenutzt.
- Diese Implikanten müssen nicht notwendigerweise Primimplikanten der einzelnen Funktionen sein.

### ➔ Bündelminimierung

#### Prinzip im KV-Diagramm:

Man sucht nach Implikanten, die in mehreren KV-Diagrammen vorkommen.

Diese **Koppelterme** müssen für mehrere Funktionen nur einmal realisiert werden und sparen dadurch Kosten.

## Primkoppelterm

- Koppelterme müssen keine Primimplikanten sein (siehe Beispiel).
- Sie lassen sich im allgemeinen weiter vereinfachen, jedoch für jede Funktion in unterschiedlicher Weise.
- Koppelterme, die gleichzeitig Primimplikanten sind, heißen **Primkoppelterme**.

## Allgemeine Problematik

- Die Anzahl der Primimplikanten kann exponentiell mit der Anzahl der Eingabevariablen steigen ( $3^n/n$  Primimplikanten)
- Das Überdeckungsproblem ist NP-vollständig
  - Es besteht wenig Hoffnung, einen Algorithmus zu finden, der das Problem in einer Zeit löst, die polynomial mit der Anzahl der Variablen wächst.  
Meist liegt exponentielles Wachstum vor.

### ➔ Heuristische Verfahren

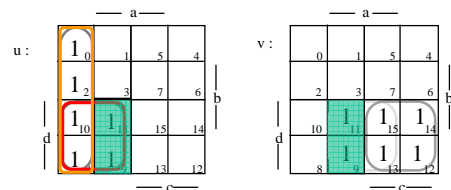
Hierbei werden nicht notwendigerweise minimale Lösungen mit akzeptablem Zeit- und Speicherbedarf erzeugt.

# Vergleich der Verfahren

- Gegenüber dem Quine-McCluskey-Verfahren hat die **Consensus-Methode** die Vorteile, dass sie nur mit Würfeln zu arbeiten braucht und nur eine Liste verwendet, die durch laufende Streichungen kurz gehalten werden kann.
- Erfahrung: Bei sehr unvollständige Schaltfunktionen ist das **Nelson-Verfahren** am schnellsten, weil die Anfangsüberdeckung des **Consensus-Verfahrens** infolge der vielen Freistellenblöcke sehr umfangreich wird.
- Bei ganz oder fast vollständigen Funktionen ist das **Consensus-Verfahren** vorteilhaft.

## Bündelminimierung

### Beispiel:



Ohne Koppelterme:

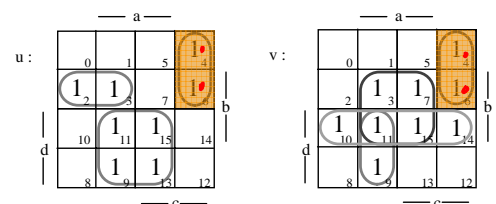
$$u = \bar{a}\bar{c}\bar{v}cd \quad v = ad\bar{v}cd$$

Mit Koppeltermen:

$$u = \bar{a}\bar{c}\bar{v}acd \quad v = cd\bar{v}acd$$

## Primkoppelterm

### Beispiel:



$$u = ad\bar{v}acd\bar{v}b\bar{c}\bar{d}$$

$$v = bd\bar{v}acd\bar{v}ab\bar{v}acd$$

$$\bar{a}\bar{c}\bar{d} \text{ ist Primkoppelterm}$$

## Heuristische Minimierung

- Heuristiken entstehen aus Experimenten und Erfahrungen und liefern mit hoher Wahrscheinlichkeit (je nach Qualität der Heuristik) eine gute wenn nicht sogar die beste Lösung.
- Heuristiken entscheiden bei der Logikminimierung z.B., in welcher Reihenfolge welche Implikanten zusammengefasst werden sollten, um eine möglichst gute Lösung zu erreichen.
- Es wird eine Lösung auf direktem Weg berechnet wird, d.h. ohne Umweg über alle Primimplikanten
  - ➔ Reduzierung von Speicherplatzbedarf und Rechenzeit

- Da der Lösungsweg jedoch durch Heuristiken bestimmt wird, die nur mit einer gewissen Wahrscheinlichkeit die richtigen Entscheidungen treffen und so in Spezialfällen "versagen", können, gibt es keine Garantie dafür, dass die erreichte Lösung die bestmögliche ist.
- Die Abweichung von der bestmöglichen Lösung ist jedoch im Verhältnis zur Größe der Lösung und vor allem zur Rechenzeitersparnis vernachlässigbar klein.

## Der ESPRESSO-Algorithmus

Man verwendet effizientere Datenstrukturen, wie z.B. die sogenannte Positional Cube Notation.

In der Positional Cube Notation wird ein Produktterm binär codiert und so als Vektor aus Nullen und Einsen dargestellt.

Diesen Vektor erhält man, indem man alle Literale des Produktterms einzeln codiert und die Ergebnisse aneinanderfügt.

a	0	1	
$\bar{a}$	1	0	
x	1	1	x: don't cares

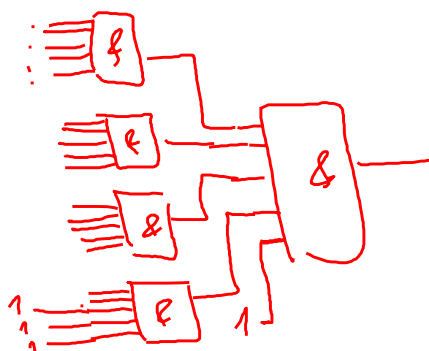
## Der ESPRESSO-Algorithmus

Da der ESPRESSO-Algorithmus relativ komplex ist, fasst man seine verschiedenen Schritte und Phasen in sogenannten **Operatoren** zusammen.

ESPRESSO unterscheidet insgesamt sechs Operatoren: EXPAND, IRREDUNDANT, REDUCE, ESSENTIALS, LAST\_GASP und COMPLEMENT.

Der Ablauf des Algorithmus kann so als Anwendung von Operatoren in einer bestimmten Reihenfolge auf bestimmte Eingabedaten verstanden werden.

UND-Gatter mit 17



- ❑ **MINI-Algorithmus:** entwickelte IBM in den 70er Jahren und für die meisten Probleme der damaligen Zeit eine sehr gute Lösung in akzeptabler Rechenzeit lieferte.
- ❑ **PRESTO-Algorithmus:** Anfang der 80er Jahre
- ❑ **ESPRESSO-Algorithmus:** von IBM und der University of California at Berkeley. Der bekannteste Algorithmus, der seit Anfang der 80er Jahre, als sich mit der nächsten VLSI-Integrationsstufe die Komplexität integrierter Schaltungen noch weiter erhöhte, auch in kommerziellen Logiksynthese-Werkzeugen eingesetzt wird.

## Der ESPRESSO-Algorithmus

Die fehlende Literal-Codierung "00" würde man nach diesem System so interpretieren, dass die entsprechende Variable einen ungültigen Wert annimmt

Jeder Produktterm wird als Zeilenvektor dargestellt, so dass eine Schaltfunktion als Matrix repräsentiert werden kann:

a b c	<u>01</u>	<u>01</u>	<u>01</u>
$\bar{a}$ b c	<u>10</u>	01	01
a $\bar{b}$ c	01	10	01
a b $\bar{c}$	01	01	10
a $\bar{b}$	01	01	<u>11</u>
$\bar{b}c$	11	10	01

a	0	1
$\bar{a}$	1	0
x	1	1

## Letzter Schritt beim Logikentwurf

**Abbildung einer abstrakten logischen Form auf realisierbare Bauelemente.**

**Beispiel:**

Erhält man als Ergebnis der Minimierung einen UND-Term mit 17 Variablen, so kann man diesen Term nicht einfach auf ein UND-Gatter mit 17 Eingängen abbilden.

**Die technische Realisierbarkeit muss als Randbedingung beim Entwurf eingehalten werden.**

**Um auch diesen Schritt zu automatisieren, werden meist Systeme verwendet, die bestimmte logische Formen nach festen Regeln auf Bausteine einer vorgegebenen Bibliothek abbilden.**