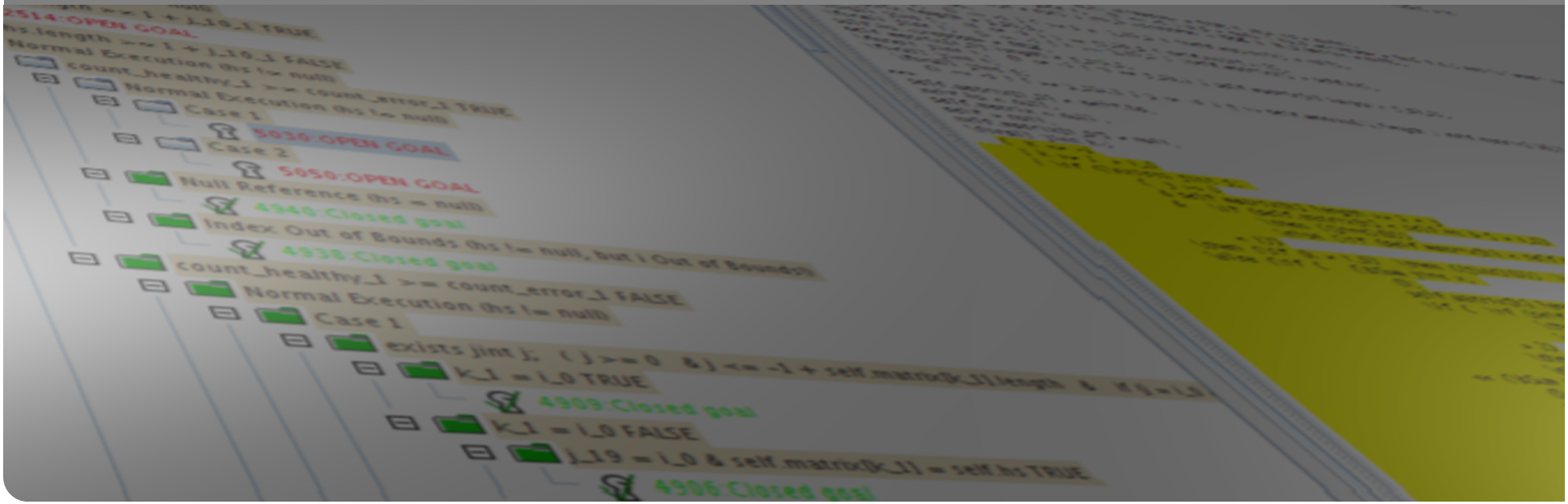
# Formale Systeme

## Prof. Dr. Bernhard Beckert, WS 2019/2020

Binary Decision Diagrams

KIT – Institut für Theoretische Informatik



# Shannon-Formeln

*Shannon-Formeln* sind aussagenlogische Formeln, die aufgebaut sind aus

- dem dreistelligen Operator *sh*

# Shannon-Formeln

*Shannon-Formeln* sind aussagenlogische Formeln, die aufgebaut sind aus

- dem dreistelligen Operator *sh*
- den Konstanten 0 und 1

# Shannon-Formeln

*Shannon-Formeln* sind aussagenlogische Formeln, die aufgebaut sind aus

- dem dreistelligen Operator *sh*
- den Konstanten 0 und 1
- Aussagevariablen $P_1, \ldots, P_n, \ldots$

# Shannon-Formeln

*Shannon-Formeln* sind aussagenlogische Formeln, die aufgebaut sind aus

- dem dreistelligen Operator *sh*
- den Konstanten 0 und 1
- Aussagevariablen $P_1, \ldots, P_n, \ldots$

Semantik von *sh* für eine Interpretation $I$

# Shannon-Formeln

*Shannon-Formeln* sind aussagenlogische Formeln, die aufgebaut sind aus

- dem dreistelligen Operator *sh*
- den Konstanten 0 und 1
- Aussagevariablen $P_1, \ldots, P_n, \ldots$

Semantik von *sh* für eine Interpretation *I*

$$val_I(sh(P_1, P_2, P_3)) = \begin{cases} val_I(P_2) & \text{falls} \quad val_I(P_1) = F \\ val_I(P_3) & \text{falls} \quad val_I(P_1) = W \end{cases}$$

# Shannon-Formeln

*Shannon-Formeln* sind aussagenlogische Formeln, die aufgebaut sind aus

- dem dreistelligen Operator *sh*
- den Konstanten 0 und 1
- Aussagevariablen $P_1, \ldots, P_n, \ldots$

Semantik von *sh* für eine Interpretation *I*

$$val_I(sh(P_1, P_2, P_3)) = \begin{cases} val_I(P_2) & \text{falls} \quad val_I(P_1) = F \\ val_I(P_3) & \text{falls} \quad val_I(P_1) = W \end{cases}$$

oder in Tabellenform:

# Shannon-Formeln

*Shannon-Formeln* sind aussagenlogische Formeln, die aufgebaut sind aus

- dem dreistelligen Operator *sh*
- den Konstanten 0 und 1
- Aussagevariablen $P_1, \ldots, P_n, \ldots$

Semantik von *sh* für eine Interpretation *I*

$$val_I(sh(P_1, P_2, P_3)) = \begin{cases} val_I(P_2) & \text{falls} \quad val_I(P_1) = F \\ val_I(P_3) & \text{falls} \quad val_I(P_1) = W \end{cases}$$

oder in Tabellenform:

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| $val_I(P_1)$ | W | W | W | W | F | F | F | F |
| $val_I(P_2)$ | W | W | F | F | **W** | **W** | **F** | **F** |
| $val_I(P_3)$ | **W** | **F** | **W** | **F** | W | F | W | F |
| $val_I(sh(P_1, P_2, P_3))$ | **W** | **F** | **W** | **F** | **W** | **W** | **F** | **F** |

# Eigenschaften des *sh*-Operators

- $sh(P_1, P_2, P_3) \leftrightarrow (\neg P_1 \wedge P_2) \vee (P_1 \wedge P_3)$

# Eigenschaften des *sh*-Operators

- $sh(P_1, P_2, P_3) \leftrightarrow (\neg P_1 \wedge P_2) \vee (P_1 \wedge P_3)$
- $sh(P_1, P_2, P_3) \leftrightarrow (\neg P_1 \vee P_3) \wedge (P_1 \vee P_2)$

# Eigenschaften des *sh*-Operators

- $sh(P_1, P_2, P_3) \leftrightarrow (\neg P_1 \wedge P_2) \vee (P_1 \wedge P_3)$
- $sh(P_1, P_2, P_3) \leftrightarrow (\neg P_1 \vee P_3) \wedge (P_1 \vee P_2)$
- $sh(P_1, P_2, P_3) \leftrightarrow (P_1 \rightarrow P_3) \wedge (\neg P_1 \rightarrow P_2)$

# Eigenschaften des *sh*-Operators

- $sh(P_1, P_2, P_3) \leftrightarrow (\neg P_1 \wedge P_2) \vee (P_1 \wedge P_3)$
- $sh(P_1, P_2, P_3) \leftrightarrow (\neg P_1 \vee P_3) \wedge (P_1 \vee P_2)$
- $sh(P_1, P_2, P_3) \leftrightarrow (P_1 \rightarrow P_3) \wedge (\neg P_1 \rightarrow P_2)$
- $\neg sh(A, B, C) \leftrightarrow sh(A, \neg B, \neg C)$

# Eigenschaften des *sh*-Operators

- $sh(P_1, P_2, P_3) \leftrightarrow (\neg P_1 \wedge P_2) \vee (P_1 \wedge P_3)$
- $sh(P_1, P_2, P_3) \leftrightarrow (\neg P_1 \vee P_3) \wedge (P_1 \vee P_2)$
- $sh(P_1, P_2, P_3) \leftrightarrow (P_1 \rightarrow P_3) \wedge (\neg P_1 \rightarrow P_2)$
- $\neg sh(A, B, C) \leftrightarrow sh(A, \neg B, \neg C)$
- $sh(0, P_2, P_3) \leftrightarrow P_2$

# Eigenschaften des *sh*-Operators

- $sh(P_1, P_2, P_3) \leftrightarrow (\neg P_1 \wedge P_2) \vee (P_1 \wedge P_3)$
- $sh(P_1, P_2, P_3) \leftrightarrow (\neg P_1 \vee P_3) \wedge (P_1 \vee P_2)$
- $sh(P_1, P_2, P_3) \leftrightarrow (P_1 \rightarrow P_3) \wedge (\neg P_1 \rightarrow P_2)$
- $\neg sh(A, B, C) \leftrightarrow sh(A, \neg B, \neg C)$
- $sh(0, P_2, P_3) \leftrightarrow P_2$
- $sh(1, P_2, P_3) \leftrightarrow P_3$

# Eigenschaften des *sh*-Operators

- $sh(P_1, P_2, P_3) \leftrightarrow (\neg P_1 \wedge P_2) \vee (P_1 \wedge P_3)$
- $sh(P_1, P_2, P_3) \leftrightarrow (\neg P_1 \vee P_3) \wedge (P_1 \vee P_2)$
- $sh(P_1, P_2, P_3) \leftrightarrow (P_1 \rightarrow P_3) \wedge (\neg P_1 \rightarrow P_2)$
- $\neg sh(A, B, C) \leftrightarrow sh(A, \neg B, \neg C)$
- $sh(0, P_2, P_3) \leftrightarrow P_2$
- $sh(1, P_2, P_3) \leftrightarrow P_3$
- $sh(P, 0, 1) \leftrightarrow P$

# Eigenschaften des *sh*-Operators

- $sh(P_1, P_2, P_3) \leftrightarrow (\neg P_1 \wedge P_2) \vee (P_1 \wedge P_3)$
- $sh(P_1, P_2, P_3) \leftrightarrow (\neg P_1 \vee P_3) \wedge (P_1 \vee P_2)$
- $sh(P_1, P_2, P_3) \leftrightarrow (P_1 \rightarrow P_3) \wedge (\neg P_1 \rightarrow P_2)$
- $\neg sh(A, B, C) \leftrightarrow sh(A, \neg B, \neg C)$
- $sh(0, P_2, P_3) \leftrightarrow P_2$
- $sh(1, P_2, P_3) \leftrightarrow P_3$
- $sh(P, 0, 1) \leftrightarrow P$
- $sh(P, 1, 0) \leftrightarrow \neg P$

# Eigenschaften des *sh*-Operators

- $sh(P_1, P_2, P_3) \leftrightarrow (\neg P_1 \wedge P_2) \vee (P_1 \wedge P_3)$
- $sh(P_1, P_2, P_3) \leftrightarrow (\neg P_1 \vee P_3) \wedge (P_1 \vee P_2)$
- $sh(P_1, P_2, P_3) \leftrightarrow (P_1 \rightarrow P_3) \wedge (\neg P_1 \rightarrow P_2)$
- $\neg sh(A, B, C) \leftrightarrow sh(A, \neg B, \neg C)$
- $sh(0, P_2, P_3) \leftrightarrow P_2$
- $sh(1, P_2, P_3) \leftrightarrow P_3$
- $sh(P, 0, 1) \leftrightarrow P$
- $sh(P, 1, 0) \leftrightarrow \neg P$
- $sh(P_1, P_2, P_2) \leftrightarrow P_2$

# Eigenschaften des *sh*-Operators

- $sh(P_1, P_2, P_3) \leftrightarrow (\neg P_1 \wedge P_2) \vee (P_1 \wedge P_3)$
- $sh(P_1, P_2, P_3) \leftrightarrow (\neg P_1 \vee P_3) \wedge (P_1 \vee P_2)$
- $sh(P_1, P_2, P_3) \leftrightarrow (P_1 \rightarrow P_3) \wedge (\neg P_1 \rightarrow P_2)$
- $\neg sh(A, B, C) \leftrightarrow sh(A, \neg B, \neg C)$
- $sh(0, P_2, P_3) \leftrightarrow P_2$
- $sh(1, P_2, P_3) \leftrightarrow P_3$
- $sh(P, 0, 1) \leftrightarrow P$
- $sh(P, 1, 0) \leftrightarrow \neg P$
- $sh(P_1, P_2, P_2) \leftrightarrow P_2$
- $sh(sh(P_1, P_2, P_3), P_4, P_5) \quad \leftrightarrow$
  $sh(P_1, sh(P_2, P_4, P_5), sh(P_3, P_4, P_5))$

# Eigenschaften des *sh*-Operators

- $sh(P_1, P_2, P_3) \leftrightarrow (\neg P_1 \wedge P_2) \vee (P_1 \wedge P_3)$
- $sh(P_1, P_2, P_3) \leftrightarrow (\neg P_1 \vee P_3) \wedge (P_1 \vee P_2)$
- $sh(P_1, P_2, P_3) \leftrightarrow (P_1 \rightarrow P_3) \wedge (\neg P_1 \rightarrow P_2)$
- $\neg sh(A, B, C) \leftrightarrow sh(A, \neg B, \neg C)$
- $sh(0, P_2, P_3) \leftrightarrow P_2$
- $sh(1, P_2, P_3) \leftrightarrow P_3$
- $sh(P, 0, 1) \leftrightarrow P$
- $sh(P, 1, 0) \leftrightarrow \neg P$
- $sh(P_1, P_2, P_2) \leftrightarrow P_2$
- $sh(sh(P_1, P_2, P_3), P_4, P_5) \quad \leftrightarrow$
  $sh(P_1, sh(P_2, P_4, P_5), sh(P_3, P_4, P_5))$
- $A \leftrightarrow sh(P, A_{P=0}, A_{P=1})$

# Normierte Shannon-Formeln

Wir fixieren eine Ordnung auf der Menge der Aussagevariablen, etwa die durch die Ordnung der Indizes gegebene.

## Definition

# Normierte Shannon-Formeln

Wir fixieren eine Ordnung auf der Menge der Aussagevariablen, etwa die durch die Ordnung der Indizes gegebene.

## Definition

1. Die Konstanten $0, 1$ sind normierte *sh*-Formeln.

# Normierte Shannon-Formeln

Wir fixieren eine Ordnung auf der Menge der Aussagevariablen, etwa die durch die Ordnung der Indizes gegebene.

## Definition

1. Die Konstanten $0, 1$ sind normierte *sh*-Formeln.
2. $sh(P_i, A, B)$ ist eine normierte *sh*-Formel wenn

# Normierte Shannon-Formeln

Wir fixieren eine Ordnung auf der Menge der Aussagevariablen, etwa die durch die Ordnung der Indizes gegebene.

## Definition

1. Die Konstanten $0, 1$ sind normierte *sh*-Formeln.
2. $sh(P_i, A, B)$ ist eine normierte *sh*-Formel wenn
   - $A$ und $B$ normierte *sh*-Formeln sind und

# Normierte Shannon-Formeln

Wir fixieren eine Ordnung auf der Menge der Aussagevariablen, etwa die durch die Ordnung der Indizes gegebene.

## Definition

1. Die Konstanten $0, 1$ sind normierte *sh*-Formeln.
2. $sh(P_i, A, B)$ ist eine normierte *sh*-Formel wenn
   - $A$ und $B$ normierte *sh*-Formeln sind und
   - für jede in $A$ oder $B$ vorkommende Variable $P_j$ gilt $i < j$.

# Normierte Shannon-Formeln

Wir fixieren eine Ordnung auf der Menge der Aussagevariablen, etwa die durch die Ordnung der Indizes gegebene.

## Definition

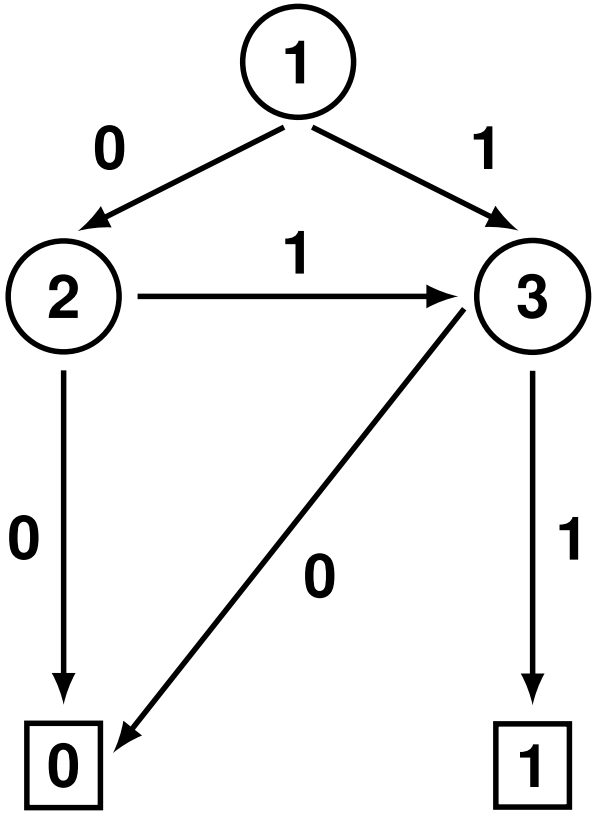
1. Die Konstanten $0, 1$ sind normierte *sh*-Formeln.
2. $sh(P_i, A, B)$ ist eine normierte *sh*-Formel wenn
   - $A$ und $B$ normierte *sh*-Formeln sind und
   - für jede in $A$ oder $B$ vorkommende Variable $P_j$ gilt $i < j$.

## Theorem

*Zu jeder aussagenlogischen Formel gibt es eine äquivalente normierte sh-Formel.*

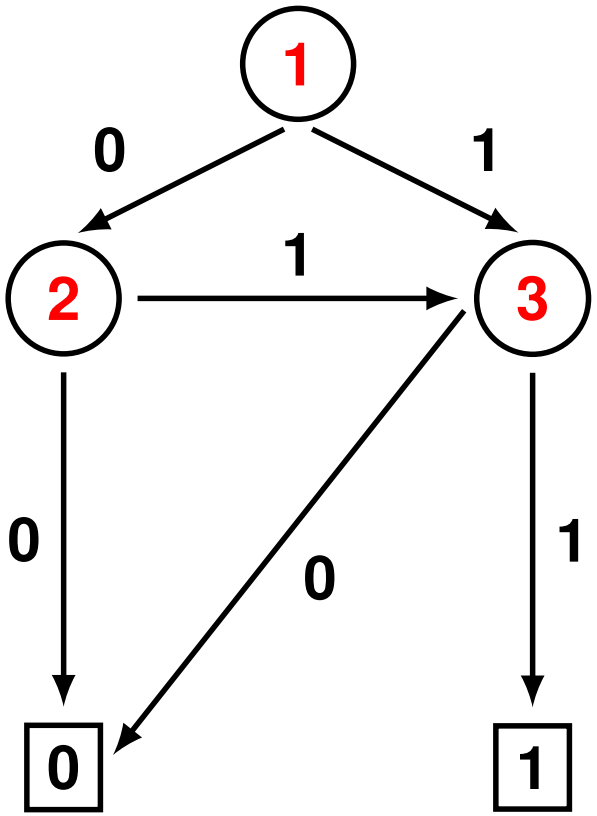
# Shannon-Graphen

# Shannon-Graphen



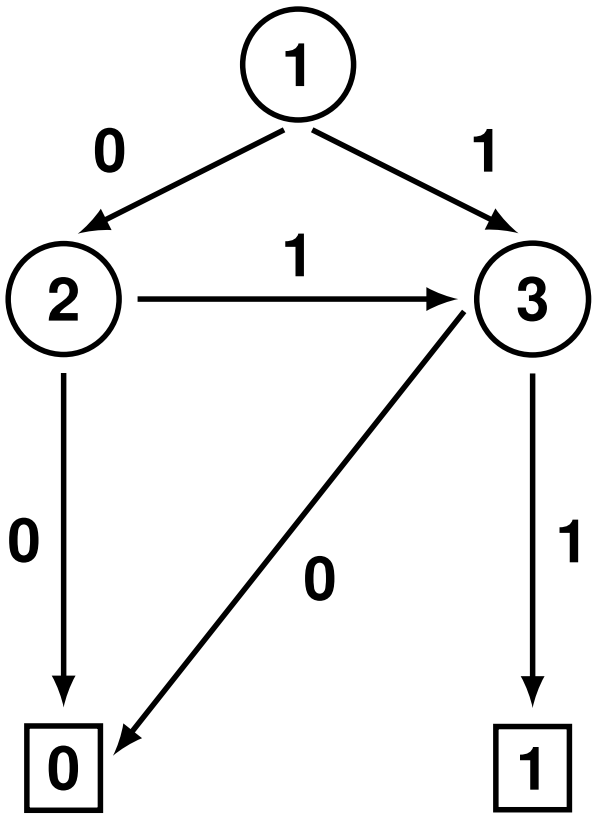

Ein *sh-Graph* ist ein gerichteter, binärer, zusammenhängender Graph.

# Shannon-Graphen



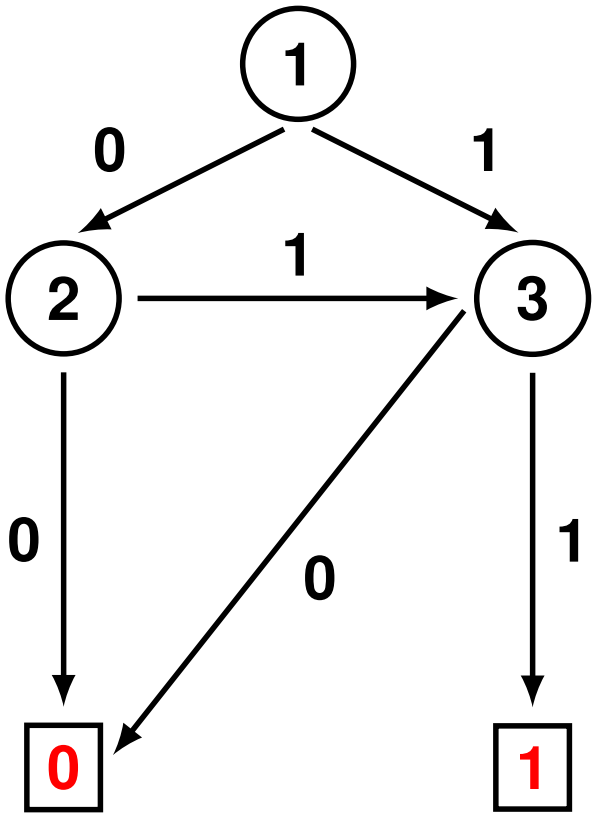

Jedem nichtterminalen Knoten $v$ ist eine natürliche Zahl $index(v)$ zugeordnet.

# Shannon-Graphen



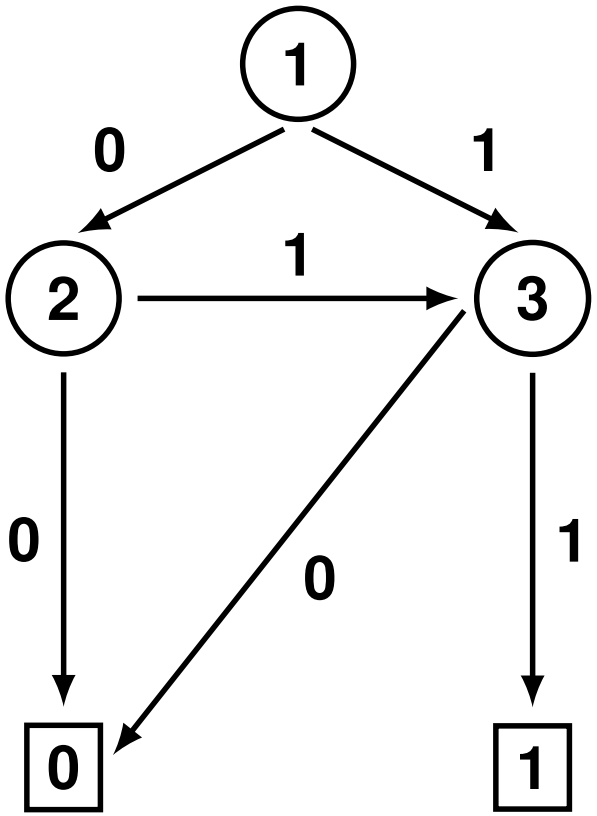

Von jedem nichtterminalen Knoten $v$ gehen zwei Kanten aus. Eine davon ist mit **0**, die andere mit **1** gekennzeichnet.

# Shannon-Graphen



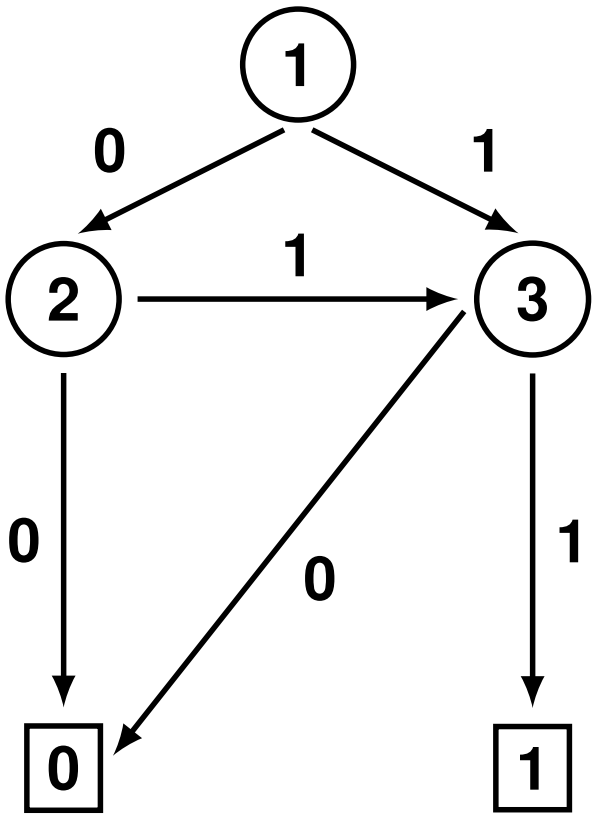

Jeder terminale Knoten $v$ ist mit **0** oder **1** versehen. Wir definieren dafür $wert(v)$ durch $wert(\mathbf{0}) = \mathbf{F}$ und $wert(\mathbf{1}) = \mathbf{W}$.

# Shannon-Graphen



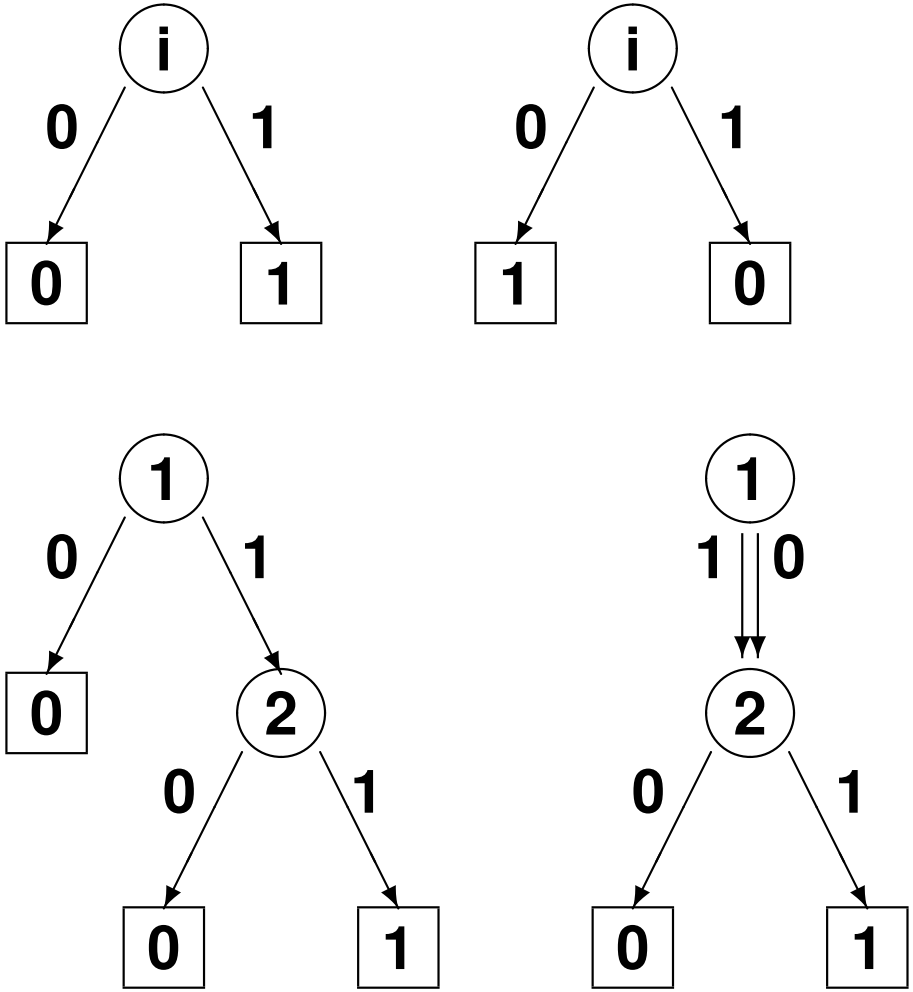

Ist der nichtterminale Knoten $w$ ein unmittelbarer Nachfolger von $v$, dann gilt $index(v) < index(w)$.

# Shannon-Graphen




Es gibt genau einen Wurzelknoten.

# Weitere Beispiele von Shannon-Graphen



# Shannon-Graphen und Boolesche Funktionen

- Jedem *sh*-Graphen $G$ kann man eine $m$-stellige Boolesche Funktion $f_G$ zuordnen, wobei $m$ die Anzahl der in $G$ vorkommenden verschiedenen Indizes $i_1, \ldots, i_m$ ist.

# Shannon-Graphen und Boolesche Funktionen

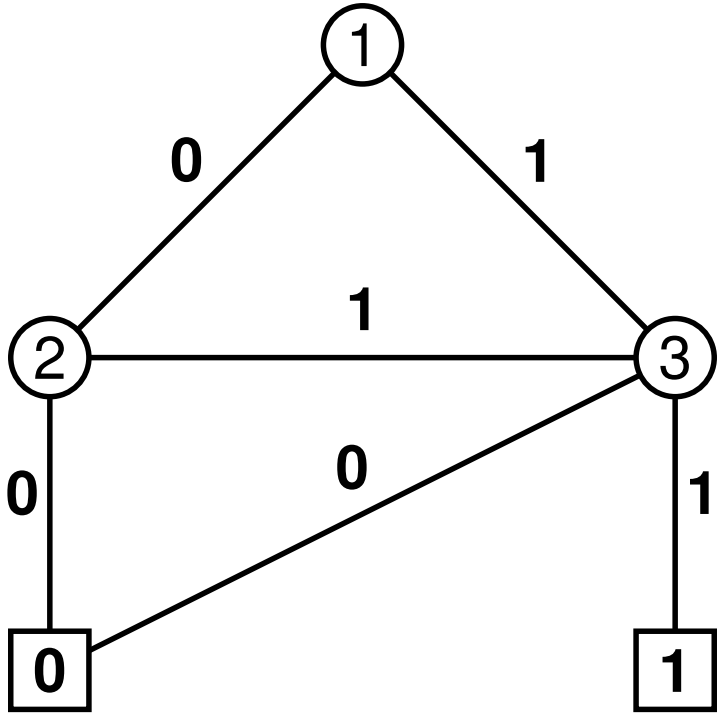
- Jedem *sh*-Graphen $G$ kann man eine $m$-stellige Boolesche Funktion $f_G$ zuordnen, wobei $m$ die Anzahl der in $G$ vorkommenden verschiedenen Indizes $i_1, \ldots, i_m$ ist.
- Wir fassen $f_G$ als eine Funktion mit den Eingabevariabeln $P_{i_1}, \ldots, P_{i_m}$ auf und bestimmen den Funktionswert $f_G(P_{i_1}, \ldots, P_{i_m})$, indem wir an der Wurzel von $G$ beginnend einen Pfad durch $G$ wählen. Am Knoten $v$ folgen wir der Kante **0**, wenn die Eingabevariable $P_{index(v)}$ den Wert $F$ hat, sonst der Kante **1**.

# Shannon-Graphen und Boolesche Funktionen

- Jedem *sh*-Graphen $G$ kann man eine $m$-stellige Boolesche Funktion $f_G$ zuordnen, wobei $m$ die Anzahl der in $G$ vorkommenden verschiedenen Indizes $i_1, \ldots, i_m$ ist.
- Wir fassen $f_G$ als eine Funktion mit den Eingabevariabeln $P_{i_1}, \ldots, P_{i_m}$ auf und bestimmen den Funktionswert $f_G(P_{i_1}, \ldots, P_{i_m})$, indem wir an der Wurzel von $G$ beginnend einen Pfad durch $G$ wählen. Am Knoten $v$ folgen wir der Kante **0**, wenn die Eingabevariable $P_{index(v)}$ den Wert $F$ hat, sonst der Kante **1**.
- Der Wert des terminalen Knotens ist der gesuchte Funktionswert.

# Shannongraph als Boolesche Funktion

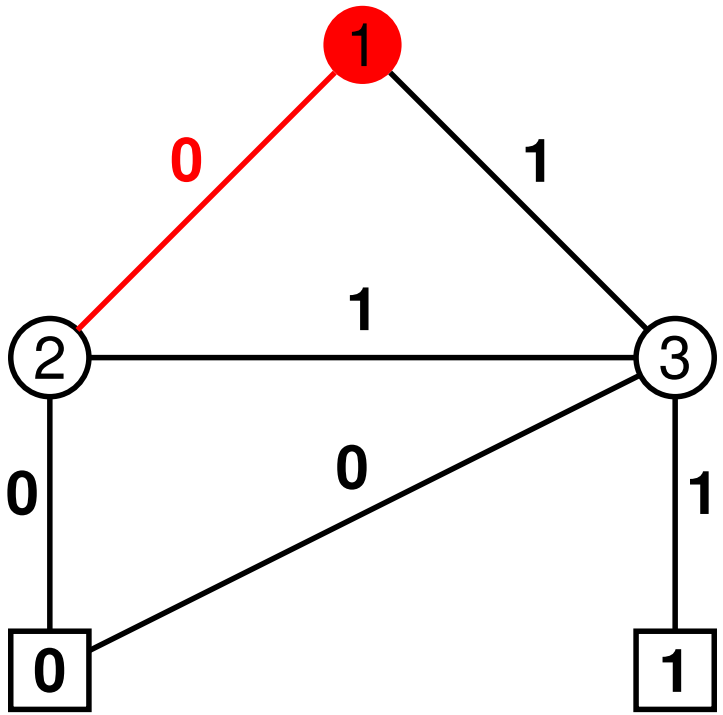
**G:**




$f_G(F, W, F) = ?$

# Shannongraph als Boolesche Funktion

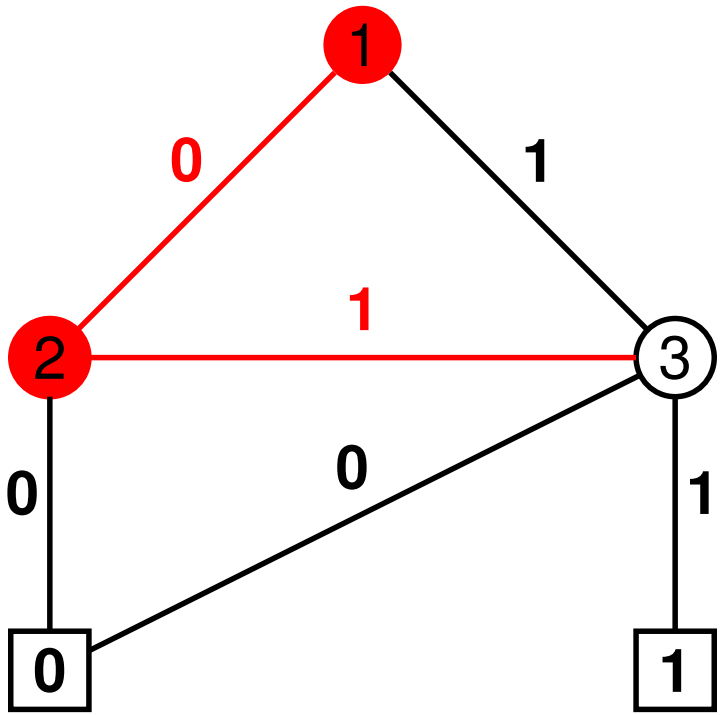
**G:**




$f_G(F, W, F) = ?$

# Shannongraph als Boolesche Funktion

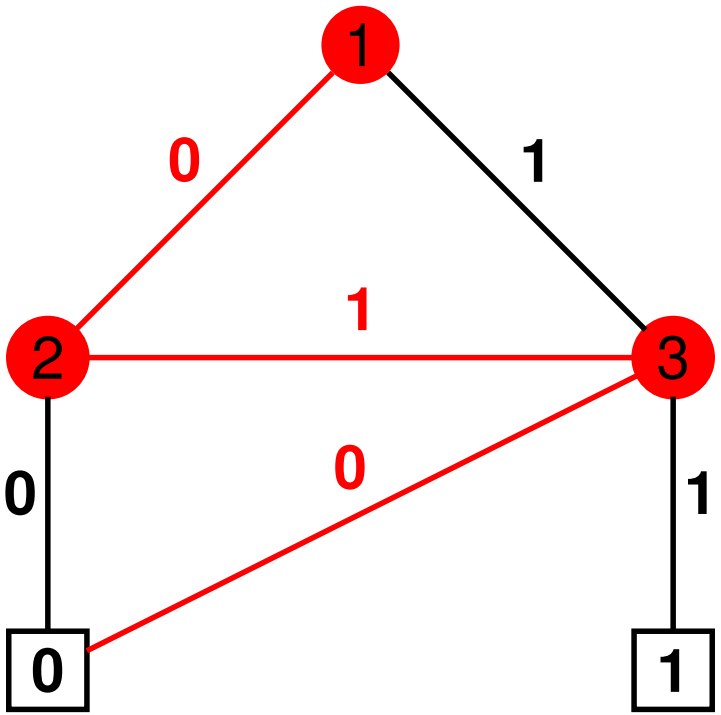
**G:**




$f_G(F, W, F) = ?$

# Shannongraph als Boolesche Funktion

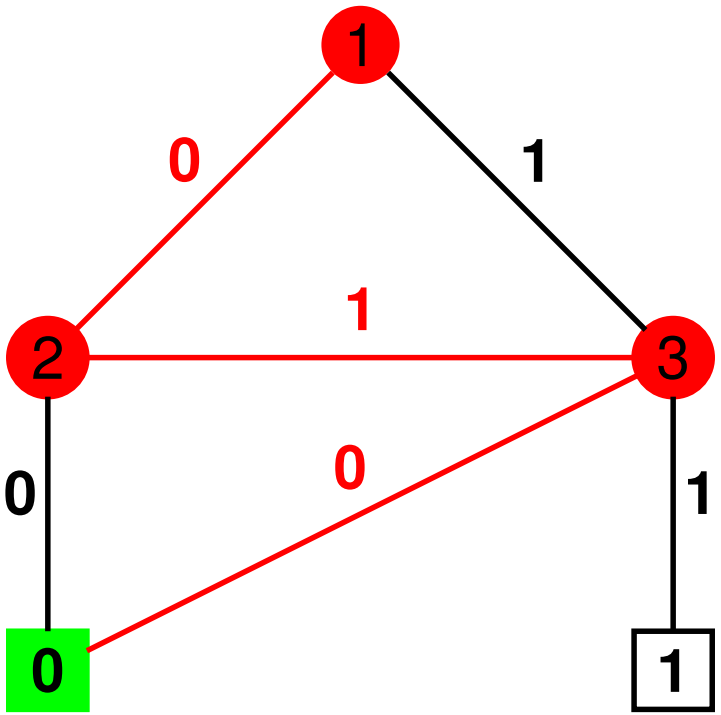
**G:**




$f_G(F, W, F) = ?$

# Shannongraph als Boolesche Funktion

**G:**




$f_G(F, W, F) = F$

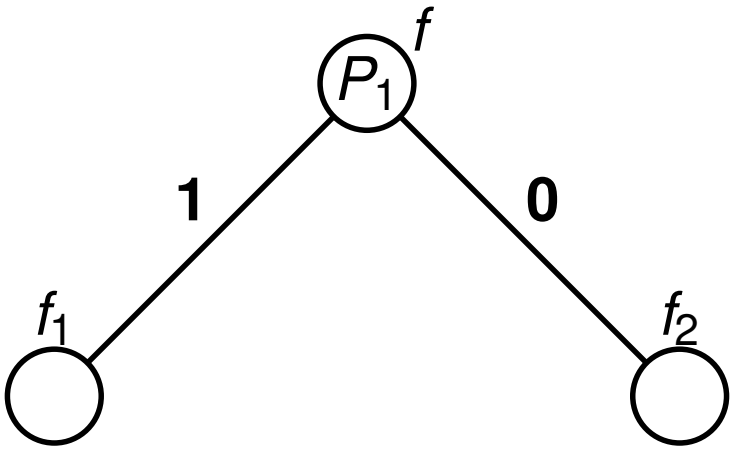
# Konstruktion von Shannon-Graphen

Beispiel

$$f(P_1, P_2, P_3) = \begin{cases} W & \text{falls } \#\{1 \leq i \leq 3 \mid P_i = W\} = 2 \\ F & \text{sonst} \end{cases}$$

# Konstruktion von Shannon-Graphen

Beispiel



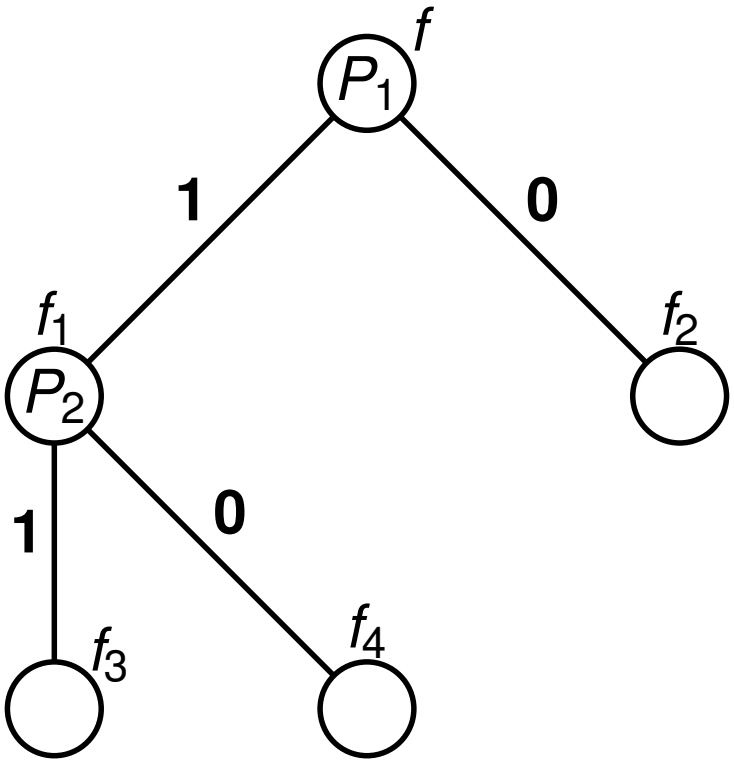

$$f(P_1, P_2, P_3) = \begin{cases} W & \text{falls } \#\{1 \leq i \leq 3 \mid P_i = W\} = 2 \\ F & \text{sonst} \end{cases}$$

$$f_1(P_2, P_3) = f(W, P_2, P_3) = \begin{cases} W & \text{falls } (P_2 = W \text{ und } P_3 = F) \text{ oder} \\ & \qquad (P_2 = F \text{ und } P_3 = W) \\ F & \text{sonst} \end{cases}$$

$$f_2(P_2, P_3) = f(F, P_2, P_3) = \begin{cases} W & \text{falls } P_2 = P_3 = W \\ F & \text{sonst} \end{cases}$$

# Konstruktion von Shannon-Graphen

Beispiel




$$f(P_1, P_2, P_3) = \begin{cases} W & \text{falls } \#\{1 \leq i \leq 3 \mid P_i = W\} = 2 \\ F & \text{sonst} \end{cases}$$

$$f_1(P_2, P_3) = f(W, P_2, P_3) = \begin{cases} W & \text{falls } (P_2 = W \text{ und } P_3 = F) \text{ oder} \\ & \qquad (P_2 = F \text{ und } P_3 = W) \\ F & \text{sonst} \end{cases}$$

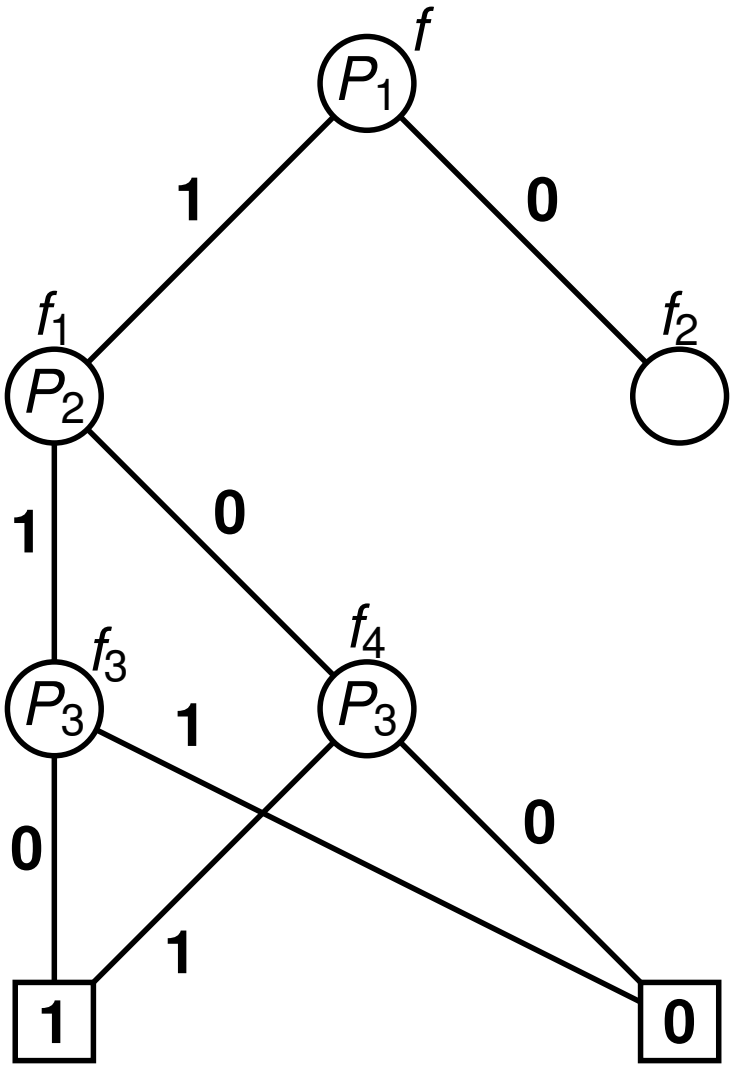
$$f_2(P_2, P_3) = f(F, P_2, P_3) = \begin{cases} W & \text{falls } P_2 = P_3 = W \\ F & \text{sonst} \end{cases}$$

$$f_3(P_3) = f(W, W, P_3) = \neg P_3$$

$$f_4(P_3) = f(W, F, P_3) = P_3$$

# Konstruktion von Shannon-Graphen

Beispiel




$f(P_1, P_2, P_3) =$

$$\begin{cases} W & \text{falls } \#\{1 \leq i \leq 3 \mid P_i = W\} = 2 \\ F & \text{sonst} \end{cases}$$

$f_1(P_2, P_3) = f(W, P_2, P_3) =$

$$\begin{cases} W & \text{falls } (P_2 = W \text{ und } P_3 = F) \text{ oder} \\ & \qquad (P_2 = F \text{ und } P_3 = W) \\ F & \text{sonst} \end{cases}$$

$f_2(P_2, P_3) = f(F, P_2, P_3) =$

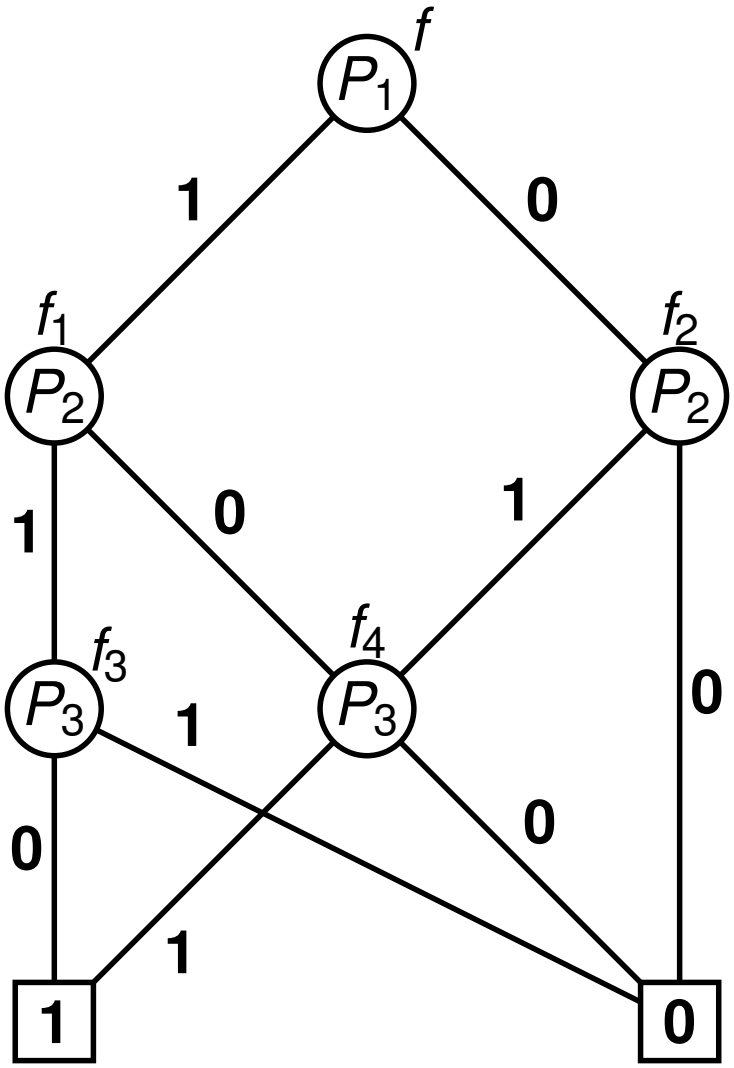
$$\begin{cases} W & \text{falls } P_2 = P_3 = W \\ F & \text{sonst} \end{cases}$$

$f_3(P_3) = f(W, W, P_3) = \neg P_3$

$f_4(P_3) = f(W, F, P_3) = P_3$

# Konstruktion von Shannon-Graphen

Beispiel




$f(P_1, P_2, P_3) =$

$$\begin{cases} W & \text{falls } \#\{1 \leq i \leq 3 \mid P_i = W\} = 2 \\ F & \text{sonst} \end{cases}$$

$f_1(P_2, P_3) = f(W, P_2, P_3) =$

$$\begin{cases} W & \text{falls } (P_2 = W \text{ und } P_3 = F) \text{ oder} \\ & \qquad (P_2 = F \text{ und } P_3 = W) \\ F & \text{sonst} \end{cases}$$

$f_2(P_2, P_3) = f(F, P_2, P_3) =$

$$\begin{cases} W & \text{falls } P_2 = P_3 = W \\ F & \text{sonst} \end{cases}$$

$f_3(P_3) = f(W, W, P_3) = \neg P_3$

$f_4(P_3) = f(W, F, P_3) = P_3$

$f_5(P_3) = f(F, F, P_3) = F$

$f_6(P_3) = f(F, W, P_3) = P_3 = f_4(P_3)$

# Shannon-Graphen vs normierte Shannon-Formeln

Es gibt eine offensichtliche **Korrespondenz** zwischen **Shannon-Graphen** und **normierten Shannon-Formeln**:

$n$-te Variable entspricht Knoten mit Index $n$.

Von jetzt an betrachten wir nur noch Shannon-Graphen.

# Reduzierte Shannon-Graphen

## Definition

Ein *sh*-Graph heißt *reduziert*, wenn

1. es keine zwei Knoten $v$ und $w$ ($v \neq w$) gibt, so daß der in $v$ verwurzelte Teilgraph $G_v$ mit dem in $w$ verwurzelten Teilgraph $G_w$ isomorph ist.

# Reduzierte Shannon-Graphen

## Definition

Ein *sh*-Graph heißt *reduziert*, wenn

1. es keine zwei Knoten $v$ und $w$ ($v \neq w$) gibt, so daß der in $v$ verwurzelte Teilgraph $G_v$ mit dem in $w$ verwurzelten Teilgraph $G_w$ isomorph ist.
2. es keinen Knoten $v$ gibt, so dass die beiden von $v$ ausgehenden Kanten zum selben Nachfolgerknoten führen.

# Reduzierte Shannon-Graphen

## Definition

Ein *sh*-Graph heißt *reduziert*, wenn

1. es keine zwei Knoten $v$ und $w$ ($v \neq w$) gibt, so daß der in $v$ verwurzelte Teilgraph $G_v$ mit dem in $w$ verwurzelten Teilgraph $G_w$ isomorph ist.
2. es keinen Knoten $v$ gibt, so dass die beiden von $v$ ausgehenden Kanten zum selben Nachfolgerknoten führen.

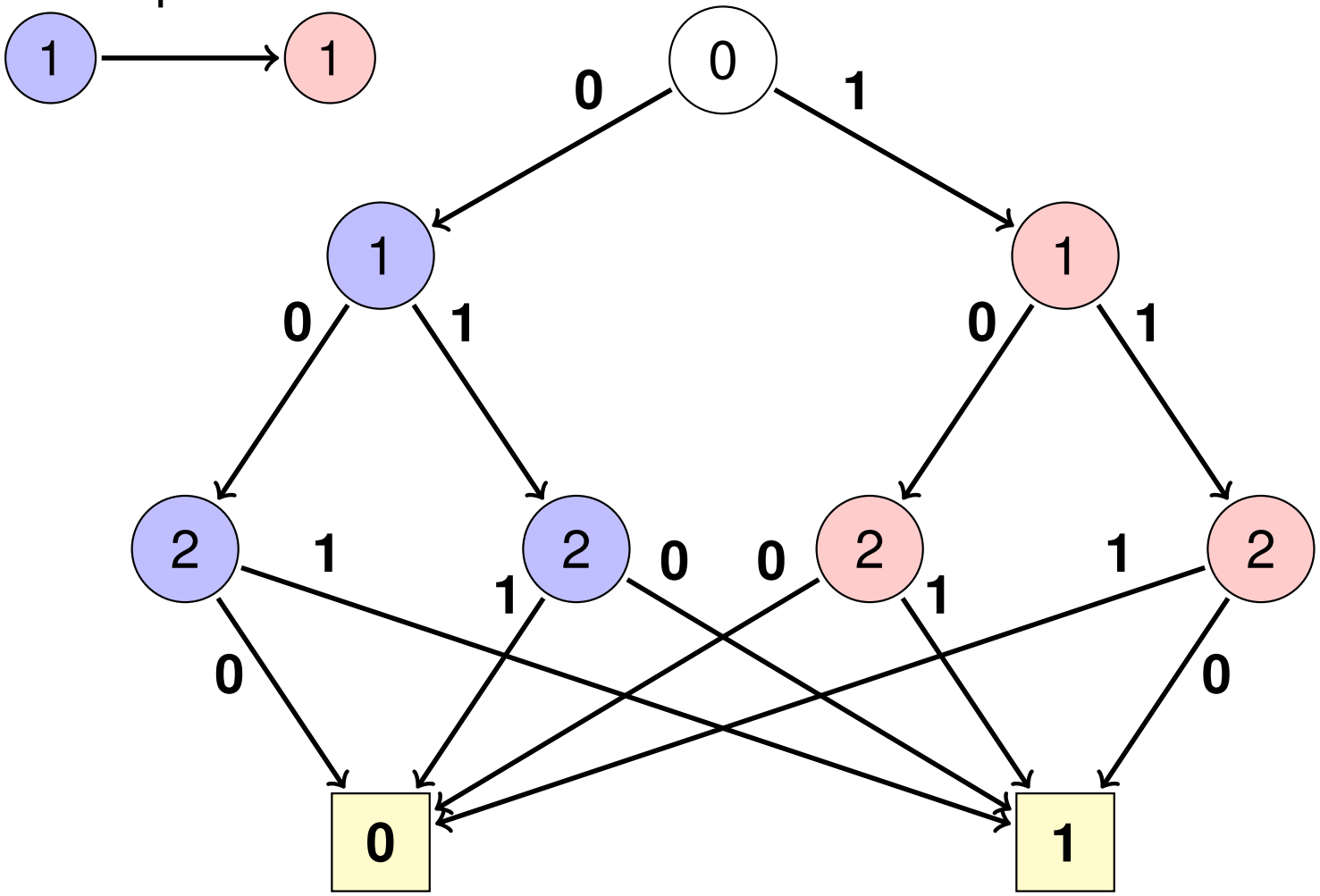
Ein reduzierter Shannongraph heißt auch *ordered binary decision diagram*: (O)BDD.

# Einfachstes Beispiel isomorpher Shannon-Graphen

# Beispiel für Reduktion

Elimination isomorpher Subgraphen

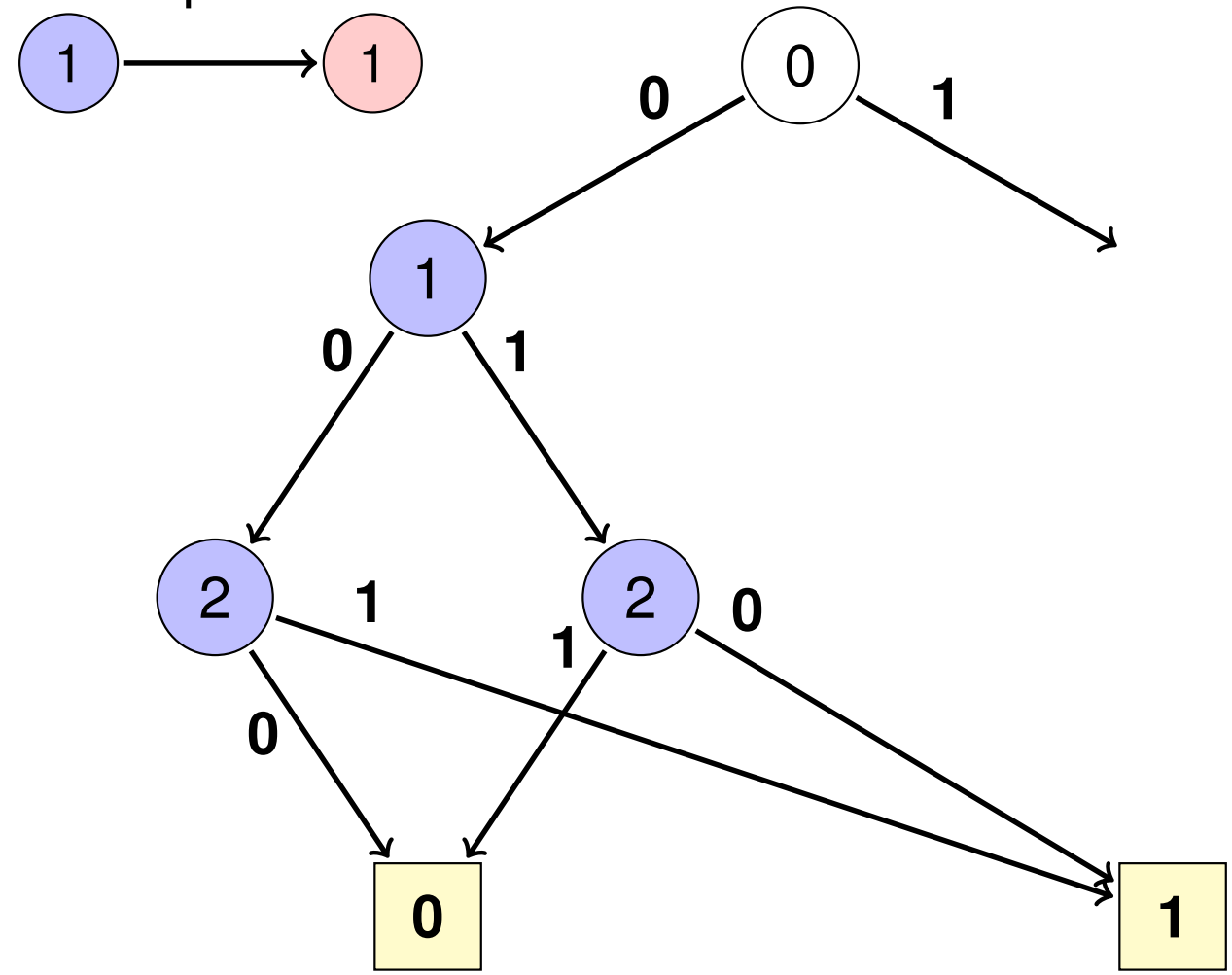
Isomorphismus $\pi$ :



# Beispiel für Reduktion

Elimination isomorpher Subgraphen

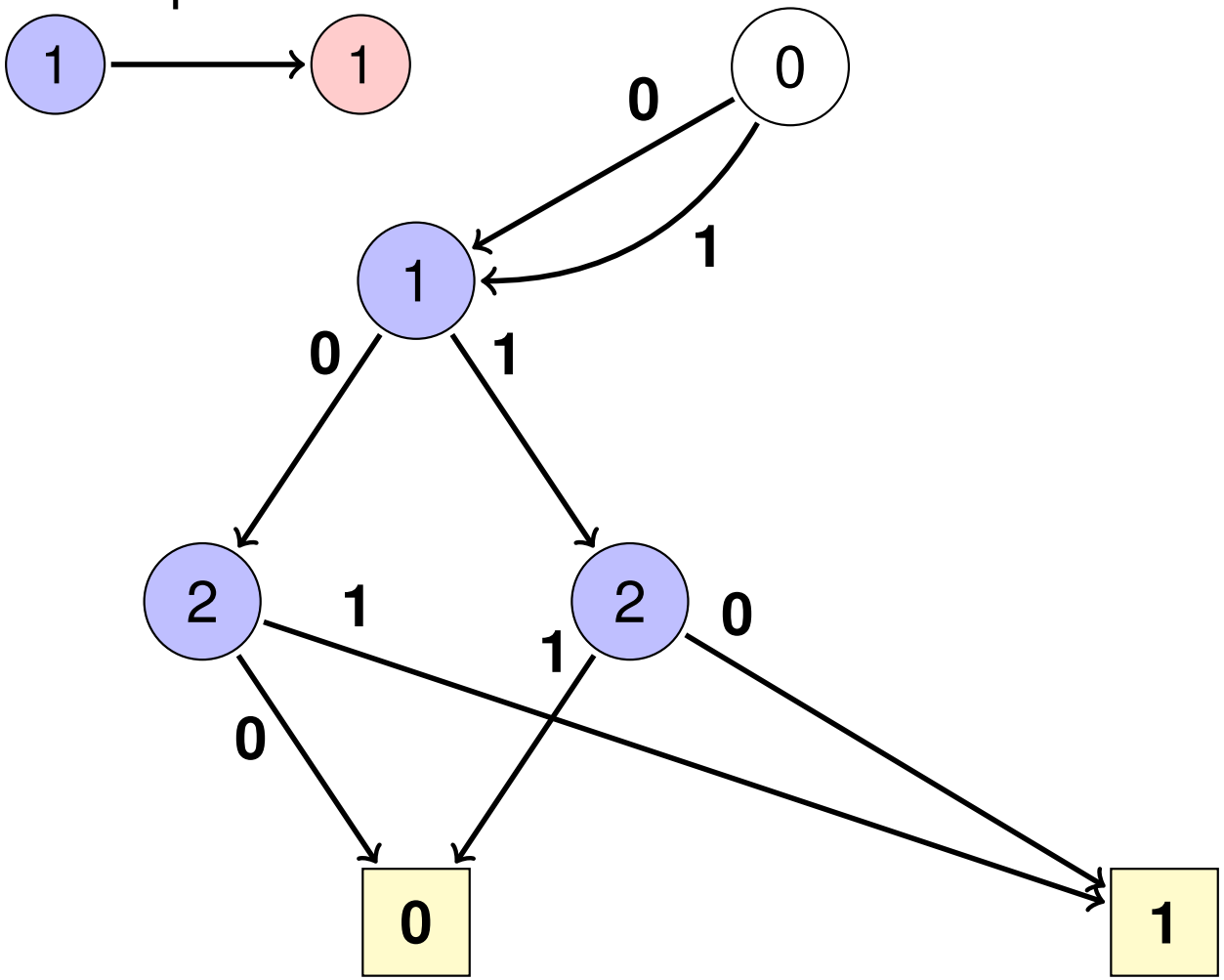
Isomorphismus $\pi$ :



# Beispiel für Reduktion

Elimination doppelter Kanten

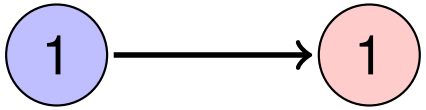
Isomorphismus $\pi$ :

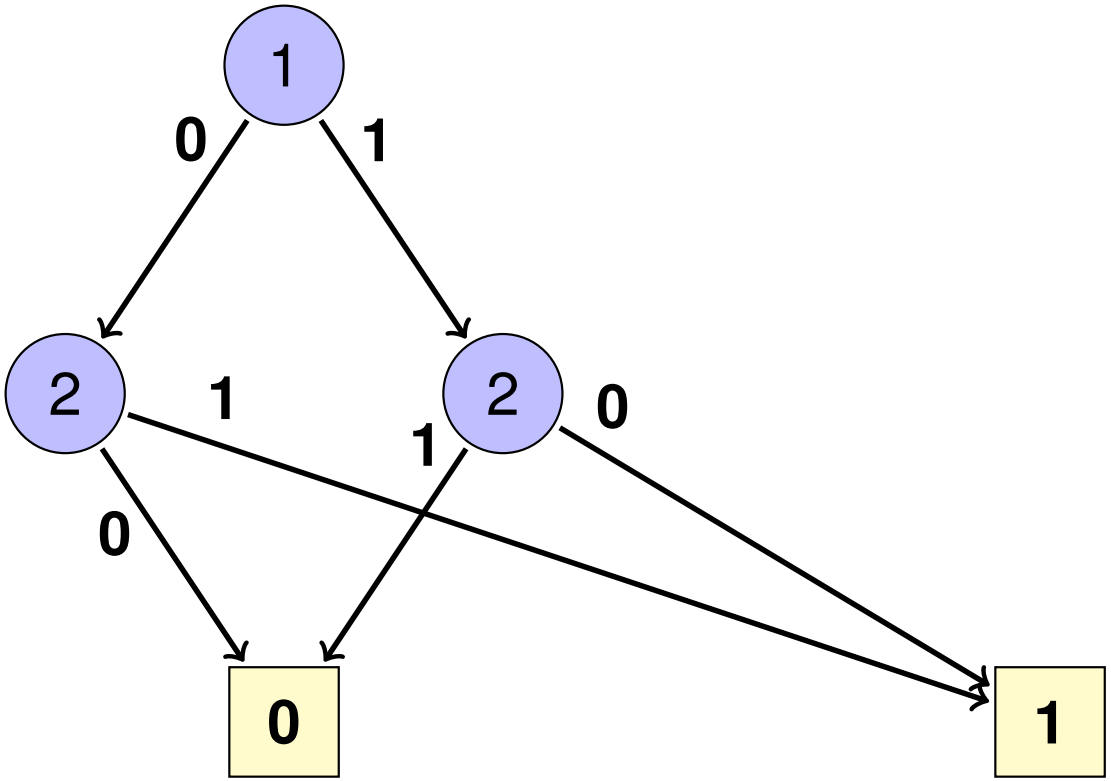


# Beispiel für Reduktion

Elimination doppelter Kanten

Isomorphismus $\pi$ :



# Isomorphie von Shannon-Graphen

## Definition

Seien zwei *sh*-Graphen $H, G$ gegeben. Ihre Knotenmengen seien $V_1, V_2$.
$H, G$ heißen zueinander *isomorph* ($H \cong G$) genau dann, wenn es eine bijektive Abbildung $\pi$ von $V_1$ nach $V_2$ gibt mit:

# Isomorphie von Shannon-Graphen

## Definition

Seien zwei *sh*-Graphen $H, G$ gegeben. Ihre Knotenmengen seien $V_1, V_2$.
$H, G$ heißen zueinander *isomorph* ($H \cong G$) genau dann, wenn es eine bijektive Abbildung $\pi$ von $V_1$ nach $V_2$ gibt mit:

1. $index(k) = index(\pi(k))$ für jeden Nichtterminalknoten $k \in V_1$

# Isomorphie von Shannon-Graphen

## Definition

Seien zwei *sh*-Graphen $H, G$ gegeben. Ihre Knotenmengen seien $V_1, V_2$.
$H, G$ heißen zueinander *isomorph* ($H \cong G$) genau dann, wenn es eine bijektive Abbildung $\pi$ von $V_1$ nach $V_2$ gibt mit:

1. $index(k) = index(\pi(k))$ für jeden Nichtterminalknoten $k \in V_1$
2. $wert(k) = wert(\pi(k))$ für jeden Terminalknoten $k \in V_1$

# Isomorphie von Shannon-Graphen

## Definition

Seien zwei *sh*-Graphen $H, G$ gegeben. Ihre Knotenmengen seien $V_1, V_2$.
$H, G$ heißen zueinander *isomorph* ($H \cong G$) genau dann, wenn es eine bijektive Abbildung $\pi$ von $V_1$ nach $V_2$ gibt mit:

1. $index(k) = index(\pi(k))$ für jeden Nichtterminalknoten $k \in V_1$
2. $wert(k) = wert(\pi(k))$ für jeden Terminalknoten $k \in V_1$
3. Für jeden Nichtterminalknoten $k \in V_1$, dessen **0**-Kante/**1**-Kante zu dem Knoten $k_0/k_1$ führt, gilt: die **0**-Kante von $\pi(k)$ führt zu $\pi(k_0)$, die **1**-Kante zu $\pi(k_1)$.

# Ein Kriterium für Reduziertheit

## Theorem

*Sei $G$ ein Shannongraph, so daß für jedes Paar von Knoten $v, w$ gilt*

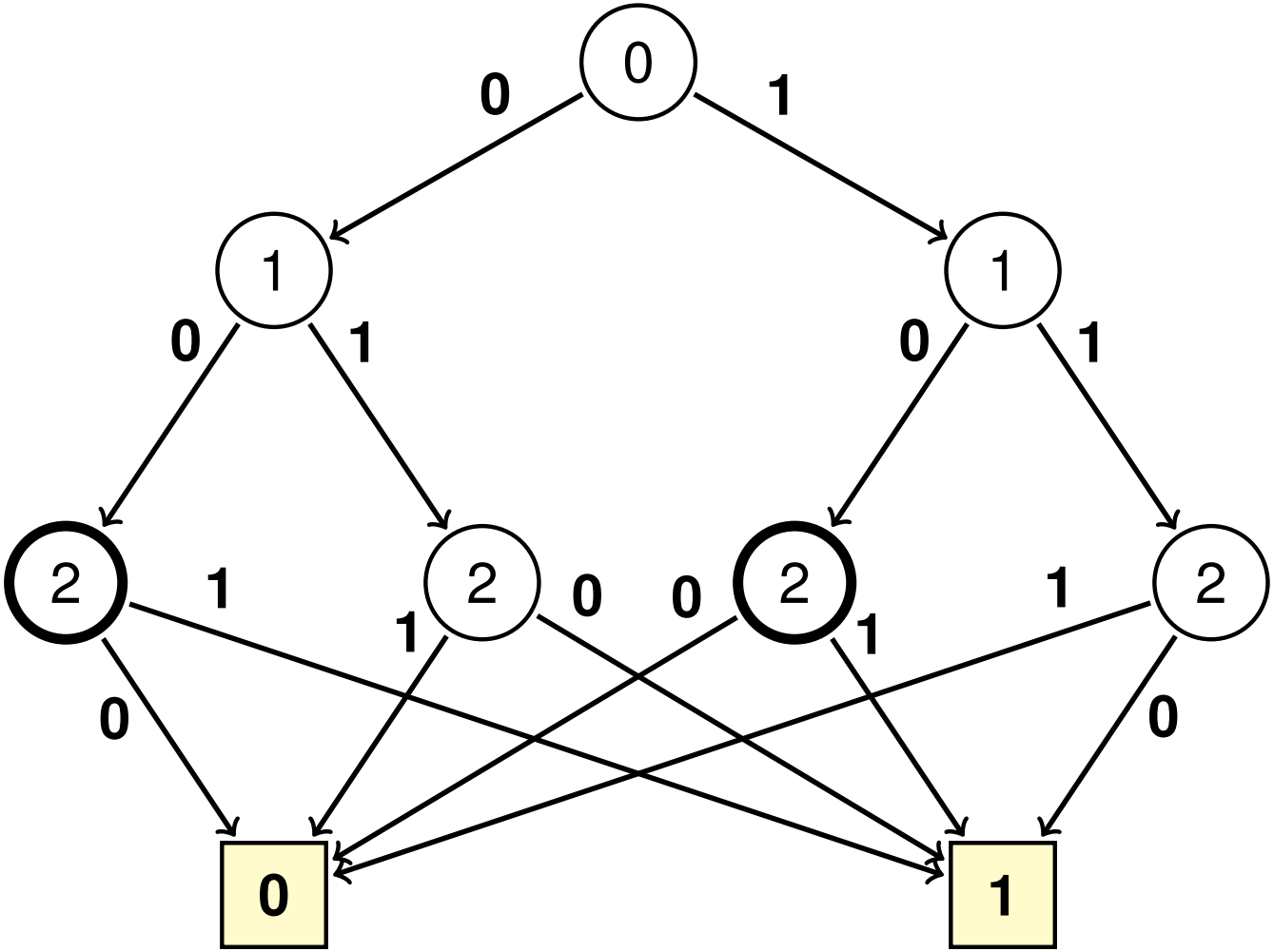
$$\begin{array}{ll} \textit{wenn} & \textit{index}(v) = \textit{index}(w), \\ & \textit{die } \mathbf{1}\textit{-Nachfolger von } v \textit{ und } w \textit{ identisch sind und} \\ & \textit{die } \mathbf{0}\textit{-Nachfolger von } v \textit{ und } w \textit{ identisch sind} \\ \textit{dann} & v = w \end{array}$$

*Dann erfüllt $G$ die Bedingung (1) aus der Definition reduzierter Shannongraphen, d.h. für jedes Paar $x, y$ von Knoten gilt*

$$\begin{array}{ll} \textit{wenn} & G_x \textit{ isomorph zu } G_y \textit{ ist} \\ \textit{dann} & x = y \end{array}$$

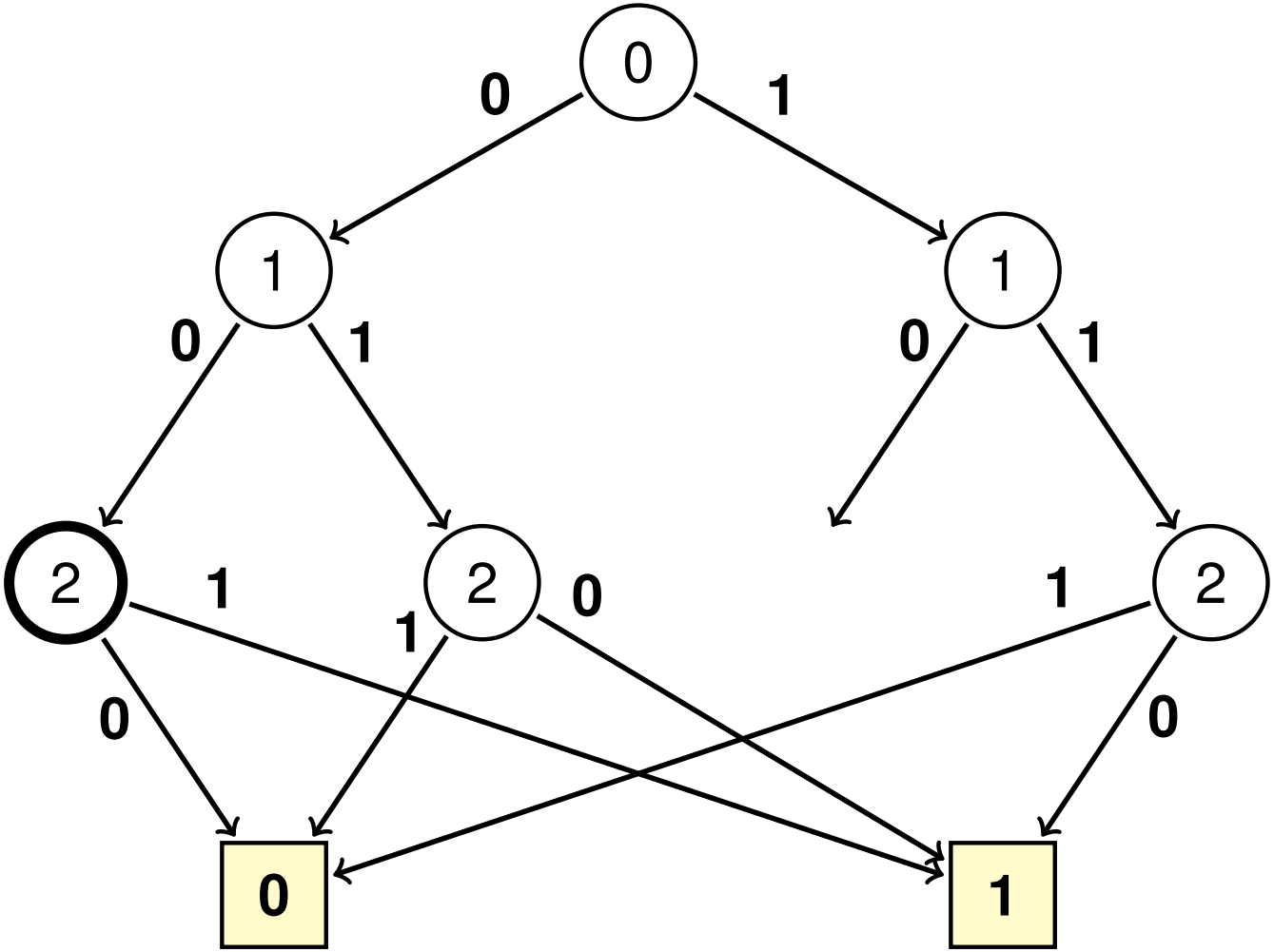
# Beispiel für Reduktion

Nun inkrementell



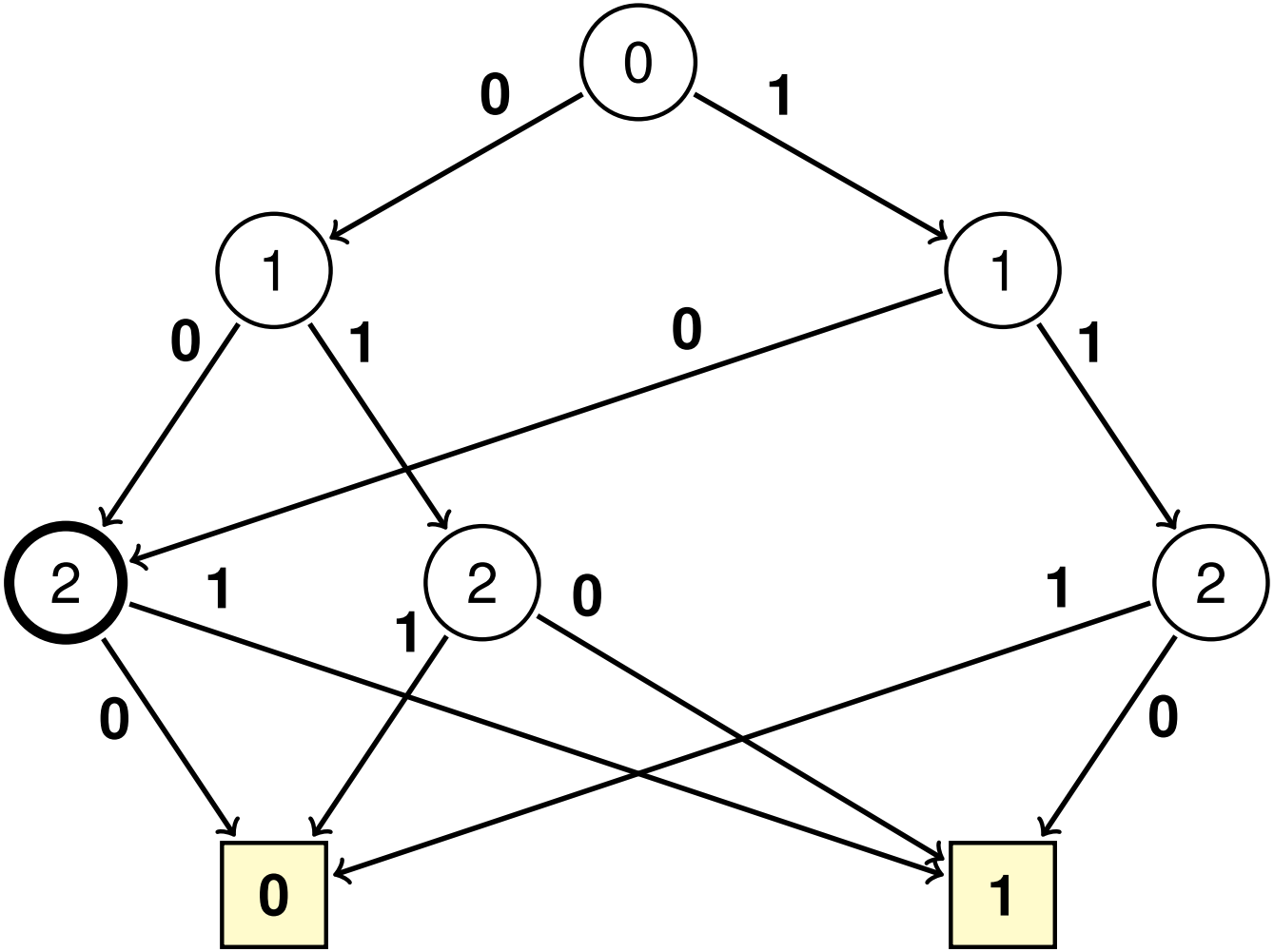
# Beispiel für Reduktion

Nun inkrementell



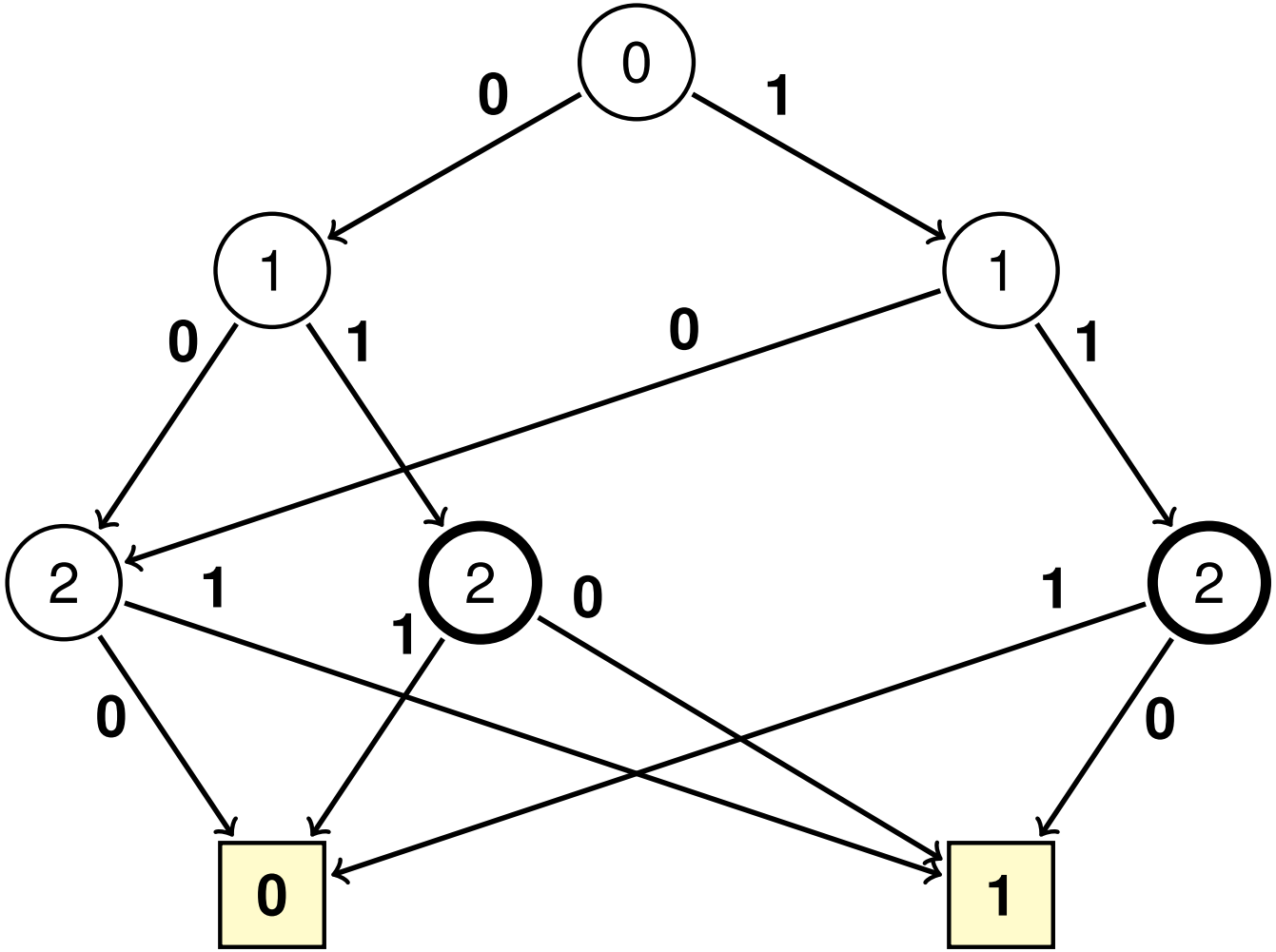
# Beispiel für Reduktion

Nun inkrementell



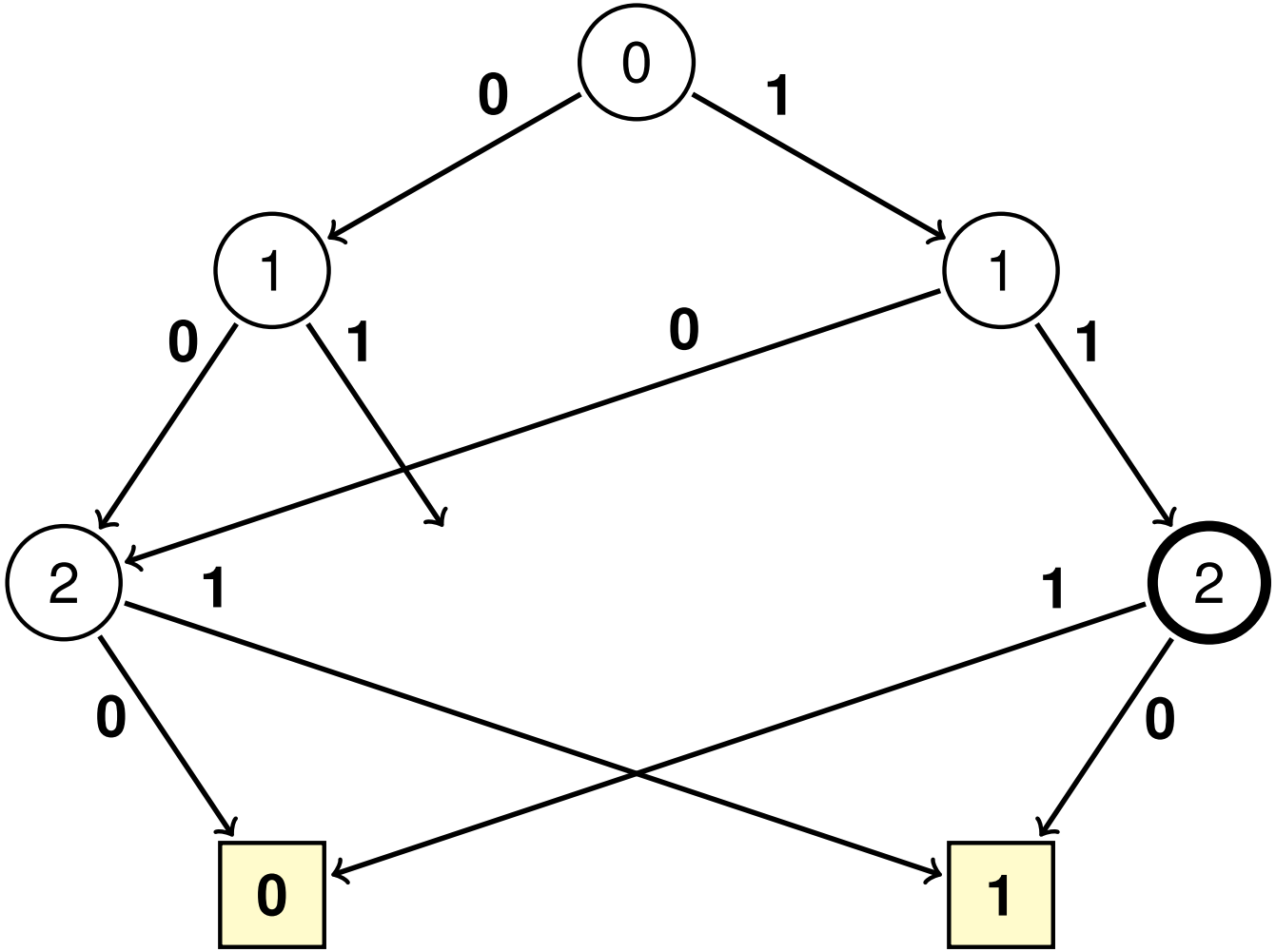
# Beispiel für Reduktion

Nun inkrementell



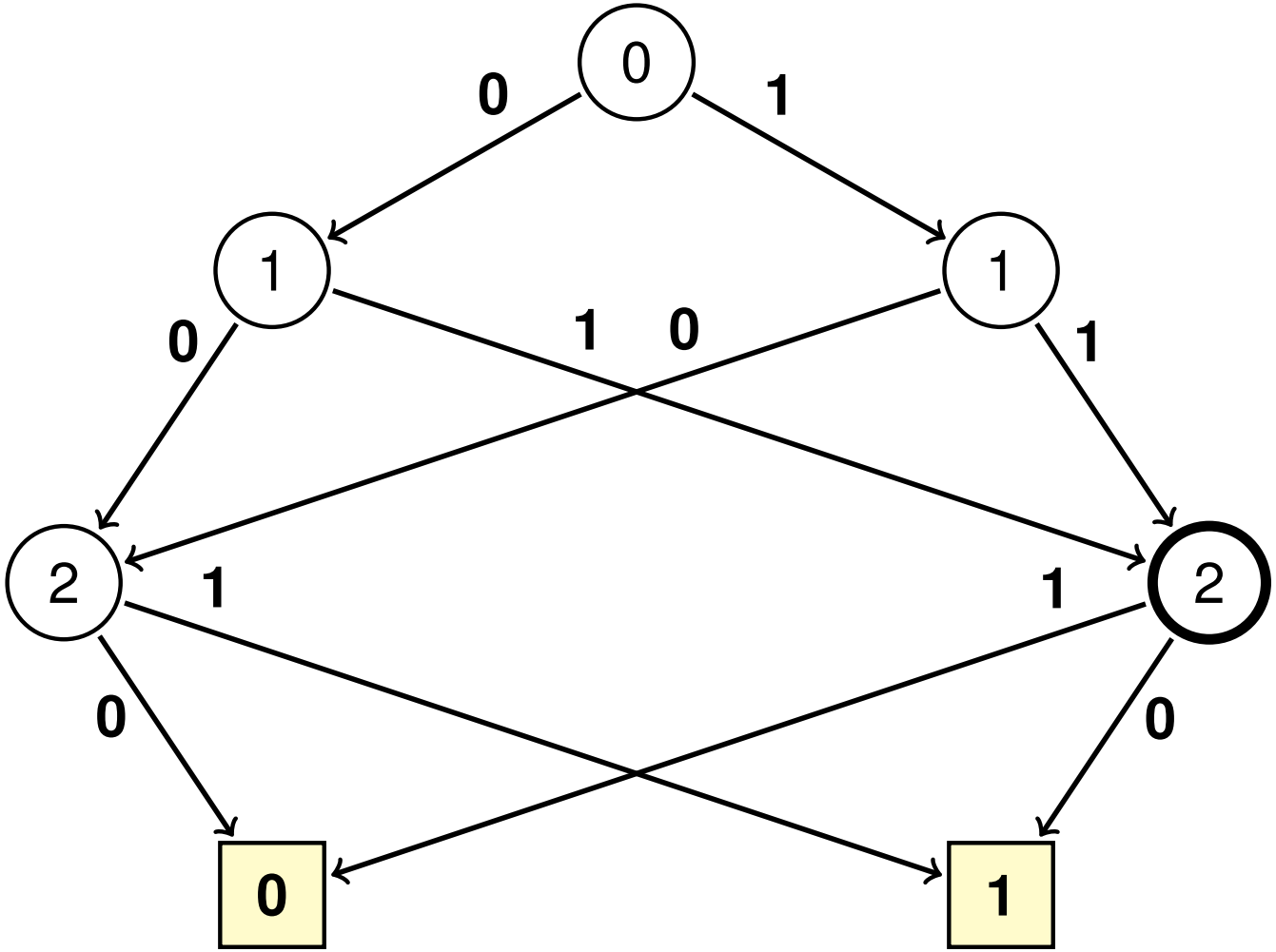
# Beispiel für Reduktion

Nun inkrementell



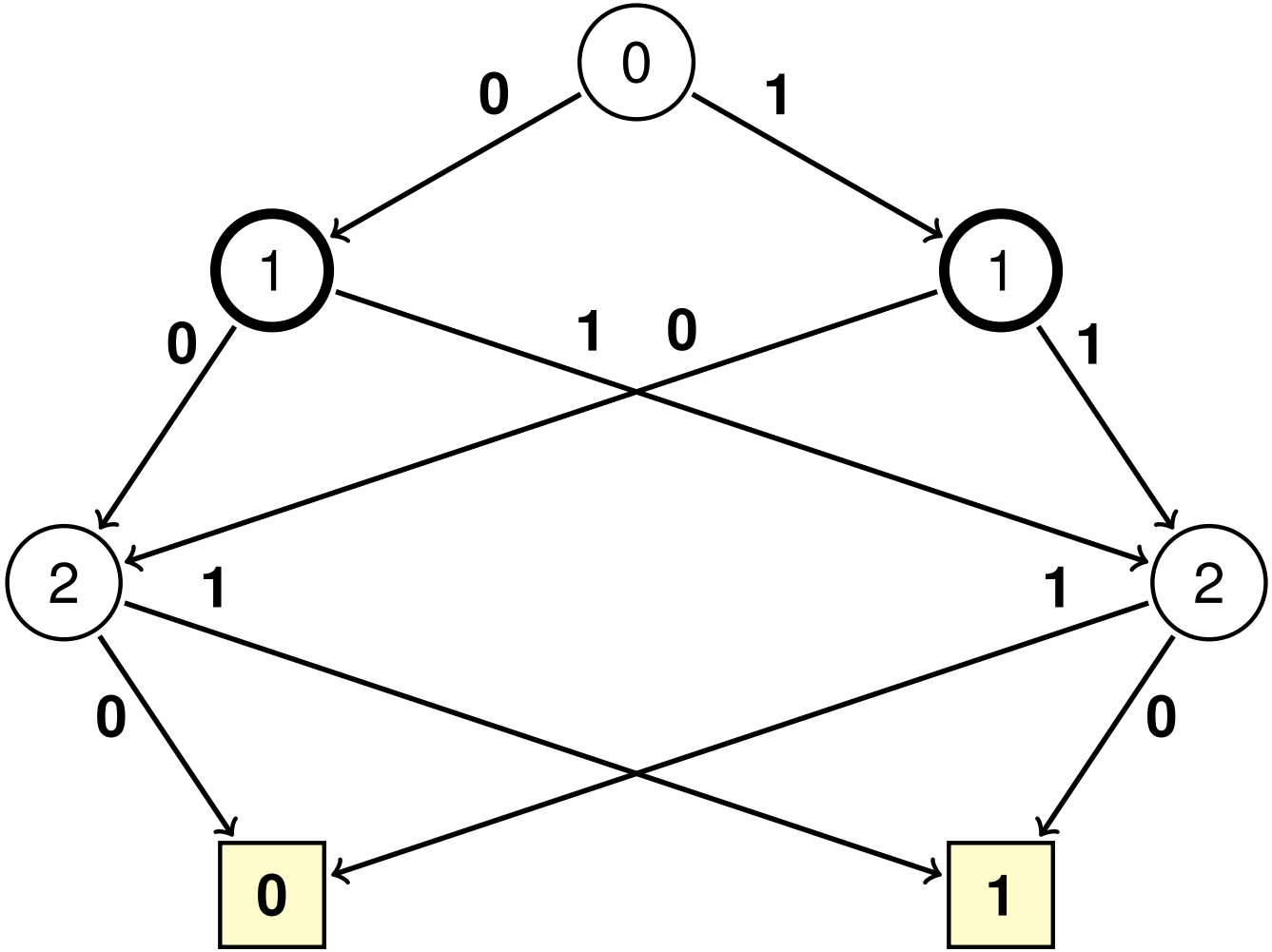
# Beispiel für Reduktion

Nun inkrementell



# Beispiel für Reduktion

Nun inkrementell



# Eindeutigkeit reduzierter Shannon-Graphen

**Theorem**

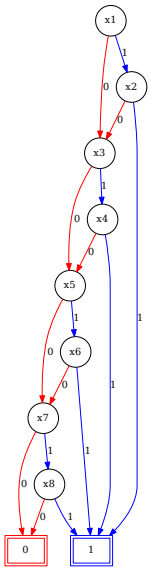
*Sind $G, H$ reduzierte sh-Graphen zu $\Sigma = \{P_1, ..., P_n\}$, dann gilt*

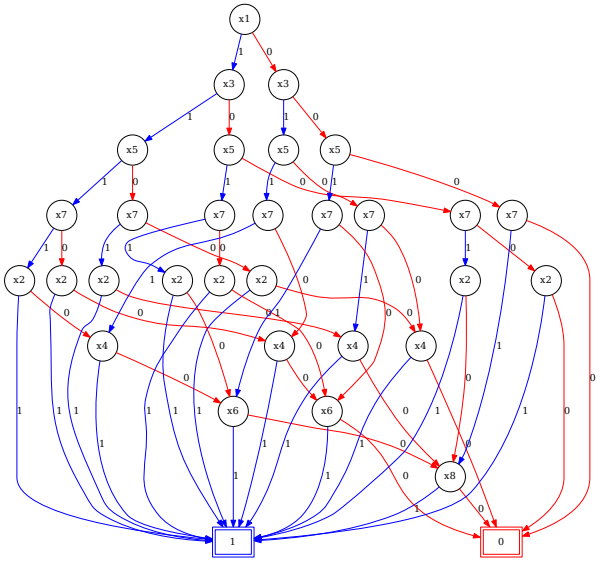
$$f_G = f_H \Leftrightarrow G \cong H.$$

*(Zu jeder Booleschen Funktion $f$ gibt es bis auf Isomorphie genau einen reduzierten sh-Graphen $H$ mit $f = f_H$).*

# BDD-Größe in Abhängigkeit von Variablenordnung

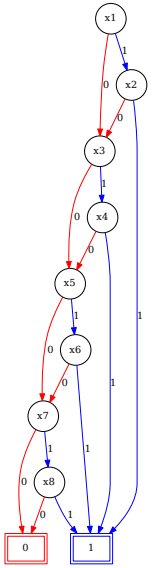
Zwei BDDs für $(x_1 \wedge x_2) \vee (x_3 \wedge x_4) \vee (x_5 \wedge x_6) \vee (x_7 \wedge x_8)$

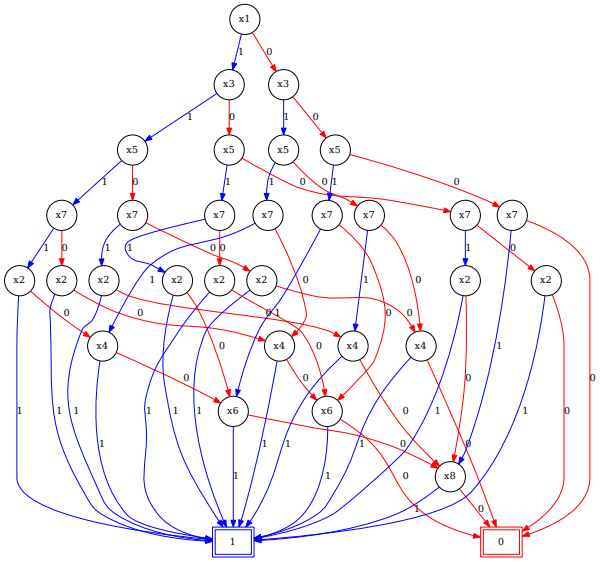


# BDD-Größe in Abhängigkeit von Variablenordnung

Zwei BDDs für $(x_1 \wedge x_2) \vee (x_3 \wedge x_4) \vee (x_5 \wedge x_6) \vee (x_7 \wedge x_8)$




Ordnung:

$x_1 < x_2 < x_3 < x_4 <$
$x_5 < x_6 < x_7 < x_8$

Ordnung:

$x_1 < x_3 < x_5 < x_7 <$
$x_2 < x_4 < x_6 < x_8$

# Eine harte Nuß

[BDD für Multiplikationen]

- $X$ enthalte $2k$ Variablen $\{x_0, \ldots, x_{k-1}, y_0, \ldots, y_{k-1}\}$

# Eine harte Nuß

[BDD für Multiplikationen]

- $X$ enthalte $2k$ Variablen $\{x_0, \ldots, x_{k-1}, y_0, \ldots, y_{k-1}\}$
- $x = x_0 \ldots x_{k-1}$ und $y = y_0 \ldots y_{k-1}$ bezeichnen $k$-stellige Binärzahlen.

# Eine harte Nuß

[BDD für Multiplikationen]

- $X$ enthalte $2k$ Variablen $\{x_0, \ldots, x_{k-1}, y_0, \ldots, y_{k-1}\}$
- $x = x_0 \ldots x_{k-1}$ und $y = y_0 \ldots y_{k-1}$ bezeichnen $k$-stellige Binärzahlen.
- für $0 \leq i < 2k$ bezeichne $Mult_i$ die boolsche Funktion, die das $i$-te Bit des Produktes von $x$ mit $y$ beschreibt.

# Eine harte Nuß

[BDD für Multiplikationen]

- $X$ enthalte $2k$ Variablen $\{x_0, \ldots, x_{k-1}, y_0, \ldots, y_{k-1}\}$
- $x = x_0 \ldots x_{k-1}$ und $y = y_0 \ldots y_{k-1}$ bezeichnen $k$-stellige Binärzahlen.
- für $0 \leq i < 2k$ bezeichne $Mult_i$ die boolsche Funktion, die das $i$-te Bit des Produktes von $x$ mit $y$ beschreibt.

# Eine harte Nuß

[BDD für Multiplikationen]

- $X$ enthalte $2k$ Variablen $\{x_0, \ldots, x_{k-1}, y_0, \ldots, y_{k-1}\}$
- $x = x_0 \ldots x_{k-1}$ und $y = y_0 \ldots y_{k-1}$ bezeichnen $k$-stellige Binärzahlen.
- für $0 \leq i < 2k$ bezeichne $Mult_i$ die boolsche Funktion, die das $i$-te Bit des Produktes von $x$ mit $y$ beschreibt.

**Theorem**

*Für jede Ordnung $<$ der Variablen in $X$ gibt es einen Index $0 \leq i < 2k$, so dass der BDD $B_{Mult_i,<}$ mindestens $2^{k/8}$ Knoten besitzt.*