**Script** generated by TTT

Title: Nipkow: Theo (08.07.2019)

Date: Mon Jul 08 14:27:17 CEST 2019

Duration: 74:42 min

Pages: 108

## Von NP-schwer zu „NP-leicht“

- Bis vor ca. 20 Jahren:

  NP-vollständig = Todesurteil

- In den letzten 15 Jahren:
  Spektakuläre Fortschritte bei *Implementierung* von
  SAT-Lösern (*SAT-solver*): `http://satcompetition.org`
  Stand der Kunst: Erfüllbar: bis $10^5$ Variablen
  Unerfüllbar: bis $10^3$ Variablen



## Von NP-schwer zu „NP-leicht“

- Bis vor ca. 20 Jahren:

  NP-vollständig = Todesurteil

- In den letzten 15 Jahren:
  Spektakuläre Fortschritte bei *Implementierung* von
  SAT-Lösern (*SAT-solver*): `http://satcompetition.org`
  Stand der Kunst: Erfüllbar: bis $10^5$ Variablen
  Unerfüllbar: bis $10^3$ Variablen



## Von NP-schwer zu „NP-leicht“

- Bis vor ca. 20 Jahren:

  NP-vollständig = Todesurteil

- In den letzten 15 Jahren:
  Spektakuläre Fortschritte bei *Implementierung* von
  SAT-Lösern (*SAT-solver*): `http://satcompetition.org`
  Stand der Kunst: Erfüllbar: bis $10^5$ Variablen
  Unerfüllbar: bis $10^3$ Variablen

- Jetzt:

  NP-vollständig = Hoffnung durch SAT

## Von NP-schwer zu „NP-leicht“

- Bis vor ca. 20 Jahren:

  NP-vollständig = Todesurteil

- In den letzten 15 Jahren:
  Spektakuläre Fortschritte bei *Implementierung* von SAT-Lösern (*SAT-solver*): `http://satcompetition.org`
  Stand der Kunst: Erfüllbar: bis $10^5$ Variablen
  Unerfüllbar: bis $10^3$ Variablen

- Jetzt:

  NP-vollständig = Hoffnung durch SAT

- Paradigma:

  SAT (Logik!) als universelle Sprache
  zur Kodierung kombinatorischer Probleme

  Reduktion auf SAT manchmal schneller als problemspezifische Löser!



## Von NP-schwer zu „NP-leicht“

- Bis vor ca. 20 Jahren:

  NP-vollständig = Todesurteil

- In den letzten 15 Jahren:
  Spektakuläre Fortschritte bei *Implementierung* von SAT-Lösern (*SAT-solver*): `http://satcompetition.org`
  Stand der Kunst: Erfüllbar: bis $10^5$ Variablen
  Unerfüllbar: bis $10^3$ Variablen

- Jetzt:

  NP-vollständig = Hoffnung durch SAT

- Paradigma:

  SAT (Logik!) als universelle Sprache
  zur Kodierung kombinatorischer Probleme

  Reduktion auf SAT manchmal schneller als problemspezifische Löser! Und fast immer einfacher.



Beispiel: Reduktion von 3-Färbbarkeit (3COL) auf SAT.



Beispiel: Reduktion von 3-Färbbarkeit (3COL) auf SAT.

3COL

Gegeben: Ein ungerichteter Graph

Problem: Gibt es eine Färbung der Knoten, so dass keine benachbarten Knoten gleich gefärbt sind?

Beispiel: Reduktion von 3-Färbbarkeit (3COL) auf SAT.

3COL

Gegeben: Ein ungerichteter Graph

Problem: Gibt es eine Färbung der Knoten, so dass keine benachbarten Knoten gleich gefärbt sind?

Lineare Reduktion von Graph $(V, E)$ auf SAT Instanz:



Beispiel: Reduktion von 3-Färbbarkeit (3COL) auf SAT.

3COL

Gegeben: Ein ungerichteter Graph

Problem: Gibt es eine Färbung der Knoten, so dass keine benachbarten Knoten gleich gefärbt sind?

Lineare Reduktion von Graph $(V, E)$ auf SAT Instanz:

- Variablen $= V \times \{1, 2, 3\}$. Notatation: $x_{vi}$



Beispiel: Reduktion von 3-Färbbarkeit (3COL) auf SAT.

3COL

Gegeben: Ein ungerichteter Graph

Problem: Gibt es eine Färbung der Knoten, so dass keine benachbarten Knoten gleich gefärbt sind?

Lineare Reduktion von Graph $(V, E)$ auf SAT Instanz:

- Variablen $= V \times \{1, 2, 3\}$. Notatation: $x_{vi}$
- Interpretation: $x_{vi} = 1$ gdw Knoten $v$ hat Farbe $i$

Formel:

$$\bigwedge_{v \in V} G(x_{v1}, x_{v2}, x_{v3})$$



Beispiel: Reduktion von 3-Färbbarkeit (3COL) auf SAT.

3COL

Gegeben: Ein ungerichteter Graph

Problem: Gibt es eine Färbung der Knoten, so dass keine benachbarten Knoten gleich gefärbt sind?

Lineare Reduktion von Graph $(V, E)$ auf SAT Instanz:

- Variablen $= V \times \{1, 2, 3\}$. Notatation: $x_{vi}$
- Interpretation: $x_{vi} = 1$ gdw Knoten $v$ hat Farbe $i$

Formel:

$$\bigwedge_{v \in V} G(x_{v1}, x_{v2}, x_{v3}) \wedge \bigwedge_{(u,v) \in E} \neg(x_{u1} \wedge x_{v1} \vee x_{u2} \wedge x_{v2} \vee x_{u3} \wedge x_{v3})$$

Beispiel: Reduktion von 3-Färbbarkeit (3COL) auf SAT.

3COL

Gegeben: Ein ungerichteter Graph

Problem: Gibt es eine Färbung der Knoten, so dass keine benachbarten Knoten gleich gefärbt sind?

Lineare Reduktion von Graph $(V, E)$ auf SAT Instanz:

- Variablen $= V \times \{1, 2, 3\}$. Notatation: $x_{vi}$
- Interpretation: $x_{vi} = 1$ gdw Knoten $v$ hat Farbe $i$

Formel:

$$\bigwedge_{v \in V} G(x_{v1}, x_{v2}, x_{v3}) \ \wedge \bigwedge_{(u,v) \in E} \neg(x_{u1} \wedge x_{v1} \vee x_{u2} \wedge x_{v2} \vee x_{u3} \wedge x_{v3})$$

**Bemerkungen**

- Zeigt 3COL $\leq_p$ SAT und damit 3COL $\in$ NP.



Beispiel: Reduktion von 3-Färbbarkeit (3COL) auf SAT.

3COL

Gegeben: Ein ungerichteter Graph

Problem: Gibt es eine Färbung der Knoten, so dass keine benachbarten Knoten gleich gefärbt sind?

Lineare Reduktion von Graph $(V, E)$ auf SAT Instanz:

- Variablen $= V \times \{1, 2, 3\}$. Notatation: $x_{vi}$
- Interpretation: $x_{vi} = 1$ gdw Knoten $v$ hat Farbe $i$

Formel:

$$\bigwedge_{v \in V} G(x_{v1}, x_{v2}, x_{v3}) \ \wedge \bigwedge_{(u,v) \in E} \neg(x_{u1} \wedge x_{v1} \vee x_{u2} \wedge x_{v2} \vee x_{u3} \wedge x_{v3})$$

**Bemerkungen**

- Zeigt 3COL $\leq_p$ SAT und damit 3COL $\in$ NP.
- Zeigt nicht, dass 3COL NP-vollständig ist.

354

Eine Anwendung von $k$-Färbbarkeit: **Registerverteilung**



Eine Anwendung von $k$-Färbbarkeit: **Registerverteilung**

*Kann man in einem Programmstück $n$ Variablen so auf $k$ Register verteilen, dass jede Variable so lange in einem Register bleibt, wie sie lebendig ist?*

355

Eine Anwendung von $k$-Färbbarkeit: **Registerverteilung**

> *Kann man in einem Programmstück $n$ Variablen so auf $k$ Register verteilen, dass jede Variable so lange in einem Register bleibt, wie sie lebendig ist?*

Variable ist an einem Punkt lebendig



Eine Anwendung von $k$-Färbbarkeit: **Registerverteilung**

> *Kann man in einem Programmstück $n$ Variablen so auf $k$ Register verteilen, dass jede Variable so lange in einem Register bleibt, wie sie lebendig ist?*

Variable ist an einem Punkt lebendig
gdw sie später noch gelesen wird, ohne vorher überschrieben worden zu sein.

Reduktion auf $k$-Färbbarkeit:



Eine Anwendung von $k$-Färbbarkeit: **Registerverteilung**

> *Kann man in einem Programmstück $n$ Variablen so auf $k$ Register verteilen, dass jede Variable so lange in einem Register bleibt, wie sie lebendig ist?*

Variable ist an einem Punkt lebendig
gdw sie später noch gelesen wird, ohne vorher überschrieben worden zu sein.

Reduktion auf $k$-Färbbarkeit:

| | | |
|---|---|---|
| Variable | $=$ | Knoten |
| $u$ und $v$ verbunden | $=$ | $u \neq v$ und es gibt einen Programmpunkt, an dem $u$ und $v$ lebendig sind |
| Farbe | $=$ | Register |



Eine Anwendung von $k$-Färbbarkeit: **Registerverteilung**

> *Kann man in einem Programmstück $n$ Variablen so auf $k$ Register verteilen, dass jede Variable so lange in einem Register bleibt, wie sie lebendig ist?*

Variable ist an einem Punkt lebendig
gdw sie später noch gelesen wird, ohne vorher überschrieben worden zu sein.

Reduktion auf $k$-Färbbarkeit:

| | | |
|---|---|---|
| Variable | $=$ | Knoten |
| $u$ und $v$ verbunden | $=$ | $u \neq v$ und es gibt einen Programmpunkt, an dem $u$ und $v$ lebendig sind |
| Farbe | $=$ | Register |
| $k$-Färbung | $=$ | Zuordnung eines Registers zu jeder Variablen |

Sowohl $k$-Färbbarkeit als auch Registerverteilung ist für $k \geq 3$ NP-vollständig.

Mehr Information: Vorlesung *Programmoptimierung*

Lemma 6.19
*NICHTÄQUIVALENZ* $\leq_p$ *SAT und SAT* $\leq_p$ *NICHTÄQUIVALENZ.*



Lemma 6.19
*NICHTÄQUIVALENZ* $\leq_p$ *SAT und SAT* $\leq_p$ *NICHTÄQUIVALENZ.*

Beweis:
NICHTÄQUIVALENZ $\leq_p$ SAT:

$$f(F_1, F_2) = (F_1 \wedge \neg F_2) \vee (\neg F_1 \wedge F_2)$$



Lemma 6.19
*NICHTÄQUIVALENZ* $\leq_p$ *SAT und SAT* $\leq_p$ *NICHTÄQUIVALENZ.*

Beweis:
NICHTÄQUIVALENZ $\leq_p$ SAT:

$$f(F_1, F_2) = (F_1 \wedge \neg F_2) \vee (\neg F_1 \wedge F_2)$$

SAT $\leq_p$ NICHTÄQUIVALENZ:



Lemma 6.19
*NICHTÄQUIVALENZ* $\leq_p$ *SAT und SAT* $\leq_p$ *NICHTÄQUIVALENZ.*

Beweis:
NICHTÄQUIVALENZ $\leq_p$ SAT:

$$f(F_1, F_2) = (F_1 \wedge \neg F_2) \vee (\neg F_1 \wedge F_2)$$

SAT $\leq_p$ NICHTÄQUIVALENZ:

$$f(F) = (F, x \wedge \neg x)$$

□

*2. Bounded Model Checking*

Hardware Entscheide, ob eine Schaltung mit Zustand für alle Eingaben innerhalb von 10 Zyklen ein bestimmtes Verhalten (nicht) hat.



*2. Bounded Model Checking*

Hardware Entscheide, ob eine Schaltung mit Zustand für alle Eingaben innerhalb von 10 Zyklen ein bestimmtes Verhalten (nicht) hat.

Software Entscheide, ob ein Programm für alle Eingaben innerhalb von 10 Schritten ein bestimmtes Verhalten (nicht) hat.



*2. Bounded Model Checking*

Hardware Entscheide, ob eine Schaltung mit Zustand für alle Eingaben innerhalb von 10 Zyklen ein bestimmtes Verhalten (nicht) hat.

Software Entscheide, ob ein Programm für alle Eingaben innerhalb von 10 Schritten ein bestimmtes Verhalten (nicht) hat.
Variablen müssen auf sehr kleine Wertebereiche eingeschränkt werden!



## 6.4 Weitere NP-vollständige Probleme

Wie zeigt man, dass ein (weiteres) Problem $B$ NP-vollständig ist?

## 6.4 Weitere NP-vollständige Probleme

Wie zeigt man, dass ein (weiteres) Problem $B$ NP-vollständig ist?

- Zeige $B \in$ NP (meist trivial)



## 6.4 Weitere NP-vollständige Probleme

Wie zeigt man, dass ein (weiteres) Problem $B$ NP-vollständig ist?

- Zeige $B \in$ NP (meist trivial)
- Zeige $A \leq_p B$ für ein NP-vollständiges Problem $A$.



## 6.4 Weitere NP-vollständige Probleme

Wie zeigt man, dass ein (weiteres) Problem $B$ NP-vollständig ist?

- Zeige $B \in$ NP (meist trivial)
- Zeige $A \leq_p B$ für ein NP-vollständiges Problem $A$.

Lemma 6.20
*Ist $A$ NP-vollständig, so ist $B \in$ NP ebenfalls NP-vollständig, falls $A \leq_p B$.*



## 6.4 Weitere NP-vollständige Probleme

Wie zeigt man, dass ein (weiteres) Problem $B$ NP-vollständig ist?

- Zeige $B \in$ NP (meist trivial)
- Zeige $A \leq_p B$ für ein NP-vollständiges Problem $A$.

Lemma 6.20
*Ist $A$ NP-vollständig, so ist $B \in$ NP ebenfalls NP-vollständig, falls $A \leq_p B$.*

Beweis:
Folgt direkt aus der Transitivität von $\leq_p$:

Warum will man wissen, dass ein Problem $B$ NP-vollständig ist?



Warum will man wissen, dass ein Problem $B$ NP-vollständig ist?
Um sicher zu sein, dass

- ein polynomieller Algorithmus für $B$ ein Durchbruch wäre
- und daher wahrscheinlich nicht existiert.

Praktische Instanzen von $B$ könnten trotzdem (zB mit SAT) „effizient“ lösbar sein.



Viele Varianten von SAT sind ebenfalls NP-vollständig:

Definition 6.21

- Eine Formel ist in Konjunktiver Normalform (KNF) gdw sie eine Konjunktion von Klauseln ist: $K_1 \wedge \cdots \wedge K_n$



Viele Varianten von SAT sind ebenfalls NP-vollständig:

Definition 6.21

- Eine Formel ist in Konjunktiver Normalform (KNF) gdw sie eine Konjunktion von Klauseln ist: $K_1 \wedge \cdots \wedge K_n$
- Eine Klausel ist eine Disjunktion von Literalen: $L_1 \vee \cdots \vee L_m$

Viele Varianten von SAT sind ebenfalls NP-vollständig:

Definition 6.21

- Eine Formel ist in Konjunktiver Normalform (KNF) gdw sie eine Konjunktion von Klauseln ist: $K_1 \wedge \cdots \wedge K_n$
- Eine Klausel ist eine Disjunktion von Literalen: $L_1 \vee \cdots \vee L_m$
- Ein Literal ist eine (evtl. negierte) Variable.
- Eine Formel ist in 3KNF gdw jede Klausel $\leq 3$ Literale enthält.

Dh eine KNF ist ein Konjunktion von Disjunktionen von (evtl negierten) Variablen.
Bsp: $(x_9 \vee \neg x_2) \wedge (\neg x_7 \vee x_1 \vee x_6)$



Viele Varianten von SAT sind ebenfalls NP-vollständig:

Definition 6.21

- Eine Formel ist in Konjunktiver Normalform (KNF) gdw sie eine Konjunktion von Klauseln ist: $K_1 \wedge \cdots \wedge K_n$
- Eine Klausel ist eine Disjunktion von Literalen: $L_1 \vee \cdots \vee L_m$
- Ein Literal ist eine (evtl. negierte) Variable.
- Eine Formel ist in 3KNF gdw jede Klausel $\leq 3$ Literale enthält.

Dh eine KNF ist ein Konjunktion von Disjunktionen von (evtl negierten) Variablen.
Bsp: $(x_9 \vee \neg x_2) \wedge (\neg x_7 \vee x_1 \vee x_6)$

3KNF-SAT

Gegeben: Ein Formel in 3KNF

Problem: Ist die Formel erfüllbar?



Viele Varianten von SAT sind ebenfalls NP-vollständig:

Definition 6.21

- Eine Formel ist in Konjunktiver Normalform (KNF) gdw sie eine Konjunktion von Klauseln ist: $K_1 \wedge \cdots \wedge K_n$
- Eine Klausel ist eine Disjunktion von Literalen: $L_1 \vee \cdots \vee L_m$
- Ein Literal ist eine (evtl. negierte) Variable.
- Eine Formel ist in 3KNF gdw jede Klausel $\leq 3$ Literale enthält.

Dh eine KNF ist ein Konjunktion von Disjunktionen von (evtl negierten) Variablen.
Bsp: $(x_9 \vee \neg x_2) \wedge (\neg x_7 \vee x_1 \vee x_6)$

3KNF-SAT

Gegeben: Ein Formel in 3KNF

Problem: Ist die Formel erfüllbar?

Offensichtlich gilt 3KNF-SAT $\in$ NP.



Satz 6.22
*3KNF-SAT ist NP-vollständig.*

Beweis:
Wir zeigen SAT $\leq_p$ 3KNF-SAT mit einer polynomiellen Reduktion $F \mapsto F'$ so dass $F'$ in 3KNF ist und

$$F \text{ ist erfüllbar } \Leftrightarrow \ F' \text{ ist erfüllbar}$$

Viele Varianten von SAT sind ebenfalls NP-vollständig:

Definition 6.21

- Eine Formel ist in Konjunktiver Normalform (KNF) gdw sie eine Konjunktion von Klauseln ist: $K_1 \wedge \cdots \wedge K_n$
- Eine Klausel ist eine Disjunktion von Literalen: $L_1 \vee \cdots \vee L_m$
- Ein Literal ist eine (evtl. negierte) Variable.
- Eine Formel ist in 3KNF gdw jede Klausel $\leq 3$ Literale enthält.

Dh eine KNF ist ein Konjunktion von Disjunktionen von (evtl. negierten) Variablen.
Bsp: $(x_9 \vee \neg x_2) \wedge (\neg x_7 \vee x_1 \vee x_6)$

3KNF-SAT

Gegeben: Ein Formel in 3KNF

Problem: Ist die Formel erfüllbar?

Offensichtlich gilt 3KNF-SAT $\in$ NP.
Aber vielleicht ist 3KNF-SAT einfacher als SAT?



Warum will man wissen, dass ein Problem $B$ NP-vollständig ist?



Satz 6.22
*3KNF-SAT ist NP-vollständig.*

Beweis:
Wir zeigen SAT $\leq_p$ 3KNF-SAT mit einer polynomiellen Reduktion $F \mapsto F'$ so dass $F'$ in 3KNF ist und

$$F \text{ ist erfüllbar } \Leftrightarrow \ F' \text{ ist erfüllbar}$$

NB $F$ und $F'$ sind erfüllbarkeitsäquivalent, aber nicht notwendigerweise auch äquivalent.



Satz 6.22
*3KNF-SAT ist NP-vollständig.*

Beweis:
Wir zeigen SAT $\leq_p$ 3KNF-SAT mit einer polynomiellen Reduktion $F \mapsto F'$ so dass $F'$ in 3KNF ist und

$$F \text{ ist erfüllbar } \Leftrightarrow \ F' \text{ ist erfüllbar}$$

NB $F$ und $F'$ sind erfüllbarkeitsäquivalent, aber nicht notwendigerweise auch äquivalent.

1. Transformiere $F$ in Negations-Normalform (NNF) durch fortgesetze Anwendung der de Morganschen Gesetze

$$\begin{aligned} \neg(A \wedge B) &= \neg A \vee \neg B \\ \neg(A \vee B) &= \neg A \wedge \neg B \\ \neg\neg A &= A \end{aligned}$$

von links nach rechts.

## Satz 6.22

*3KNF-SAT ist NP-vollständig.*

## Beweis:

Wir zeigen SAT $\leq_p$ 3KNF-SAT mit einer polynomiellen Reduktion $F \mapsto F'$ so dass $F'$ in 3KNF ist und

$$F \text{ ist erfüllbar } \Leftrightarrow \ F' \text{ ist erfüllbar}$$

NB $F$ und $F'$ sind erfüllbarkeitsäquivalent, aber nicht notwendigerweise auch äquivalent.

1. Transformiere $F$ in Negations-Normalform (NNF) durch fortgesetze Anwendung der de Morganschen Gesetze

$$\begin{aligned} \neg(A \wedge B) &= \neg A \vee \neg B \\ \neg(A \vee B) &= \neg A \wedge \neg B \\ \neg\neg A &= A \end{aligned}$$

von links nach rechts. $F_1$ ist Resultat.



## Beweis (Forts.):

Für $F_1$ gilt: $\neg$ nur noch direkt vor Variablen.



## Beweis (Forts.):

Für $F_1$ gilt: $\neg$ nur noch direkt vor Variablen.

2. Betrachte $F_1$ als Baum, wobei die Literale Blätter sind.
Ordne jedem inneren Knoten eine neue Variable $\in \{y_0, y_1, \dots\}$ zu.
Ordne dabei der Wurzel von $F_1$ die Variable $y_0$ zu.

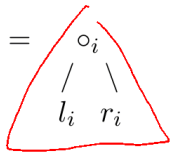


## Beweis (Forts.):

Für $F_1$ gilt: $\neg$ nur noch direkt vor Variablen.

2. Betrachte $F_1$ als Baum, wobei die Literale Blätter sind.
Ordne jedem inneren Knoten eine neue Variable $\in \{y_0, y_1, \dots\}$ zu.
Ordne dabei der Wurzel von $F_1$ die Variable $y_0$ zu.

3. Betrachte die $y_i$ als Abkürzung für die Teilbäume, an deren Wurzeln sie stehen

$y_i =$ 

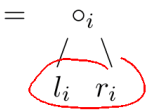

wobei $\circ_i \in \{\wedge, \vee\}$

## Beweis (Forts.):

Für $F_1$ gilt: $\neg$ nur noch direkt vor Variablen.

2. Betrachte $F_1$ als Baum, wobei die Literale Blätter sind.
Ordne jedem inneren Knoten eine neue Variable $\in \{y_0, y_1, \dots\}$ zu.
Ordne dabei der Wurzel von $F_1$ die Variable $y_0$ zu.

3. Betrachte die $y_i$ als Abkürzung für die Teilbäume, an deren Wurzeln sie stehen

$y_i =$ 


wobei $\circ_i \in \{\wedge, \vee\}$ und $l_i, r_i$ ein Literal oder eine Variable $y_j$ ist.
Beschreibe $F_1$ Knoten für Knoten:

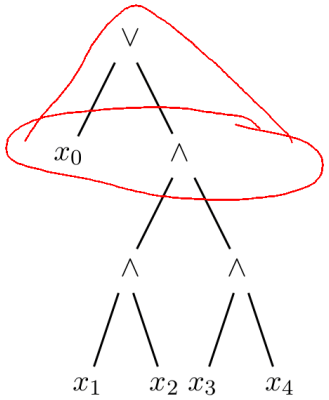
$$y_0 \wedge (y_0 \leftrightarrow (l_0 \circ_0 r_0)) \wedge (y_1 \leftrightarrow (l_1 \circ_1 r_1)) \cdots =: F_2$$



Beispiel: $F_1 = x_0 \vee ((x_1 \wedge x_2) \wedge (x_3 \wedge x_4))$

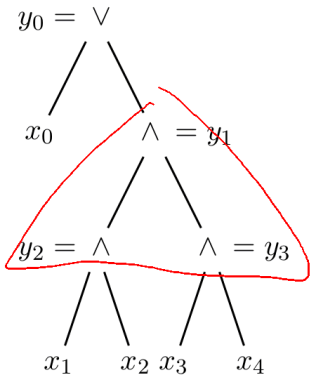


Beispiel: $F_1 = x_0 \vee ((x_1 \wedge x_2) \wedge (x_3 \wedge x_4))$





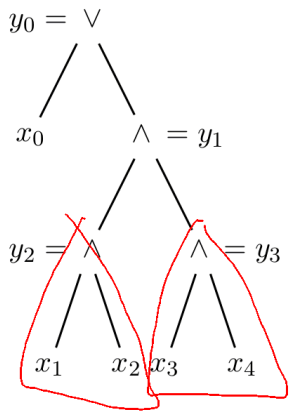

Beispiel: $F_1 = x_0 \vee ((x_1 \wedge x_2) \wedge (x_3 \wedge x_4))$




$F_2 =$

$y_0 \wedge (y_0 \leftrightarrow (x_0 \wedge y_1)) \wedge (y_1 \leftrightarrow (y_2 \wedge y_3))$

Beispiel: $F_1 = x_0 \vee ((x_1 \wedge x_2) \wedge (x_3 \wedge x_4))$

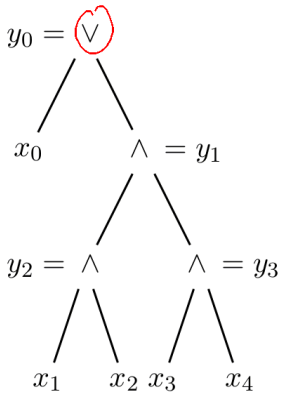



$F_2 =$

$y_0 \wedge (y_0 \leftrightarrow (x_0 \wedge y_1)) \wedge (y_1 \leftrightarrow (y_2 \wedge y_3)) \wedge (y_2 \leftrightarrow (x_1 \wedge x_2))$



Beispiel: $F_1 = x_0 \vee ((x_1 \wedge x_2) \wedge (x_3 \wedge x_4))$




$F_2 =$

$y_0 \wedge (y_0 \leftrightarrow (x_0 \wedge y_1)) \wedge (y_1 \leftrightarrow (y_2 \wedge y_3)) \wedge (y_2 \leftrightarrow (x_1 \wedge x_2)) \wedge (y_3 \leftrightarrow (x_3 \wedge x_4))$

364

## Beweis (Forts.):

$F_1$ erf. $\implies$ $F_2$ erf.: $y_i$ bekommt Wert seines Teilbaums.



## Beweis (Forts.):

$F_1$ erf. $\implies$ $F_2$ erf.: $y_i$ bekommt Wert seines Teilbaums.
$F_2$ erf. $\implies$ $F_1$ erf.: klar

365

## Beweis (Forts.):

$F_1$ erf. $\Longrightarrow$ $F_2$ erf.: $y_i$ bekommt Wert seines Teilbaums.
$F_2$ erf. $\Longrightarrow$ $F_1$ erf.: klar

4. Transformiere jede Äquivalenz in 3KNF:

$$(a \leftrightarrow (b \vee c)) \quad \mapsto \quad (a \vee \neg b) \wedge (a \vee \neg c) \wedge (\neg a \vee b \vee c)$$
$$(a \leftrightarrow (b \wedge c)) \quad \mapsto \quad (\neg a \vee b) \wedge (\neg a \vee c) \wedge (a \vee \neg b \vee \neg c)$$



## Beweis (Forts.):

$F_1$ erf. $\Longrightarrow$ $F_2$ erf.: $y_i$ bekommt Wert seines Teilbaums.
$F_2$ erf. $\Longrightarrow$ $F_1$ erf.: klar

4. Transformiere jede Äquivalenz in 3KNF:

$$(a \leftrightarrow (b \vee c)) \quad \mapsto \quad (a \vee \neg b) \wedge (a \vee \neg c) \wedge (\neg a \vee b \vee c)$$
$$(a \leftrightarrow (b \wedge c)) \quad \mapsto \quad (\neg a \vee b) \wedge (\neg a \vee c) \wedge (a \vee \neg b \vee \neg c)$$

Ergebnis ist $F'$.



## Beweis (Forts.):

$F_1$ erf. $\Longrightarrow$ $F_2$ erf.: $y_i$ bekommt Wert seines Teilbaums.
$F_2$ erf. $\Longrightarrow$ $F_1$ erf.: klar

4. Transformiere jede Äquivalenz in 3KNF:

$$(a \leftrightarrow (b \vee c)) \quad \mapsto \quad (a \vee \neg b) \wedge (a \vee \neg c) \wedge (\neg a \vee b \vee c)$$
$$(a \leftrightarrow (b \wedge c)) \quad \mapsto \quad (\neg a \vee b) \wedge (\neg a \vee c) \wedge (a \vee \neg b \vee \neg c)$$

Ergebnis ist $F'$.

Jede Transformation ist in polynomieller Zeit (in $|F|$) berechenbar.
Beispiel: Bei Transformation in NNF nimmt mit jedem Schritt

Summe der $|G|$ für alle Teilformeln $\neg G$ von $F$

ab.



## Beweis (Forts.):

$F_1$ erf. $\Longrightarrow$ $F_2$ erf.: $y_i$ bekommt Wert seines Teilbaums.
$F_2$ erf. $\Longrightarrow$ $F_1$ erf.: klar

4. Transformiere jede Äquivalenz in 3KNF:

$$(a \leftrightarrow (b \vee c)) \quad \mapsto \quad (a \vee \neg b) \wedge (a \vee \neg c) \wedge (\neg a \vee b \vee c)$$
$$(a \leftrightarrow (b \wedge c)) \quad \mapsto \quad (\neg a \vee b) \wedge (\neg a \vee c) \wedge (a \vee \neg b \vee \neg c)$$

Ergebnis ist $F'$.

Beweis (Forts.):

Für $F_1$ gilt: $\neg$ nur noch direkt vor Variablen.

2. Betrachte $F_1$ als Baum, wobei die Literale Blätter sind.



Da jede Formel in 3KNF auch in KNF ist:

Korollar 6.23
*KNF-SAT ist NP-vollständig.*

Kann man wie folgt die NP-Vollständigkeit von KNF-SAT zeigen?

Man zeigt SAT $\leq_p$ KNF-SAT
indem man jede Formel in KNF transformiert.



Da jede Formel in 3KNF auch in KNF ist:

Korollar 6.23
*KNF-SAT ist NP-vollständig.*

Kann man wie folgt die NP-Vollständigkeit von KNF-SAT zeigen?

Man zeigt SAT $\leq_p$ KNF-SAT
indem man jede Formel in KNF transformiert.



Da jede Formel in 3KNF auch in KNF ist:

Korollar 6.23
*KNF-SAT ist NP-vollständig.*

Kann man wie folgt die NP-Vollständigkeit von KNF-SAT zeigen?

Man zeigt SAT $\leq_p$ KNF-SAT
indem man jede Formel in KNF transformiert.

Satz 6.24
*2KNF-SAT* $\in P$

Ohne Beweis

## MENGENÜBERDECKUNG (MÜ)

Gegeben: Teilmengen $T_1, \ldots, T_n \subseteq M$ einer endlichen Menge $M$ und eine Zahl $k \leq n$.



## MENGENÜBERDECKUNG (MÜ)

Gegeben: Teilmengen $T_1, \ldots, T_n \subseteq M$ einer endlichen Menge $M$ und eine Zahl $k \leq n$.

Problem: Gibt es $i_1, \ldots, i_k \in \{1, \ldots, n\}$ mit $M = T_{i_1} \cup \cdots \cup T_{i_k}$?



## MENGENÜBERDECKUNG (MÜ)

Gegeben: Teilmengen $T_1, \ldots, T_n \subseteq M$ einer endlichen Menge $M$ und eine Zahl $k \leq n$.

Problem: Gibt es $i_1, \ldots, i_k \in \{1, \ldots, n\}$ mit $M = T_{i_1} \cup \cdots \cup T_{i_k}$?

Beispiel 6.25

$$\begin{array}{lllllll} T_1 & = & \{1,2\} & & T_2 & = & \{1,3\} \\ T_3 & = & \{3,4\} & & T_4 & = & \{3,5\} \\ M & = & \{1,2,3,4,5\} & & k & = & 3 \end{array}$$



## MENGENÜBERDECKUNG (MÜ)

Gegeben: Teilmengen $T_1, \ldots, T_n \subseteq M$ einer endlichen Menge $M$ und eine Zahl $k \leq n$.

Problem: Gibt es $i_1, \ldots, i_k \in \{1, \ldots, n\}$ mit $M = T_{i_1} \cup \cdots \cup T_{i_k}$?

Beispiel 6.25

$$\begin{array}{lllllll} T_1 & = & \{1,2\} & & T_2 & = & \{1,3\} \\ T_3 & = & \{3,4\} & & T_4 & = & \{3,5\} \\ M & = & \{1,2,3,4,5\} & & k & = & 3 \end{array}$$

Anwendung: Zulieferer

$M$ Menge der Teile, die eine Firma einkaufen muss

$T_i$ Menge der Teile, die Zulieferer $i$ anbietet

Kann die Firma ihre Bedürfnisse mit $k$ Zulieferern abdecken?

## MENGENÜBERDECKUNG (MÜ)

Gegeben: Teilmengen $T_1, \ldots, T_n \subseteq M$ einer endlichen Menge $M$ und eine Zahl $k \leq n$.

Problem: Gibt es $i_1, \ldots, i_k \in \{1, \ldots, n\}$ mit $M = T_{i_1} \cup \cdots \cup T_{i_k}$?

### Beispiel 6.25

$$\begin{array}{lllllll} T_1 & = & \{1,2\} & & T_2 & = & \{1,3\} \\ T_3 & = & \{3,4\} & & T_4 & = & \{3,5\} \\ M & = & \{1,2,3,4,5\} & & k & = & 3 \end{array}$$

Anwendung: Zulieferer

$M$ Menge der Teile, die eine Firma einkaufen muss

$T_i$ Menge der Teile, die Zulieferer $i$ anbietet

Kann die Firma ihre Bedürfnisse mit $k$ Zulieferern abdecken?

### Fakt 6.26

*MÜ $\in$ NP.*



### Satz 6.27

*MÜ ist NP-vollständig.*

### Satz 6.27

*MÜ ist NP-vollständig.*

Beweis: KNF-SAT $\leq_p$ MÜ.

Sei $F = K_1 \wedge \cdots \wedge K_m$ in KNF, mit Variablen $x_1, \ldots, x_l$.



### Satz 6.27

*MÜ ist NP-vollständig.*

Beweis: KNF-SAT $\leq_p$ MÜ.

Sei $F = K_1 \wedge \cdots \wedge K_m$ in KNF, mit Variablen $x_1, \ldots, x_l$.

Wir konstruieren eine Menge $M$, Teilmengen $T_1, \ldots, T_n$ und eine Zahl $k$

**Satz 6.27**
*MÜ ist NP-vollständig.*

**Beweis:** KNF-SAT $\leq_p$ MÜ.

Sei $F = K_1 \wedge \cdots \wedge K_m$ in KNF, mit Variablen $x_1, \ldots, x_l$.

Wir konstruieren eine Menge $M$, Teilmengen $T_1, \ldots, T_n$ und eine Zahl $k$

$$M := \{1, \ldots, m, m+1, \ldots, m+l\}$$



**Satz 6.27**
*MÜ ist NP-vollständig.*

**Beweis:** KNF-SAT $\leq_p$ MÜ.

Sei $F = K_1 \wedge \cdots \wedge K_m$ in KNF, mit Variablen $x_1, \ldots, x_l$.

Wir konstruieren eine Menge $M$, Teilmengen $T_1, \ldots, T_n$ und eine Zahl $k$

$$\begin{aligned} M &:= \{1, \ldots, m, m+1, \ldots, m+l\} \\ T_i &:= \{j \mid x_i \text{ kommt in } K_j \text{ positiv vor}\} \cup \{m+i\} \end{aligned}$$



**Satz 6.27**
*MÜ ist NP-vollständig.*

**Beweis:** KNF-SAT $\leq_p$ MÜ.

Sei $F = K_1 \wedge \cdots \wedge K_m$ in KNF, mit Variablen $x_1, \ldots, x_l$.

Wir konstruieren eine Menge $M$, Teilmengen $T_1, \ldots, T_n$ und eine Zahl $k$

$$\begin{aligned} M &:= \{1, \ldots, m, m+1, \ldots, m+l\} \\ T_i &:= \{j \mid x_i \text{ kommt in } K_j \text{ positiv vor}\} \cup \{m+i\} \\ T_{l+i} &:= \{j \mid x_i \text{ kommt in } K_j \text{ negativ vor}\} \cup \{m+i\} \end{aligned}$$



**Satz 6.27**
*MÜ ist NP-vollständig.*

**Beweis:** KNF-SAT $\leq_p$ MÜ.

Sei $F = K_1 \wedge \cdots \wedge K_m$ in KNF, mit Variablen $x_1, \ldots, x_l$.

Wir konstruieren eine Menge $M$, Teilmengen $T_1, \ldots, T_n$ und eine Zahl $k$

$$\begin{aligned} M &:= \{1, \ldots, m, m+1, \ldots, m+l\} \\ T_i &:= \{j \mid x_i \text{ kommt in } K_j \text{ positiv vor}\} \cup \{m+i\} \\ T_{l+i} &:= \{j \mid x_i \text{ kommt in } K_j \text{ negativ vor}\} \cup \{m+i\} \\ n &:= 2l \end{aligned}$$

**Satz 6.27**
*MÜ ist NP-vollständig.*

**Beweis:** KNF-SAT $\leq_p$ MÜ.

Sei $F = K_1 \wedge \cdots \wedge K_m$ in KNF, mit Variablen $x_1, \ldots, x_l$.

Wir konstruieren eine Menge $M$, Teilmengen $T_1, \ldots, T_n$ und eine Zahl $k$

$$\begin{aligned} M &:= \{1, \ldots, m, m+1, \ldots, m+l\} \\ T_i &:= \{j \mid x_i \text{ kommt in } K_j \text{ positiv vor}\} \cup \{m+i\} \\ T_{l+i} &:= \{j \mid x_i \text{ kommt in } K_j \text{ negativ vor}\} \cup \{m+i\} \\ n &:= 2l \\ k &:= l \end{aligned}$$



**Satz 6.27**
*MÜ ist NP-vollständig.*

**Beweis:** KNF-SAT $\leq_p$ MÜ.

Sei $F = K_1 \wedge \cdots \wedge K_m$ in KNF, mit Variablen $x_1, \ldots, x_l$.

Wir konstruieren eine Menge $M$, Teilmengen $T_1, \ldots, T_n$ und eine Zahl $k$

$$\begin{aligned} M &:= \{1, \ldots, m, m+1, \ldots, m+l\} \\ T_i &:= \{j \mid x_i \text{ kommt in } K_j \text{ positiv vor}\} \cup \{m+i\} \\ T_{l+i} &:= \{j \mid x_i \text{ kommt in } K_j \text{ negativ vor}\} \cup \{m+i\} \\ n &:= 2l \\ k &:= l \end{aligned}$$

$F$ ist erfülbar
gdw
$M$ wird durch $l$ der Teilmengen $T_1, \ldots, T_l, T_{l+1}, \ldots, T_{2l}$ überdeckt

368

Sei $F = K_1 \wedge \cdots \wedge K_m$ in KNF, mit Variablen $x_1, \ldots, x_l$.

$$\begin{aligned} M &:= \{1, \ldots, m+l\} \\ T_i &:= \{j \mid x_i \text{ kommt in } K_j \text{ positiv vor}\} \cup \{m+i\} \\ T'_i &:= \{j \mid x_i \text{ kommt in } K_j \text{ negativ vor}\} \cup \{m+i\} \end{aligned}$$



**Satz 6.27**
*MÜ ist NP-vollständig.*

**Beweis:** KNF-SAT $\leq_p$ MÜ.

Sei $F = K_1 \wedge \cdots \wedge K_m$ in KNF, mit Variablen $x_1, \ldots, x_l$.

Wir konstruieren eine Menge $M$, Teilmengen $T_1, \ldots, T_n$ und eine Zahl $k$

$$\begin{aligned} M &:= \{1, \ldots, m, m+1, \ldots, m+l\} \\ T_i &:= \{j \mid x_i \text{ kommt in } K_j \text{ positiv vor}\} \cup \{m+i\} \\ T_{l+i} &:= \{j \mid x_i \text{ kommt in } K_j \text{ negativ vor}\} \cup \{m+i\} \\ n &:= 2l \\ k &:= l \end{aligned}$$

$F$ ist erfülbar
gdw
$M$ wird durch $l$ der Teilmengen $T_1, \ldots, T_l, T_{l+1}, \ldots, T_{2l}$ überdeckt

Sei $F = K_1 \wedge \cdots \wedge K_m$ in KNF, mit Variablen $x_1, \ldots, x_l$.

$$
\begin{aligned}
M &:= \{1, \ldots, m+l\} \\
T_i &:= \{j \mid x_i \text{ kommt in } K_j \text{ positiv vor}\} \cup \{m+i\} \\
T'_i &:= \{j \mid x_i \text{ kommt in } K_j \text{ negativ vor}\} \cup \{m+i\}
\end{aligned}
$$



Sei $F = K_1 \wedge \cdots \wedge K_m$ in KNF, mit Variablen $x_1, \ldots, x_l$.

$$
\begin{aligned}
M &:= \{1, \ldots, m+l\} \\
T_i &:= \{j \mid x_i \text{ kommt in } K_j \text{ positiv vor}\} \cup \{m+i\} \\
T'_i &:= \{j \mid x_i \text{ kommt in } K_j \text{ negativ vor}\} \cup \{m+i\}
\end{aligned}
$$

Beispiel: $(x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee x_2 \vee x_4)$

$\Longrightarrow$

$$T_1 = \{1, 2+1\} \qquad T'_1 = \{2, 2+1\}$$



Sei $F = K_1 \wedge \cdots \wedge K_m$ in KNF, mit Variablen $x_1, \ldots, x_l$.

$$
\begin{aligned}
M &:= \{1, \ldots, m+l\} \\
T_i &:= \{j \mid x_i \text{ kommt in } K_j \text{ positiv vor}\} \cup \{m+i\} \\
T'_i &:= \{j \mid x_i \text{ kommt in } K_j \text{ negativ vor}\} \cup \{m+i\}
\end{aligned}
$$

Beispiel: $(x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee x_2 \vee x_4)$

$\Longrightarrow$

$$
\begin{aligned}
T_1 &= \{1, 2+1\} & T'_1 &= \{2, 2+1\} \\
T_2 &= \{1, 2, 2+2\} & T'_2 &= \{2+2\}
\end{aligned}
$$



Sei $F = K_1 \wedge \cdots \wedge K_m$ in KNF, mit Variablen $x_1, \ldots, x_l$.

$$
\begin{aligned}
M &:= \{1, \ldots, m+l\} \\
T_i &:= \{j \mid x_i \text{ kommt in } K_j \text{ positiv vor}\} \cup \{m+i\} \\
T'_i &:= \{j \mid x_i \text{ kommt in } K_j \text{ negativ vor}\} \cup \{m+i\}
\end{aligned}
$$

Beispiel: $(x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee x_2 \vee x_4)$

$\Longrightarrow$

$$
\begin{aligned}
T_1 &= \{1, 2+1\} & T'_1 &= \{2, 2+1\} \\
T_2 &= \{1, 2, 2+2\} & T'_2 &= \{2+2\} \\
T_3 &= \{1, 2+3\} & T'_3 &= \{2+3\} \\
T_3 &= \{2, 2+4\} & T'_4 &= \{2+4\}
\end{aligned}
$$

Überdeckung: $T'_1, T_2, T3', T4'$

Sei $F = K_1 \wedge \cdots \wedge K_m$ in KNF, mit Variablen $x_1, \ldots, x_l$.

$$
\begin{aligned}
M &:= \{1, \ldots, m+l\} \\
T_i &:= \{j \mid x_i \text{ kommt in } K_j \text{ positiv vor}\} \cup \{m+i\} \\
T'_i &:= \{j \mid x_i \text{ kommt in } K_j \text{ negativ vor}\} \cup \{m+i\}
\end{aligned}
$$

Beispiel: $(x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee x_2 \vee x_4)$
$\Longrightarrow$

$$
\begin{aligned}
T_1 &= \{1, 2+1\} & T'_1 &= \{2, 2+1\} \\
T_2 &= \{1, 2, 2+2\} & T'_2 &= \{2+2\} \\
T_3 &= \{1, 2+3\} & T'_3 &= \{2+3\} \\
T_3 &= \{2, 2+4\} & T'_4 &= \{2+4\}
\end{aligned}
$$

Überdeckung: $T'_1, T_2, T3', T4' \approx x_1 = 0, x_2 = 1, x_3 = 0, x4 = 0$



Beweis (Forts.):

Sei $F = K_1 \wedge \cdots \wedge K_m$ in KNF, mit Variablen $x_1, \ldots, x_l$.

$$
\begin{aligned}
M &:= \{1, \ldots, m+l\} \\
T_i &:= \{j \mid x_i \text{ kommt in } K_j \text{ positiv vor}\} \cup \{m+i\} \\
T'_i &:= \{j \mid x_i \text{ kommt in } K_j \text{ negativ vor}\} \cup \{m+i\}
\end{aligned}
$$



Sei $F = K_1 \wedge \cdots \wedge K_m$ in KNF, mit Variablen $x_1, \ldots, x_l$.

$$
\begin{aligned}
M &:= \{1, \ldots, m+l\} \\
T_i &:= \{j \mid x_i \text{ kommt in } K_j \text{ positiv vor}\} \cup \{m+i\} \\
T'_i &:= \{j \mid x_i \text{ kommt in } K_j \text{ negativ vor}\} \cup \{m+i\}
\end{aligned}
$$

Beispiel: $(x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee x_2 \vee x_4)$
$\Longrightarrow$

$$
\begin{aligned}
T_1 &= \{1, 2+1\} & T'_1 &= \{2, 2+1\} \\
T_2 &= \{1, 2, 2+2\} & T'_2 &= \{2+2\} \\
T_3 &= \{1, 2+3\} & T'_3 &= \{2+3\} \\
T_3 &= \{2, 2+4\} & T'_4 &= \{2+4\}
\end{aligned}
$$



Sei $F = K_1 \wedge \cdots \wedge K_m$ in KNF, mit Variablen $x_1, \ldots, x_l$.

$$
\begin{aligned}
M &:= \{1, \ldots, m+l\} \\
T_i &:= \{j \mid x_i \text{ kommt in } K_j \text{ positiv vor}\} \cup \{m+i\} \\
T'_i &:= \{j \mid x_i \text{ kommt in } K_j \text{ negativ vor}\} \cup \{m+i\}
\end{aligned}
$$

## Beweis (Forts.):

Sei $F = K_1 \wedge \cdots \wedge K_m$ in KNF, mit Variablen $x_1, \ldots, x_l$.

$$\begin{aligned} M &:= \{1, \ldots, m+l\} \\ T_i &:= \{j \mid x_i \text{ kommt in } K_j \text{ positiv vor}\} \cup \{m+i\} \\ T_i' &:= \{j \mid x_i \text{ kommt in } K_j \text{ negativ vor}\} \cup \{m+i\} \end{aligned}$$

„⇒“ Gegeben $\sigma$ mit $\sigma(F) = 1$.

$$U_i := \text{if } \sigma(x_i) = 1 \text{ then } T_i \text{ else } T_i'$$



## Beweis (Forts.):

Sei $F = K_1 \wedge \cdots \wedge K_m$ in KNF, mit Variablen $x_1, \ldots, x_l$.

$$\begin{aligned} M &:= \{1, \ldots, m+l\} \\ T_i &:= \{j \mid x_i \text{ kommt in } K_j \text{ positiv vor}\} \cup \{m+i\} \\ T_i' &:= \{j \mid x_i \text{ kommt in } K_j \text{ negativ vor}\} \cup \{m+i\} \end{aligned}$$

„⇒“ Gegeben $\sigma$ mit $\sigma(F) = 1$.

$$U_i := \text{if } \sigma(x_i) = 1 \text{ then } T_i \text{ else } T_i'$$

Beh.: Die $U_i$ überdecken $M$.



## Beweis (Forts.):

Sei $F = K_1 \wedge \cdots \wedge K_m$ in KNF, mit Variablen $x_1, \ldots, x_l$.

$$\begin{aligned} M &:= \{1, \ldots, m+l\} \\ T_i &:= \{j \mid x_i \text{ kommt in } K_j \text{ positiv vor}\} \cup \{m+i\} \\ T_i' &:= \{j \mid x_i \text{ kommt in } K_j \text{ negativ vor}\} \cup \{m+i\} \end{aligned}$$

„⇒“ Gegeben $\sigma$ mit $\sigma(F) = 1$.

$$U_i := \text{if } \sigma(x_i) = 1 \text{ then } T_i \text{ else } T_i'$$

Beh.: Die $U_i$ überdecken $M$.

Sei $j \in M$.

1. $1 \leq j \leq m$:



## Beweis (Forts.):

Sei $F = K_1 \wedge \cdots \wedge K_m$ in KNF, mit Variablen $x_1, \ldots, x_l$.

$$\begin{aligned} M &:= \{1, \ldots, m+l\} \\ T_i &:= \{j \mid x_i \text{ kommt in } K_j \text{ positiv vor}\} \cup \{m+i\} \\ T_i' &:= \{j \mid x_i \text{ kommt in } K_j \text{ negativ vor}\} \cup \{m+i\} \end{aligned}$$

„⇒“ Gegeben $\sigma$ mit $\sigma(F) = 1$.

$$U_i := \text{if } \sigma(x_i) = 1 \text{ then } T_i \text{ else } T_i'$$

Beh.: Die $U_i$ überdecken $M$. Sei $j \in M$.

1. $1 \leq j \leq m$: Sei $l$ (= $x_i$ oder $\neg x_i$) ein Literal in $K_j$ mit $\sigma(l) = 1$.

## Beweis (Forts.):

Sei $F = K_1 \wedge \cdots \wedge K_m$ in KNF, mit Variablen $x_1, \ldots, x_l$.

$$\begin{aligned} M &:= \{1, \ldots, m+l\} \\ T_i &:= \{j \mid x_i \text{ kommt in } K_j \text{ positiv vor}\} \cup \{m+i\} \\ T_i' &:= \{j \mid x_i \text{ kommt in } K_j \text{ negativ vor}\} \cup \{m+i\} \end{aligned}$$

„⇒“ Gegeben $\sigma$ mit $\sigma(F) = 1$.

$$U_i := \text{if } \sigma(x_i) = 1 \text{ then } T_i \text{ else } T_i'$$

Beh.: Die $U_i$ überdecken $M$. Sei $j \in M$.

1. $1 \leq j \leq m$: Sei $l$ ($= x_i$ oder $\neg x_i$) ein Literal in $K_j$ mit $\sigma(l) = 1$.
$\Rightarrow j \in$ if $x_i$ kommt positiv in $K_j$ vor then $T_i$ else $T_i'$ per def.



## Beweis (Forts.):

Sei $F = K_1 \wedge \cdots \wedge K_m$ in KNF, mit Variablen $x_1, \ldots, x_l$.

$$\begin{aligned} M &:= \{1, \ldots, m+l\} \\ T_i &:= \{j \mid x_i \text{ kommt in } K_j \text{ positiv vor}\} \cup \{m+i\} \\ T_i' &:= \{j \mid x_i \text{ kommt in } K_j \text{ negativ vor}\} \cup \{m+i\} \end{aligned}$$

„⇒“ Gegeben $\sigma$ mit $\sigma(F) = 1$.

$$U_i := \text{if } \sigma(x_i) = 1 \text{ then } T_i \text{ else } T_i'$$

Beh.: Die $U_i$ überdecken $M$. Sei $j \in M$.

1. $1 \leq j \leq m$: Sei $l$ ($= x_i$ oder $\neg x_i$) ein Literal in $K_j$ mit $\sigma(l) = 1$.
$\Rightarrow j \in$ if $x_i$ kommt positiv in $K_j$ vor then $T_i$ else $T_i'$
$\Rightarrow j \in$ if $\sigma(x_i) = 1$ then $T_i$ else $T_i'$



## Beweis (Forts.):

Sei $F = K_1 \wedge \cdots \wedge K_m$ in KNF, mit Variablen $x_1, \ldots, x_l$.

$$\begin{aligned} M &:= \{1, \ldots, m+l\} \\ T_i &:= \{j \mid x_i \text{ kommt in } K_j \text{ positiv vor}\} \cup \{m+i\} \\ T_i' &:= \{j \mid x_i \text{ kommt in } K_j \text{ negativ vor}\} \cup \{m+i\} \end{aligned}$$

„⇒“ Gegeben $\sigma$ mit $\sigma(F) = 1$.

$$U_i := \text{if } \sigma(x_i) = 1 \text{ then } T_i \text{ else } T_i'$$

Beh.: Die $U_i$ überdecken $M$. Sei $j \in M$.

1. $1 \leq j \leq m$: Sei $l$ ($= x_i$ oder $\neg x_i$) ein Literal in $K_j$ mit $\sigma(l) = 1$.
$\Rightarrow j \in$ if $x_i$ kommt positiv in $K_j$ vor then $T_i$ else $T_i'$
$\Rightarrow j \in$ if $\sigma(x_i) = 1$ then $T_i$ else $T_i'$
$\Rightarrow j \in U_i$



## Beweis (Forts.):

Sei $F = K_1 \wedge \cdots \wedge K_m$ in KNF, mit Variablen $x_1, \ldots, x_l$.

$$\begin{aligned} M &:= \{1, \ldots, m+l\} \\ T_i &:= \{j \mid x_i \text{ kommt in } K_j \text{ positiv vor}\} \cup \{m+i\} \\ T_i' &:= \{j \mid x_i \text{ kommt in } K_j \text{ negativ vor}\} \cup \{m+i\} \end{aligned}$$

„⇒“ Gegeben $\sigma$ mit $\sigma(F) = 1$.

$$U_i := \text{if } \sigma(x_i) = 1 \text{ then } T_i \text{ else } T_i'$$

Beh.: Die $U_i$ überdecken $M$. Sei $j \in M$.

1. $1 \leq j \leq m$: Sei $l$ ($= x_i$ oder $\neg x_i$) ein Literal in $K_j$ mit $\sigma(l) = 1$.
$\Rightarrow j \in$ if $x_i$ kommt positiv in $K_j$ vor then $T_i$ else $T_i'$
$\Rightarrow j \in$ if $\sigma(x_i) = 1$ then $T_i$ else $T_i'$
$\Rightarrow j \in U_i$

2. $m+1 \leq j \leq m+k$
$\Rightarrow j \in U_{j-m}$ denn $m + (j-m) = j$

## Beweis (Forts.):

Sei $F = K_1 \wedge \cdots \wedge K_m$ in KNF, mit Variablen $x_1, \ldots, x_l$.

$$\begin{aligned} M &:= \{1, \ldots, m+l\} \\ T_i &:= \{j \mid x_i \text{ kommt in } K_j \text{ positiv vor}\} \cup \{m+i\} \\ T_i' &:= \{j \mid x_i \text{ kommt in } K_j \text{ negativ vor}\} \cup \{m+i\} \end{aligned}$$



## Beweis (Forts.):

Sei $F = K_1 \wedge \cdots \wedge K_m$ in KNF, mit Variablen $x_1, \ldots, x_l$.

$$\begin{aligned} M &:= \{1, \ldots, m+l\} \\ T_i &:= \{j \mid x_i \text{ kommt in } K_j \text{ positiv vor}\} \cup \{m+i\} \\ T_i' &:= \{j \mid x_i \text{ kommt in } K_j \text{ negativ vor}\} \cup \{m+i\} \end{aligned}$$

„$\Leftarrow$“ Sei $U_1, \ldots, U_n \in \{T_1, \ldots, T_n'\}$ eine Überdeckung vom $M$.

$\Rightarrow$ Es gibt für alle $1 \leq i \leq n$ genau eine Menge $U_i \in \{T_i, T_i'\}$ mit $m+i \in U_i$.



## Beweis (Forts.):

Sei $F = K_1 \wedge \cdots \wedge K_m$ in KNF, mit Variablen $x_1, \ldots, x_l$.

$$\begin{aligned} M &:= \{1, \ldots, m+l\} \\ T_i &:= \{j \mid x_i \text{ kommt in } K_j \text{ positiv vor}\} \cup \{m+i\} \\ T_i' &:= \{j \mid x_i \text{ kommt in } K_j \text{ negativ vor}\} \cup \{m+i\} \end{aligned}$$

„$\Rightarrow$“ Gegeben $\sigma$ mit $\sigma(F) = 1$.

$$U_i := \text{ if } \sigma(x_i) = 1 \text{ then } T_i \text{ else } T_i'$$

Beh.: Die $U_i$ überdecken $M$. Sei $j \in M$.

1. $1 \leq j \leq m$: Sei $l$ ($= x_i$ oder $\neg x_i$) ein Literal in $K_j$ mit $\sigma(l) = 1$.
$\Rightarrow j \in$ if $x_i$ kommt positiv in $K_j$ vor then $T_i$ else $T_i'$
$\Rightarrow j \in$ if $\sigma(x_i) = 1$ then $T_i$ else $T_i'$



Das **Minimierungsproblem**

Gegeben: Teilmengen $T_1, \ldots, T_n \subseteq M$ einer endlichen Menge $M$

Problem: Finde das kleinste $k$, so dass $M$ von $k$ der Teilmengen überdeckt wird.

kann auf das Entscheidungsproblem „reduziert“ werden:

Das **Minimierungsproblem**

Gegeben: Teilmengen $T_1, \ldots, T_n \subseteq M$ einer endlichen Menge $M$

Problem: Finde das kleinste $k$, so dass $M$ von $k$ der Teilmengen überdeckt wird.

kann auf das Entscheidungsproblem „reduziert“ werden:

> Finde kleinstes $k$ durch binäre Suche im Intervall $[1, n]$
> mit $O(\log n)$ Aufrufen von MÜ.



Das **Minimierungsproblem**

Gegeben: Teilmengen $T_1, \ldots, T_n \subseteq M$ einer endlichen Menge $M$

Problem: Finde das kleinste $k$, so dass $M$ von $k$ der Teilmengen überdeckt wird.

kann auf das Entscheidungsproblem „reduziert“ werden:

> Finde kleinstes $k$ durch binäre Suche im Intervall $[1, n]$
> mit $O(\log n)$ Aufrufen von MÜ.

Kann man MÜ in Zeit $O(f(n))$ entscheiden,
dann kann man das kleinste $k$ in Zeit $O(f(n) \cdot \log n)$ berechnen.

$$f(n) \text{ polynomiell} \implies f(n) \cdot \log n \text{ polynomiell}$$
$$f(n) \text{ exponentiell} \implies f(n) \cdot \log n \text{ exponentiell}$$



Die Berechnung einer **Lösung** von

Gegeben: Teilmengen $\vec{T} := T_1, \ldots, T_n \subseteq M$ einer endlichen Menge $M$, und eine Zahl $k \leq n$.

Problem: Finde eine Überdeckung von $M$ durch $k$ der Teilmengen.



Die Berechnung einer **Lösung** von

Gegeben: Teilmengen $\vec{T} := T_1, \ldots, T_n \subseteq M$ einer endlichen Menge $M$, und eine Zahl $k \leq n$.

Problem: Finde eine Überdeckung von $M$ durch $k$ der Teilmengen.

kann auf das Entscheidungsproblem reduziert werden:

Die Berechnung einer **Lösung** von

Gegeben: Teilmengen $\vec{T} := T_1, \ldots, T_n \subseteq M$ einer endlichen Menge $M$, und eine Zahl $k \leq n$.

Problem: Finde eine Überdeckung von $M$ durch $k$ der Teilmengen.

kann auf das Entscheidungsproblem reduziert werden:

**if** $(\vec{T}, M, k) \notin$ MÜ **then** output("nicht lösbar")
**else**
  $\ddot{U} := \emptyset$



Die Berechnung einer **Lösung** von

Gegeben: Teilmengen $\vec{T} := T_1, \ldots, T_n \subseteq M$ einer endlichen Menge $M$, und eine Zahl $k \leq n$.

Problem: Finde eine Überdeckung von $M$ durch $k$ der Teilmengen.

kann auf das Entscheidungsproblem reduziert werden:

**if** $(\vec{T}, M, k) \notin$ MÜ **then** output("nicht lösbar")
**else**
  $\ddot{U} := \emptyset$
  **for** $i := 1$ **to** $n$ **do**



Die Berechnung einer **Lösung** von

Gegeben: Teilmengen $\vec{T} := T_1, \ldots, T_n \subseteq M$ einer endlichen Menge $M$, und eine Zahl $k \leq n$.

Problem: Finde eine Überdeckung von $M$ durch $k$ der Teilmengen.

kann auf das Entscheidungsproblem reduziert werden:

**if** $(\vec{T}, M, k) \notin$ MÜ **then** output("nicht lösbar")
**else**
  $\ddot{U} := \emptyset$
  **for** $i := 1$ **to** $n$ **do**
    **if** $(\vec{T} - T_i, M, k) \in$ MÜ
    **then** $\vec{T} := \vec{T} - T_i$



Die Berechnung einer **Lösung** von

Gegeben: Teilmengen $\vec{T} := T_1, \ldots, T_n \subseteq M$ einer endlichen Menge $M$, und eine Zahl $k \leq n$.

Problem: Finde eine Überdeckung von $M$ durch $k$ der Teilmengen.

kann auf das Entscheidungsproblem reduziert werden:

**if** $(\vec{T}, M, k) \notin$ MÜ **then** output("nicht lösbar")
**else**
  $\ddot{U} := \emptyset$
  **for** $i := 1$ **to** $n$ **do**
    **if** $(\vec{T} - T_i, M, k) \in$ MÜ
    **then** $\vec{T} := \vec{T} - T_i$
    **else** $\ddot{U} := \ddot{U} \cup \{T_i\}$

CLIQUE

Gegeben: Ungerichteter Graph $G = (V, E)$ und Zahl $k \in \mathbb{N}$.

Problem: Besitzt $G$ eine „Clique“ der Größe mindestens $k$?
Dh eine Teilmenge $V' \subseteq V$ mit $|V'| \geq k$
und alle $u, v \in V'$ mit $u \neq v$ sind benachbart.