Informatik III

Albert-Ludwigs-Universität Freiburg
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

Christian Schindelhauer

Wintersemester 2006/07

23. Vorlesung

25.01.2007

NP-Vollständigkeit

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- **Definition:**
  - Eine Sprache S ist NP-vollständig, wenn:
  - $S \in NP$
  - S ist NP-schwierig, d.h. für alle $L \in NP$: $L \leq_{m,p} S$

- **Lemma:**
  - Sei S eine NP-vollständige Sprache.
  - Dann ist eine Sprache T ist NP-vollständig, wenn:
  - $T \in NP$
  - $S \leq_{m,p} T$

NP-Vollständigkeit

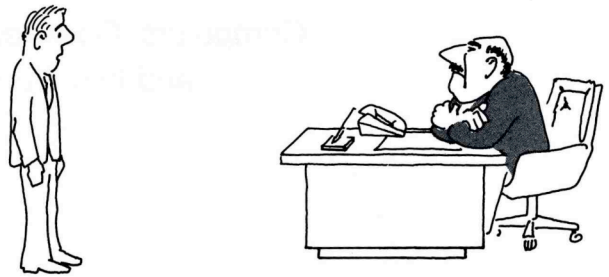
Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- **Gegeben ein unbekanntes NP-Problem X, sollte man**
  - nicht nur nach einem Algorithmus mit polynomieller Laufzeit forschen
    - $X \in P$
  - sondern auch nach einem NP-Vollständigskeitsbeweis

- **Beweisideen nicht unbedingt naheliegend …**

- **Beispiele für NP-Vollständigkeitsbeweise**
  - VERTEX-COVER ist NP-vollständig
  - (U)HAMPATH ist NP-vollständig
  - SUBSET-SUM ist NP-vollständig

specifications, and the bandersnatch department is already 13 components behind schedule. You certainly don't want to return to his office and report:



"I can't find an efficient algorithm, I guess I'm just too dumb."

To avoid serious damage to your position within the company, it would be much better if you could prove that the bandersnatch problem is *inherently* intractable, that no algorithm could possibly solve it quickly. You then could stride confidently into the boss's office and proclaim:

"I can't find an efficient algorithm, because no such algorithm is possible!"

Unfortunately, proving inherent intractability can be just as hard as finding efficient algorithms. Even the best theoreticians have been stymied in their attempts to obtain such proofs for commonly encountered hard problems. However, having read this book, you have discovered something almost as good. The theory of NP-completeness provides many straightforward techniques for proving that a given problem is "just as hard" as a large number of other problems that are widely recognized as being difficult and that have been confounding the experts for years. Armed with these techniques, you might be able to prove that the bandersnatch problem is NP-complete and, hence, that it is equivalent to all these other hard problems. Then you could march into your boss's office and announce:

"I can't find an efficient algorithm, but neither can all these famous people."

At the very least, this would inform your boss that it would do no good to fire you and hire another expert on algorithms.

Of course, our own bosses would frown upon our writing this book if its sole purpose was to protect the jobs of algorithm designers. Indeed, discovering that a problem is NP-complete is usually just the beginning of work on that problem. The needs of the bandersnatch department won't disappear overnight simply because their problem is known to be NP-complete. However, the knowledge that it is NP-complete does provide valuable information about what lines of approach have the potential of being most productive. Certainly the search for an efficient, exact algorithm should be accorded low priority. It is now more appropriate to concentrate on other, less ambitious, approaches. For example, you might look for efficient algorithms that solve various special cases of the general problem. You might look for algorithms that, though not guaranteed to run quickly, seem likely to do so most of the time. Or you might even relax the problem somewhat, looking for a fast algorithm that merely finds designs that

Wiederholung: 3-SAT

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- **Definition:**
  - 3-SAT = { ψ | ψ ist eine erfüllbare Formel in 3-CNF}
  - z.B.:

- **3-SAT ist NP-vollständig**

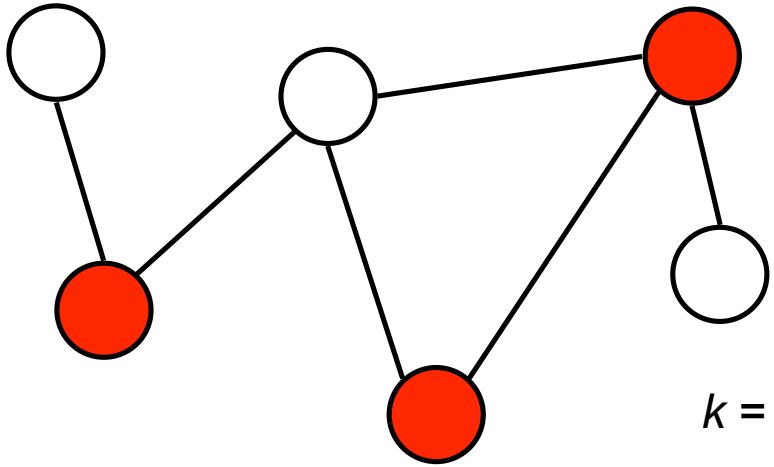
$$\psi = (x_1 \vee x_1 \vee x_2) \wedge (\overline{x_1} \vee \overline{x_2} \vee \overline{x_2}) \wedge (\overline{x_1} \vee x_2 \vee x_2)$$

VERTEX-COVER

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

➢ **Definition:**

- Eine *Knotenüberdeckung* eines ungerichteten Graphs G
- ist eine Teilmenge seiner Knoten, so dass jede Kante von
- einem dieser Knoten berührt wird.



- VERTEX-COVER = { (G, *k*) | G ist ein ungerichteter Graph mit einer *k*-Knotenüberdeckung }

VERTEX-COVER

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

Theorem:

– VERTEX-COVER ist NP-vollständig

Beweis:

– VERTEX-COVER $\in$ NP:

- k-Knotenüberdeckung dient als Zertifikat c
  - Größe offensichtlich polynomiell in Eingabelänge
- Verifizierer A(G=(V, E), k, c)
  - Prüfe, ob c Kodierung von $U \subseteq V$, wobei $|U| \leq k$
  - Für alle Knoten $u \in U$ markiere alle Kanten $\{a, b\} \in E$ mit $u \in \{a, b\}$
  - Sind alle Kanten markiert, akzeptiere. Sonst verwerfe.
- Laufzeit von A polynomiell in der Eingabelänge

– z.z.: VERTEX-COVER ist NP-schwierig

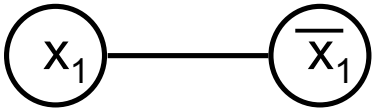
VERTEX-COVER

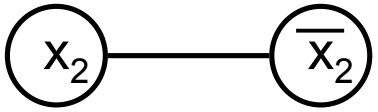
Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- **VERTEX-COVER ist NP-schwierig**
- **Beweis durch 3-SAT $\leq$m,p VERTEX-COVER**
  - Idee:

$$\psi = (x_1 \vee x_1 \vee x_2) \wedge (\overline{x_1} \vee \overline{x_2} \vee \overline{x_2}) \wedge (\overline{x_1} \vee x_2 \vee x_2)$$

  - Jede Variable aus ψ abbilden auf Knotenpaar, das für positive bzw. negative Belegung steht:

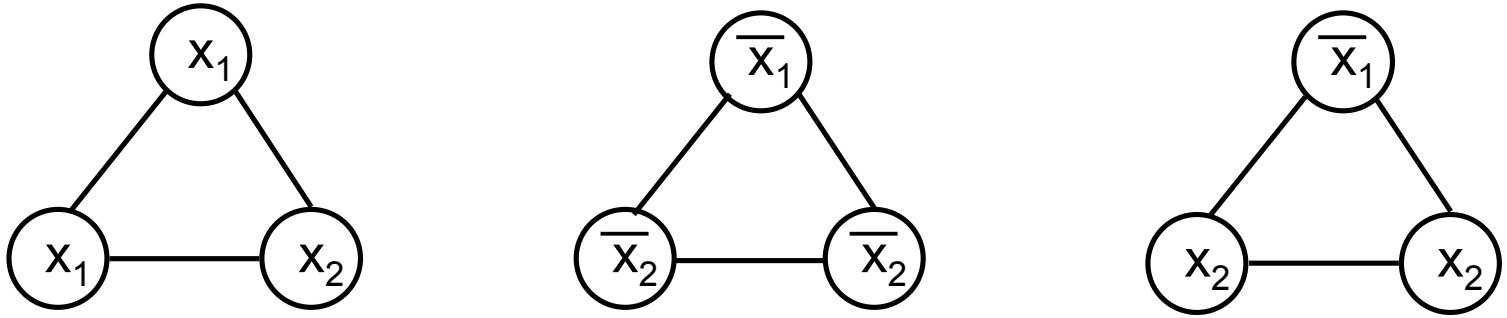




VERTEX-COVER

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

– Jede Klausel aus ψ abbilden auf Knotentripel, das den Literalen entspricht:

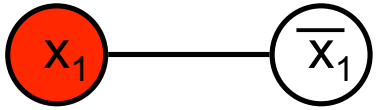


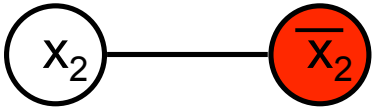
$$\psi = (x_1 \vee x_1 \vee x_2) \wedge (\overline{x_1} \vee \overline{x_2} \vee \overline{x_2}) \wedge (\overline{x_1} \vee x_2 \vee x_2)$$

VERTEX-COVER

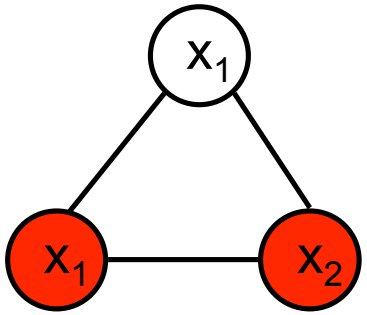
Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

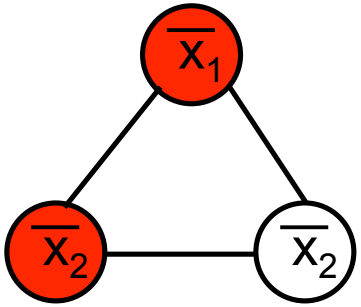
- ψ habe m Variablen und n Klauseln. Wähle $k = m + 2n$
- k-Knotenüberdeckung z.B.

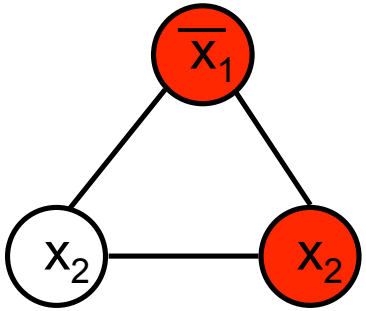




$k = 8$



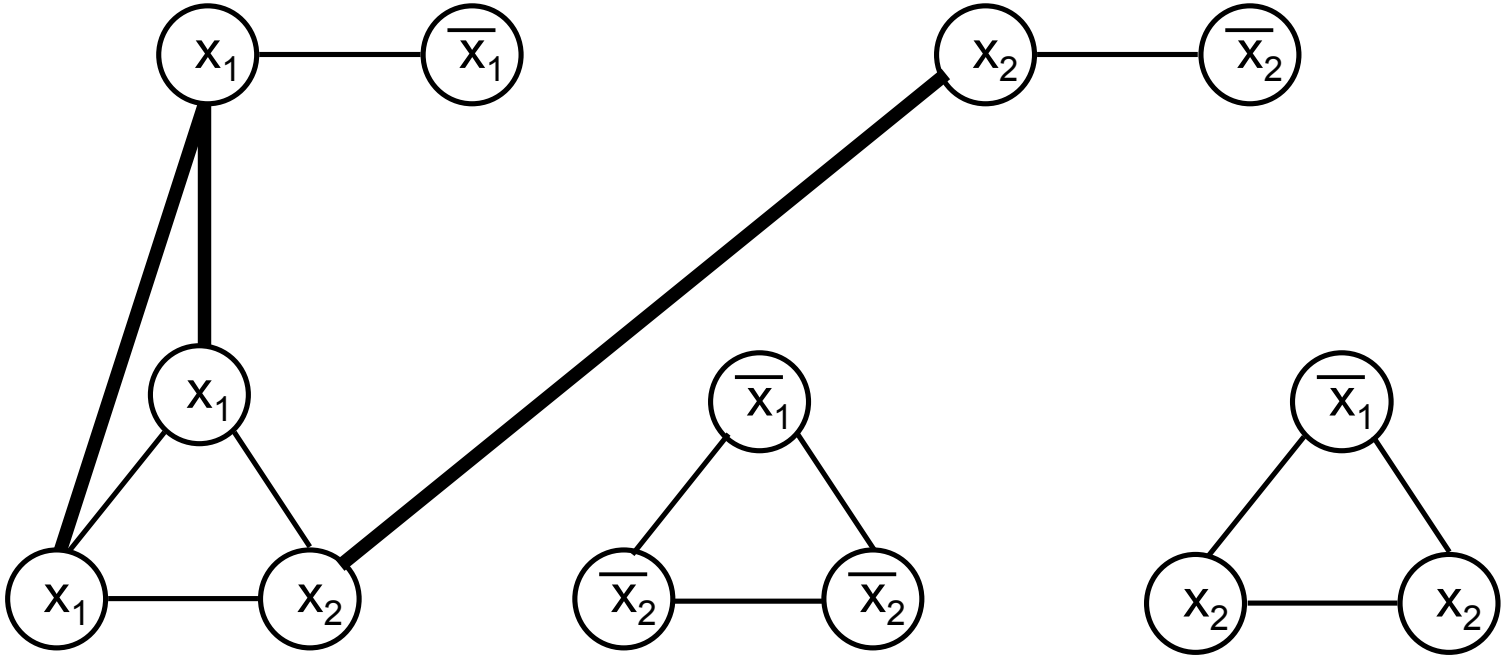




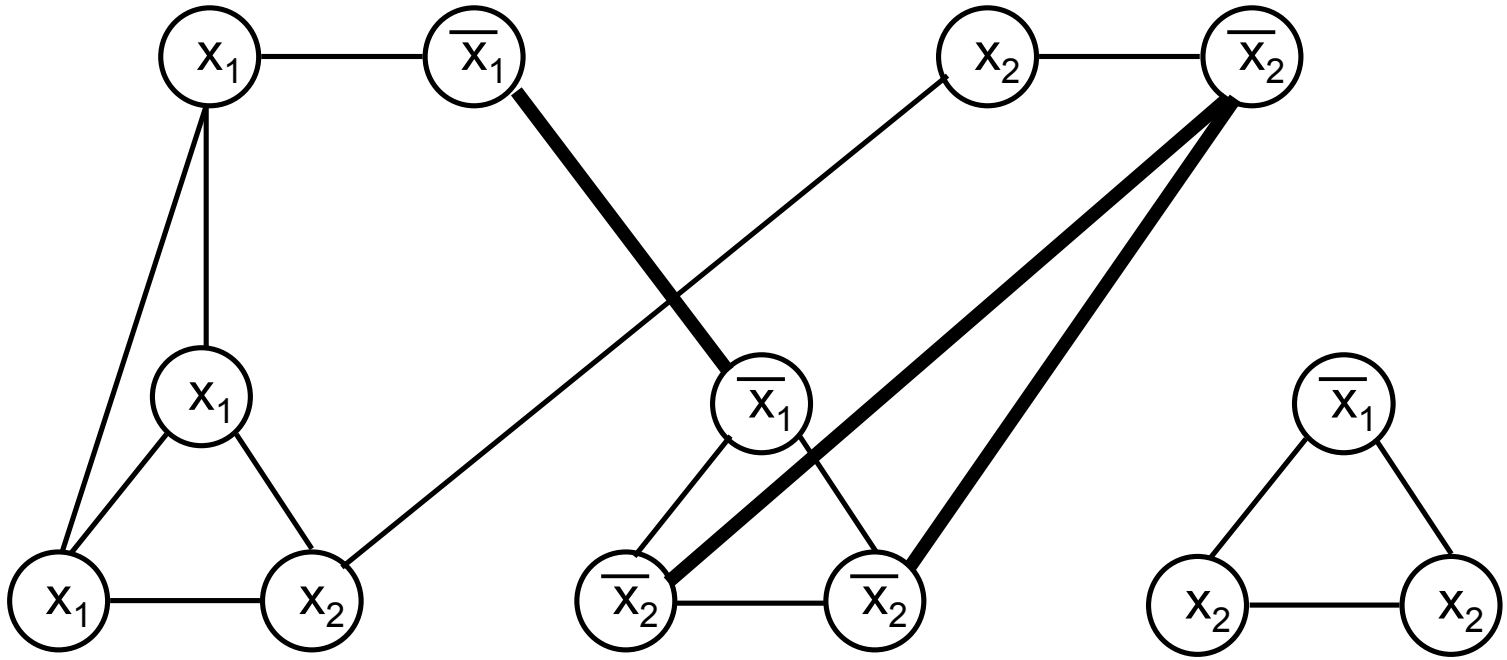
VERTEX-COVER

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

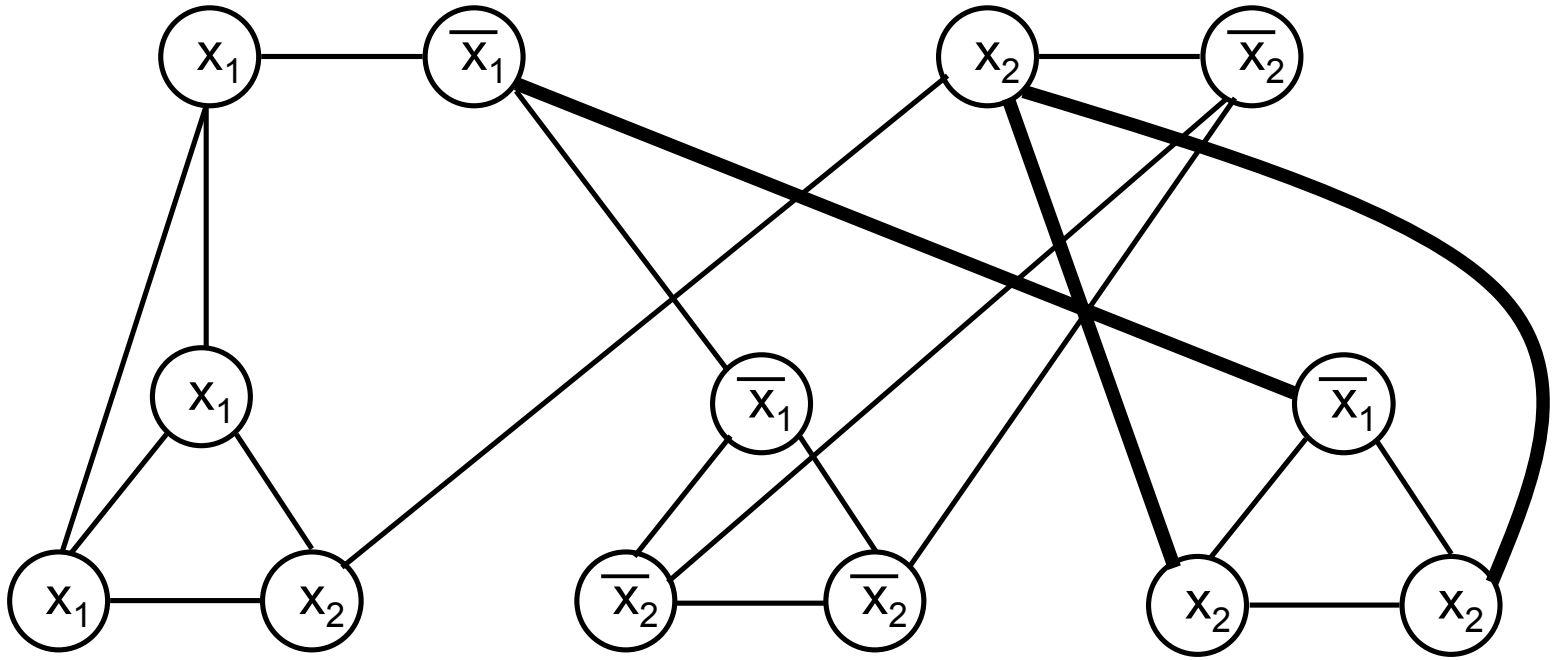
– Verbinde ‚Literale' der Knotentripel mit den entsprechenden Knoten der Variablen-Knotenpaare



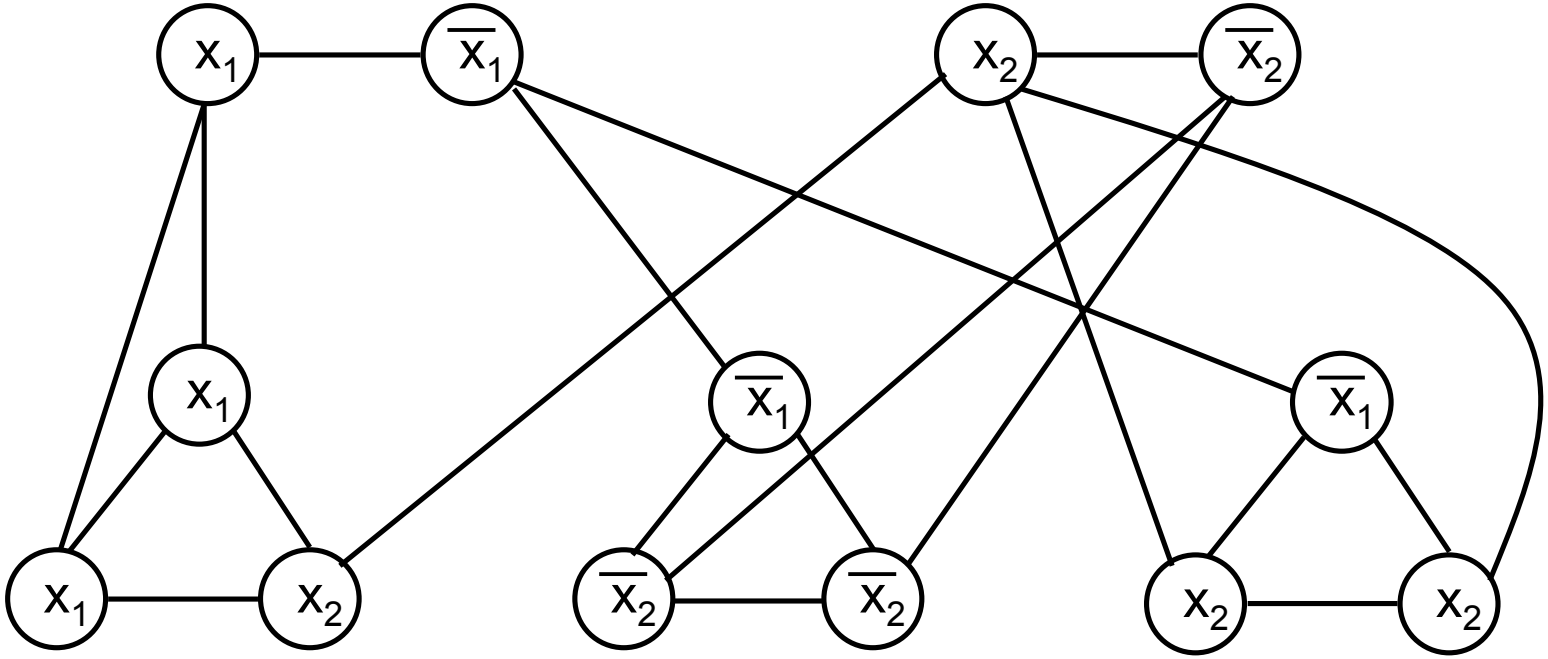
– Verbinde ‚Literale' der Knotentripel mit den entsprechenden Knoten der Variablen-Knotenpaare



– Verbinde ‚Literale' der Knotentripel mit den entsprechenden Knoten der Variablen-Knotenpaare



– z.z.: ψ ist erfüllbar genau dann, wenn G eine k-Knotenüberdeckung hat



$$\psi = (x_1 \vee x_1 \vee x_2) \wedge (\overline{x_1} \vee \overline{x_2} \vee \overline{x_2}) \wedge (\overline{x_1} \vee x_2 \vee x_2)$$

VERTEX-COVER

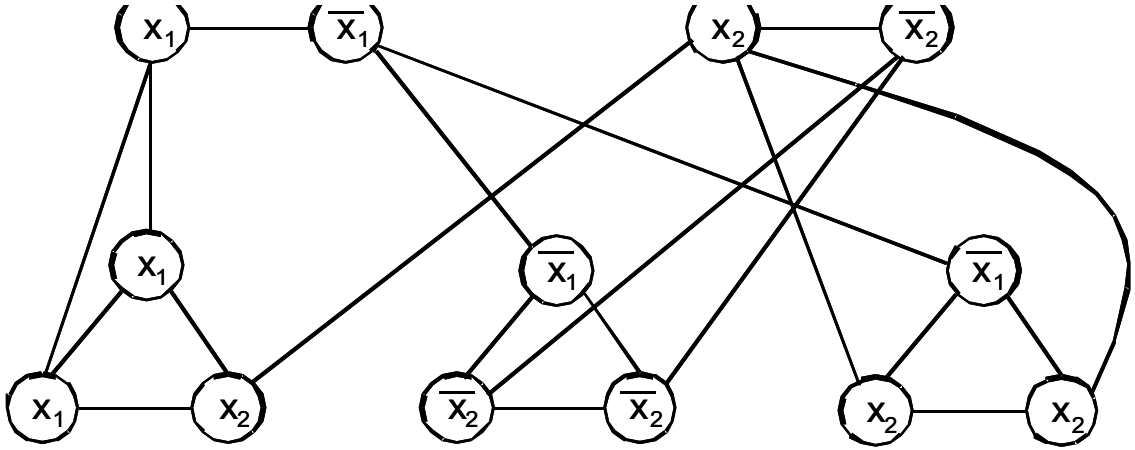
Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- **ψ ist erfüllbar → G hat eine *k*-Knotenüberdeckung**
  - Wähle die Knotenüberdeckung wie folgt:
  - Für jede in der erfüllenden Belegung von ψ mit „wahr“ belegte Variable x_i wähle im entsprechenden Knotenpaar in G den Knoten x_i; falls x_i mit „falsch“ belegt ist den entsprechenden negierten Knoten.
  - ψ ist erfüllbar
    → jede Klausel von ψ ist erfüllbar
    → jede Klausel enthält maximal zwei falsche Literale. Wähle diese Knoten. (Falls weniger als zwei Literale falsch sind, fülle mit beliebigen wahren Literalen auf.)
  - Gewählte Knotenüberdeckung hat offensichtlich Größe *k*

VERTEX-COVER

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

– Gewählte k-Knotenüberdeckung berührt alle Kanten:
  - Kanten der Knotenpaare, da je einer der beiden gewählt
  - Kanten der Knotentripel, da je zwei der drei gewählt
  - Verbindungskanten zu erfüllten Literalen in Variablen-Knotenpaaren abgedeckt
  - Verbindungskanten zu nicht erfüllten Literalen über Knotentripel abgedeckt



$$\psi = (x_1 \vee x_1 \vee x_2) \wedge (\overline{x_1} \vee \overline{x_2} \vee \overline{x_2}) \wedge (\overline{x_1} \vee x_2 \vee x_2)$$

VERTEX-COVER

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

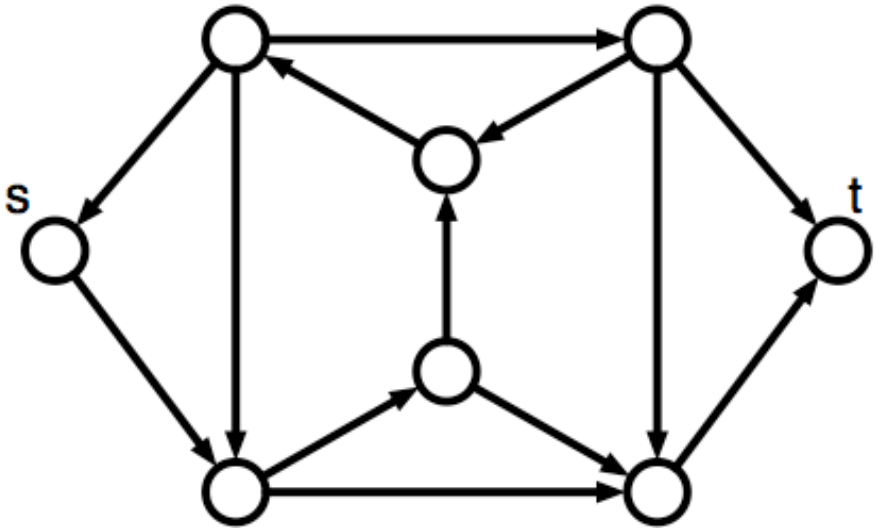
- **G hat eine *k*-Knotenüberdeckung → ψ ist erfüllbar**
  - In jedem Variablen-Knotenpaar muss genau ein Knoten der Überdeckung angehören. Belege die Variable in ψ dementsprechend.
  - In jedem Klausel-Knotentripel müssen genau zwei Knoten Teil der Überdeckung sein. Die zu dem dritten Knoten gehörende Verbindungskante muss daher durch einen Variablen-Knoten überdeckt sein
    → das entsprechende Literal ist wahr
    → die Klausel ist erfüllt

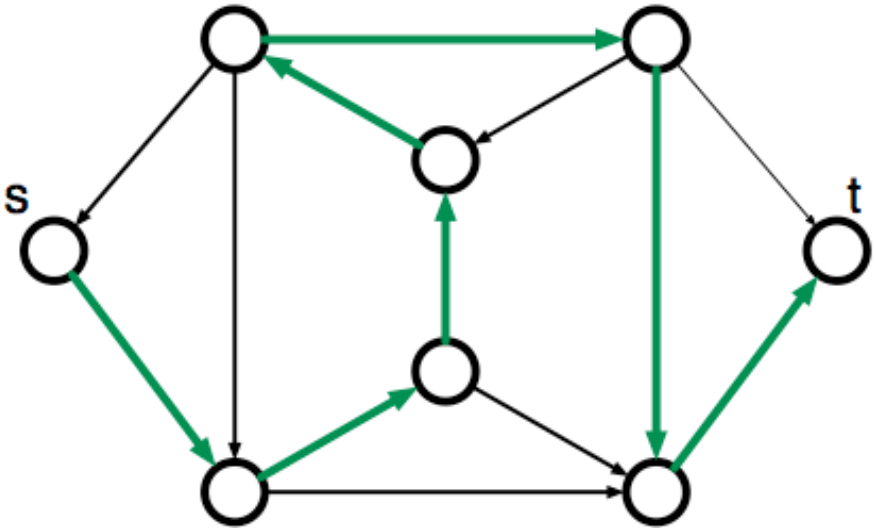
- **Theorem:**
  - VERTEX-COVER ist NP-vollständig

Hamiltonsche Pfade

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- **Definition:**
  - HAMPATH = { (G, *s*, *t*) | Der gerichtete Graph G enthält einen Weg von *s* nach *t*, der jeden Knoten genau einmal besucht. }





Hamiltonsche Pfade

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- **Theorem:**
  - HAMPATH ist NP-vollständig

- **Beweis:**
  - HAMPATH $\in$ NP:
    - Hamiltonscher Pfad (s, v_1, v_2, ..., v_{n-2}, t) dient als Zertifikat c (Größe offensichtlich polynomiell in Eingabelänge)
    - Verifizierer A(G=(V, E), s, t, c)
      - Prüfe, ob c Kodierung einer Permutation der Knoten (s, v_1, v_2, ..., v_{n-2}, t) ist
      - Für je zwei aufeinander folgende Knoten (x_1, x_2) $\in$ c, prüfe, ob es eine gerichtete Kante (x_1, x_2) $\in$ E gibt. Falls nicht, verwerfe.
      - Akzeptiere.
    - Laufzeit von A polynomiell in der Eingabelänge
  - z.z.: HAMPATH ist NP-schwierig

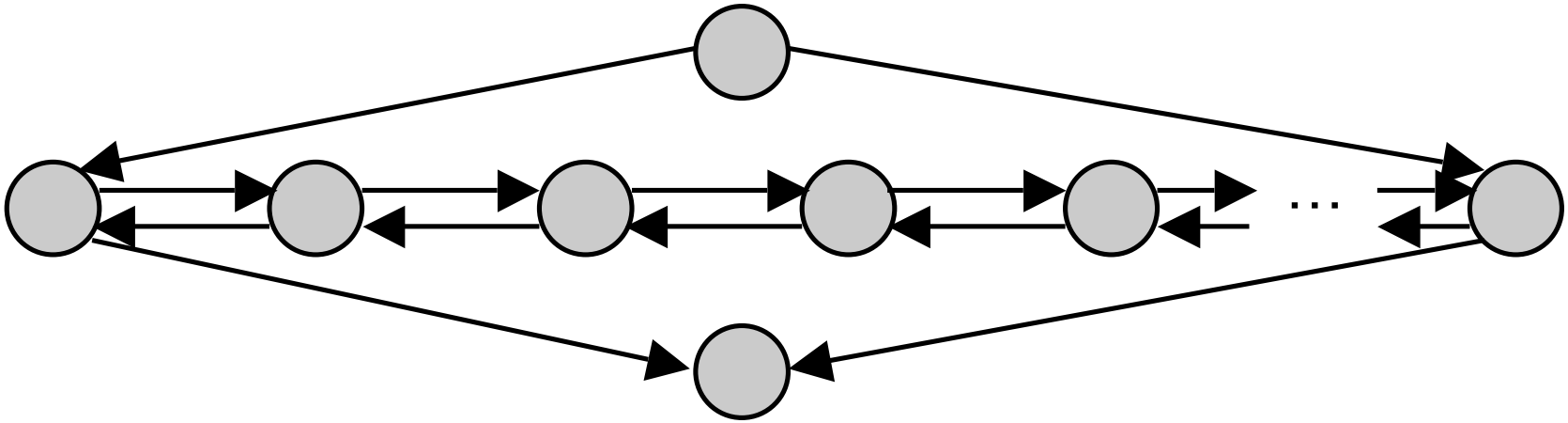
Hamiltonsche Pfade

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

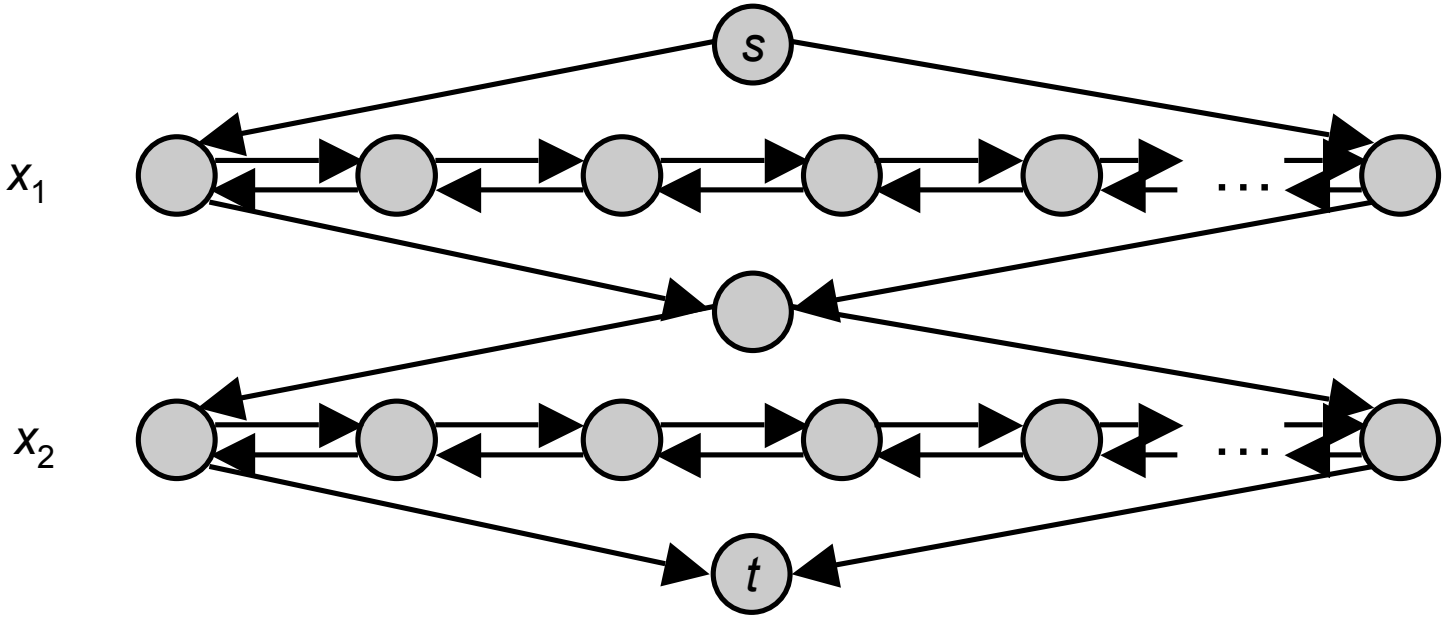
- **HAMPATH ist NP-schwierig**
- **Beweis durch 3-SAT $\leq$m,p HAMPATH**
  - Idee:

$$\psi = (x_1 \vee x_1 \vee x_2) \wedge (\overline{x_1} \vee \overline{x_2} \vee \overline{x_2}) \wedge (\overline{x_1} \vee x_2 \vee x_2)$$

  - Jede Variable x_i aus ψ abbilden auf rautenartige Knotenstruktur mit einer horizontalen Knotenzeile

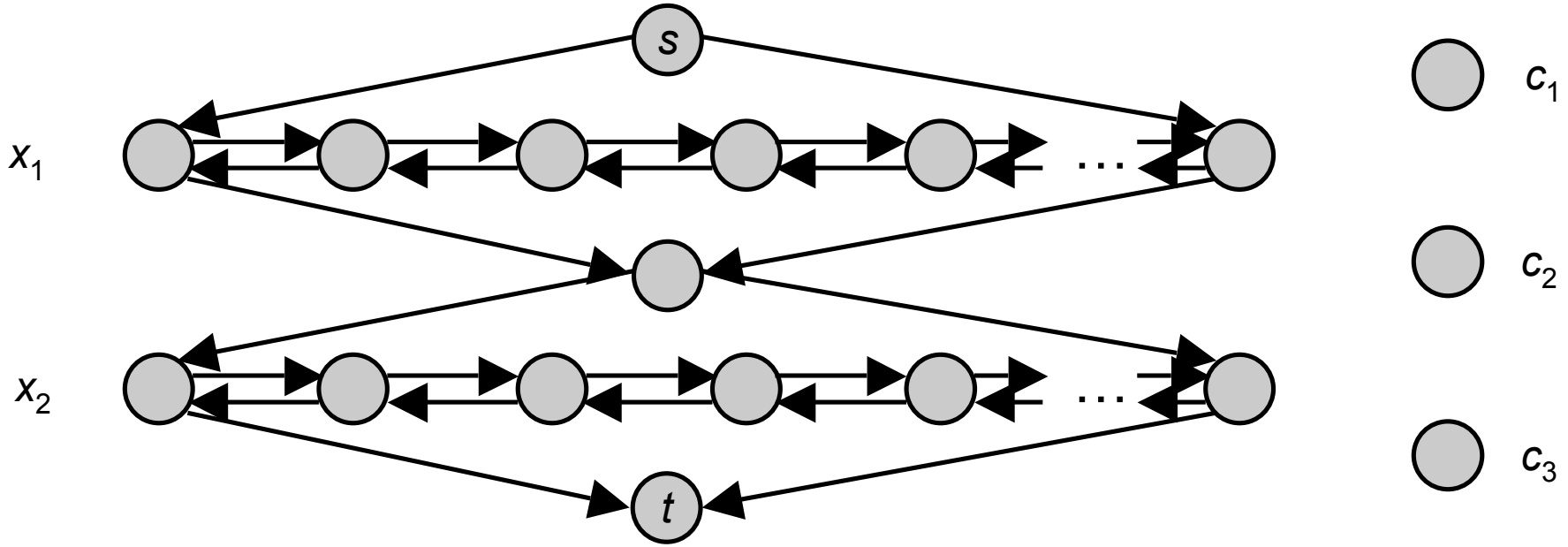


- Rautenförmige Knotenkonstrukte für die einzelnen Variablen xi verketten
- Startknoten s, Endknoten t

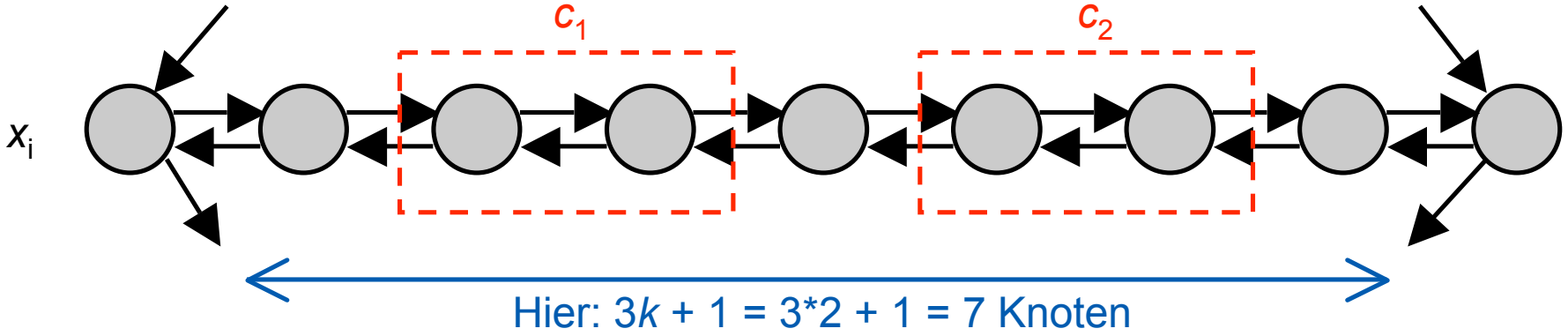


– Klauseln c_i aus ψ abbilden auf separate Knoten

$$\psi = (x_1 \vee x_1 \vee x_2) \wedge (\overline{x_1} \vee \overline{x_2} \vee \overline{x_2}) \wedge (\overline{x_1} \vee x_2 \vee x_2)$$



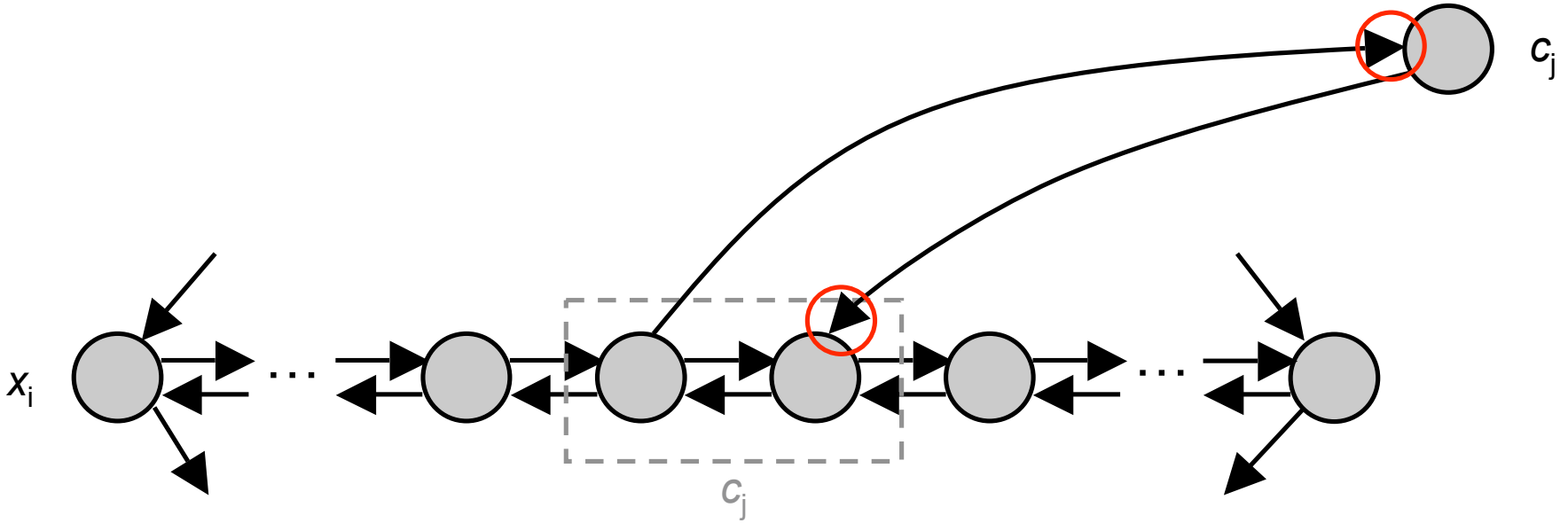
– Horizontale Knotenzeile einer Knotenstruktur im Detail:
  - Für jede Klausel c_1 ... c_k ein Knotenpaar
  - „Trenner-Knoten“ zwischen den einzelnen Knotenpaaren, sowie am Anfang und Ende der Knotenzeile
  - d.h. 3k + 1 Knoten im Inneren der Rauten-Knotenstruktur



Hamiltonsche Pfade

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

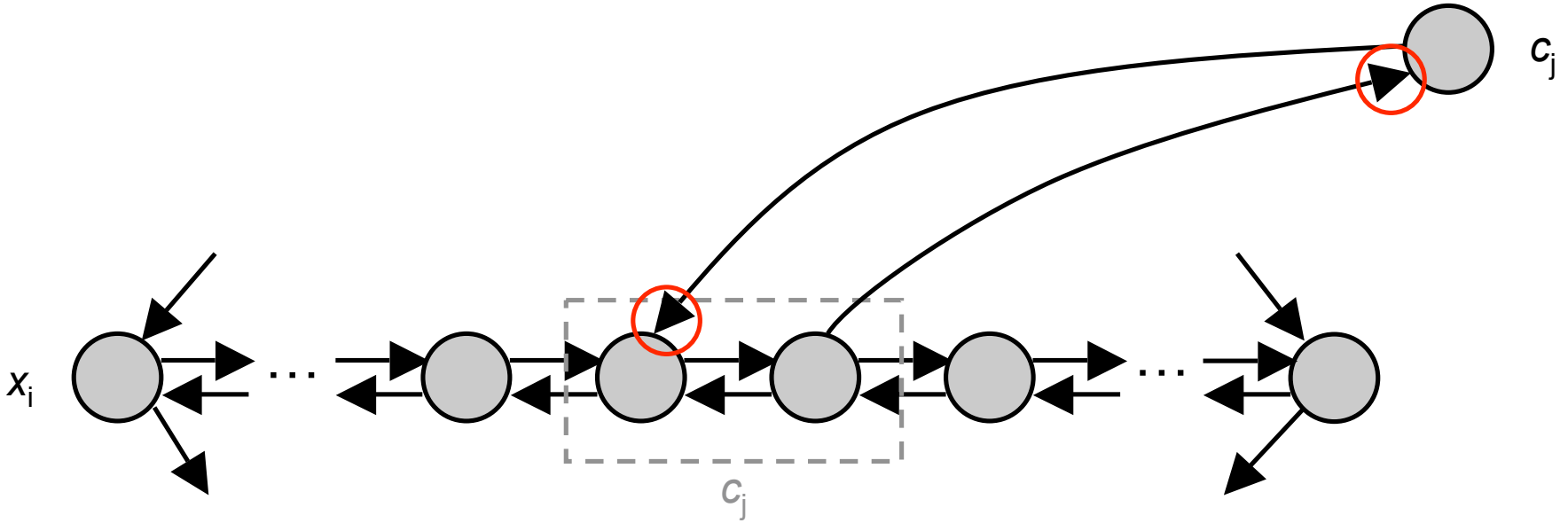
– Zusätzliche Kanten zu den Klausel-Knoten
  • Falls Klausel c_j die Variable x_i enthält:



Hamiltonsche Pfade

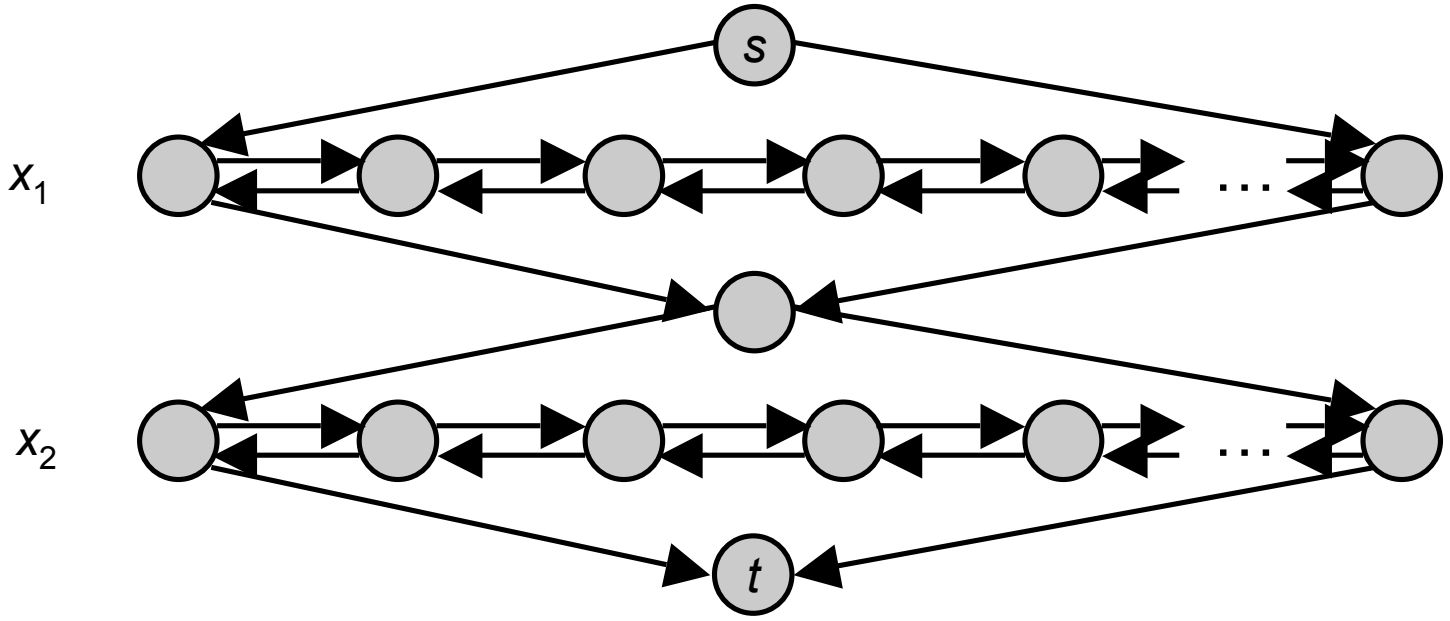
Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

– Zusätzliche Kanten zu den Klausel-Knoten
  • Falls Klausel c_j die Variable $\overline{x_i}$ enthält:

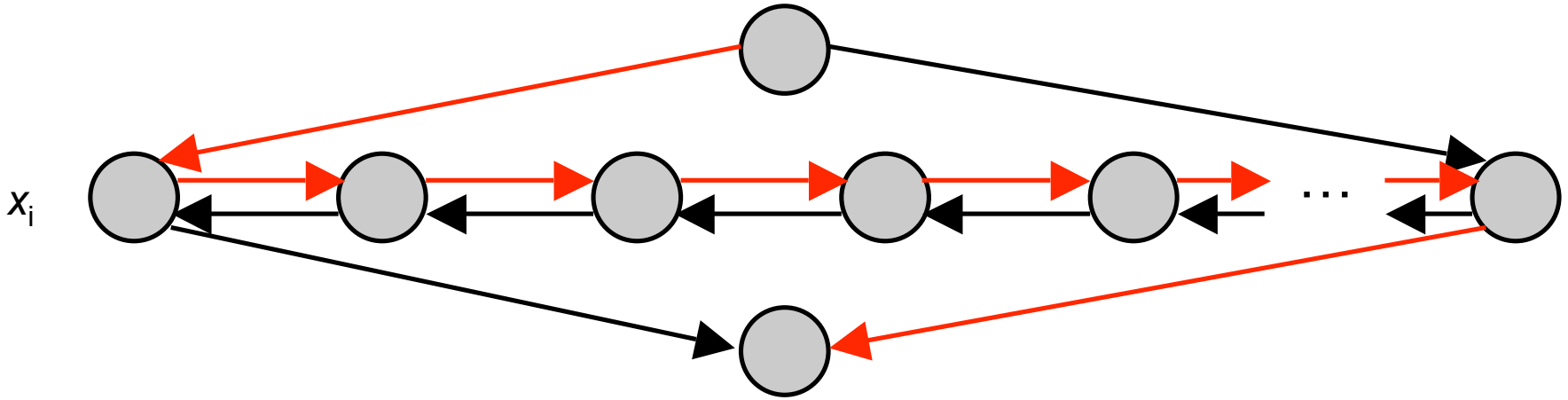


Hamiltonsche Pfade

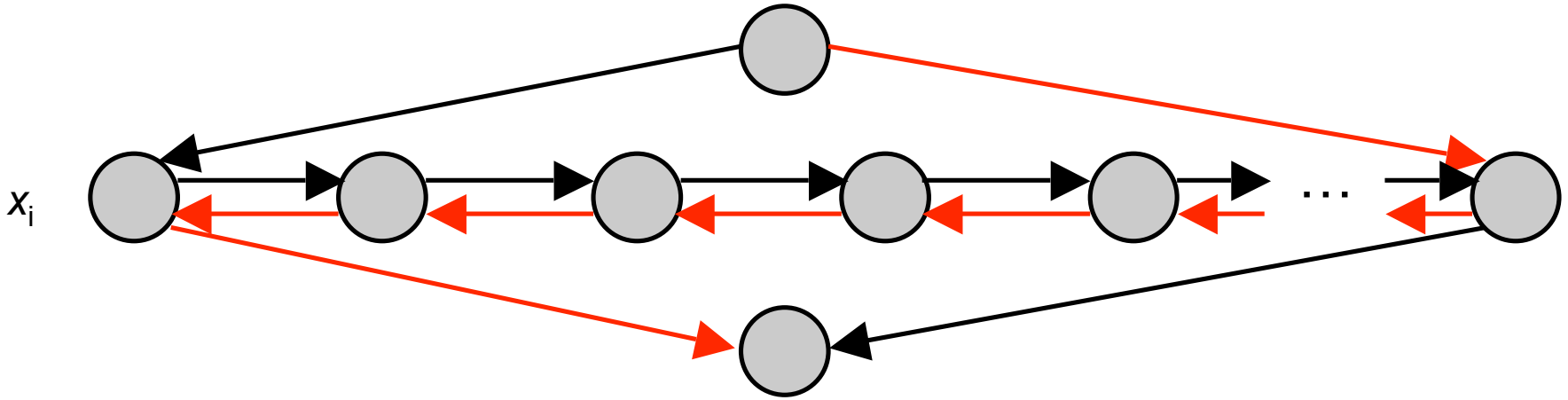
- **ψ ist erfüllbar → G besitzt einen Hamiltonschen Pfad**
  - Vernachlässige zunächst die separaten Klausel-Knoten
  - Dann gibt es einen Pfad von *s* nach *t*, der der Reihe nach alle rautenförmigen Knotenstrukturen durchläuft



– Durchlaufe dabei eine Rautenstruktur „zick-zack“, wenn die Variable x_i in der erfüllenden Belegung von ψ mit „wahr“ belegt ist, d.h.



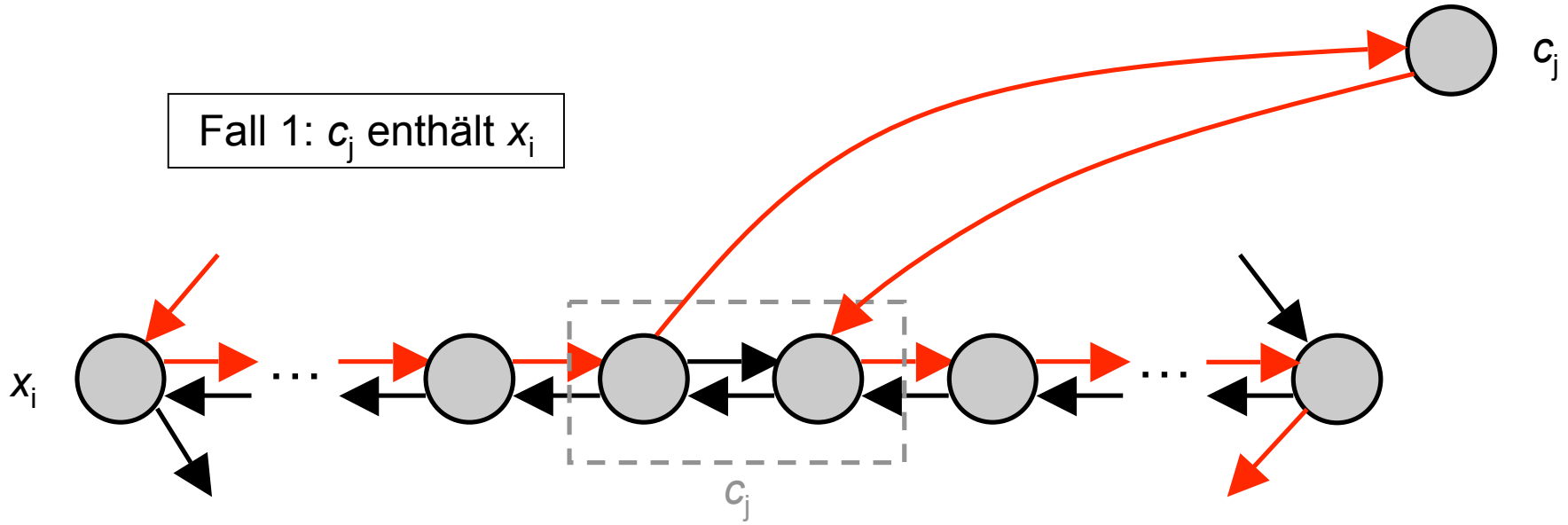
– Durchlaufe dabei eine Rautenstruktur „zack-zick“, wenn die Variable x_i in der erfüllenden Belegung von ψ mit „falsch“ belegt ist, d.h.

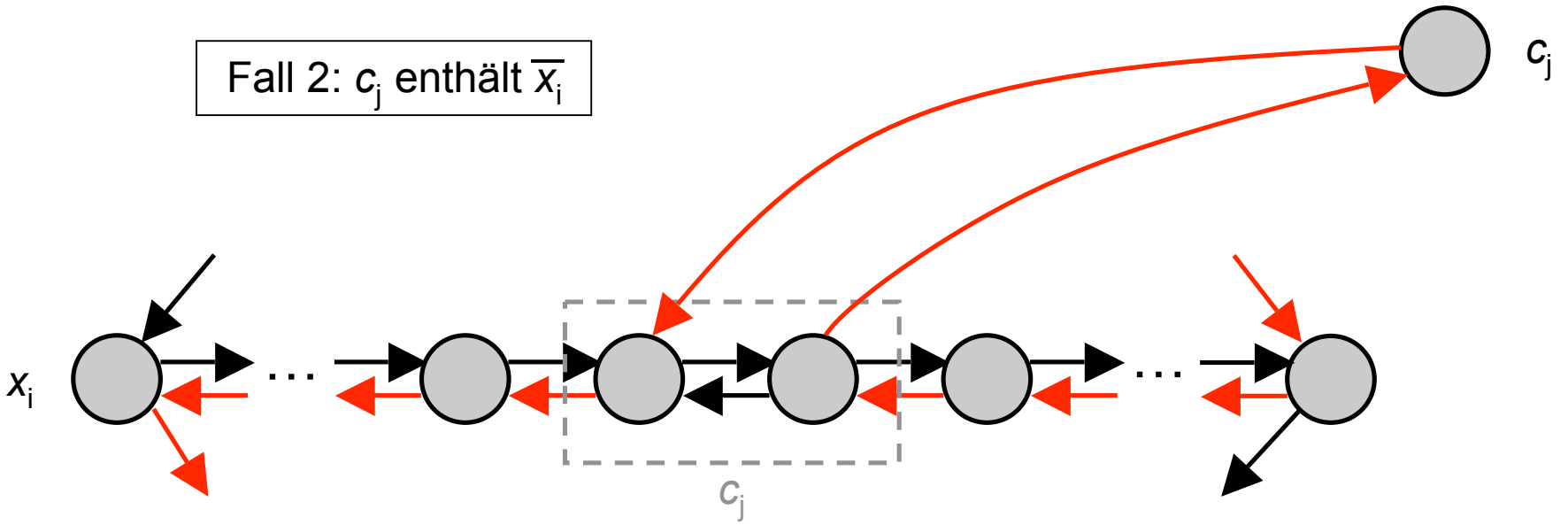


Hamiltonsche Pfade

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

- Binde separate Klausel-Knoten in Pfad ein
- ψ ist erfüllbar → jede Klausel von ψ ist erfüllbar
  → jede Klausel enthält mindestens ein wahres Literal
- Wähle in jeder Klausel *genau ein* wahres Literal und baue Umweg über Klausel-Knoten in Pfad ein





– Durch Wahl des Wegs „zick-zack“ bzw. „zack-zick“ abhängig von der Variablenbelegung, zeigen die Kanten zu den Klauselknoten für wahre Literale stets in die richtige Richtung
→ Der beschriebe Pfad von *s* nach *t* existiert und ist ein Hamiltonscher Pfad

Ende der 23. Vorlesung

Albert-Ludwigs-Universität Freiburg
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

Christian Schindelhauer

Wintersemester 2006/07

23. Vorlesung

25.01.2007