Algorithmen & Komplexität

Johannes Lengler
Institut für Theoretische Informatik

Teil 2

Berechenbarkeit und Komplexität

Berechenbarkeit und Komplexität

Berechenbarkeit:

Welche Probleme lassen sich überhaupt berechnen?

Komplexität:

Welche Probleme lassen sich effizient berechnen?

P und NP

Wir betrachten Funktionen der Form $f : \{0,1\}^* \to \{0,1\}$.

Äquivalent: Betrachten Sprachen $L \subseteq \{0,1\}^*$.

Wir bezeichnen f (bzw. L) auch als *Entscheidungsproblem*.

Definition: $L \subseteq \{0,1\}^*$

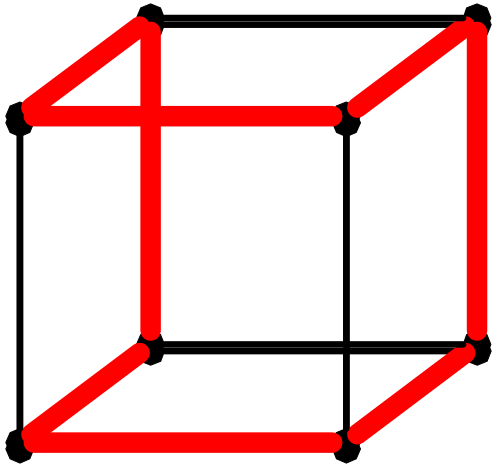
Eine Sprache ist in P, falls es einen Algorithmus A gibt, der in polynomieller Zeit mit A(x)=L(x) terminiert.

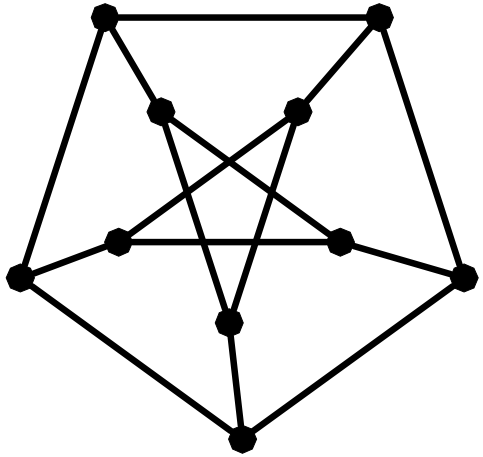
Probleme in P

Probleme in P:

- ZUSAMMENHANG
  $L = \{G \mid \ G \text{ zusammenhängender Graph}\}$
- DURCHMESSER
  $L = \{(G, b) \mid G \text{ Graph}, b \in \mathbb{R}, \text{alle Knotenpaare haben Distanz} \leq b\}$
- PLANARITÄT
  $L = \{G \mid G \text{ kann “überkreuzungsfrei” in der Ebene gezeichnet werden}\}$
- PRIMES
  $L = \{p \in \mathbb{N} \mid p \text{ prim}\}$

Beispiel für NP: Hamiltonkreis





kein Hamiltonkreis …

P und NP

Wir betrachten Funktionen der Form $f : \{0,1\}^* \to \{0,1\}$.

Äquivalent: Betrachten Sprachen $L \subseteq \{0,1\}^*$.

Wir bezeichnen f (bzw. L) auch als *Entscheidungsproblem*.

Definition: $L \subseteq \{0,1\}^*$

Eine Sprache ist in P, falls es einen Algorithmus A gibt, der in polynomieller Zeit mit A(x)=L(x) terminiert.

Eine Sprache L ist in NP (*nicht-deterministisch polynomiell*), falls es einen Verifizierer V gibt, sodass V in polynomieller Zeit terminiert, und sodass

$$L(x) = 1 \iff \exists p \in \{0,1\}^* \text{ mit } |p| = \text{poly}(|x|) : V(x,p) = 1$$

und $L(x) = 0 \iff \forall p \in \{0,1\}^* : V(x,p) = 0.$

Probleme in NP

Probleme in NP:

- alle Probleme in P
- GRAPH-ISOMORHPHISMUS

  $L = \{(G, H) \mid G \text{ und } H \text{ sind isomorphe Graphen}\}$
- 3-COL

  $L = \{G \mid \ G \text{ ist mit 3 Farben färbbar}\}$
- CLIQUE

  $L = \{(G, m) \mid \text{Es gibt } m \text{ paarweise adjazente Knoten in } G\}$
- HAMILTONKREIS

  $L = \{G \mid G \text{ hat einen Kreis, der jeden Knoten genau einmal besucht}\}$
- NULLSTELLE-MOD-N

  $L = \{(F, n) \mid F \text{ ist ein Polynom, das über } \mathbb{Z}/n\mathbb{Z} \text{ eine Nullstelle hat}\}$

  z.B.: $(x^2 + 1,\ 41) \in L, \qquad (x^2 + y^2 + z^2 - 7,\ 8) \notin L$
- SAT (SATISFIABILITY)

  $L = \{(F, n) \mid F \text{ ist erfüllbare aussagenlogische Formel in K-Normalform}\}$

  z.B.: $(x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee x_2 \vee \overline{x}_3) \wedge (\overline{x}_1) \wedge (\overline{x}_2) \notin L$

Sprachen ausserhalb NP.

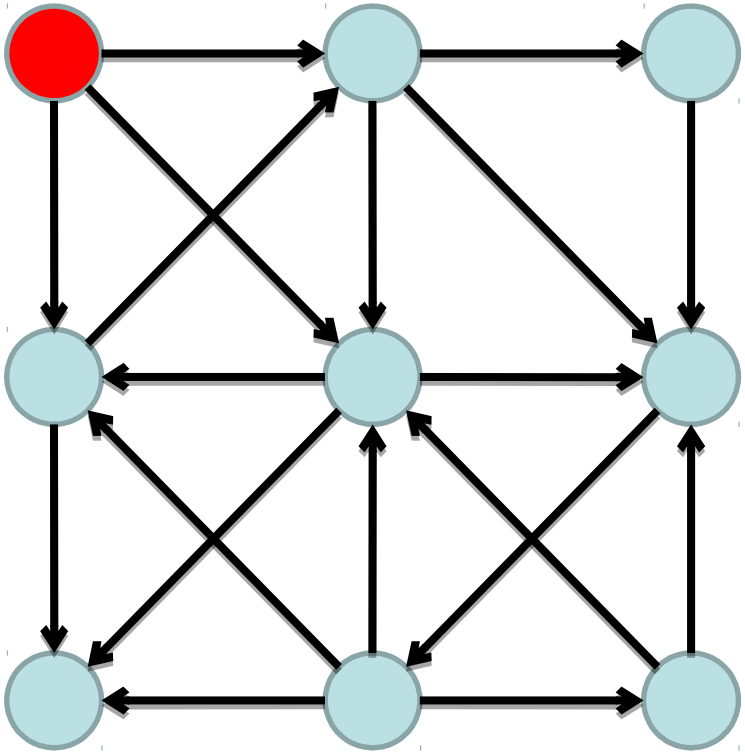
Die folgende Sprache ist *vermutlich* nicht in NP.

Betrachte folgendes Spiel:

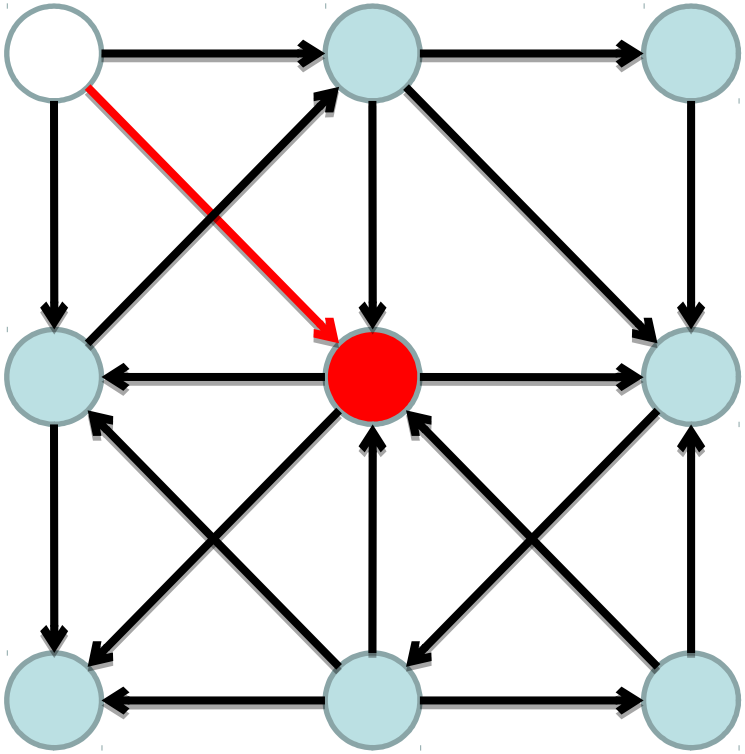
- Gegeben ein gerichteter Graph und ein Knoten v mit einer Kugel
- Abwechslungsweise bewegen die Spieler die Kugel auf zuvor nicht besuchte Knoten.
- Wer nicht mehr ziehen kann verliert.

$$L = \{(G, v) \mid \text{Startspieler hat Gewinnstrategie auf } (G, v)\}$$

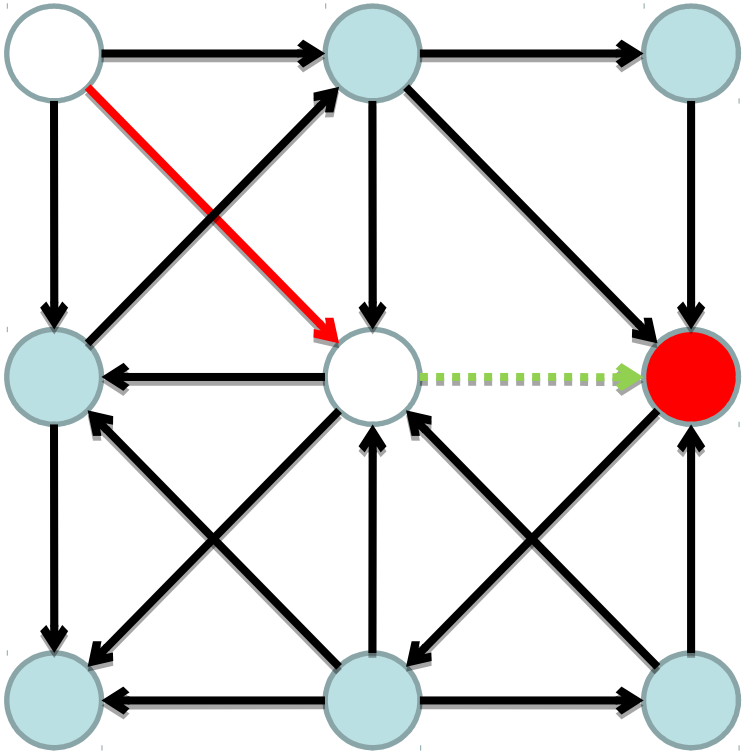
Sprachen ausserhalb NP



Sprachen ausserhalb NP



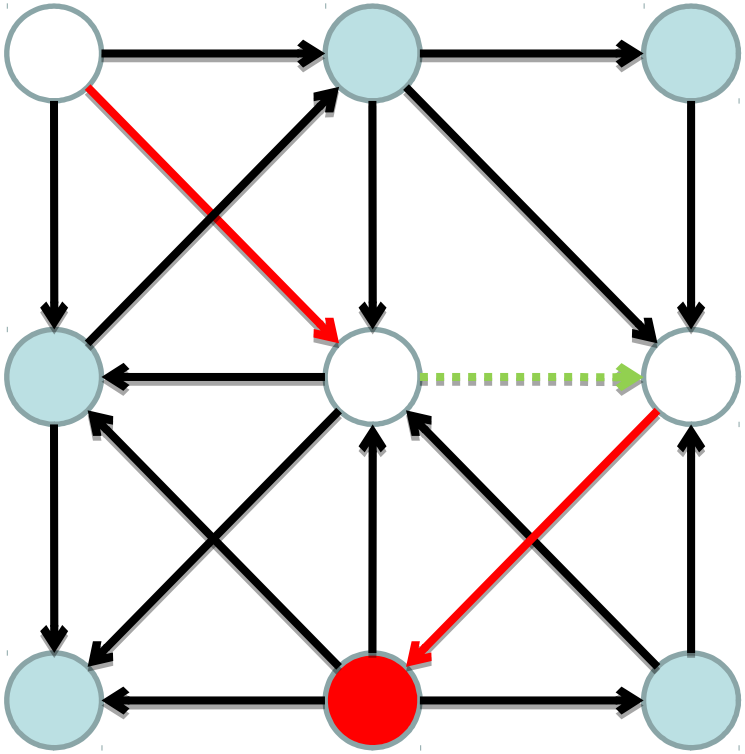
Sprachen ausserhalb NP



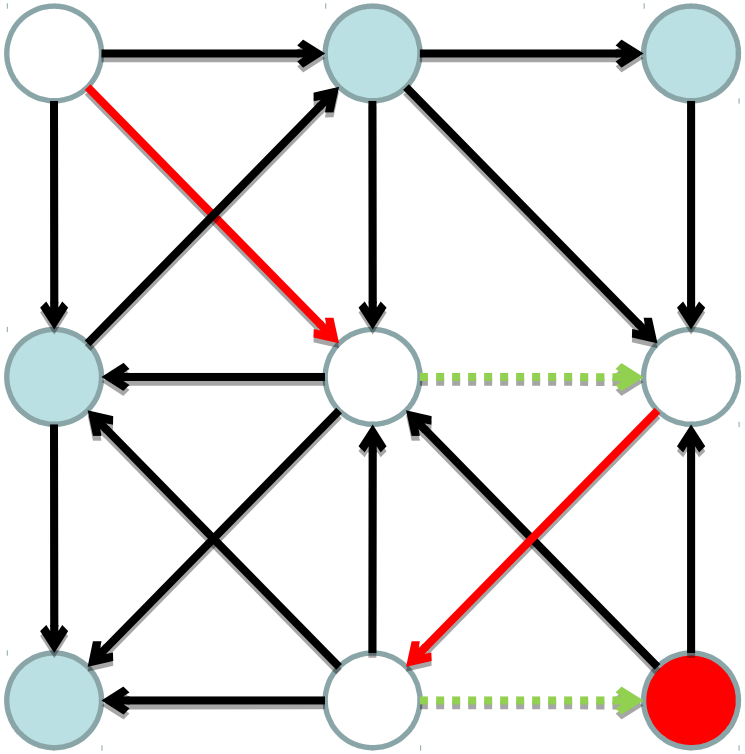
Alice

Bob

Sprachen ausserhalb NP



Sprachen ausserhalb NP

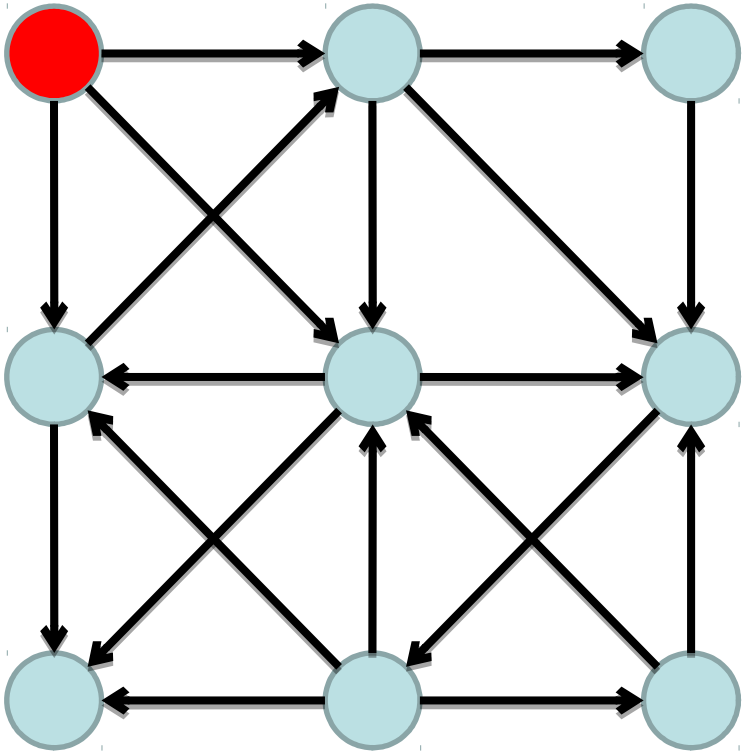


Alice

Bob

Bob gewinnt

Sprachen ausserhalb NP



Behauptung: Bei optimalem Spiel gewinnt Bob (für diesen Graphen).

Im Allgemeinen gibt es *vermutlich* keinen “kurzen Beweis” für die Aussage “Bob gewinnt”.

Komplexitätstheorie

P = *effizient entscheidbare Probleme*

NP = *(einseitig) effizient verifizierbare Probleme*

$P \stackrel{?}{=} NP$ → *1 Million US-$ (Clay-Foundation)*

[eines von sieben Millennium-Problemen]

Reduzierbarkeit

Definition:

Seien $f, g : \{0,1\}^* \to \{0,1\}$ Entscheidungsprobleme. Wir sagen, dass *f* *polynomiell reduzierbar* auf *g* ist, $f \leq_p g$, falls es eine Funktion $\kappa : \{0,1\}^* \to \{0,1\}^*$ gibt, die in polynomieller Zeit berechnet werden kann, mit:

$$\forall x \in \{0,1\}^* : \; f(x) = g(\kappa(x)).$$

Lemma:

Ist $f \leq_p g$, und ist *g* in P, so ist auch *f* in P.

NP-vollständig

Definition:

Ein Entscheidungsproblem *g* heisst *NP-schwer* genau dann, wenn $f \leq_p g$ für alle Probleme f in NP gilt. Ist zusätzlich g in NP, so heisst g *NP-vollständig*.

Lemma:

Ist $f \leq_p g$, und ist *f* NP-schwer, so ist auch *g* NP-schwer.

- Wir werden sehen: Es gibt NP-vollständige Probleme.
- Wenn Sie für ein(!) solches Problem einen polynomiellen Algorithmus finden, haben sie P=NP gezeigt!!

Probleme in NP

Probleme in NP:

- alle Probleme in P
- GRAPH-ISOMORHPHISMUS

  $L = \{(G, H) \mid G \text{ und } H \text{ sind isomorphe Graphen}\}$
- 3-COL

  $L = \{G \mid \ G \text{ ist mit 3 Farben färbbar}\}$
- CLIQUE

  $L = \{(G, m) \mid \text{Es gibt } m \text{ paarweise adjazente Knoten in } G\}$
- HAMILTONKREIS

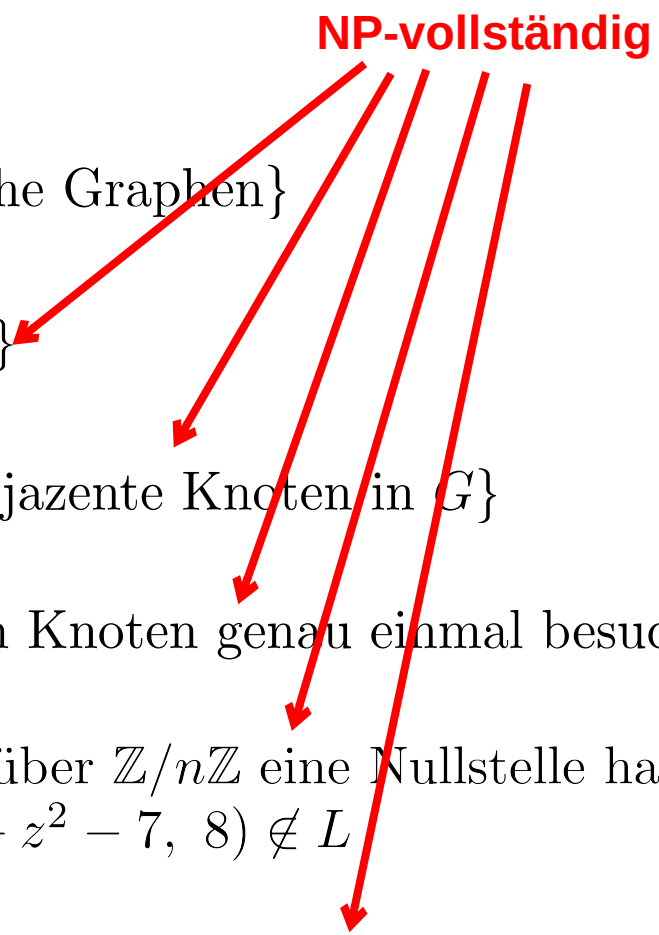
  $L = \{G \mid G \text{ hat einen Kreis, der jeden Knoten genau einmal besucht}\}$
- NULLSTELLE-MOD-N

  $L = \{(F, n) \mid F \text{ ist ein Polynom, das über } \mathbb{Z}/n\mathbb{Z} \text{ eine Nullstelle hat}\}$

  z.B.: $(x^2 + 1,\ 41) \in L, \qquad (x^2 + y^2 + z^2 - 7,\ 8) \notin L$
- SAT (SATISFIABILITY)

  $L = \{(F, n) \mid F \text{ ist erfüllbare aussagenlogische Formel in K-Normalform}\}$

  z.B.: $(x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee x_2 \vee \overline{x}_3) \wedge (\overline{x}_1) \wedge (\overline{x}_2) \notin L$



SAT ist NP-vollständig

Theorem (Cook-Levin, 1971):

- SAT ist NP-vollständig.
- 3-SAT ist NP-vollständig.

SATISFIABILITY

- *Boolesche Variable*: Variable, die nur Werte aus {0,1} annimmt (False/True)

- **rekursive Definition**: *Boolesche Formel* über $X = \{x_1, \ldots, x_n\}$:

  - Jede Boolesche Variable x_i und jede Konstante in {0,1} ist eine Boolesche Formel
  - Sind F_1 und F_2 Boolesche Formeln, dann auch

    - die Konjunktion $F_1 \wedge F_2$,
    - die Disjunktion $F_1 \vee F_2$,
    - die Negation $\neg F_1$ (auch $\overline{F}_1$).

- Eine *Belegung* der Variablen X ist eine Abb. $X \to \{0,1\}$. Eine *erfüllende Belegung* ist eine Belegung, für die sich die Formel zu 1 evaluiert.

SATISFIABILITY

Disjunktive Normalform (DNF):

$$F = C_1 \vee C_2 \vee \ldots \vee C_m,$$

wobei jedes C_i von der Form $C_i = L_{i_1} \wedge L_{i_2} \wedge \ldots \wedge L_{i_\ell}$ ist,

$$L_j \in \{x_1 \ldots, x_n, \overline{x}_1, \ldots, \overline{x}_n\}.$$

Konjunktive Normalform (KNF):

$$F = C_1 \wedge C_2 \wedge \ldots \wedge C_m,$$

wobei jedes C_i von der Form $C_i = L_{i_1} \vee L_{i_2} \vee \ldots \vee L_{i_\ell}$ ist.

$$L_j \in \{x_1 \ldots, x_n, \overline{x}_1, \ldots, \overline{x}_n\}.$$

Klauseln

Literale

$$(x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee x_2 \vee \overline{x}_3) \wedge (\overline{x}_1) \wedge (\overline{x}_2)$$

z.B.:

SATISFIABILITY

- BOOLEANFORMULA:

  Gegeben: Variablen $x_1,\ldots,x_n$ und Boolesche Formel F über $x_1,\ldots,x_n$
  Frage: Gibt es eine erfüllende Belegung?

  SATISFIABILITY (SAT)

  Gegeben: Variablen $x_1,\ldots,x_n$ und Boolesche Formel F in KNF
  Frage: Gibt es eine erfüllende Belegung?

  K-SATISFIABILITY (k-SAT)

  Gegeben: Variablen $x_1,\ldots,x_n$ und Boolesche Formel F in KNF, wobei jede Klausel höchstens k Literale enthält.
  Frage: Gibt es eine erfüllende Belegung?

$$k\text{-SAT} \leq_p \text{SAT} \leq_p \text{BOOLEANFORMULA}$$

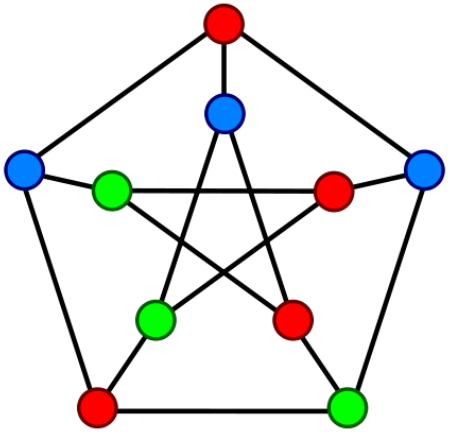
SAT ist NP-vollständig

Theorem (Cook-Levin, 1971):

- SAT ist NP-vollständig.
- 3-SAT ist NP-vollständig.

3-COL

Definition: Ein Graph G=(V,E) ist *3-färbbar*, wenn es eine Abbildung $f : V \to \{0, 1, 2\}$ gibt, sodass je zwei benachbarte Knoten auf verschiedene Werte abgebildet werden.



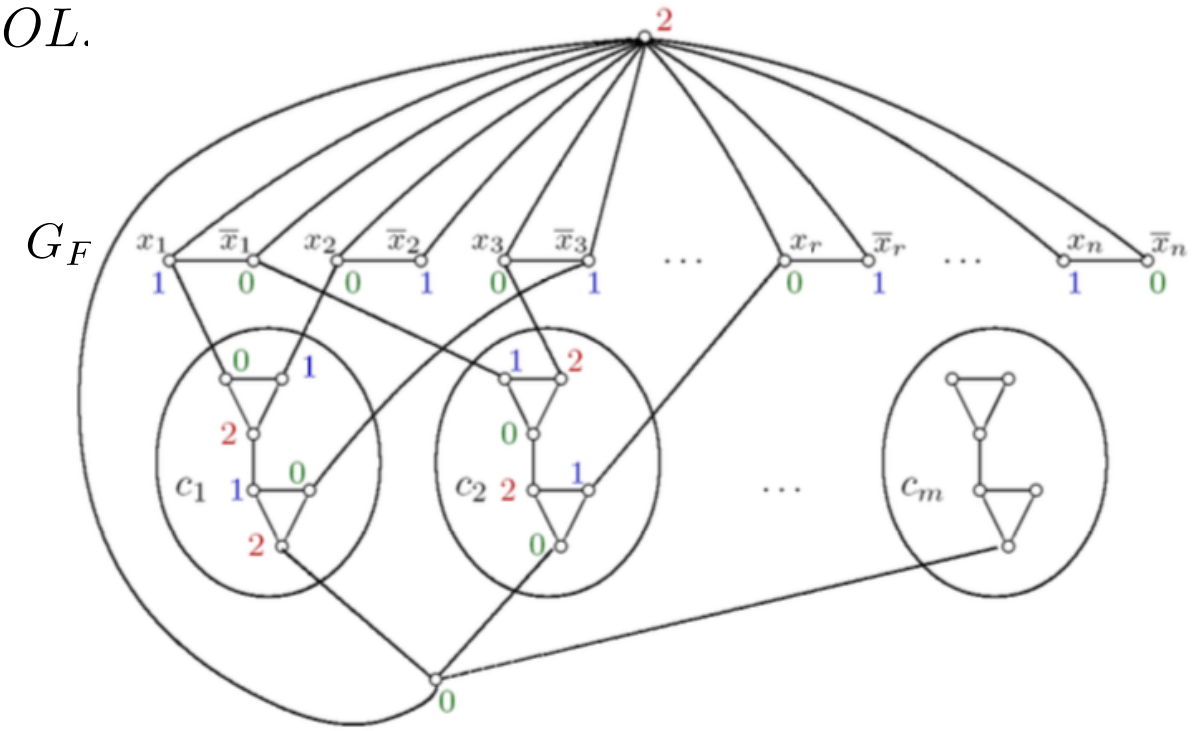
3-COL: $L = \{\text{Graph } G \mid \ G \text{ ist 3-färbbar}\}$

Satz: $3\text{-}SAT \leq_p 3\text{-}COL.$

Korollar: 3-COL ist NP-vollständig.

3-COL

Satz: $3\text{-}SAT \leq_p 3\text{-}COL.$



$$F = (x_1 \vee x_2 \vee \overline{x}_3) \wedge (\overline{x}_1 \vee x_3 \vee x_r) \wedge \ldots$$

Behauptung: Der so konstruierte Graph ist genau dann drei-färbbar, wenn die Formel F erfüllbar ist.

SAT ist NP-vollständig

Theorem (Cook-Levin):

- SAT ist NP-vollständig.
- 3-SAT ist NP-vollständig.

Lemma:

Sei A ein Algorithmus mit r Zeilen und seien p(n), q(n), s(n) Polynome, sodass der Algorithmus für jeden Input der Länge n

- nach höchstens s(n) Schritten mit 0 oder 1 terminiert;
- nur die Speicherzellen $M_1, \ldots, M_{p(n)}$ benutzt;
- in jede Speicherzelle höchstens q(n) Bits schreibt;

Dann gibt es für jedes n eine (3-)SAT-Formel F der Grösse poly(n), die Variablen $y_1, \ldots, y_n, z_1, \ldots, z_m$ enthält, sodass:
Für jedes $I = I_1 \ldots I_n \in \{0,1\}^n$ sei F_I die Formel, die wir durch Einsetzen von $y_1 := I_1, \ldots, y_n := I_n$ aus F erhalten. Dann ist

$$A \text{ gibt 1 aus für Input } I \iff F_I \text{ erfüllbar.}$$

Ausserdem ist F in polynomieller Zeit berechenbar.

NP-vollständig

Definition:

Ein Entscheidungsproblem *g* heisst *NP-schwer* genau dann, wenn $f \leq_p g$ für alle Probleme f in NP gilt. Ist zusätzlich g in NP, so heisst g *NP-vollständig*.

- Wir werden sehen: Es gibt (jede Menge) NP-vollständige Probleme.
- Wenn Sie für ein(!) solches Problem einen polynomiellen Algorithmus finden, haben sie P=NP gezeigt!!

Lemma:

Ist $f \leq_p g$, und ist *f* NP-schwer, so ist auch *g* NP-schwer.

Probleme in NP

Probleme in NP:

- alle Probleme in P
- GRAPH-ISOMORHPHISMUS

  $L = \{(G, H) \mid G \text{ und } H \text{ sind isomorphe Graphen}\}$
- 3-COL

  $L = \{G \mid \ G \text{ ist mit 3 Farben färbbar}\}$
- CLIQUE

  $L = \{(G, m) \mid \text{Es gibt } m \text{ paarweise adjazente Knoten in } G\}$
- HAMILTONKREIS

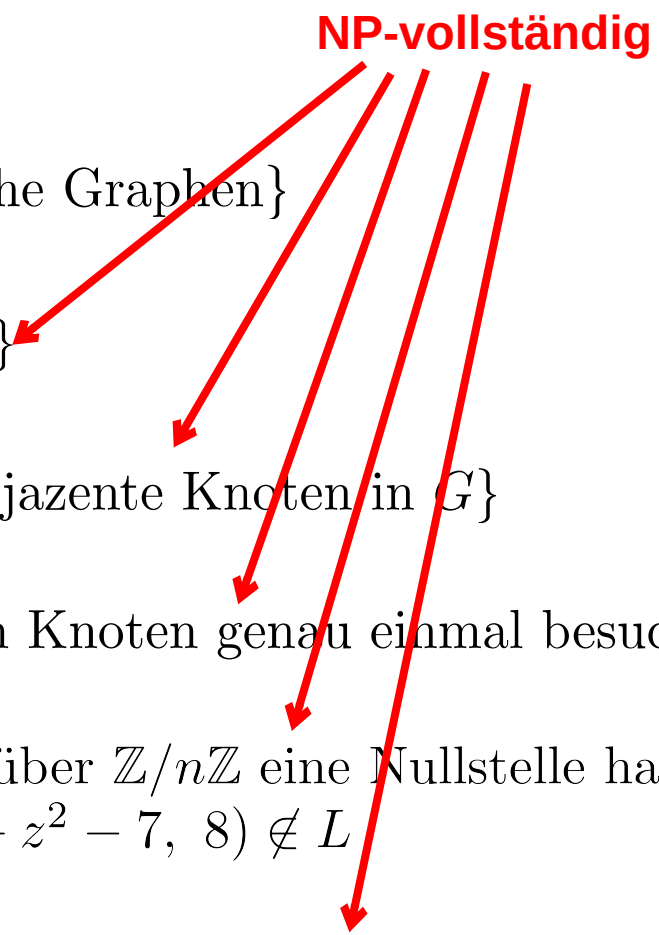
  $L = \{G \mid G \text{ hat einen Kreis, der jeden Knoten genau einmal besucht}\}$
- NULLSTELLE-MOD-N

  $L = \{(F, n) \mid F \text{ ist ein Polynom, das über } \mathbb{Z}/n\mathbb{Z} \text{ eine Nullstelle hat}\}$

  z.B.: $(x^2 + 1,\ 41) \in L, \qquad (x^2 + y^2 + z^2 - 7,\ 8) \notin L$
- SAT (SATISFIABILITY)

  $L = \{(F, n) \mid F \text{ ist erfüllbare aussagenlogische Formel in K-Normalform}\}$

  z.B.: $(x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee x_2 \vee \overline{x}_3) \wedge (\overline{x}_1) \wedge (\overline{x}_2) \notin L$



NP-vollständig

Theorem (Cook-Levin, 1971):

- SAT ist NP-vollständig

Theorem (Karp 1972):

Die folgenden 21 Probleme sind NP-vollständig:

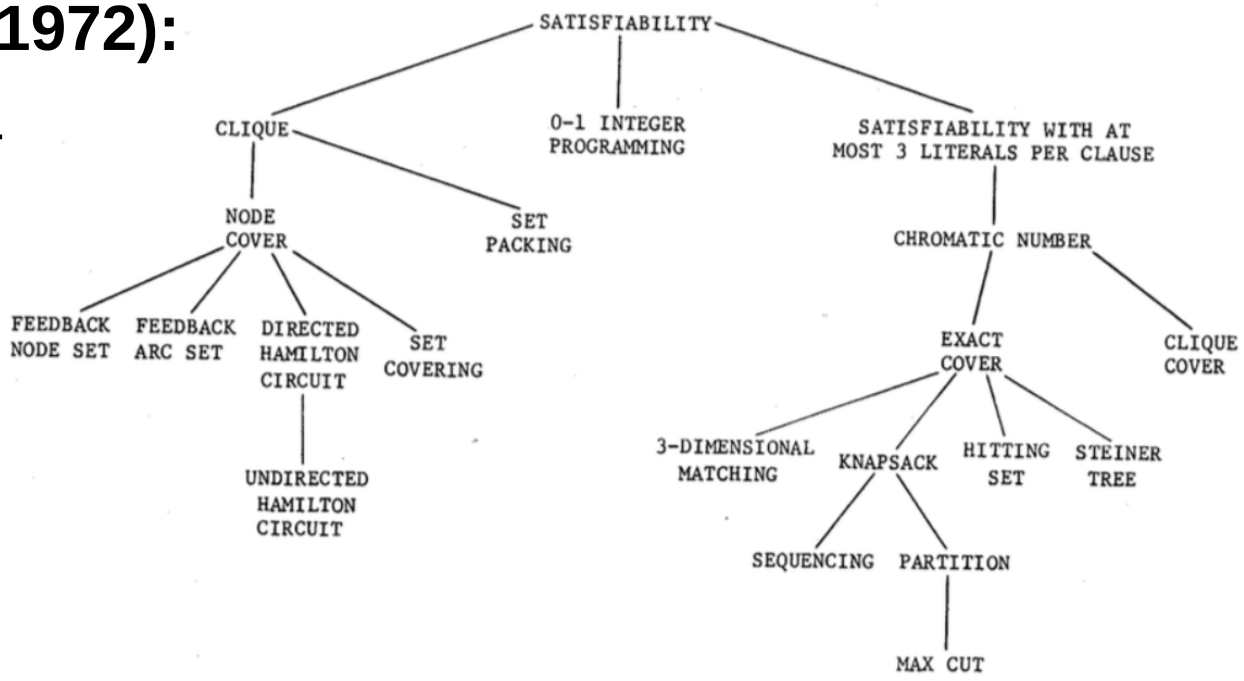


FIGURE 1 - Complete Problems

96

RICHARD M. KARP

NP-vollständig

Classic Nintendo Games are (Computationally) Hard

Greg Aloupis* Erik D. Demaine† Alan Guo†‡ Giovanni Viglietta§

February 10, 2015

Abstract

We prove NP-hardness results for five of Nintendo's largest video game franchises: Mario, Donkey Kong, Legend of Zelda, Metroid, and Pokémon. Our results apply to generalized versions of Super Mario Bros. 1–3, The Lost Levels, and Super Mario World; Donkey Kong Country 1–3; all Legend of Zelda games; all Metroid games; and all Pokémon role-playing games. In addition, we prove PSPACE-completeness of the Donkey Kong Country games and several Legend of Zelda games.