Richard Kaye: "Minesweeper is NP-complete"

Vortrag von Marcus Daum

08.11.2007

Überblick

etwas Komplexitätstheorie

- P und NP
- Erfüllbarkeitsproblem der Aussagenlogik (SAT)

Das Spiel Minesweeper

boolesche Formeln in Minesweeper

P und NP

$A \in P :\Longleftrightarrow A$ wird von det. TM in Polynomialzeit entschieden

P und NP

$A \in P :\Longleftrightarrow A$ wird von det. TM in Polynomialzeit entschieden

$A \in NP :\Longleftrightarrow A$ wird von nichtdet. TM in Polynomialzeit entschieden

P und NP

$A \subseteq \Sigma^*$ und $B \subseteq \Gamma^*$

A heisst **auf** B **in Polyomialzeit reduzierbar** ($A \leq_p B$) $:\Longleftrightarrow$
$\exists$ *berechenbares* $f : \Sigma^* \rightarrow \Gamma^*$ mit:

$$\forall w \in \Sigma^* : \quad w \in A \Leftrightarrow f(w) \in B$$

f muss von det. polynomiell zeitbeschränkter Turingmaschine berechnet werden können.

P und NP

$A \subseteq \Sigma^*$ und $B \subseteq \Gamma^*$

A heisst **auf** B **in Polyomialzeit reduzierbar** $(A \leq_p B) :\Longleftrightarrow$
$\exists$ *berechenbares* $f : \Sigma^* \rightarrow \Gamma^*$ mit:

$$\forall w \in \Sigma^* : \quad w \in A \Leftrightarrow f(w) \in B$$

f muss von det. polynomiell zeitbeschränkter Turingmaschine berechnet werden können.

B heisst **NP-hart** $:\Longleftrightarrow \ \forall A \in NP : \ A \leq_p B$

B heisst **NP-vollständig** $:\Longleftrightarrow \ B \in NP$ und B ist NP-hart.

SAT

SAT (von engl. "satisfiable") ist berühmtes Entscheidungsproblem der Informatik.

Sei φ boolesche Formel, die aus $\neg, \wedge$ und Variablen aufgebaut ist.

$$SAT = \{\varphi \mid \exists \text{Belegung } f : \; f(\varphi) = 1\}$$

SAT

1971 zeigte Stephen A. Cook, dass *SAT* NP-vollständig ist.
(Satz von Cook)

Leonid Levin zeigte unabhängig davon dasselbe 1973.

SAT

1971 zeigte Stephen A. Cook, dass *SAT* NP-vollständig ist.
(Satz von Cook)

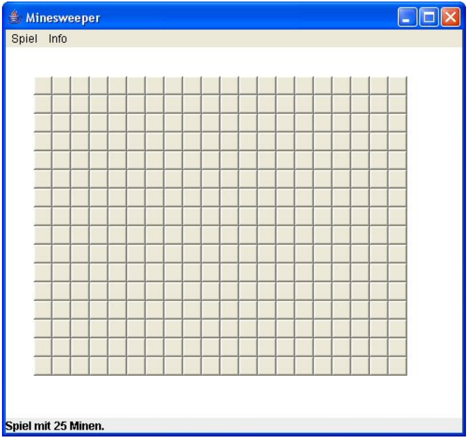
Leonid Levin zeigte unabhängig davon dasselbe 1973.

Wir zeigen eine Reduktion von *SAT* auf das Minesweeper-Problem *MINES*.

Minesweeper

Ziel:
Lokalisation aller im Spiel befindlichen Minen

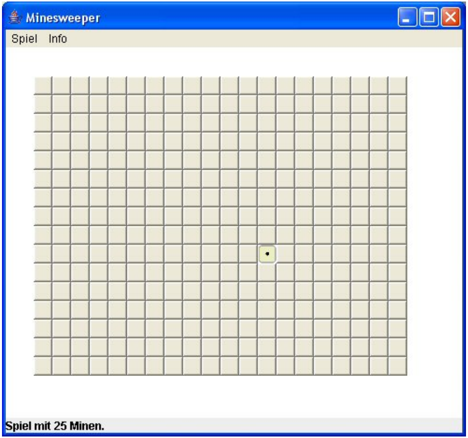
Anfangskonfiguration:
unbekannte Verteilung von Minen auf einem $m \cdot n$-Spielfeld



Minesweeper

einzelner Zug:
Betreten eines Feldes

Reaktion:
aufdecken und Bekanntgabe der Anzahl der Minen in Nachbarschaft



MINES

gegeben $c : [m] \times [n] \dashrightarrow \{0, \ldots, 8\}$

existiert $C : [m] \times [n] \longrightarrow \{0, \ldots, 8\} \cup \{*\}$ mit

- $c \subseteq C$
- (Konsistenz) $\forall (i,j) \in [m] \times [n]$ mit $C(i,j) \neq *$:

$$C(i,j) = \sum_{(k,l) \in N(i,j)} \delta((k,l))$$

MINES

gegeben $c: [m] \times [n] \dashrightarrow \{0, \ldots, 8\}$

existiert $C: [m] \times [n] \longrightarrow \{0, \ldots, 8\} \cup \{*\}$ mit

- $c \subseteq C$
- (Konsistenz) $\forall (i,j) \in [m] \times [n]$ mit $C(i,j) \neq *$:

$$C(i,j) = \sum_{(k,l) \in N(i,j)} \delta((k,l))$$

$$N(i,j) = \{(k,l) \in [m] \times [n] \mid |k-i| \leq 1, |l-j| \leq 1\} \setminus \{(i,j)\}$$

$$\delta((k,l)) = \begin{cases} 1 & \text{wenn } C((k,l)) = * \\ 0 & \text{sonst} \end{cases}$$

MINES

gegeben $c: \ [m] \times [n] \dashrightarrow \{0, \ldots, 8\}$

existiert $C: \ [m] \times [n] \longrightarrow \{0, \ldots, 8\} \cup \{*\}$ mit

- $c \subseteq C$
- (Konsistenz) $\forall (i,j) \in [m] \times [n]$ mit $C(i,j) \neq *$:

$$C(i,j) = \sum_{(k,l) \in N(i,j)} \delta((k,l))$$

$$N(i,j) = \{(k,l) \in [m] \times [n] \ \mid \ |k-i| \leq 1, |l-j| \leq 1\} \setminus \{(i,j)\}$$

$$\delta((k,l)) = \begin{cases} 1 & \text{wenn } C((k,l)) = * \\ 0 & \text{sonst} \end{cases}$$

$$\mathit{MINES} = \{c \mid \exists C \text{ mit } c \subseteq C \text{ und Konsistenz } \}$$

$MINES \in NP$

Algorithmus, der *MINES* entscheidet:

gegeben sei partielle Funktion c

1. rate $C: [m] \times [n] \longrightarrow \{0, \ldots, 8\} \cup \{*\}$ mit $c \subseteq C$
2. überprüfe
   $\forall (i,j) \in [m] \times [n]$ mit $C(i,j) \neq * :\ C(i,j) = \sum_{n \in N(i,j)} \delta(n)$

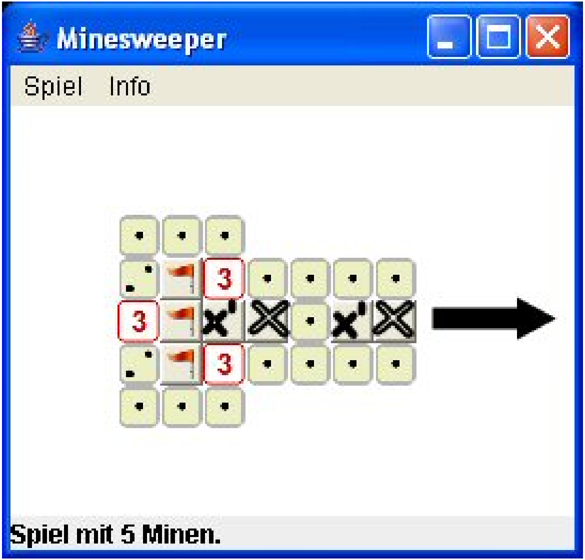
MINES ist NP-hart

Jetzt wollen wir zeigen, dass *MINES* NP-hart ist.

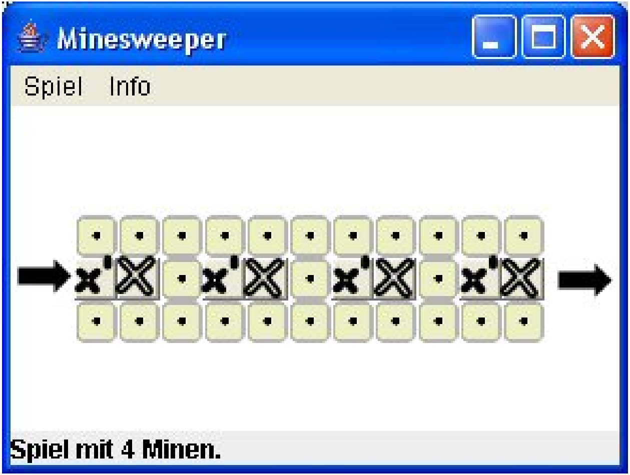
Dazu kodieren wir boolesche Formeln als Minesweeperkonfigurationen.

Die Formeln stellen wir uns dazu als Schaltkreise vor.

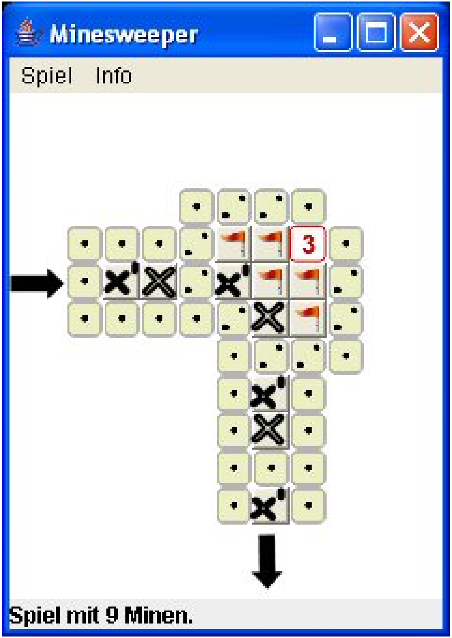
Terminator



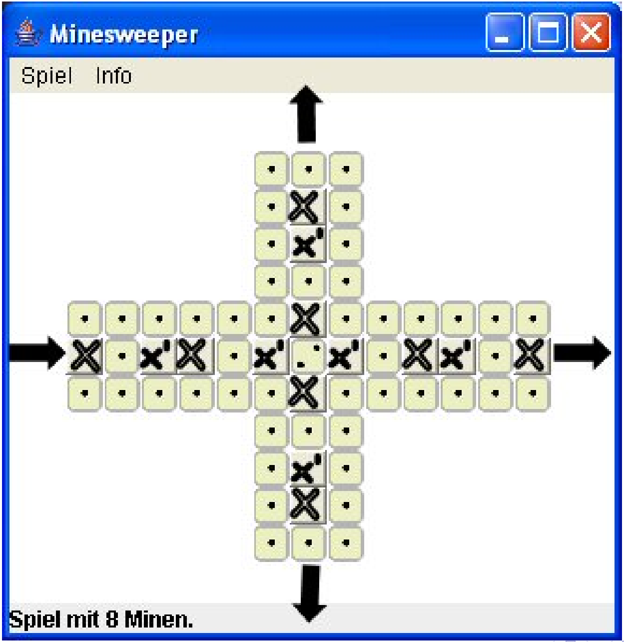
Kabel



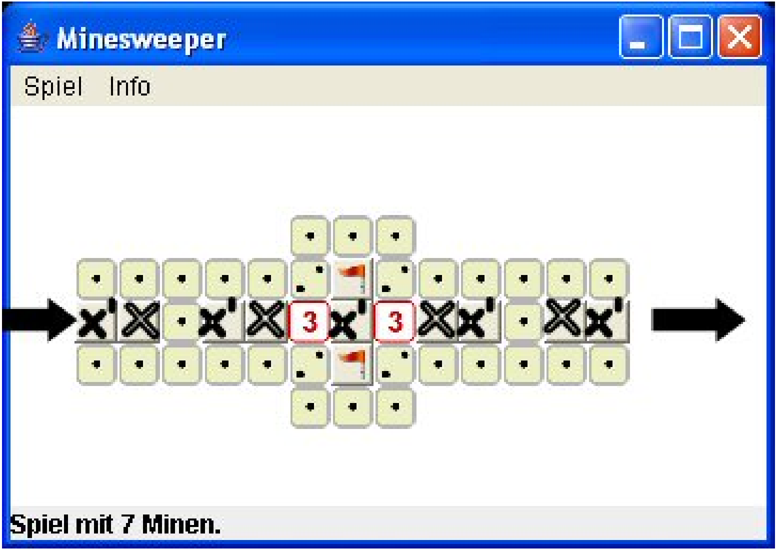
Ecke



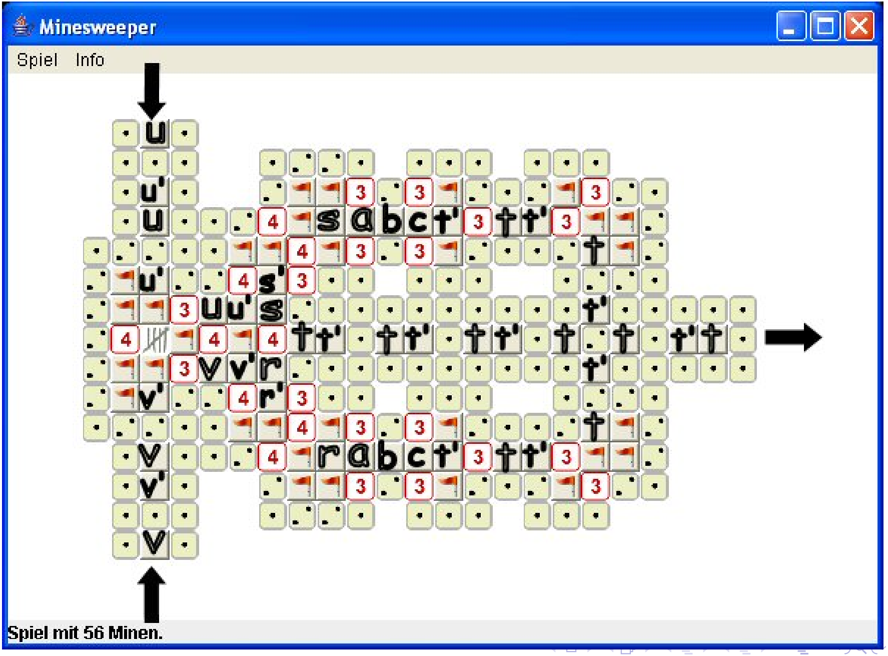
Splitter



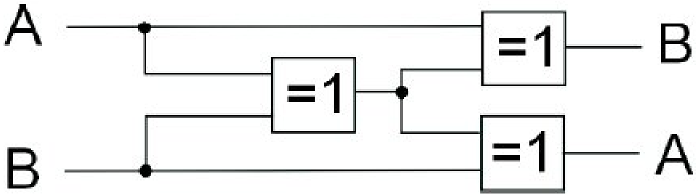
NOT



AND



Crossover



$SAT \leq_p MINES$

Sei φ boolesche Formel.

1. stelle φ als Minesweeperkonfiguration c dar
2. setze Ausgabefeld von φ auf TRUE
3. entscheide *MINES* für c

$SAT \leq_p MINES$

Sei φ boolesche Formel.

1. stelle φ als Minesweeperkonfiguration c dar
2. setze Ausgabefeld von φ auf TRUE
3. entscheide *MINES* für c

$$\implies \quad c \in MINES \iff \varphi \in SAT$$

SAT $\leq_p$ *MINES*

Sei φ boolesche Formel.

1. stelle φ als Minesweeperkonfiguration c dar
2. setze Ausgabefeld von φ auf TRUE
3. entscheide *MINES* für c

$$\Longrightarrow \quad c \in MINES \Longleftrightarrow \varphi \in SAT$$

Also: **Minesweeper ist NP-vollständig**

qed

Infinite Versions of Minesweeper are Turing-Complete

Richard Kaye zeigte im Mai 2007, dass Minesweeper auf einem $\mathbb{N} \times \mathbb{N}$-Spielfeld Turing-vollständig ist.

Es gibt Regeln R mit:

1. für alle TM P und Wörter w existiert endliche Minesweeperkonfiguration c mit:
   $\exists C$ mit $c \subseteq C \iff P$ hält bei Eingabe w nicht
2. es gibt TM P, so dass für alle endlichen c gilt:
   $\exists C$ mit $c \subseteq C \iff P$ hält bei Eingabe von c nicht