Grundlagen der Theoretischen Informatik

Till Mossakowski

Fakultät für Informatik
Otto-von-Guericke Universität
Magdeburg

Wintersemester 2014/15

Satz: $\textsc{domino} \preceq_{\mathrm{P}} \textsc{sat}$.

Beweis:

Zu gegebenen $\mathcal{D} = (D, f_0, H, V, \ell)$ mit $D = \{d_1, \ldots, d_k\}$ konstruieren wir eine Boolesche Formel ϕ. Für alle $-\ell \leq m \leq \ell$ und $0 \leq n \leq \ell + 1$ und $d \in D$ gibt es eine Variable $X_{m,n,d}$. Diese Variable wird mit 1 belegt werden genau dann wenn $f(m,n) = d$.

Um sicherzustellen, dass jede Zelle (m,n) überkachelt wird, fügen wir für alle $-\ell \leq m \leq \ell$ und $0 \leq n \leq \ell + 1$ die Klausel

$$(X_{m,n,d_1} \vee X_{m,n,d_2} \vee \ldots \vee X_{m,n,d_k})$$

zu ϕ hinzu.

Um sicherzustellen, dass jede Zelle (m, n) höchstens einmal überkachelt wird, fügen wir für alle $-\ell \leq m \leq \ell$ und $0 \leq n \leq \ell + 1$ und $d \neq d'$ die Klausel

$$(\overline{X_{m,n,d}} \vee \overline{X_{m,n,d'}})$$

zu ϕ hinzu.

Um die unterste Kachelreihe festzulegen, fügen wir für $-\ell \leq m \leq \ell$ die Klausel

$$(X_{m,0,f_0(m)})$$

zu ϕ hinzu.

Für alle $-\ell \leq m < \ell$ und $0 \leq n \leq \ell + 1$ und $(d, d') \in (D \times D) - H$ fügen wir die Klausel

$$(\overline{X_{m,n,d}} \vee \overline{X_{m+1,n,d'}})$$

zu ϕ hinzu. Dies garantiert, dass H beachtet wird.

Für alle $-\ell \leq m \leq \ell$ und $0 \leq n \leq \ell$ und $(d, d') \in (D \times D) - V$ fügen wir die Klausel

$$(\overline{X_{m,n,d}} \vee \overline{X_{m,n+1,d'}})$$

zu ϕ hinzu. Dies garantiert, dass V beachtet wird.

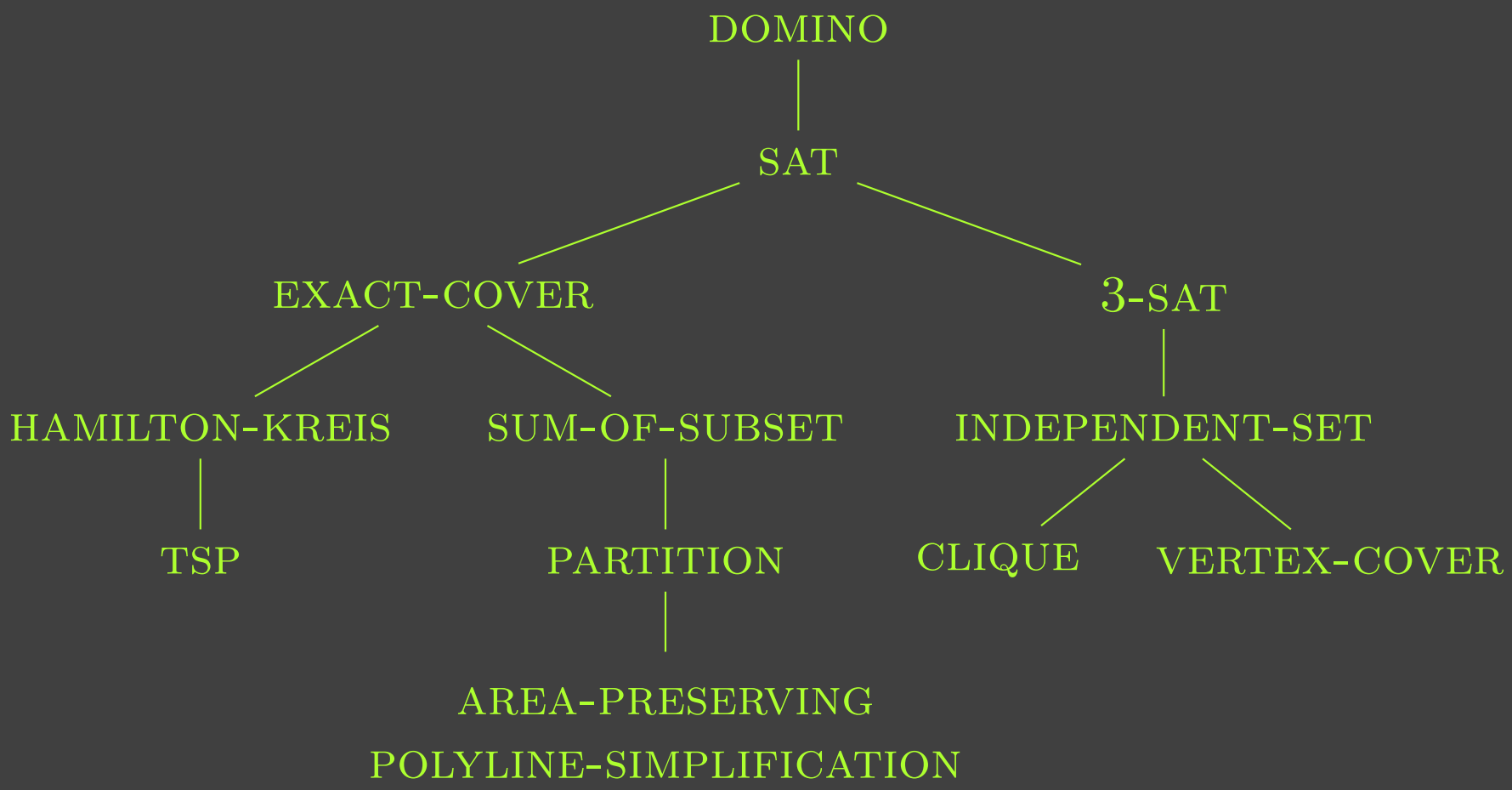
Eine erfüllende Belegung der Variablen entspricht nach unserer Konstruktion einer $\mathcal{D}$-Kachelung und umgekehrt.

Ferner ist die zugehörige Reduktionsfunktion, die ein eingeschränktes Kachelproblem wie eben beschrieben auf eine Boolesche Formel in konjunktiver Normalform abbildet, in Polynomialzeit berechenbar. ■

Da $\text{SAT} \in \text{NP}$, folgt

Satz: SAT *ist* NP*-vollständig.*

Weitere NP-vollständige Probleme



$3\text{-SAT} = \{\langle \phi \rangle \mid \phi$ ist eine Boolesche Formel in konjunktiver Normalform, in der alle Klauseln aus genau 3 Literalen bestehen, und ϕ ist erfüllbar$\}$

Satz: $\text{SAT} \preceq_{\text{P}} 3\text{-SAT}$.

Beweisskizze:

Sei ϕ eine Formel in konjunktiver Normalform. ϕ wird in polynomieller Zeit zu einer Formel ϕ' transformiert, die genau dann erfüllbar ist, wenn ϕ erfüllbar ist, und in der jede Klausel genau drei Literale enthält.

Zu kurze Klauseln werden durch schon vorhandene Literale aufgefüllt.

Jede Klausel $c = (\lambda_1 \vee \lambda_2 \vee \cdots \vee \lambda_k)$ mit $k > 3$ wird ersetzt durch die Klauseln

$$
\begin{aligned}
&(\lambda_1 \vee \lambda_2 \vee y_1),\\
&(\overline{y_1} \vee \lambda_3 \vee y_2),\\
&(\overline{y_2} \vee \lambda_4 \vee y_3),\\
&\qquad\vdots\\
&(\overline{y_{k-4}} \vee \lambda_{k-2} \vee y_{k-3}) \text{ und}\\
&(\overline{y_{k-3}} \vee \lambda_{k-1} \vee \lambda_k)
\end{aligned}
$$

Beispiel:

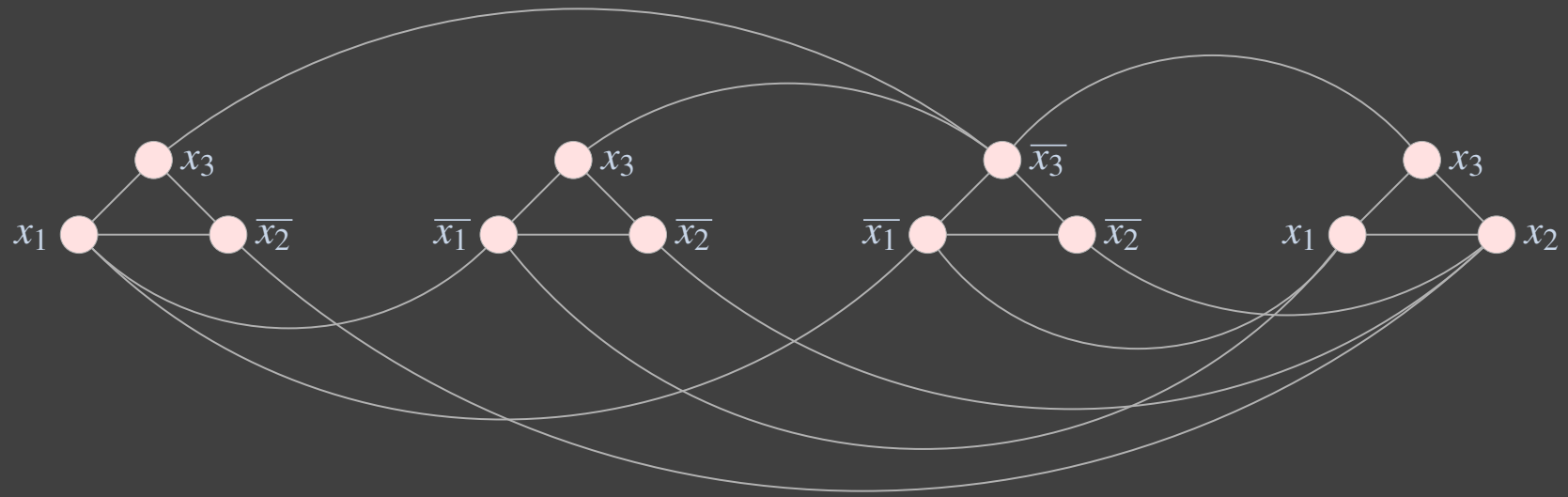
$$
\begin{aligned}
&(x_1 \vee x_3) \\
\wedge\ &(\overline{x_2} \vee x_3 \vee x_4 \vee \overline{x_5}) \\
\wedge\ &(\overline{x_6}) \\
\wedge\ &(\overline{x_1} \vee \overline{x_4} \vee x_5) \\
\wedge\ &(x_2 \vee \overline{x_3} \vee x_4 \vee \overline{x_5} \vee x_6)
\end{aligned}
$$

$$
\begin{aligned}
&(x_1 \vee x_3 \vee x_3) \\
\wedge\ &(\overline{x_2} \vee x_3 \vee y_{21}) \\
\wedge\ &(\overline{y_{21}} \vee x_4 \vee \overline{x_5}) \\
\wedge\ &(\overline{x_6} \vee \overline{x_6} \vee \overline{x_6}) \\
\wedge\ &(\overline{x_1} \vee \overline{x_4} \vee x_5) \\
\wedge\ &(x_2 \vee \overline{x_3} \vee y_{51}) \\
\wedge\ &(\overline{y_{51}} \vee x_4 \vee y_{52}) \\
\wedge\ &(\overline{y_{52}} \vee \overline{x_5} \vee x_6)
\end{aligned}
$$

Satz: 3-SAT $\preceq_{\mathrm{P}}$ INDEPENDENT-SET.

Illustration der Beweisidee an einem Beispiel:

$$(x_1 \vee \overline{x_2} \vee x_3) \wedge (\overline{x_1} \vee \overline{x_2} \vee x_3) \wedge (\overline{x_1} \vee \overline{x_2} \vee \overline{x_3}) \wedge (x_1 \vee x_2 \vee x_3)$$



Satz: $\textsc{sat} \preceq_{\mathrm{P}} \textsc{exact-cover}$.

Beweis: Zu jeder Booleschen Formel ϕ lässt sich eine Probleminstanz $(U, \mathcal{F})$ von $\textsc{exact-cover}$ angeben, die genau dann eine exakte Überdeckung besitzt, wenn ϕ erfüllbar ist.

Sei ϕ eine Boolesche Formel mit Klauseln $c_1, \ldots, c_\ell$ in den Variablen $X_1, \ldots, X_n$. Mit λ_{jk}, $k = 1, 2, \ldots, m_j$, bezeichnen wir das k-te Literal in der Klausel c_j, die aus m_j Literalen besteht.

$$(\lambda_{11} \vee \lambda_{12} \vee \cdots \vee \lambda_{1m_1}) \wedge \cdots \wedge (\cdots \vee \lambda_{jk} \vee \cdots) \wedge \cdots \wedge (\lambda_{\ell 1} \vee \cdots \vee \lambda_{\ell m_\ell})$$

Die zu ϕ gehörige Probleminstanz $(U, \mathcal{F})$ ist folgende:

$$
\begin{aligned}
U &= \{X_i \mid 1 \le i \le n\} \\
&\cup \{c_j \mid 1 \le j \le \ell\} \\
&\cup \{p_{jk} \mid 1 \le j \le \ell, 1 \le k \le m_j\}
\end{aligned}
$$

$$
\begin{aligned}
\mathcal{F} &= \{\, \{p_{jk}\} \mid 1 \le j \le \ell, 1 \le k \le m_j \,\} \\
&\cup \{\, \{X_i\} \cup \{p_{jk} \mid \lambda_{jk} = \overline{X_i}\} \mid 1 \le i \le n \,\} \\
&\cup \{\, \{X_i\} \cup \{p_{jk} \mid \lambda_{jk} = X_i\} \mid 1 \le i \le n \,\} \\
&\cup \{\, \{c_j, p_{jk}\} \mid 1 \le j \le \ell, 1 \le k \le m_j \,\}
\end{aligned}
$$

Um X_i in U zu überdecken, muss genau eine der beiden folgenden Mengen in $\mathcal{C}$ sein:

$$\begin{aligned} T_{i,\top} &= \{X_i\} \cup \{p_{jk} \mid \lambda_{jk} = \overline{X_i}\} \\ T_{i,\bot} &= \{X_i\} \cup \{p_{jk} \mid \lambda_{jk} = X_i\} \end{aligned}$$

Um c_j in U zu überdecken, muss für genau ein $k \in \{1, \ldots, m_j\}$ die Menge $\{c_j, p_{jk}\}$ in $\mathcal{C}$ sein.

Durch das Vorhandensein obiger Mengen in $\mathcal{C}$ sind bereits einige p_{jk} überdeckt. Anderweitig nicht überdeckte p_{jk} können durch $\{p_{jk}\}$ in $\mathcal{C}$ überdeckt werden.

Eine ϕ erfüllende Belegung Ψ der Variablen liefert wie folgt eine exakte Überdeckung $\mathcal{C}$:

$$\begin{aligned} \Psi(X_i) = 1 &\Rightarrow T_{i,\top} \in \mathcal{C} \\ \Psi(X_i) = 0 &\Rightarrow T_{i,\bot} \in \mathcal{C} \end{aligned}$$

Wird $T_{i,\top}$ zu $\mathcal{C}$ hinzugefügt, so werden mit X_i auch alle λ_{jk}, die zu $\overline{X_i}$ korrespondieren, überdeckt. Diese können also nicht mehr bei der Überdeckung der c_j benutzt werden.

Wird $T_{i,\bot}$ zu $\mathcal{C}$ hinzugefügt, so werden mit X_i auch alle λ_{jk}, die zu X_i korrespondieren, überdeckt. Diese können also nicht mehr bei der Überdeckung der c_j benutzt werden.

Für jede Klausel c_j wird nun ein p_{jk} ausgewählt, dessen zugehöriges Literal erfüllt ist.

$\{p_{jm}\}$ wird zu $\mathcal{C}$ hinzugefügt wenn das zugehörige Literal erfüllt ist, aber nicht zusammen mit c_j ausgewählt wurde.

Umgekehrt liefert jede exakte Überdeckung $\mathcal{C}$ eine Belegung Ψ der Variablen:

$$\begin{aligned} T_{i,\top} \in \mathcal{C} &\Rightarrow \Psi(X_i) = 1 \\ T_{i,\bot} \in \mathcal{C} &\Rightarrow \Psi(X_i) = 0 \end{aligned}$$

Da jedes X_i überdeckt werden muss, wird jede Variable belegt.

Da jedes c_j überdeckt werden muss, gibt es genau ein $\{c_j, p_{jk}\} \in \mathcal{C}$:

$$\begin{aligned} \lambda_{jk} &= X_i \\ &\Rightarrow T_{i,\bot} \notin \mathcal{C} \\ &\Rightarrow T_{i,\top} \in \mathcal{C} \\ &\Rightarrow \Psi(X_i) = 1 \end{aligned} \qquad \begin{aligned} \lambda_{jk} &= \overline{X_i} \\ &\Rightarrow T_{i,\top} \notin \mathcal{C} \\ &\Rightarrow T_{i,\bot} \in \mathcal{C} \\ &\Rightarrow \Psi(X_i) = 0 \\ &\Rightarrow \Psi(\overline{X_i}) = 1 \end{aligned}$$

Also wird in jeder Klausel mindestens ein Literal erfüllt.

Eine entsprechende Reduktion τ kann in polynomieller Zeit berechnet werden.

Beispiel:

$$(x_1 \vee x_3) \wedge (\overline{x_2} \vee x_3 \vee x_4 \vee \overline{x_5}) \wedge (\overline{x_1} \vee \overline{x_4} \vee x_5) \wedge (x_2 \vee \overline{x_3} \vee x_5)$$

$$\begin{aligned}
U &= \{x_1, x_2, x_3, x_4, x_5\} \cup \{c_1, c_2, c_3, c_4\} \\
&\cup \{p_{11}, p_{12}, p_{21}, p_{22}, p_{23}, p_{24}, p_{31}, p_{32}, p_{33}, p_{41}, p_{42}, p_{43}\}
\end{aligned}$$

$$\begin{aligned}
\mathcal{F} &= \{\{p_{11}\}, \{p_{12}\}, \{p_{21}\}, \{p_{22}\}, \{p_{23}\}, \{p_{24}\}\} \\
&\cup \{\{p_{31}\}, \{p_{32}\}, \{p_{33}\}, \{p_{41}\}, \{p_{42}\}, \{p_{43}\}\} \\
&\cup \{\{x_1, p_{31}\}, \{x_2, p_{21}\}, \{x_3, p_{42}\}, \{x_4, p_{32}\}, \{x_5, p_{24}\}\} \\
&\cup \{\{x_1, p_{11}\}, \{x_2, p_{41}\}, \{x_3, p_{12}, p_{22}\}, \{x_4, p_{23}\}, \{x_5, p_{33}, p_{43}\}\} \\
&\cup \{\{c_1, p_{11}\}, \{c_1, p_{12}\}, \{c_2, p_{21}\}, \{c_2, p_{22}\}, \{c_2, p_{23}\}, \{c_2, p_{24}\}\} \\
&\cup \{\{c_3, p_{31}\}, \{c_3, p_{32}\}, \{c_3, p_{33}\}, \{c_4, p_{41}\}, \{c_4, p_{42}\}, \{c_4, p_{43}\}\}
\end{aligned}$$

Beispiel:

$$(x_1 \vee x_3) \wedge (\overline{x_2} \vee x_3 \vee x_4 \vee \overline{x_5}) \wedge (\overline{x_1} \vee \overline{x_4} \vee x_5) \wedge (x_2 \vee \overline{x_3} \vee x_5)$$

$$\begin{aligned}
U &= \{x_1, x_2, x_3, x_4, x_5\} \cup \{c_1, c_2, c_3, c_4\} \\
&\cup \{p_{11}, p_{12}, p_{21}, p_{22}, p_{23}, p_{24}, p_{31}, p_{32}, p_{33}, p_{41}, p_{42}, p_{43}\}
\end{aligned}$$

$$\begin{aligned}
\mathcal{F} &= \{\{p_{11}\}, \{p_{12}\}, \{p_{21}\}, \{p_{22}\}, \{p_{23}\}, \{p_{24}\}\} \\
&\cup \{\{p_{31}\}, \{p_{32}\}, \{p_{33}\}, \{p_{41}\}, \{p_{42}\}, \{p_{43}\}\} \\
&\cup \{\{x_1, p_{31}\}, \{x_2, p_{21}\}, \{x_3, p_{42}\}, \{x_4, p_{32}\}, \{x_5, p_{24}\}\} \\
&\cup \{\{x_1, p_{11}\}, \{x_2, p_{41}\}, \{x_3, p_{12}, p_{22}\}, \{x_4, p_{23}\}, \{x_5, p_{33}, p_{43}\}\} \\
&\cup \{\{c_1, p_{11}\}, \{c_1, p_{12}\}, \{c_2, p_{21}\}, \{c_2, p_{22}\}, \{c_2, p_{23}\}, \{c_2, p_{24}\}\} \\
&\cup \{\{c_3, p_{31}\}, \{c_3, p_{32}\}, \{c_3, p_{33}\}, \{c_4, p_{41}\}, \{c_4, p_{42}\}, \{c_4, p_{43}\}\}
\end{aligned}$$

Da 3-SAT, INDEPENDENT-SET, CLIQUE, VERTEX-COVER und EXACT-COVER alle in NP liegen, folgt aus dem bisher Gezeigten

Satz: 3-SAT *ist* NP-*vollständig.*

Satz: INDEPENDENT-SET *ist* NP-*vollständig.*

Satz: CLIQUE *ist* NP-*vollständig.*

Satz: VERTEX-COVER *ist* NP-*vollständig.*

Satz: EXACT-COVER *ist* NP-*vollständig.*

Satz: EXACT-COVER $\preceq_P$ SUM-OF-SUBSET.

Beweisskizze:

$(\{u_1, \ldots, u_n\}, \{S_1, \ldots, S_m\})$

$$a_i = \sum_{u_j \in S_i} m^{j-1} \qquad K = \sum_{j=1}^{n} m^{j-1}$$

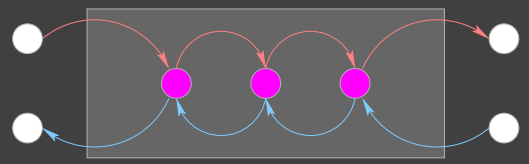
Beispiel: $U = \{u_1, u_2, u_3, u_4, u_5\}$

$$
\begin{array}{lcll}
S_1 & = & \{u_1, u_2\} & a_1 : \quad 00011_6 \\
S_2 & = & \{u_2, u_3, u_5\} & a_2 : \quad 10110_6 \\
S_3 & = & \{u_3, u_4\} & a_3 : \quad 01100_6 \\
S_4 & = & \{u_1\} & a_4 : \quad 00001_6 \\
S_5 & = & \{u_1, u_4\} & a_5 : \quad 01001_6 \\
S_6 & = & \{u_5\} & a_6 : \quad 10000_6 \\
 & & & \\
 & & & K : \quad 11111_6
\end{array}
$$

Satz: EXACT-COVER $\preceq_P$ HAMILTON-KREIS.

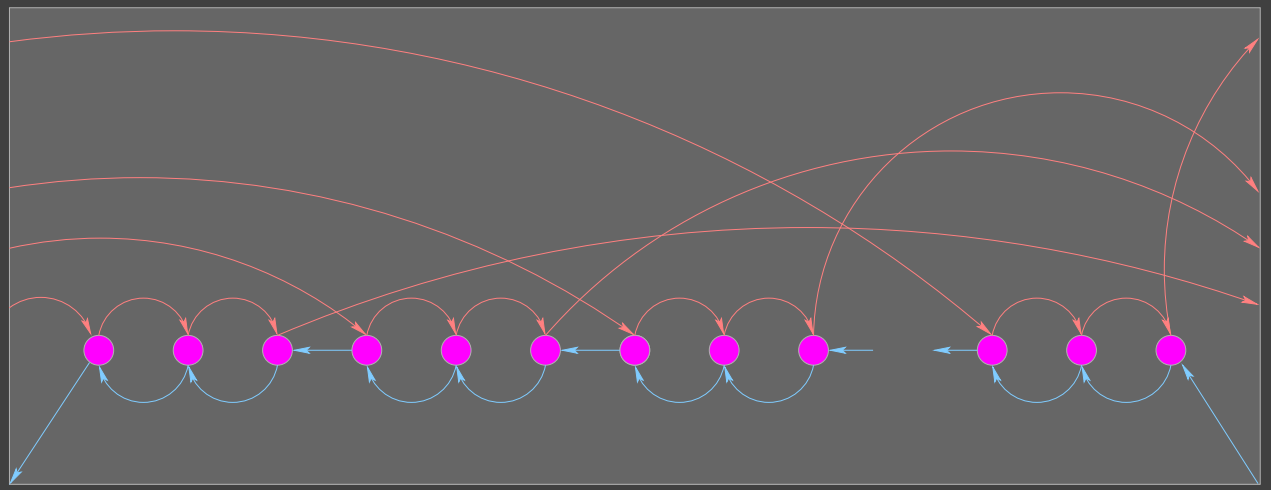
Beweisskizze:
Zu $U = \{u_1, \ldots, u_n\}$ und $\mathcal{F} = \{S_1, \ldots, S_m\}$ konstruieren wir einen gerichteten Graphen G, der genau dann einen Hamiltonkreis besitzt, wenn $(U, \mathcal{F})$ eine exakte Überdeckung besitzt.

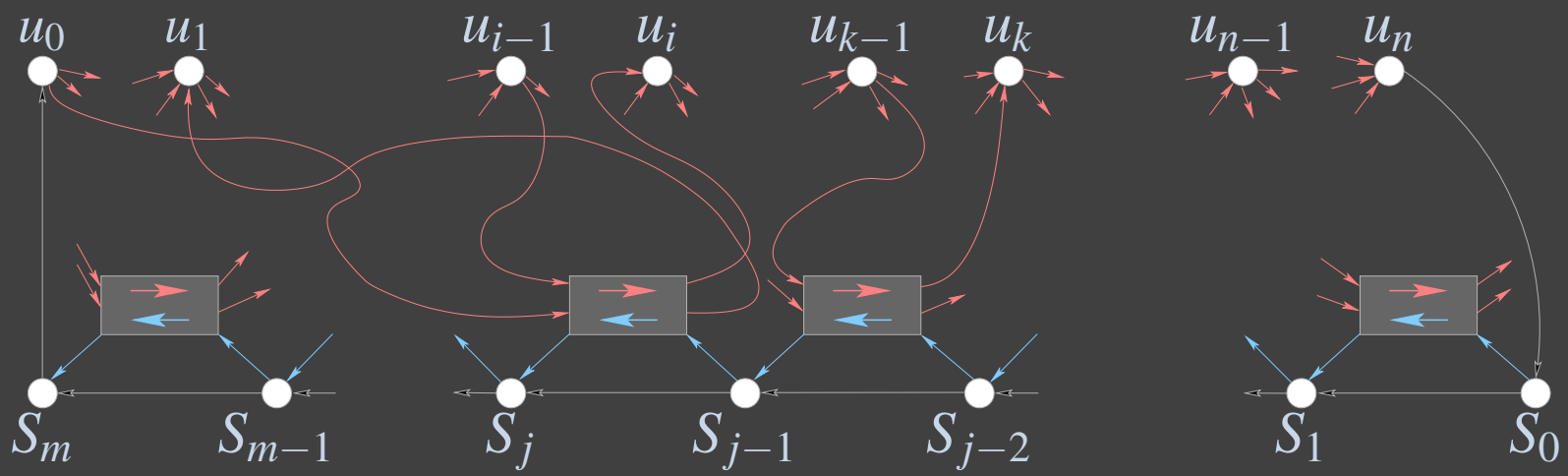
G enthält Konstrukte der folgenden Art als Teilgraphen:



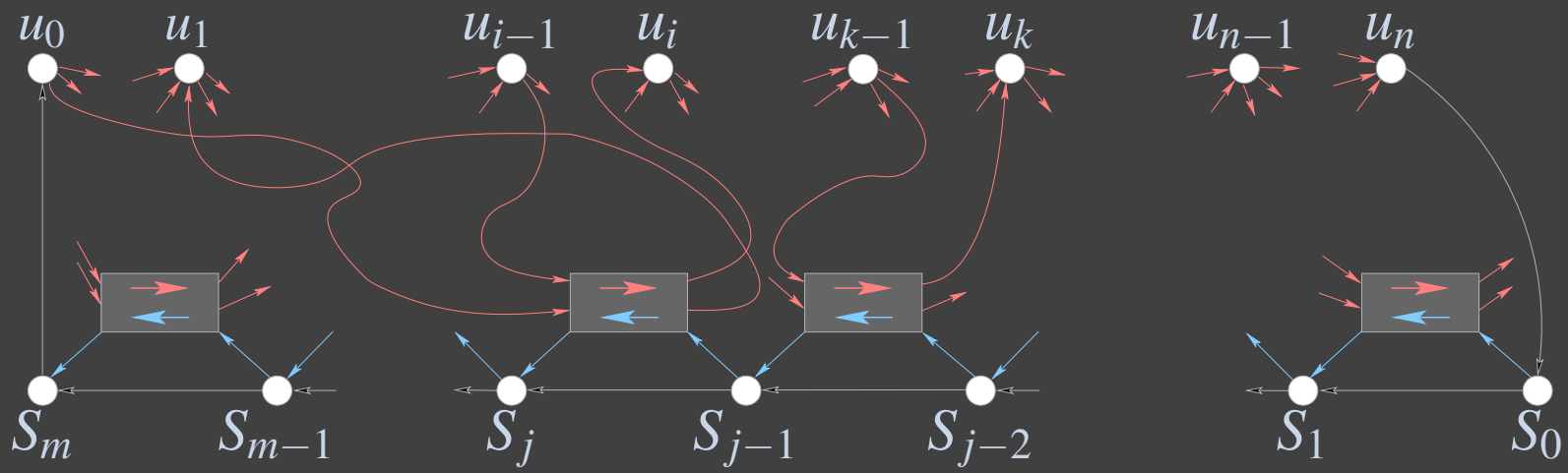
Jeder Hamiltonkreis muss entweder den kompletten roten Pfad benutzen oder den kompletten blauen.

Dieses Konstrukt kann so erweitert werden, dass jeder Hamiltonkreis entweder alle roten Pfade benutzt oder nur den blauen. Ein solches Konstrukt nennen wir *Gadget*.

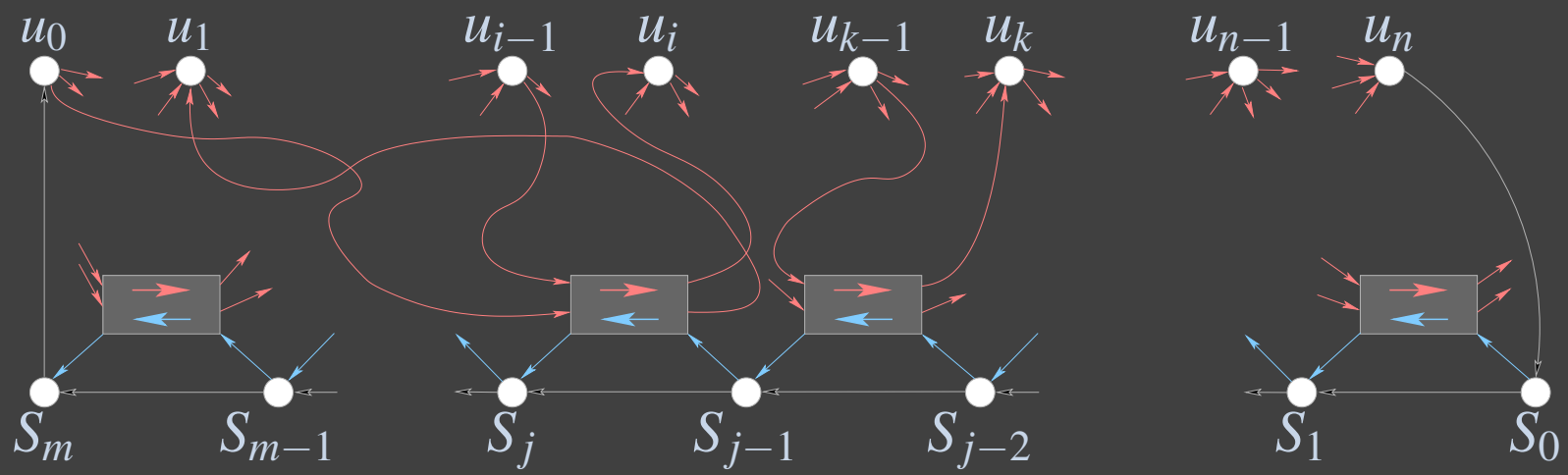




In G gibt es zu jedem $u_i \in U$ einen Knoten und zu jedem $S_j \in \mathcal{F}$ einen Knoten. Ferner gibt es zwei weitere Knoten, die wir mit u_0 und S_0 bezeichnen. Es gibt eine Kante von u_n nach S_0 und eine von S_m nach u_0. Von S_{j-1} nach S_j gibt es eine gerichtete Kante und einen blauen Pfad durch ein Gadget.



Von u_{i-1} nach u_i gibt es soviele rote Pfade wie es Teilmengen in $\mathcal{F}$ gibt, die u_i enthalten. Genau dann wenn u_i zu S_j gehört, führt ein roter Pfad durch das Gadget, durch das der blaue Pfad von S_{j-1} nach S_j führt. Die Anzahl der roten Pfade durch das Gadget zwischen S_{j-1} und S_j ist also $|S_j|$.



Der blaue Pfad durch das Gadget zwischen S_{j-1} und S_j wird benutzt genau dann wenn $S_j \notin \mathcal{C}$. Wird der blaue Pfad nach S_j nicht benutzt, dann muss für alle $u_i \in S_j$ der in u_i endende rote Pfad benutzt werden. u_i wird über genau einen roten Pfad erreicht, gehört also zu genau einer der Mengen in $\mathcal{C}$.

Zu gegebenem $(U, \mathcal{F})$ kann der Graph G in Polynomialzeit konstruiert werden. ■

Da EXACT-COVER NP-vollständig ist und ferner HAMILTON-KREIS $\in$ NP, folgt

Satz: HAMILTON-KREIS *ist* NP-*vollständig.*

und mit HAMILTON-KREIS $\preceq_{\mathrm{P}}$ TSP und TSP $\in$ NP schließlich

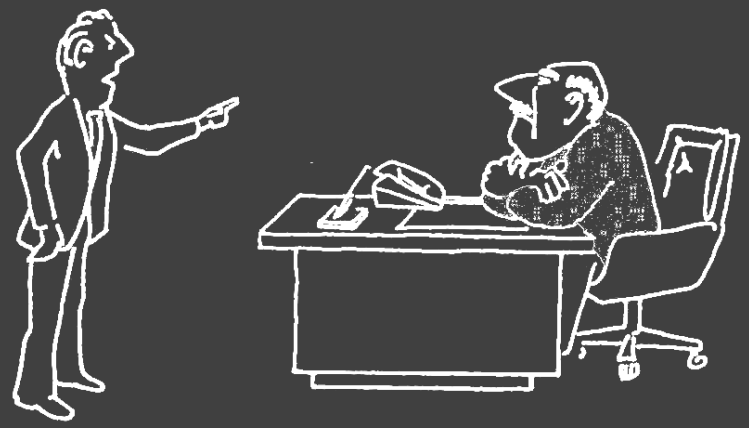
Satz: TSP *ist* NP-*vollständig.*

A short story

Big Boss $\mathscr{BB}$ hat in seinem traditionsreichen Wirtschaftsunternehmen schon sehr früh erfolgreich auf Elektronische Datenverarbeitung gesetzt. So kam es, dass in seinem Unternehmen noch immer eine ganze Reihe Programme im Einsatz sind, die in COBOL geschrieben wurden. Nun sind unter diesen einige, die mitunter in eine Endlosschleife geraten, zumindest sieht es danach aus.

Big Boss $\mathscr{BB}$ hat sich deshalb entschlossen, ein Programm erstellen zu lassen, das für COBOL-Programme testet, ob sie bei irgendeiner Eingabe in eine Endlosschleife geraten, und bittet nun Sie, dieses Programm zu schreiben.

Dank ihres Wissens aus der Theoretischen Informatik können sie $\mathscr{BB}$ jedoch gleich antworten



„Einen Algorithmus für ein solches Programm kann es nicht geben, denn damit könnte man das Halteproblem lösen!“

und $\mathscr{BB}$ wird nach Alternativen suchen müssen.

Another short story

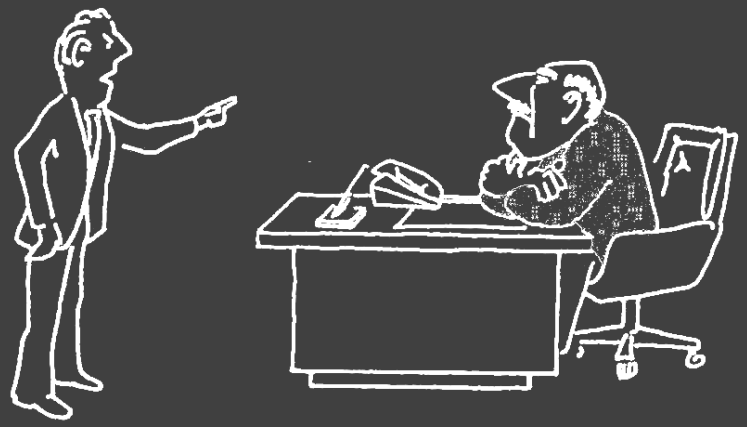
Nehmen wir an, Big Boss $\mathscr{BB}$ hat ein lukratives Projekt an Land gezogen, zu dessem erfolgreichen Abschluss ein exakter effizienter Algorithmus zur Lösung eines kombinatorischen Optimierungsproblems $\mathcal{O}$ benötigt wird. Nun hat er sie als besten Informatiker in seinem Team damit beauftragt, einen solchen effizienten Algorithmus für $\mathcal{O}$ zu entwickeln und zu implementieren.

Doch auch nach monatelangem Suchen und Einsatz all ihrer an der Technischen Universität für angewandte Wissenschaften erlernten Ingenieurtugenden und nach Durchsicht all ihrer im Studium besprochenen Beispielprojekte haben sie noch keinen effizienten exakten Algorithmus gefunden. Wie sie sich auch drehen und wenden, bei allem was sie sich überlegt haben ist die Laufzeit stets exponentiell in der Größe der Eingabe und das ist schon für recht kleine Eingaben nicht mehr tragbar.

Da würden ihnen nun auch die im Studium sehr ausgiebig erlernten Schlüsselkompetenzen bei der Vorstellung ihrer Ergebnisse nicht wirklich weiterhelfen ☺ Sie fürchten, zu $\mathscr{BB}$ sagen zu müssen:

„Ich bin einfach zu blöd, einen
effizienten Algorithmus für $\mathcal{O}$ zu finden“

Sehr viel lieber würden sie ja zu $\mathscr{BB}$ gehen und sagen:

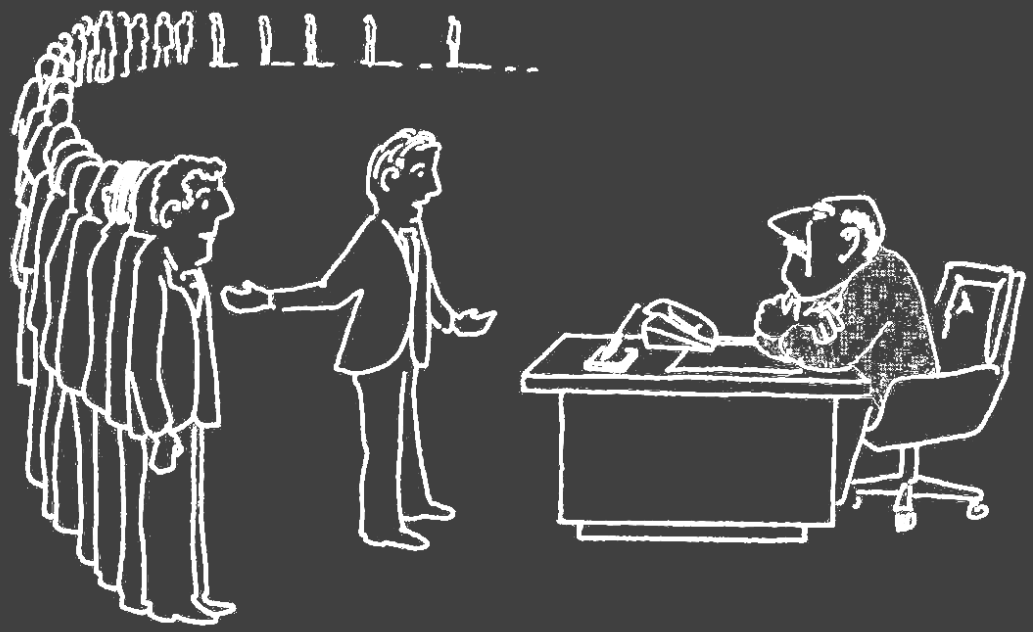


„Ich habe keinen effizienten Algorithmus gefunden,
weil es einen effizienten Algorithmus für $\mathcal{O}$ nicht gibt“

Aber auch das können sie nicht, denn es ist ihnen nicht gelungen, das nachzuweisen. Natürlich hat sich zuvor noch nie jemand mit ihrem Problem $\mathcal{O}$ auseinandergesetzt, schließlich ist das Projekt von $\mathscr{BB}$ ja höchst innovativ!

Doch schließlich können sie nachweisen, dass ihr Problem $\mathcal{O}$ ein NP-vollständiges Problem ist, indem sie $\mathcal{O} \in \mathrm{NP}$ zeigen und eine Polynomialzeitreduktion von einem bekannten NP-vollständigen Problem auf $\mathcal{O}$ konstruieren.

Nun können sie vor $\mathscr{BB}$ treten und sagen:



„Ich habe keinen effizienten Algorithmus gefunden,
aber alle diese berühmten schlauen Leute auch nicht!“

$\mathscr{BB}$ wird nun einsehen, dass es keinen Sinn hat, sie zu feuern und einen anderen schlauen Informatiker einzustellen. Statt nach einem effizienten Algorithmus, der bei allen Eingaben eine optimale Lösung findet, wird er sie nun nach einem Approximations-algorithmus suchen lassen oder nach Verfahren, die zwar nicht immer, aber doch sehr häufig schnell eine optimale Lösung finden ...

Quelle: [M.R. Garey, D.S. Johnson. *Computers and Intractability*. Freeman]