Theoretische Informatik 2

SoSe 2024

Berechenbarkeit und Komplexität

Prof. Dr. Sebastian Siebertz
AG Theoretische Informatik
MZH, Raum 3160
siebertz@uni-bremen.de

Universität
Bremen

Wiederholung

Satz von Cook und Levin

SAT ist NP-vollständig.

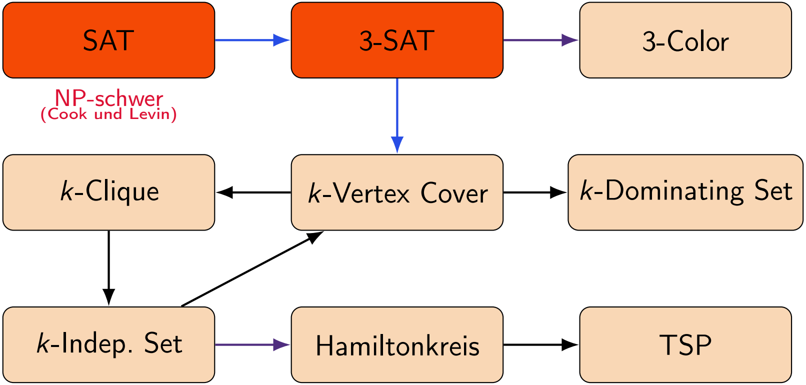
Lemma

Ist L_1 NP*-schwer und gilt* $L_1 \leq_p L_2$*, so ist auch* L_2 NP*-schwer.*

Satz

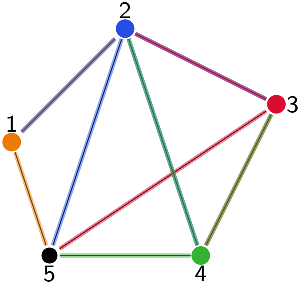
3-SAT ist NP-vollständig.

NP-vollständige Probleme

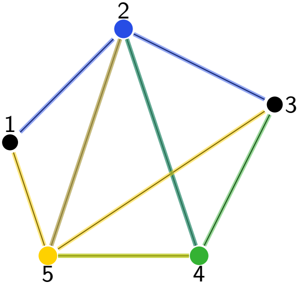


Vertex Cover

Gegeben: Graph $G = (V, E)$, natürliche Zahl k.
Frage: Gibt es eine Teilmenge $U \subseteq V$ mit $|U| \leq k$ und $\{u, v\} \cap U \neq \emptyset$ für alle $\{u, v\} \in E$?



$\{1, 2, 3, 4\}$ ist
VC der Größe 4



$\{2, 4, 5\}$ ist
VC der Größe 3

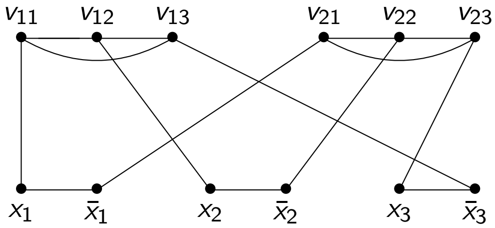
Vertex Cover

Satz

Vertex Cover ist NP-vollständig.

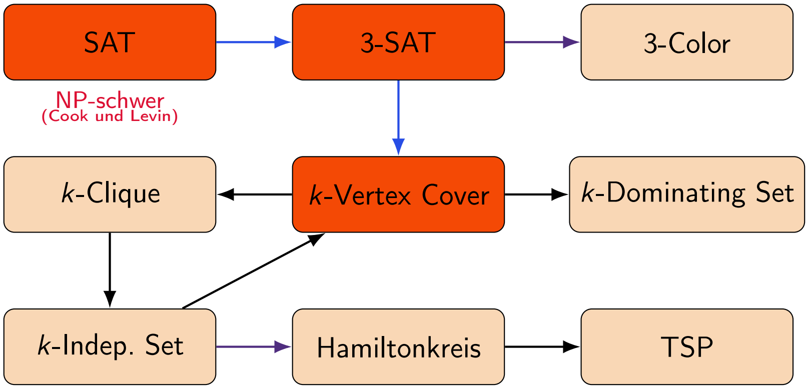
- 3-SAT $\leq_p$ Vertex Cover:

$$(X_1 \lor X_2 \lor \neg X_3) \land (\neg X_1 \lor \neg X_2 \lor X_3)$$



- 3-SAT Instanz mit n Variablen und c Klauseln wird übersetzt in Vertex Cover Instanz mit $2n + 3c$ Knoten und $2n + 6c$ Kanten.

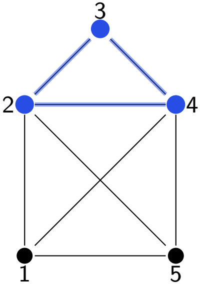
NP-vollständige Probleme



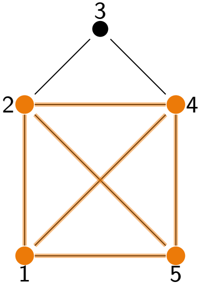
Definition: Clique

Gegeben: Graph $G = (V, E)$, natürliche Zahl k.

Frage: Gibt es eine Teilmenge $C \subseteq V$ mit $|C| \geq k$ und für alle $u, v \in U$ gilt $\{u, v\} \in E$?



$\{2, 3, 4\}$ ist
Clique der Größe 3



$\{1, 2, 4, 5\}$ ist
Clique der Größe 4

- Wollen zeigen: Clique ist NP-vollständig.

Clique ist NP-vollständig

Clique ist in NP:

- Rate Clique $C \subseteq V$ der Größe mindestens k.
- Verifiziere C: Iteriere über alle Knotenpaare $u, v \in C$ und teste, ob $\{u, v\} \in E$. Möglich in Zeit $\mathcal{O}(|C|^2) \subseteq \mathcal{O}(|V|^2)$.

Clique ist NP-schwer: Vertex Cover $\leq_p$ Clique.

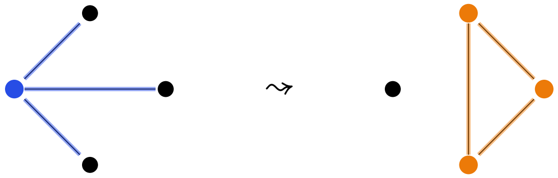
- Sei $G = (V, E)$ ein Graph. Wir definieren das Komplement von G als $\bar{G} = (V, \binom{V}{2} \setminus E)$; d. h., wir flippen alle Kanten.
  - Folgende Funktion f ist die gesuchte Reduktion:

$$f(G, k) = (\bar{G}, |V| - k).$$

Korrektheitsbeweis der Reduktion

Wollen zeigen: $f(G,k) = (\bar{G}, |V| - k)$ ist eine Polynomialzeitreduktion von Vertex Cover auf Clique.

- Sei $G = (V, E)$ ein Graph und $U \subseteq V$ ein VC mit $|U| \leq k$.
- Dann verläuft keine Kante zwischen zwei Knoten in $\bar{U} = V \setminus U$ (ansonsten wäre eine solche Kante nicht abgedeckt).
- Also ist $\bar{U}$ im Komplement $\bar{G}$ vollständig verbunden.
- Da $|\bar{U}| = |V| - |U| \geq |V| - k$, ist $C = \bar{U}$ eine Clique der Größe mindestens $|V| - k$ in $\bar{G}$.



Korrektheitsbeweis der Reduktion

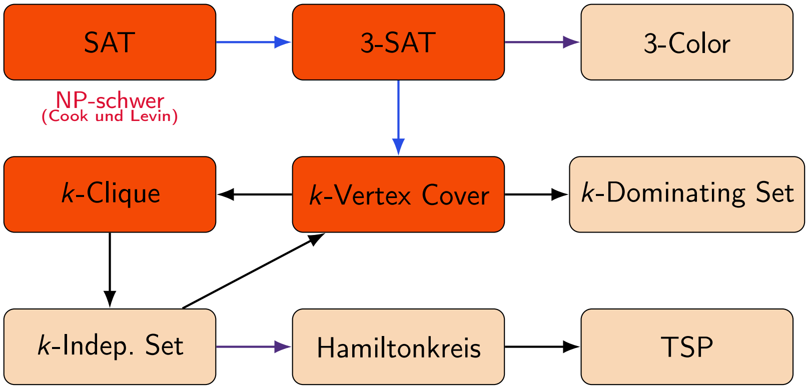
- Sei nun C eine Clique in $\bar{G}$ mit $|C| \geq |V| - k$.
- Dann verläuft in G keine Kante zwischen zwei Knoten in C.
- Kein Knoten aus C ist notwendig, um alle Kanten in G abzudecken (jede hat min. einen Endpunkt außerhalb von C).
- Alle anderen Knoten bilden also ein Vertex Cover. Die Menge $U = V \setminus C$ ist ein VC in G mit $|U| \leq |V| - (|V| - k) = k$.

Ferner ist f in Polynomialzeit berechenbar:

- Berechne $\bar{G}$: Iteration über alle $\mathcal{O}(|V|^2)$ Knotenpaare.
- Berechne $|V| - k$: in konstanter Zeit möglich.

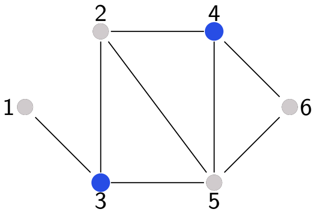
▸ Clique ist NP-vollständig

NP-vollständige Probleme

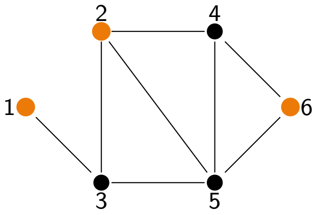


Definition: Independent Set

Gegeben: Graph $G = (V, E)$, natürliche Zahl k.
Frage: Gibt es eine Teilmenge $I \subseteq V$ mit $|I| \geq k$ und für alle $u, v \in I$ gilt $\{u, v\} \notin E$?



$\{3, 4\}$ ist
IS der Größe 2



$\{1, 2, 6\}$ ist
IS der Größe 3

- Wollen zeigen: Independent Set ist NP-vollständig.

Independent Set ist NP-vollständig

Independent Set ist in NP (sehr ähnlich zu Clique):

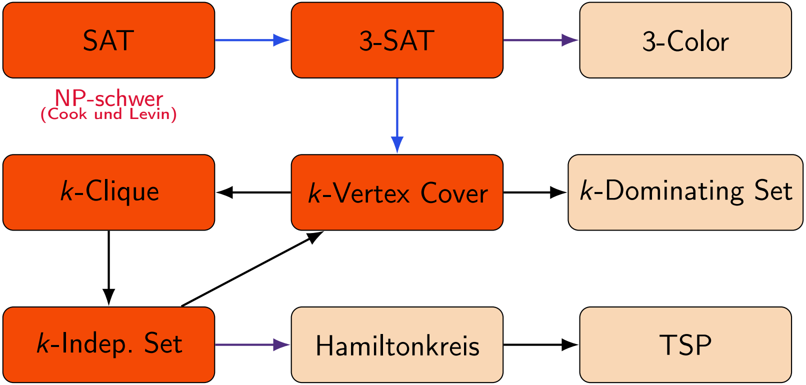
- Rate Independent Set $I \subseteq V$ der Größe mindestens k.
- Verifiziere I: Iteriere über alle Knotenpaare $u, v \in I$ und teste, ob $\{u, v\} \notin E$. Dies ist auch in $\mathcal{O}(|I|^2) \subseteq \mathcal{O}(|V|^2)$ machbar.

Independent Set ist NP-schwer: Clique $\leq_p$ Independent Set.

- Jede Clique in G ist ein Independent Set in $\bar{G}$.
  - Folgende Funktion f ist also die gesuchte Reduktion:

$$f(G, k) = (\bar{G}, k)$$

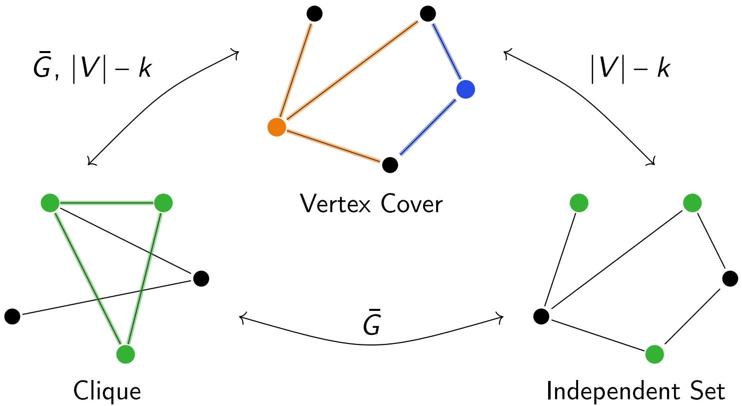
NP-vollständige Probleme



Vertex Cover, Clique und Independent Set

Der Korrektheitsbeweis ist analog zu Vertex Cover $\leq_p$ Clique.

- All diese Probleme sind wechselseitig aufeinander reduzierbar.

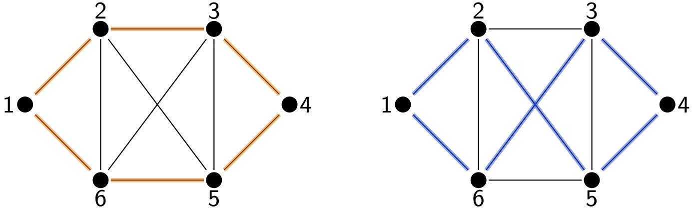


- All diese Probleme sind NP-vollständig

Definition: Hamiltonkreis

Gegeben: Graph $G = (V, E)$.

Frage: Gibt es in G einen Hamiltonkreis, also einen Rundweg, der jeden Knoten genau einmal besucht?

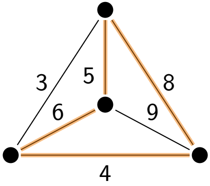


- Ohne Beweis (siehe Folien der letzten VL): Hamiltonkreis ist NP-vollständig.

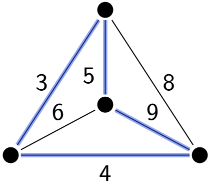
Definition: Travelling Salesperson Problem (TSP)

Gegeben: Graph $G = (V, E)$ mit Kantengewichten $w : E \to \mathbb{N}$, natürliche Zahl k.

Frage: Gibt es in G einen Hamiltonkreis $v_1 v_2 \dots v_n v_1$ mit $w(v_n, v_1) + \sum_{i<n} w(v_i, v_{i+1}) \leq k$?



Summe: 23



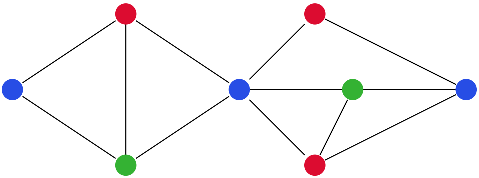
Summe: 21

- Triviale Reduktion von Hamilton Cycle: TSP ist NP-vollständig.

Definition: 3-Color

Gegeben: Graph $G = (V, E)$

Frage: Existiert eine Färbung $c : V \to [3]$, sodass für alle Kanten $\{u, v\} \in E$ gilt: $c(u) \neq c(v)$?



- 3-Color ist NP-vollständig.

3-Color ist NP-vollständig

3-Color ist in NP:

- Rate Färbung $c : V \to [3]$.
- Verifiziere c: Iteriere über alle Kanten $\{u, v\} \in E$ und teste, ob $c(u) \neq c(v)$ gilt. Dies ist in Zeit $\mathcal{O}(|E|)$ möglich.

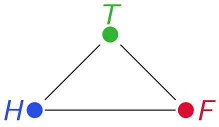
NP-Schwere:

- Wir zeigen 3-SAT $\leq_p$ 3-Color.

Schritte der Reduktion

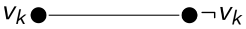
Sei $\varphi = (L_{11} \vee L_{12} \vee L_{13}) \wedge \cdots \wedge (L_{n1} \vee L_{n2} \vee L_{n3})$

Schritt 1: Erstelle eine Palette mit Knoten T, F, H:



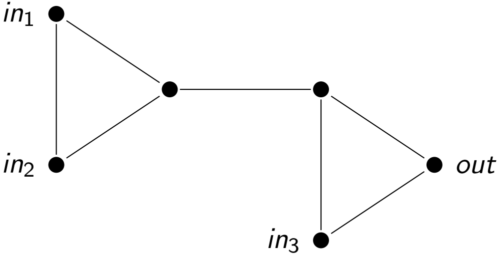
- Stellen Farben für True, False, und Help bereit.

Schritt 2: Verbinde für jede Variable x_k neue Knoten v_k und $\neg v_k$ und verbinde sie alle mit H:



Schritte der Reduktion

Schritt 3: Für jede Klausel $(L_{i1} \vee L_{i2} \vee L_{i3})$ ein „Oder-Gadget“:



- Verbinde in_j mit v_k, falls $L_{ij} = x_k$, und mit $\neg v_k$ sonst.
- „Output“ *out* kann mit T gefärbt werden gdw. mindestens ein „Input“ (ein v_k bzw. $\neg v_k$) mit T gefärbt kann.

Schritte der Reduktion

Schritt 4: Komponenten verbinden

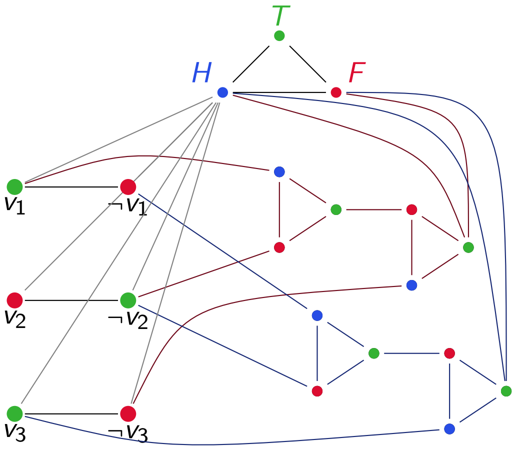
- Verbinde *in*-Knoten der Oder-Gadgets wie eben beschrieben.
- Verbinde *out*-Knoten sowohl mit F als auch mit H.
  - Alle Klauseln sollen wahr gemacht werden.
- Verbinde alle v_k mit H und alle $\neg v_k$ mit H.
  - Entweder v_k oder $\neg v_k$ ist wahr.

Insgesamt haben wir also

- Palette mit Knoten T, F, H und Knoten zwei $v_k, \neg v_k$ pro x_k.
- Oder-Gadgets für alle Klauseln $(L_{i1} \vee L_{i2} \vee L_{i3})$.
  - Kanten $\{in_j, v_k\}$, falls $L_{ij} = x_k$, und $\{in_j, \neg v_k\}$, falls $L_{ij} = x_k$.
  - Kanten $\{out, T\}, \{out, H\}$ für alle Ausgänge out.
  - Kanten $\{v_k, H\}, \{\neg v_k, H\}$ für alle Variablen x_k.

Beispiel

Eingabe: $\varphi = (x_1 \vee \neg x_2 \vee \neg x_3) \wedge (\neg x_1 \vee \neg x_2 \vee x_3)$



Korrektheit der Reduktion

Sei $\varphi = (L_{11} \vee L_{12} \vee L_{13}) \wedge \cdots \wedge (L_{n1} \vee L_{n2} \vee L_{n3})$ erfüllbar.

- Sei $\mathfrak{I}$ eine erfüllendeBelegung.
- Wenn $\mathfrak{I}(x_i) = 1$, färbe v_i mit T und $\neg v_i$ mit F.
- Ansonsten färbe v_i mit F und $\neg v_i$ mit T.
- Nun liegt an jedem Oder-Gadget ein Knoten mit Farbe T an.
- Die Gadgets können so gefärbt werden, dass der Output T ist.

Diese Färbung ist eine valide 3-Färbung,

- Der Ausgabegraph ist also 3-färbbar.

Korrektheit der Reduktion

Sei umgekehrt $c : V \to [3]$ eine Färbung des Ausgabegraphen.

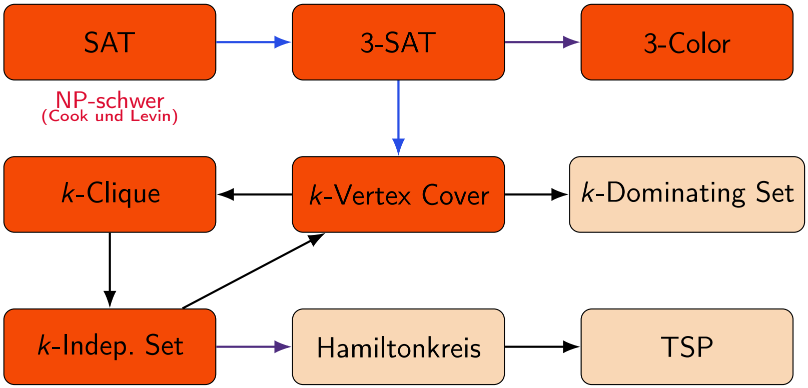
- Es gilt: $c(v_k) = T$ und $c(\neg v_k) = F$, oder andersherum.
  - Wähle $\mathfrak{I}(x_k) = 1$, falls $c(v_k) = T$; sonst $\mathfrak{I}(x_k) = 0$.
- Außerdem: alle Outputknoten der Gadgets sind mit T gefärbt, also auch mindestens ein Input pro Klausel.

Somit gilt: $\mathfrak{I}$ ist wohldefiniert und erfüllt alle Klauseln

- Also ist φ erfüllbar.

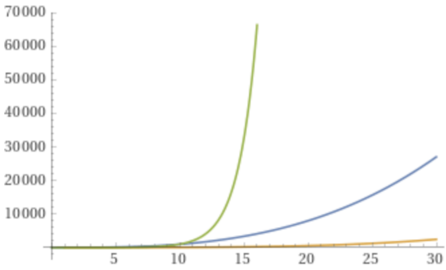
- 3-Color ist NP-vollständig.

NP-vollständige Probleme



Die Exponential Time Hypothesis

- Wir kennen keine effizienten Algorithmen für NP-schwere Probleme.
- Wir können aber auch nicht beweisen, dass solche nicht existieren.
- Wir nehmen P $\neq$ NP an: dann können NP-schwere Probleme nicht in Polynomialzeit gelöst werden.
- Welche superpolynomielle Laufzeit wird benötigt?



  - Funktionen n^3 vs $n^{\log\log n}$ vs 2^n

Wie schnell können wir SAT lösen?

- Sei φ eine SAT-Formel in KNF mit N Variablen und M Klauseln.
- Trivialer Algorithmus: teste für alle möglichen 2^N Belegungen, ob sie φ erfüllen.
  - Laufzeit $2^N \cdot (N+M)^{\mathcal{O}(1)}$
- Etwas besser für 3-SAT:
  - Betrachte beliebige noch nicht erfüllte Klausel und teste rekursiv für die Variablenbelegungen der 3 Variablen der Klausel, ob sie die Formel erfüllen.
  - Es gibt höchstens 7 erfüllende Belegungen dieser Klausel.
  - Kleiner Gewinn gegenüber den möglichen $2^3 = 8$ Belegungen führt zu verbesserten Laufzeit von $7^{N/3} \cdot (N+M)^{\mathcal{O}(1)} = 1.913^N \cdot (N+M)^{\mathcal{O}(1)}$.
- Schnellster bekannter Algorithmus für 3-SAT: $1.308^N \cdot (N+M)^{\mathcal{O}(1)}$

Die Exponential Time Hypothesis (ETH)

Exponential Time Hypothesis (ETH)

3-SAT kann nicht in Zeit $2^{o(N)} \cdot (N+M)^{\mathcal{O}(1)}$ gelöst werden.

- 1999 von Impagliazzo, Paturi und Zane aufgestellt.
- Wenn die Exponential Time Hypothesis gilt, so kann 3-SAT sogar nicht in Zeit $2^{o(N+M)} \cdot (N+M)^{\mathcal{O}(1)}$ gelöst werden.

Untere Schranken mit Hilfe der ETH

- Untere Schranken unter der Annahme, dass ETH gilt:
- Angenommen 3-SAT $\leq_p Q$ für ein Problem Q und
  - die Reduktion übersetzt Formeln mit N Variablen und M Klauseln in Instanzen von Q der Größe $n = \mathcal{O}(N + M)$.
- Dann kann Q nicht in Zeit $2^{o(n)} \cdot n^{\mathcal{O}(1)}$ gelöst werden.
  - Sonst könnte 3-SAT in Zeit $2^{o(N+M)} \cdot (N + M)^{\mathcal{O}(1)}$ gelöst werden, indem wir
  - die Eingabe φ in Polynomialzeit $(N + M)^{\mathcal{O}(1)}$ in eine Instanz x von Q der Größe $\mathcal{O}(N + M)$ übersetzen und
  - den Algorithmus für Q auf x ausführen in Zeit $2^{o(N+M)}(N + M)^{\mathcal{O}(1)}$.
  - Insgesamt hätten wir 3-SAT gelöst in Zeit $(N + M)^{\mathcal{O}(1)} + 2^{o(N+M)}(N + M)^{\mathcal{O}(1)} = 2^{o(N+M)}(N + M)^{\mathcal{O}(1)}$, im Widerspruch zur Annahme, dass ETH gilt.

Beispiel Vertex Cover

- Vertex Cover auf Graphen mit n Knoten, m Kanten und Parameter k: Laufzeit $2^k \cdot (n+m)^{\mathcal{O}(1)}$.
  - Für jede nicht abgedeckte Kante $\{u, v\}$ teste rekursiv, ob es eine Lösung der Größe $k-1$ für $G-u$ oder $G-v$ gibt (einer der beiden Knoten muss in der Lösung liegen).
  - Dies liefert einen Rekursionsbaum mit Verzweigungsgrad 2 und Tiefe k.

Laufzeit ist asymptotisch optimal unter ETH

Wenn die Exponential Time Hypothesis gilt, so kann Vertex Cover auf Graphen mit n Knoten, m Kanten und Parameter k nicht in Zeit $2^{o(k)} \cdot (n+m)^{\mathcal{O}(1)}$ gelöst werden.

- Beweis: unsere Reduktion hat eine 3-SAT Formel mit N Variablen und M Klauseln in eine Vertex Cover Instanz mit $n = 2N + 3M$ Knoten, $m = 2N + 6M$ Kanten und Parameter $k = N + 2M$ übersetzt.

Beispiel 3-Color

- 3-Color Graphen mit n Knoten und m Kanten: Laufzeit $3^n \cdot (n+m)^{\mathcal{O}(1)} = 2^{\mathcal{O}(n)} \cdot (n+m)^{\mathcal{O}(1)}$.
  - Teste alle der 3^n möglichen 3-Färbungen.

Laufzeit ist asymptotisch optimal unter ETH

Wenn die Exponential Time Hypothesis gilt, so kann 3-Color auf Graphen mit n Knoten und m Kanten k nicht in Zeit $2^{o(n+m)} \cdot (n+m)^{\mathcal{O}(1)}$ gelöst werden.

- Beweis: unsere Reduktion hat eine 3-SAT Formel mit N Variablen und M Klauseln in 3-Color Instanz mit $n = 2N + 6M + 3$ Knoten und $m = 4N + 9M + 3$ Kanten übersetzt.

Strong Exponential Time Hypothesis

- Wir können mit dem gleichen Trick wie für 3-SAT auch die Laufzeit für q-SAT verbessern, da wir für jede noch unterfüllte Klausel nur $2^q - 1$ Belegungen testen müssen.
  - Laufzeit: $2^{c_q N} \cdot (N+M)^{\mathcal{O}(1)}$, wobei $c_q \in 1 - \Theta(\frac{1}{q})$.
  - Je größer q wird, um so näher ist die Laufzeit an $2^N \cdot (N+M)^{\mathcal{O}(1)}$.
  - $$\delta_q = \inf\{c_q \mid q\text{-SAT kann in Zeit } \mathcal{O}(2^{c_q N} \cdot (N+M)^{\mathcal{O}(1)}) \text{ gelöst werden}\}.$$

Strong Exponential Time Hypothesis (SETH)

Es gilt $\lim_{q\to\infty} \delta_q = 1$.

- Wenn die Strong Exponential Time Hypothesis gilt, so gibt es keinen Algorithmus für das allgemeine KNF-SAT Problem mit Laufzeit $(2-\varepsilon)^N \cdot (N+M)^{\mathcal{O}(1)}$ für beliebiges $\varepsilon > 0$.

Untere Schranken mit SETH

$$\text{SETH} \quad \Rightarrow \quad \text{ETH} \quad \Rightarrow \quad \text{P} \neq \text{NP}.$$

- Beispiel Dominating Set mit n Knoten und Parameter k: Laufzeit $\mathcal{O}(kn^{k+1})$:
  - Teste für jede Teilmengen D der Größe k, ob es sich um eine dominierende Menge handelt.
  - Es gibt $\binom{n}{k} \leq n^k$ solche k-elementigen Teilmengen.
  - Für jede Teilmenge D markieren wir welche Knoten dominiert werden und prüfen dann, ob alle Knoten markiert sind: Zeit $\mathcal{O}(k \cdot n)$.
  - Mit Optimierung können wir Laufzeit $\mathcal{O}(n^{k+o(1)})$ erreichen.
- SETH $\Rightarrow$ Dominating Set auf Graphen mit n Knoten und Parameter k für beliebiges festes $\varepsilon > 0$ nicht in Zeit $\mathcal{O}(n^{k-\varepsilon})$ gelöst werden.