Exakte Algorithmen für NP-schwere Probleme
Handout vom 30. Januar 2006

Klaus Meyer

7. Februar 2006

1 Überblick

In dem Papier *On The Parameterized Complexity of Exact Satisfiability Problems* werden verschiedene Varianten des Exakten Erfüllbarkeitsproblems ExactSAT unter dem Aspekt der parametrisierten Komplexität betrachtet. In dieser Sitzung wurde durch FPT-Reduktionen von MaxExactSAT auf IndependentSet und umgekehrt gezeigt, dass MaxExactSAT $W[1]$-vollständig ist. Dabei wurde auch gezeigt, dass sogar MonoMaxExactSAT $W[1]$-schwer ist, wenn Literale mehrfach in Klauseln vorkommen können.

2 Definitionen

2.1 Definition 1: FPT

Eine parametrisierte Sprache wird wie folgt definiert:

$\mathcal{L}$ ist eine *parametrisierte Sprache* gdw. $\mathcal{L} \subseteq \Sigma^* \times \mathbb{N}$ für ein Alphabet Σ. Für $(x, k) \in \mathcal{L}$ heißt x Instanz und k Parameter.

$\mathcal{L}$ heißt *fixed parameter tractable* gdw. es einen Algorithmus gibt, der in $O\left(f(k)p(|x|)\right)$ entscheidet, ob $(x, k) \in \mathcal{L}$, für eine Funktion $f : \mathbb{N} \to \mathbb{N}$ und ein Polynom p. („fixed parameter tractable“ lässt sich am besten mit „mit festem Parameter handhabbar“ übersetzen.)

FPT ist die Komplexitätsklasse aller *fixed parameter tractable*-Sprachen.

2.2 Definition 2: fpt-Reduktion

Seien $\mathcal{L} \subseteq (\Sigma^* \times \mathbb{N}), \mathcal{L}' \subseteq (\Sigma'^* \times \mathbb{N})$ parametrisierte Sprachen. Eine Abbildung $f : \mathcal{L} \to \mathcal{L}'$ heißt fpt-Reduktion (geschrieben $\mathcal{L} \leq^{\text{fpt}} \mathcal{L}'$) gdw.

1. $(x, k) \in \mathcal{L} \Leftrightarrow f(x, k) \in \mathcal{L}'$
2. es eine berechenbare Funktion $g : \mathbb{N} \to \mathbb{N}$ gibt, so dass $k' \leq g(k)$ für alle $(x, k) \in \mathcal{L}$ mit $f(x, k) = (x', k')$
3. es eine Konstante c und eine berechenbare Funktion f' gibt, so dass $f(x, k)$ in Zeit $f'(k)|x|^c$ berechnet werden kann.

2.3 Komplexitätsklassen $W[k]$

Da das Papier die Kenntnis von $W[k]$ voraussetzt, wurde das Prinzip in der Sitzung angesprochen. Dazu betrachten wir boolesche Schaltnetze mit einem Eingabevektor der Größe n, k *großen* Gattern und beliebig vielen *kleinen* Gattern. Ein Gatter gilt als klein, wenn es höchstens c Eingänge hat, für eine Konstante c. Die maximale Anzahl großer Gatter auf einem Eingabe-Ausgabe-Pfad im Schaltnetz wird *weft* genannt. (Man sollte noch bemerken, dass auch die Anzahl der Gatter insgesamt auf einem Pfad durch eine Konstante begrenzt wird, da es sonst sehr einfach wäre, nur kleine Gatter zu haben.)

Die Komplexitätsklasse $W[k]$ wird über folgendes Problem mit Parameter $l \geq 1$ definiert: Akzeptiert ein gegebenes Schaltnetz mit weft k einen Eingabevektor mit l vielen Einsen?[1]

Es ergibt sich die folgende Inklusionskette:

$$FPT \subseteq W[1] \subseteq W[2] \subseteq \cdots \subseteq W[SAT] \subseteq W[P]$$

Wichtig ist hier aber nur, dass $W[1]$ als nicht tractable gilt[2], und dass IndependentSet $W[1]$-vollständig ist.

2.4 Definition 3

Eine KNF-Formel ist eine Multimenge von Klauseln, die wiederum Multimengen von Literalen sind. Formeln heißen monoton, wenn in der Formel für jede Variable x nur entweder die Literale x oder $\overline{x}$ vorkommen, aber nicht gemischt.

[1] Neugierige finden die exakte Definition im Web, u.a. in dem Papier *The Hardness of Problems on Thin Colored Graphs*:
`http://archive.cs.uu.nl/pub/RUU/CS/techreps/CS-1995/1995-36.pdf`

[2] Es wird nur vermutet, dass $FPT \neq W[1]$.

2.5 Definition 4

Eine Belegung *exakt-erfüllt* eine Klausel C gdw. genau ein Literal in C wahr ist. Wenn mehr als ein Literal wahr ist, heißt die Klausel *übererfüllt*.

2.6 Definition 5

- MaxExactSAT ist das Problem zu entscheiden, ob es eine Belegung gibt, die mindestens k Klauseln in einer gegebenen Formel F exakt-erfüllt.

- ResMaxExactSAT ist wie MaxExactSAT, mit der zusätzlichen Bedingung, dass keine übererfüllten Klauseln erlaubt sind.

3 MaxExactSAT

3.1 Lemma 1

Behauptung: IndependentSet $\leq^{\text{fpt}}$ MaxExactSAT
Beweis:

Hierzu konstruieren wir aus einem gegebenen Graphen $G = (V, E)$ mit $V = \{v_1, \ldots, v_n\}$ eine Formel $F = \{C_1, \ldots, C_n\}$ über Variablen $x_1, \ldots, x_n, z$, und zeigen, dass das IndependentSet-Problem mit Verwendung von MaxExactSAT lösbar ist. Dazu wird für jeden Knoten v_i die Klausel C_i wie folgt definiert:

$$C_i := \{z, \overline{z}, \overline{x_i}\} \cup \bigcup_{v_j \in N(v_i)} \{x_j\}$$

(Hierbei ist $N(v_i)$ die Menge der Nachbarknoten von v_i, ohne v_i selbst.)

Da immer entweder z oder $\overline{z}$ wahr ist, ist C_i exakt-erfüllt gdw. x_i wahr ist und x_j falsch für alle Nachbarn v_j von v_i. Es bleibt noch zu zeigen, dass G ein Independent Set der Größe k hat gdw. k Klauseln in F exakt-erfüllt werden können.

„$\Rightarrow$“: Sei $I \subseteq V$ ein Independent Set der Größe k. Setze $x_j = 1$ für alle $v_j \in I$ und $x_j = 0$ sonst. Damit ist C_j exakt-erfüllt für alle $v_j \in I$.

„$\Leftarrow$“: Gegeben sei eine Belegung, die k Klauseln exakt-erfüllt. Setze $I = \{v_i \mid C_i \text{ exakt-erfüllt}\}$. Per Konstruktion der Klauseln sind für alle $v_i \in I$ alle Nachbarknoten nicht in I, und damit ist I ein Independent Set.

3.2 Lemma 2

Behauptung: IndependentSet $\leq^{\text{fpt}}$ MonoMaxExactSAT
Beweis:

Wie eben wird aus einem Graphen $G = (V, E)$ mit $V = \{v_1, \ldots, v_n\}$ eine Formel $F = \{C_1, \ldots, C_n\}$ gebastelt, aber diesmal nur über Variablen $x_1, \ldots, x_n$, und F ist monoton. Definiere die Klausel C_i für jeden Knoten v_i wie folgt:

$$C_i := \{x_i\} \cup \bigcup_{v_j \in N(v_i)} \{x_j, x_j\}$$

Wie eben ist C_i exakt-erfüllt gdw. x_i wahr ist und x_j falsch für alle Nachbarn v_j von v_i. Damit geht der Beweis analog zu Lemma 1.

Obwohl die Aussage von Lemma 1 in Lemma 2 enthalten ist, hat auch Lemma 1 seine Daseinsberechtigung: Es gilt nämlich auch dann, wenn Literale nur einfach in Klauseln vorkommen dürfen.

3.3 Lemma 3

Behauptung: MaxExactSAT $\leq^{\text{fpt}}$ IndependentSet
Beweis:

Hierzu konstruieren wir aus einer Formel $F = \{C_1, \ldots, C_m\}$ mit $C_i = \{l_{i,1}, \ldots, l_{i,t_i}\}$ einen Graphen G, so dass die Knoten, die Vorkommen von exakt-erfüllenden Literalen in F entsprechen, ein Independent Set in G bilden. Dabei nehmen wir zunächst an, dass Klauseln Mengen statt Multimengen sind.

Der Graph $G = (V, E)$ wird wie folgt definiert:

$$\begin{aligned} V &= \{v_{i,j} \mid 1 \leq i \leq m, 1 \leq j \leq t_i\} \\ E &= E_1 \cup E_2 \cup E_3 \cup E_4 \end{aligned}$$

mit

$$E_1 := \big\{\{v_{i,j}, v_{a,b}\} \mid i = a\big\} \tag{1}$$
$$E_2 := \big\{\{v_{i,j}, v_{a,b}\} \mid l_{i,j} = \overline{l_{a,b}}\big\} \tag{2}$$
$$E_3 := \big\{\{v_{i,j}, v_{a,b}\} \mid l_{i,j} = l_{a,t}, t \neq b\big\} \tag{3}$$
$$E_4 := \big\{\{v_{i,j}, v_{a,b}\} \mid l_{i,s} = \overline{l_{a,t}}, s \neq j, t \neq b\big\} \tag{4}$$

Damit entspricht jedes Vorkommen eines Literals in einer Klausel einem Knoten in G, und alle Literalvorkommen einer Klausel in F bilden wegen E_1 eine Clique in G. Das stellt sicher, dass höchstens ein Literalvorkommen der

Clique in ein Independent Set kommt. E_2, E_3 und E_4 stellen weitere Eigenschaften des Independent Sets sicher, die gleich gebraucht werden. Es ist noch zu zeigen, dass k Klauseln in F exakt-erfüllt werden können gdw. G ein Independent Set der Größe k hat.

„$\Rightarrow$": Gegeben sei eine Belegung, die k Klauseln $C_1, \ldots, C_k$ exakt-erfüllt, und seien o.B.d.A. l_{i,t_i} die eindeutigen Literale, die C_i erfüllen. Setze $I = \{v_{i,t_i} \mid i = 1, \ldots, k\}$. Damit ist $|I| = k$, und I ist ein Independent Set, denn für alle verschiedenen $v_{i,t_i}, v_{j,t_j} \in I$ gilt:

- $\{v_{i,t_i}, v_{j,t_j}\} \notin E_1$, da l_{i,t_i} und l_{j,t_j} in verschiedenen Klauseln sind.
- $\{v_{i,t_i}, v_{j,t_j}\} \notin E_2$, da $l_{i,t_i} \neq \overline{l_{j,t_j}}$, denn beide Literale sind wahr.
- $\{v_{i,t_i}, v_{j,t_j}\} \notin E_3$, da beide Literale exakt-erfüllend sind. Wenn $l_{i,t_i} \neq l_{j,t_j}$, können sie nicht gemeinsam in einer Klausel auftreten, da sonst die Klausel übererfüllt wäre.
- $\{v_{i,t_i}, v_{j,t_j}\} \notin E_4$, denn wenn ein weiteres Literal l in C_i und $\bar{l}$ in C_j vorkäme, wäre eine der beiden Klauseln übererfüllt.

„$\Leftarrow$": Gegeben sei ein Independent Set $I \subseteq V$ der Größe k, und seien $\mathcal{L} = \{l_{i,j} \mid v_{i,j} \in I\}$ die entsprechenden Literale und $X = \{x_1, \ldots, x_{k'}\}$ die entsprechenden Variablen, mit $k' \leq k$.

Sei $\mathcal{C} = \{C_i \mid \exists j : v_{i,j} \in I\}$ die Menge der Klauseln, die durch $l_{i,j}$ exakt-erfüllt werden sollen. Wegen (1) sind alle $l_{i,j}$ in verschiedenen Klauseln, also $|\mathcal{C}| = k$. (2) stellt sicher, dass alle Literale in $\mathcal{L}$ erfüllt werden können. Damit kann für X eine Belegung angegeben werden, die $\mathcal{L}$ erfüllt, und damit $\mathcal{C}$. Wegen (3) tauchen Literale aus $\mathcal{L}$ nur einmal in jeder Klausel in $\mathcal{C}$ auf, also exakt-erfüllt die Belegung die Klauseln in $\mathcal{C}$. Es bleibt noch die Frage, ob die restlichen Variablen so gesetzt werden können, dass sie keine Klausel erfüllen. Das kann nur schiefgehen, wenn für $l \notin \mathcal{L}$ sowohl l als auch $\bar{l}$ in der Formel auftreten. (4) stellt gerade sicher dass das nicht passiert, und alles ist in Butter.

Wenn Klauseln auch echte Multimengen sein können, muss bei der Reduktion zusätzlich dafür sorgen, dass mehrfach in einer Klausel auftretende Literale nicht als Knoten für das Independent Set ausgewählt werden können. Dazu entfernt man nach der Reduktion einfach alle solche Knoten, die mehrfach auftretenden Literalvorkommen entsprechen.

3.4 Satz 1

MaxExactSAT ist vollständig für $W[1]$. Das folgt direkt aus Lemma 1 und Lemma 3.

3.5 Satz 2

MonoMaxExactSAT $\in FPT$, wenn Klauseln Mengen sind. Wir hatten in dieser Sitzung nur angefangen, dies zu beweisen. Dazu hatten wir die folgenden Fälle für eine monotone Formel F betrachtet:

- In F kommt eine Variable x in mindestens k Klauseln vor. Dann gibt es eine exakt-erfüllende Belegung: Setze x auf wahr und alle anderen Variablen auf falsch.

- In F gibt es höchstens k^2+k Klauseln. Dann gibt es höchstens 2^{k^2+k} viele Variablen[3], und das Problem kann durch erschöpfende Suche gelöst werden. Zwar ist die Laufzeit exponentiell in der Anzahl der Variablen, aber diese hängt hier nur vom Parameter ab, und das ist ja in Ordnung.

In allen anderen Fällen brauchen wir nur noch Formeln behandeln, die mehr als $k^2 + k$ Klauseln haben, und in denen jede Variable in höchstens $k - 1$ Klauseln vorkommt. Dazu wird ein Algorithmus betrachtet, der immer eine Belegung findet, die k Klauseln exakt-erfüllt, und damit ist der Satz bewiesen. Dieser Algorithmus wird in der nächsten Sitzung vorgestellt.

[3]Stimmt ja gar nicht! Es können natürlich mehr sein, aber dann lässt sich F in eine äquivalente Formel mit nur 2^{k^2+k} Variablen umformen. Betrachte für jede Variable die Menge der Klauseln, in denen sie enthalten ist. Es gibt nur 2^{k^2+k} viele solche Mengen, und wenn dieselbe Menge zu mehreren Variablen gehört, tauchen diese immer gemeinsam auf, also kann man alle bis auf eine Variable weglassen.