5. Model-Checking für FO und FO+MOD auf Klassen von beschränkter lokaler Baumweite

Definition 5.1 (beschränkte lokale Baumweite)

Sei σ eine Signatur.

(a) Sei $f : \mathbb{N} \to \mathbb{N}$.

- Eine σ-Struktur $\mathcal{A}$ hat eine durch f beschränkte lokale Baumweite, falls f.a. $r \in \mathbb{N}$ und alle $a \in A$ gilt:

$$bw\left(\mathcal{N}_r^{\mathcal{A}}(a)\right) \leq f(r).$$

D.h.: Für jede Zahl r ist $f(r)$ eine obere Schranke für die Baumweite der r-Nachbarschaften von Elementen im Universum von $\mathcal{A}$.

- Eine Klasse $\mathcal{C}$ von σ-Strukturen hat eine durch f beschränkte lokale Baumweite, falls jedes $\mathcal{A} \in \mathcal{C}$ eine durch f beschränkte lokale Baumweite hat.

(b) Eine Klasse $\mathcal{C}$ von σ-Strukturen hat beschränkte lokale Baumweite, falls es eine Funktion $f : \mathbb{N} \to \mathbb{N}$ gibt, s.d. $\mathcal{C}$ eine durch f beschränkte lokale Baumweite hat.

Beispiel 5.2

Sei σ eine Signatur.

(a) Sei $w \in \mathbb{N}$.
Die Klasse $C_{bw \le w}$ aller σ-Strukturen $\mathcal{A}$ mit $bw(\mathcal{A}) \le w$ hat beschränkte lokale Baumweite; die lokale Baumweite ist beschränkt durch die Funktion $f: \mathbb{N} \to \mathbb{N}$ mit $f(r) := w$ f.a. $r \in \mathbb{N}$.

(b) Sei $d \in \mathbb{N}$.
Die Klasse $C_{Grad \le d}$ aller σ-Strukturen $\mathcal{A}$ vom Grad $\le d$ hat beschränkte lokale Baumweite; die lokale Baumweite ist beschränkt durch die Funktion $f: \mathbb{N} \to \mathbb{N}$ mit

$$f(r) := \nu_d(r) \underset{\substack{\text{Def. in} \\ \text{Lemma 0.2}}}{=} 1 + d \cdot \sum_{i=0}^{r-1} (d-1)^i$$

Beweis: Gemäß Lemma 0.2 gilt für jede σ-Struktur $\mathcal{A}$ vom Grad $\le d$, jedes $r \in \mathbb{N}$ und jedes $a \in A$, dass $|N_r^{\mathcal{A}}(a)| \le \nu_d(r)$ ist.

Gemäß Beispiel 4.4 ist

$$bw(\mathcal{N}_r^{\mathcal{A}}(a)) \le |N_r^{\mathcal{A}}(a)| - 1.$$

Insbes. gilt also: $bw(\mathcal{N}_r^{\mathcal{A}}(a)) \le \nu_d(r)$. $\square$

(c) (Hier ohne Beweis) Die Klasse C_{planar} aller σ-Strukturen $\mathcal{A}$, deren Gaifman-Graph planar ist, hat beschränkte lokale Baumweite.

(d) Aus (b) (bzw analog auch aus (c)) folgt direkt, dass die Klasse $\{ G_{n \times n} : n \in \mathbb{N}_{\geq 1} \}$ aller quadratischen Gitter beschränkte lokale Baumweite besitzt.

Ziel für den Rest dieses Kapitels ist, das folgende algorithmische Metatheorem zu beweisen.

<u>Theorem 5.3</u>

Sei σ eine Signatur, sei φ ein FO+MOD[σ]-Satz und sei $\mathcal{C}$ eine Klasse von σ-Strukturen mit lokal beschränkter Baumweite.
Dann ist das Problem

$\mathrm{EVAL}_{\varphi, \mathcal{C}}$	
<u>Eingabe</u>:	$\mathcal{A} \in \mathcal{C}$
<u>Frage</u>:	Gilt $\mathcal{A} \models \varphi$?

in Pseudo-Linearzeit lösbar, d.h. für jedes $k \in \mathbb{N}_{\geq 1}$ gibt es einen Algorithmus, der das Problem $\mathrm{EVAL}_{\varphi, \mathcal{C}}$ in Zeit $O(|A|^{1+\frac{1}{k}})$ löst.

<u>Bemerkung:</u>
Die Aussage von Theorem 5.3 für FO statt FO+MOD wurde von Frick und Grohe (2001) bewiesen; die Verallgemeinerung zu FO+MOD erfolgte in einer Arbeit von Kuske und Schweikardt (ICALP 2018).

Der Beweis von Theorem 5.3 verwendet das im Folgenden vorgestellte Konzept der Nachbarschaftsüberdeckungen.

Definition 5.4 (Nachbarschaftsüberdeckung)

Sei σ eine Signatur, sei $\mathcal{A}$ eine σ-Struktur, sei $r, s \in \mathbb{N}$.

Eine (r, s)-Nachbarschaftsüberdeckung (kurz: Nbü) von $\mathcal{A}$ ist eine Familie $\mathcal{N}$ von Teilmengen von A, s.d. gilt:

(1) F.a. $a \in A$ ex. $N \in \mathcal{N}$ s.d. $N_r^{\mathcal{A}}(a) \subseteq N$,

und

(2) f.a. $N \in \mathcal{N}$ ex. $b \in A$ s.d. $N \subseteq N_s^{\mathcal{A}}(b)$.

Die Größe $\|\mathcal{N}\|$ von $\mathcal{N}$ ist definiert als

$$\|\mathcal{N}\| := \sum_{N \in \mathcal{N}} |N|.$$

Lemma 5.5 (Lemma von Peleg, 1993)

Sei $k \in \mathbb{N}_{\geq 1}$.

Es gibt einen Algorithmus, der bei Eingabe eines ungerichteten Graphen G und einer Zahl $r \in \mathbb{N}_{\geq 1}$ eine $(r, 2kr)$-Nbü $\mathcal{N}$ von G der Größe

$$\|\mathcal{N}\| \leq |V(G)|^{1+1/k} \quad \text{in Zeit } O\Big(\sum_{N \in \mathcal{N}} \|G[N]\|\Big)$$

berechnet.

Hierbei ist die Größe $\|H\|$ eines ungerichteten Graphen H definiert als $\|H\| := |V(H)| + |E(H)|$.

<u>Beweis:</u> Sei $k \in \mathbb{N}_{\geq 1}$ fest.

Bei Eingabe von G und r geht der Algorithmus wie folgt vor.

$\mathcal{N}_0 := \emptyset$, $V_0 := V(G)$, $i := 0$, $n := |V(G)|$

while $V_i \neq \emptyset$ do

  $i := i+1$

  Wähle ein beliebiges $b \in V_{i-1}$

  $N_{i,0} := \{b\}$, $j := 0$

  repeat

    $j := j+1$

    $L_{i,j} := N_r^G(N_{i,j-1}) \cap V_{i-1}$

    $N_{i,j} := N_r^G(L_{i,j})$

  until $|N_{i,j}| \leq n^{1/k} \cdot |N_{i,j-1}|$

  $j_i := j$

  $N_i := N_{i,j_i}$

  $\mathcal{N}_i := \mathcal{N}_{i-1} \cup \{N_i\}$

  $V_i := V_{i-1} \setminus L_{i,j_i}$

end while

Ausgabe: $\mathcal{N} := \mathcal{N}_i$; $i_{max} := i$

Wir analysieren im Folgenden einen Lauf des Algorithmus bei Eingabe von G und r. Es sei $n := |V(G)|$.

<u>Behauptung 1</u>: Für jede Zahl i gilt: Wenn es einen i-ten Durchlauf durch die while-Schleife gibt, dann gibt es innerhalb des i-ten Durchlaufs durch die while-Schleife höchstens k Durchläufe durch die repeat-Schleife, und am Ende des i-ten Durchlaufs durch die while-Schleife ist $j_i \in \{1,\dots,k\}$, $|W_i| \le i$, $|V_i| \le n-i$ und $V_i \subsetneq V_{i-1}$ und $W_i \supseteq W_{i-1}$.

<u>Beweis</u>: Gemäß Zeile 5 des Algorithmus ist $|N_{i,0}| = 1$.
Gemäß Zeilen 6-10 gilt für jede Zahl $j \geq 1$, für die es einen j-ten und einen $(j+1)$-ten Durchlauf durch die repeat-Schleife gibt, dass $|N_{i,j}| > n^{1/k} \cdot |N_{i,j-1}|$.
Per Induktion ergibt sich, dass für jede solche Zahl j gilt:

$$|N_{i,j}| > n^{1/k} \cdot |N_{i,j-1}| > n^{1/k} \cdot n^{1/k} \cdot |N_{i,j-2}| > \dots > (n^{1/k})^j \cdot \underbrace{|N_{i,0}|}_{=1} = n^{j/k}$$

Außerdem ist $N_{i,j} \subseteq V(G)$ und somit $|N_{i,j}| \le n$.

Also gilt: $n \geqslant |N_{i,j}| > n^{j/k}$.

Daher muss $j<k$ sein. D.h.: Für jede Zahl $j \geq 1$, für die es einen j-ten und einen $(j+1)$-ten Durchlauf durch die repeat-Schleife gibt, ist $j \le k-1$. Es gibt also höchstens k Durchläufe durch die repeat-Schleife. Insbes. ist am Ende des i-ten Durchlaufs durch die while-Schleife $j_i \in \{1,\dots,k\}$.

Außerdem gilt gemäß Zeile 1: $|W_0| = 0$ und $|V_0| = n$; und per Induktion nach i gilt gemäß Zeile 13: $|W_i| \le |W_{i-1}| + 1 \underset{\text{Ind.ann}}{\le} (i-1)+1 = i$.

Gemäß Zeile 14 gilt: $|V_i| = |V_{i-1}| - |L_{i,j_i} \cap V_{i-1}| \underset{\text{Ind.ann.}}{\le} n-(i-1) - |L_{i,j_i} \cap V_{i-1}|$

Um den Beweis von Beh 1 abzuschließen reicht es also zu zeigen, dass $|L_{i,j_i} \cap V_{i-1}| \geqslant 1$ ist. Sei $b \in V_{i-1}$ dasjenige Element, das zu Beginn des i-ten Durchlaufs durch die while-Schleife in Zeile 4 gewählt wird.

Per Induktion nach j zeigen wir, dass f.a. $j \in \{1,\dots,j_i\}$ gilt: $b \in L_{i,j}$:

<u>$j=1$</u>: $L_{i,1} \underset{\text{Zeile 8}}{=} N_r^G(N_{i,0}) \cap V_{i-1} \underset{\text{Zeile 5}}{=} N_r^G(b) \cap V_{i-1} \ni b$. ✓

$\underline{j-1 \to j}$: $L_{i,j} \underset{\text{Zeile 8}}{=} N_r^G(N_{i,j-1}) \cap V_{i-1}$ und

$$N_{i,j-1} \underset{\text{Zeile 9}}{=} N_r^G(L_{i,j-1})$$

Gemäß Ind.annahme ist $b \in L_{i,j-1}$ — also gilt auch: $b \in N_{i,j-1}$ und $b \in N_r^G(N_{i,j-1})$. Da außerdem $b \in V_{i-1}$ ist, gilt: $b \in L_{i,j}$. ✓

Insbes. haben wir gezeigt, dass $b \in L_{i,j_i} \cap V_{i-1}$.

$\square_{\text{Beh 1}}$.

Beachte: Aus Beh 1 folgt insbes., dass der Algorithmus nach max. n Durchläufen durch die while-Schleife terminiert.
Im Folgenden seien $\mathcal{W}$ und i_{max} die Ausgabe des Algorithmus.

$\boxed{\text{Behauptung 2: F.a. } a \in V(G) \text{ ex } N \in \mathcal{W} \text{ s.d. } N_r^G(a) \subseteq N}$

Beweis: Betrachte ein beliebiges $a \in V(G)$. Laut Beh. 1 und den Zeilen 1 und 2 des Algorithmus ist

$$V(G) = V_0 \supsetneq V_1 \supsetneq \dots \supsetneq V_{i_{max}} = \emptyset.$$

Somit existiert ein $i \in \{1, \dots, i_{max}\}$ s.d. $a \in V_{i-1} \setminus V_i$.
Wir betrachten den i-ten Durchlauf durch die while-Schleife.
Gemäß Zeile 14 ist $a \in L_{i,j_i}$. Gemäß Zeilen 9 und 11-13 ist

$$N_i = N_{i,j_i} = N_r^G(L_{i,j_i}) \in \mathcal{W}_i$$

Insbes ist also $N_r^G(a) \subseteq N_i$ (da $a \in L_{i,j_i}$).
Und wegen $\mathcal{W}_i \subseteq \mathcal{W}$ ist $N_i \in \mathcal{W}$.

$\square_{\text{Beh 2}}$.

Behauptung 3: F.a. $N \in \mathcal{W}$ ex. $b \in V(G)$ s.d. $N \subseteq N_{2kr}^{G}(b)$

Beweis: Betrachte ein beliebiges $N \in \mathcal{W}$.
Gemäß Aufbau des Algorithmus gibt es ein $i \in \{1, \dots, i_{max}\}$
s.d. $N = N_i$
Sei b das zu Beginn des i-ten Durchlaufs durch die while-Schleife in Zeile 4 gewählte Element aus V_{i-1}.
Aus Beh 1 wissen wir, dass $j_i \in \{1, \dots, k\}$ ist.
Per Induktion nach j zeigen wir, dass f.a. $j \in \{1, \dots, j_i\}$ gilt:

$$N_{i,j} \subseteq N_{2jr}^{G}(b)$$

– daraus folgt dann direkt Beh 3, da $j_i \leq k$ und $N_i = N_{i,j_i}$ ist.

$j=1$: $N_{i,1} = N_r^G(L_{i,1})$ für $L_{i,1} = N_r^G(b) \cap V_{i-1}$. Somit ist

$$N_{i,1} = N_r^G(N_r^G(b) \cap V_{i-1}) \subseteq N_r^G(N_r^G(b)) = N_{2r}^G(b). \checkmark$$

$j-1 \to j$: Sei $j \in \{2, \dots, j_i\}$. Gemäß Induktionsannahme ist
$N_{i,j-1} \subseteq N_{2(j-1)r}^G(b)$. zu zeigen: $N_{i,j} \subseteq N_{2jr}^G(b)$.

Gemäß Zeilen 8 und 9 des Algorithmus ist

$$\begin{aligned} N_{i,j} &= N_r^G(N_r^G(N_{i,j-1}) \cap V_{i-1}) \\ &\subseteq N_r^G(N_r^G(N_{i,j-1})) \\ &= N_{2r}^G(N_{i,j-1}) \\ &\underset{\text{Ind.ann.}}{\subseteq} N_{2r}^G(N_{2(j-1)r}^G(b)) \\ &= N_{2jr}^G(b) \checkmark \end{aligned}$$

$\square$ Beh 3

Behauptung 4: F.a. $i_1, i_2 \in \{1, \ldots, i_{max}\}$ mit $i_1 \neq i_2$ gilt:

$$N_{i_1, j_{i_1}-1} \cap N_{i_2, j_{i_2}-1} = \emptyset$$

Beweis: Sei oBdA $i_1 < i_2$.

Wir gehen in 2 Schritten vor:

Schritt 1: Zeige, dass $N_{i_1, j_{i_1}-1} \cap N_r^G(V_{i_1}) = \emptyset$

Schritt 2: Zeige, dass $N_{i_2, j_{i_2}-1} \subseteq N_r^G(V_{i_2-1})$

Beh. 4 folgt dann direkt, da wegen $i_1 \leq i_2 - 1$ gilt: $V_{i_2-1} \subseteq V_{i_1}$.
Es reicht also, Schritt 1 und Schritt 2 zu beweisen.

Zu Schritt 1:
Gemäß Zeilen 8 und 14 des Algo. für $i := i_1$ gilt:

$$L_{i_1, j_{i_1}} = N_r^G(N_{i_1, j_{i_1}-1}) \cap V_{i_1-1} \quad \text{und} \quad V_{i_1} = V_{i_1-1} \setminus L_{i_1, j_{i_1}}$$

Somit ist $N_r^G(N_{i_1, j_{i_1}-1}) \cap V_{i_1} = \emptyset$,

d.h.: $N_{i_1, j_{i_1}-1} \cap N_r^G(V_{i_1}) = \emptyset$. ✓(Schritt 1)

Zu Schritt 2:
Sei b das Element aus V_{i_2-1}, das zu Beginn des i_2-ten Durchlaufs durch die while-Schleife in Zeile 4 gewählt wurde.

Dann ist insbes. $N_{i_2,0} \underset{\text{Zeile 5}}{=} \{b\} \subseteq N_r^G(V_{i_2-1})$.

Und gemäß den Zeilen 6–10 des Algorithmus gilt f.a. $j \in \{1, \ldots, j_{i_2}\}$:

$$L_{i_2, j} \subseteq V_{i_2-1} \quad \text{und} \quad N_{i_2, j} = N_r^G(L_{i_2, j}) \subseteq N_r^G(V_{i_2-1}).$$

Insbes. für $j := j_{i_2} - 1$ folgt: $N_{i_2, j_{i_2}-1} \subseteq N_r^G(V_{i_2-1})$ ✓(Schritt 2)

Insgesamt schließt dies den Beweis von Beh. 4 ab

□ Beh. 4

Behauptung 5 Es gilt: $\|\mathcal{N}\| \leq n^{1+1/k}$

Beweis:

Gemäß Aufbau des Algorithmus ist $\mathcal{N} = \{N_1, \dots, N_{i_{max}}\}$

Gemäß Zeilen 10-12 des Algorithmus gilt für jedes $i \in \{1, \dots, i_{max}\}$:

$$|N_i| \leq n^{1/k} \cdot |N_{i,j_i - 1}|.$$

Somit gilt:

$$\|\mathcal{N}\| = \sum_{i=1}^{i_{max}} |N_i| \leq \sum_{i=1}^{i_{max}} n^{1/k} \cdot |N_{i,j_i-1}| = n^{1/k} \cdot \sum_{i=1}^{i_{max}} |N_{i,j_i-1}|$$

Aus Beh. 4 folgt, dass $\sum_{i=1}^{i_{max}} |N_{i,j_i-1}| = \left| \bigcup_{i=1}^{i_{max}} N_{i,j_i-1} \right|$.

Außerdem ist $\bigcup_{i=1}^{i_{max}} N_{i,j_i-1} \subseteq V(G)$, also $\left| \bigcup_{i=1}^{i_{max}} N_{i,j_i-1} \right| \leq n$.

Insgesamt erhalten wir:

$$\|\mathcal{N}\| \leq n^{1/k} \cdot n = n^{1+1/k}$$

$\square_{\text{Beh 5}}$

Beachte:

Aus Beh 1 erhalten wir, dass der Algorithmus bei jeder Eingabe (G, r) terminiert.

Aus Beh 2 und Beh 3 folgt, dass die Ausgabe $\mathcal{N}$ des Algorithmus eine $(r, 2kr)$-Nbü von G ist.

Beh 5 besagt, dass $\mathcal{N}$ die Größe $\|\mathcal{N}\| \leq |V(G)|^{1+1/k}$ hat.

Um den Beweis von Lemma 5.5 abzuschließen müssen wir noch zeigen, dass der Algorithmus Laufzeit $O\left(\sum_{N \in \mathcal{N}} \|G[N]\|\right)$ hat.

Dazu genügt es, Folgendes zu zeigen:

<u>Behauptung 6</u> Für jedes $i \in \{1, \dots, i_{max}\}$ gilt:
Der i-te Durchlauf durch die while-Schleife hat Laufzeit $O(\|G[N_i]\|)$.

<u>Beweis:</u> Übung!

Insgesamt schließt dies den Beweis von Lemma 5.5 ab

$\square$ Lemma 5.5

Durch Kombination von Lemma 5.5 und Satz 4.10 erhalten wir:

<u>Korollar 5.6</u>
Sei σ eine Signatur, sei C eine Klasse von σ-Strukturen mit beschränkter lokaler Baumweite und sei $k, r \in \mathbb{N}_{\geq 1}$.

Dann gibt es einen Algorithmus, der bei Eingabe einer σ-Struktur $\mathcal{A} \in C$ eine $(r, 2kr)$-Nbü $\mathcal{N}$ von $\mathcal{A}$ der Größe $\|\mathcal{N}\| \leq |A|^{1+1/k}$ in Zeit $O(|A|^{1+\frac{1}{k}})$ berechnet.

<u>Beweis:</u> Sei $f: \mathbb{N} \to \mathbb{N}$ so dass C eine durch f beschränkte lokale Baumweite hat. Sei $w := f(2kr)$.
Sei $c := c(w)$ gemäß Satz 4.10 gewählt, so dass für jeden ungerichteten Graphen G' der Baumweite $\leq w$ gilt:
$\|G'\| \underset{\text{Def.}}{=} |V(G')| + |E(G')| \underset{\text{Satz 4.10}}{\leq} c \cdot |V(G')|$.

Unser Algorithmus berechnet zunächst den Gaifman-Graphen G von $\mathcal{A}$ und nutzt dann den Algorithmus aus dem Lemma von Peleg (Lemma 5.5), um eine $(r, 2kr)$-Nbü

$\mathcal{W}$ von G zu berechnen. Gemäß Def. 5.4 ist dies auch eine $(r, 2kr)$-Nbü von $\mathcal{A}$.

Gemäß Lemma 5.5 hat $\mathcal{W}$ die Größe $\|\mathcal{W}\| \leq |A|^{1+1/k}$ und wird in Zeit $O\left(\sum_{N \in \mathcal{W}} \|G[N]\|\right)$ berechnet.

Da $\mathcal{W}$ eine $(r, 2kr)$-Nbü von G ist, gilt für jedes $N \in \mathcal{W}$: ex $b \in A$ s.d. $N \subseteq N^{G}_{2kr}(b) = N^{\mathcal{A}}_{2kr}(b)$. Somit ist

$$bw(G[N]) = bw(\mathcal{A}[N]) \leq bw\left(\mathcal{N}^{\mathcal{A}}_{2kr}(b)\right) \leq f(2kr) = w$$

$\mathcal{A} \in C$, und C hat durch f beschränkte lokale Baumweite

Gemäß Satz 4.10 ist daher $\|G[N]\| \leq c \cdot |N|$.
Die Laufzeit zur Berechnung von $\mathcal{W}$ ist somit

$$O\left(\sum_{N \in \mathcal{W}} \|G[N]\|\right) = O\left(\sum_{N \in \mathcal{W}} c \cdot |N|\right) = O\left(c \cdot \sum_{N \in \mathcal{W}} |N|\right)$$

$$= O(c \cdot \|\mathcal{W}\|) = O\left(c \cdot |A|^{1+1/k}\right) = O\left(|A|^{1+1/k}\right).$$

Um den Beweis abzuschließen muss nur noch gezeigt werden, dass bei Eingabe von $\mathcal{A} \in C$ der Gaifman-Graph G von $\mathcal{A}$ in Zeit $O\left(|A|^{1+1/k}\right)$ berechnet werden kann.

Details dazu: Übung!

$\square$ Korollar 5.6