1 Orakelkonstruktion DisjNP, UP, und Q (O_1)

U.a. Verbesserung der Konstruktion von Khaniki ($\mathsf{CON}^N \wedge \mathrm{TFNP} = \mathrm{FP}$, i.e Q).

Konstruktion wie bei DG.

Sei $e(0) = 2, e(i+1) = 2^{e(i)}$. Sei hier $\{H_m\}_{m\in\mathbb{N}}$ eine Familie von paarweise disjunkten, unendlichen Teilmenge von $e(\mathbb{N})$. (Ebenen H_m gehören zur Zeugensprache bzgl. DisjNP-Maschinenpaar M_a, M_b.) Starte mit PSPACE-vollständiger Menge C welche keine Wörter der Länge $e(\cdot)$ enthält. Definiere folgende Zeugensprachen:

$$A_m^O := \{0^n \mid n \in H_m, \text{existiert } x \in \Sigma^n \text{ mit } x \in O \text{ und } x \text{ endet mit } 0\}$$
$$B_m^O := \{0^n \mid n \in H_m, \text{existiert } x \in \Sigma^n \text{ mit } x \in O \text{ und } x \text{ endet mit } 1\}$$
$$C_m^O := \{0^n \mid n \in H_m, \text{existiert } x \in \Sigma^n \text{ mit } x \in O\}$$

Fakt: wenn $|O \cap \Sigma^n| \leq 1$ für alle $n \in H_m$, dann $(A_m^O, B_m^O) \in \mathrm{DisjUP}^O$.
Wenn $|O \cap \Sigma^n| \leq 1$ für alle $n \in H_m$, dann $C_m^O \in \mathrm{UP}^O$.

Idee: erreiche entweder dass M_a, M_b nicht disjunkt akzeptieren (Task $\tau^1_{a,b}$), oder dass das Zeugenpaar (A_m, B_m) nicht auf $(L(M_a), L(M_b))$ reduzierbar ist (Task $\tau^1_{a,b,r}$ für Transduktor F_r).

Symmetrisch: erreiche das M_a nicht kategorisch akzeptiert (Task τ^3_a), oder dass die Zeugensprache C_m nicht auf $L(M_a)$ reduzierbar ist (Task $\tau^2_{a,r}$ für Transduktor T_r).

Gleichzeitig versuchen wir für möglichst viele M_j erreichen, dass diese nicht total sind. (Task τ^2_j) Am Ende sind die verbleibenden totalen Maschinen M_j^O sehr speziell, denn sie sind auch für gewisse Teilmengen von O total. In Kombination mit dem Fakt dass $\mathrm{P}^C = \mathrm{PSPACE}^C$ können wir relevante Wörter in $O - C$ errechnen und so einen akzeptierenden Weg von $M_j^O(x)$ ausgeben – damit erzielen wir Q.

Sei wie üblich $t \in \mathcal{T}$ wenn der Definitionsbereich endlich ist, nur die Tasks der Form $\tau^1_j, \tau^2_{a,b}, \tau^3_a$ enthält, t diese Tasks auf $\mathbb{N}$ abbildet, und injektiv auf dem Support ist.

Ein Orakel $w \in \Sigma^*$ ist t-valide wenn $t \in \mathcal{T}$ und folgendes gilt:

- V1 Wenn $x < |w|$ und $|x| \notin e(\mathbb{N})$, dann gilt $x \in w \iff x \in C$.
  (Orakel w und C stimmen auf Wörtern mit Länge $\neq e(\cdot)$ überein.)
- V2 Für alle $n = e(i)$ gilt $|w \cap \Sigma^n| \leq 2$.
  (Orakel w ist dünn auf den Ebenen der Länge $e(\cdot)$.)
- V3 Wenn $t(\tau^1_j) = 0$, dann existiert ein z sodass $M_j^w(z)$ definitiv ablehnt.
  ($L(M_j) \neq \Sigma^*$ relativ zum finalen Orakel.)
- V4 Wenn $t(\tau^2_{a,b}) = 0$, dann existiert ein z sodass $M_a^w(z)$ und $M_b^w(z)$ definitiv akzeptieren.
  (Wenn $t(\tau^2_{a,b}) = 0$, dann $L(M_a) \cap L(M_b) \neq \emptyset$ relativ zum finalen Orakel.)
- V5 Wenn $0 < t(\tau^2_{a,b}) = m$, dann gilt für alle $n \in H_m$ dass $|\Sigma^n \cap w| \leq 1$.
  (Wenn $0 < t(\tau^2_{a,b}) = m$, dann $(A_m, B_m) \in \mathrm{DisjNP}$.)
- V6 Wenn $t(\tau^3_a) = 0$, dann existiert ein z sodass $M_a^w(z)$ definitiv auf zwei Rechenwegen akzeptiert.
  (Wenn $t(\tau^3_a) = 0$, dann $L(M_a) \notin \mathrm{UP}$ relativ zum finalen Orakel.)
- V7 Wenn $0 < t(\tau^3_a) = m$, dann gilt für alle $n \in H_m$ dass $|\Sigma^n \cap w| \leq 1$.
  (Wenn $0 < t(\tau^3_a) = m$, dann $C_m \in \mathrm{UP}$.)

Sei T eine abzählbare Aufzählung der o.g. Tasks sodass $\tau^2_{a,b,r}$ immer nach $\tau^2_{a,b}$ kommt, sowie $\tau^3_{a,r}$ immer nach τ^3_a kommt.

[. . . Üblicher Text zur stufenweisen Erweiterung von w_s und t_s . . .]

Wir definieren nun Stufe $s > 0$, diese startet mit einem $t_{s-1} \in \mathcal{T}$ und eine t_{s-1}-validen Orakel w_{s-1} welche nun den kleinsten Task bearbeitet, welcher noch in T ist. Dieser wird unmittelbar nach der Bearbeitung aus T entfernt. In der Bearbeitung wird das Orakel strikt verlängert.

- τ^1_j: Setze $t' = t_{s-1} \cup \{\tau^1_j \mapsto 0\}$. Existiert ein t'-valides Orakel $v \supsetneq w_{s-1}$, dann setze $t_s := t'$ und $w_s := v$.

Ansonsten setze $t_s := t_{s-1}$ und setze $w_s := w_{s-1}y$ für geeignetes $y \in \{0,1\}$ sodass w_s auch t_s-valide ist. (Das ist möglich nach Behauptung 1.1.)

- $\tau^2_{a,b}$: Setze $t' = t_{s-1} \cup \{\tau^2_{a,b} \mapsto 0\}$. Existiert ein t'-valides Orakel $v \sqsupsetneq w_{s-1}$, dann setze $t_s := t'$ und $w_s := v$. Entferne außerdem alle Tasks der Form $\tau^2_{a,b,r}$ von T.

  Ansonsten wähle ein hinreichend großes $m \notin \operatorname{img}(t_s)$ sodass w_s kein Wort der Länge $\min H_m$ definiert. Setze $t_s := t_{s-1} \cup \{\tau^2_{a,b} \mapsto m\}$; damit ist w_{s-1} auch t_s-valide. Setze $w_s := w_{s-1}y$ für geeignetes $y \in \{0,1\}$ sodass w_s auch t_s-valide ist. (Das ist möglich nach Behauptung 1.1.)

- $\tau^2_{a,b,r}$: Wir wissen dass $t_{s-1}(\tau^2_{a,b}) = m > 0$. Setze $t_s = t_{s-1}$ und wähle ein t_s-valides Orakel $w_s \sqsupsetneq w_{s-1}$ sodass bezüglich einem $n \in \mathbb{N}$ eine der folgenden Aussagen gilt:
  - $0^n \in A^v_m$ für alle $v \sqsupseteq w_s$ und $M_a(F_r(0^n))$ lehnt relativ zu w_s definitiv ab.
  - $0^n \in B^v_m$ für alle $v \sqsupseteq w_s$ und $M_b(F_r(0^n))$ lehnt relativ zu w_s definitiv ab.

  (Das ist möglich nach Behauptung 1.2.)

- τ^3_a: Setze $t' = t_{s-1} \cup \{\tau^3_{a,b} \mapsto 0\}$. Existiert ein t'-valides Orakel $v \sqsupsetneq w_{s-1}$, dann setze $t_s := t'$ und $w_s := v$. Entferne außerdem alle Tasks der Form $\tau^3_{a,b,r}$ von T.

  Ansonsten wähle ein hinreichend großes $m \notin \operatorname{img}(t_s)$ sodass w_s kein Wort der Länge $\min H_m$ definiert. Setze $t_s := t_{s-1} \cup \{\tau^3_{a,b} \mapsto m\}$; damit ist w_{s-1} auch t_s-valide. Setze $w_s := w_{s-1}y$ für geeignetes $y \in \{0,1\}$ sodass w_s auch t_s-valide ist. (Das ist möglich nach Behauptung 1.1.)

- $\tau^3_{a,r}$: Wir wissen dass $t_{s-1}(\tau^3_a) = m > 0$. Setze $t_s = t_{s-1}$ und wähle ein t_s-valides Orakel $w_s \sqsupsetneq w_{s-1}$ sodass bezüglich einem $n \in \mathbb{N}$ eine der folgenden Aussagen gilt:
  - $0^n \in C^v_m$ für alle $v \sqsupseteq w_s$ und $M_a(F_r(0^n))$ lehnt relativ zu w_s definitiv ab.
  - $0^n \notin C^v_m$ für alle $v \sqsupseteq w_s$ und $M_b(F_r(0^n))$ akzeptiert relativ zu w_s definitiv.

  (Das ist möglich nach Behauptung 1.3.)

Behauptung 1.1. *Für jedes $t \in \mathcal{T}$ und jedes t-valide w existiert ein $b \in \{0,1\}$ sodass wb auch t-valide ist.*

Behauptung 1.2. *Die Bearbeitung eines Tasks $\tau^2_{a,b,r}$ ist möglich: gilt $t_{s-1}(\tau^2_{a,b}) = m > 0$, dann lässt sich w_{s-1} so zu t_{s-1}-validem $u \sqsupsetneq w_{s-1}$ erweitern, dass eine der o.g. Fälle eintritt.*

Skizze. Widerspruchsbeweis. Erweitere w_{s-1} mit Behauptung 1.1 so weit zu u, dass genau alle Wörter der Länge $< n = e(i) \in H_m$ definiert sind, wobei das i hinreichend groß gewählt wird. Sei $u(X), X \subseteq \Sigma^n$ das Orakel was entsteht, wenn die Ebene $e(i)$ mit genau den Wörtern aus X gefüllt wird, bzw. $u(X) = u \cup X \cup C$. Beob. dass $u(X), |X| \leq 1$ auch t_{s-1}-valide ist.

Nach Annahme gilt

- für gerades $\alpha \in \Sigma^n$ gilt $0^n \in A^{u(\{\alpha\})}_m$ und daher akzeptiert $M_a(F_r(0^n))$ definitiv relativ zu $u(\{\alpha\})$.
- für ungerades $\beta \in \Sigma^n$ gilt $0^n \in B^{u(\{\alpha\})}_m$ und daher akzeptiert $M_b(F_r(0^n))$ definitiv relativ zu $u(\{\beta\})$.

Kombinatorische Standardmethoden zeigen dann, dass relativ zu $u(\{\alpha,\beta\})$ mit geeignetem geraden α, ungeradem β sowohl $M_a(F_r(0^n))$ also auch $M_b(F_r(0^n))$ relativ zu $u(\{\alpha,\beta\})$ akzeptieren. Damit wäre aber auch $u(\{\alpha,\beta\})$ ein geeignetes Orakel in der Bearbeitung von Task $\tau^2_{a,b}$ und wir hätten $t_{s-1}(\tau^2_{a,b}) = 0$. □

Behauptung 1.3. *Die Bearbeitung eines Tasks $\tau^3_{a,r}$ ist möglich: gilt $t_{s-1}(\tau^3_{a,b}) = m > 0$, dann lässt sich w_{s-1} so zu t_{s-1}-validem $u \sqsupsetneq w_{s-1}$ erweitern, dass eine der o.g. Fälle eintritt.*

Skizze. Widerspruchsbeweis symmetrisch zu Behauptunng 2. Betrachte wieder die identisch definierten Orakel $u(X)$. Nach Annahme gilt

- es gilt $0^n \in C_m^{u(\emptyset)}$ und daher lehnt $M_a(F_r(0^n))$ definitiv relativ zu $u(\emptyset)$ ab. [Das ist wichtig um zu zeigen dass es zwei *unterschiedliche* Berechnungen gibt.]
- für gerades $\alpha \in \Sigma^n$ gilt $0^n \in C_m^{u(\{\alpha\})}$ und daher akzeptiert $M_a(F_r(0^n))$ definitiv relativ zu $u(\{\alpha\})$.
- für ungerades $\beta \in \Sigma^n$ gilt $0^n \in C_m^{u(\{\beta\})}$ und daher akzeptiert $M_a(F_r(0^n))$ definitiv relativ zu $u(\{\beta\})$.

Sei für $\xi \in \Sigma^n$ die Menge Q_ξ die Menge an Orakelfragen auf dem akzeptierenden Rechenweg von $M_a(F_r(0^n))$ relativ zu $u(\{\xi\})$. Es gilt $\xi \in Q_\xi$, denn andernfalls würde $u(\emptyset)$ und $u(\{\xi\})$ auf Q_ξ übereinstimmen und wir hätten dass auch $M_a(F_r(0^n))$ relativ zu $u(\emptyset)$ akzeptiert. Das widerspricht der Annahme.

Kombinatorische Standarmethoden zeigen dann, dass es ein gerades α, ungerades β gibt mit $\alpha \notin Q_\beta$, $\beta \notin Q_\alpha$ und so $M_a(F_r(0^n))$ relativ zu $u(\{\alpha, \beta\})$ auf zwei Rechenwegen akzeptiert, je mit Orakelfragen Q_α und Q_β. Diese Rechenwege sind nicht gleich, da $\alpha \in Q_\alpha$ nach obiger Behauptung, aber $\beta \notin Q_\beta$.

Damit wäre aber auch $u(\{\alpha, \beta\})$ ein geeignetes Orakel in der Bearbeitung von Task τ_a^3 und wir hätten $t_{s-1}(\tau_a^3) = 0$. □

Damit ist die Konstruktion möglich. Sei $O = \lim_{s\to\infty} w_s$.

Behauptung 1.4. *Kein Paar aus* DisjNP^O *ist* $\leq_{\mathrm{m}}^{\mathrm{pp}}$*-hart für* DisjUP.

Behauptung 1.5. *Keine Menge aus* UP^O *ist* $\leq_{\mathrm{m}}^{\mathrm{p}}$*-vollständig.*

Behauptung 1.6. *Sei M_j eine totale Maschine, d.h. $L(M^O) = \Sigma^*$. Es existiert eine Länge n mit folgender Eigenschaft: falls $T \subseteq O$ mit O auf Wörtern der Länge $\neq e(\cdot)$ und Wörtern $\leq n$ übereinstimmt, dann $L(M_j^T) = \Sigma^*$.*

Skizze. Sei s die Stufe bei der τ_j^1 bearbeitet wurde. Setze $n = |w_{s-1}|$. Wir zeigen nun, dass dieses n die behauptete Eigenschaft erfüllt. Angenommen, dies gilt nicht, dann existiert ein $T \subseteq O$ dass mit O auf Wörtern der Länge $\neq e(\cdot)$ und Wörtern $\leq n$ übereinstimmt, aber für ein Wort z lehnt $M_j^T(z)$ ab. Sei $t' = t_{s-1} \cup \{\tau_j^1 \mapsto 0\}$ und sei $v = T \cap \Sigma^m$ wobei $m \geq p_j(|x|), n+1$. Beob. dass $M_j^v(z)$ definitiv ablehnt.

Wir zeigen, das v auch t'-valide ist; damit wäre v eine geeignete Erweiterung in Stufe s und wir hätten $t_s = t'$. Das bedeutet nach V3, dass $M_j^{w_s}(z)$ definitiv ablehnt, damit auch $M_j^O(z)$ ablehnt, was der Voraussetzung widerspricht.

Wir zeigen dass v auch t'-valide ist: V1, V2, V5, V7 sind sofort erfüllt nach Definition von T als Teilmenge von O bzw. übereinstimmend mit O auf Wörtern der Länge $\neq e(\cdot)$. Da $v \supsetneq w_{s-1}$, sind auch V4 und V6 erfüllt, außer der neue V3-Fall von $t'(\tau_j^2) = 0$.

Aber auch hier erfüllen wir entsprechende Instanz von V3, da ja $M_j^v(z)$ definitiv ablehnt. □

Behauptung 1.7. *Sei M_j eine totale Maschine, d.h. $L(M_j^O) = \Sigma^*$. Dann existiert eine Funktion $g \in \mathrm{FP}^O$ sodass $g(x)$ einen akzeptierenden Rechenweg von $M_j^O(x)$ ausgibt. Damit gilt nach Definition die Hypothese Q relativ zu O.*

Skizze. Es reicht aus, dass $g \in \mathrm{FP}^O$ nur Wörter hinreichender Länge verarbeiten muss. Sei n hinreichend groß, sodass diese vorige Behauptung 1.6 erfüllt ist. Damit gilt: wenn $T \subseteq O$ mit O auf Wörtern der Länge $\neq e(\cdot)$ und Wörtern der Länge $\leq n$ übereinstimmt, dann $L(M_j^T) = \Sigma^*$.

Sei also nun ein solches x gegeben. Wir werden obige Eigenschaft ausnutzen und iterativ eine Menge $D \subseteq O$ an Orakelwörtern der Länge $e(\cdot)$ aufbauen, welche für die Berechnung $M_j^O(z)$ relevant ist, bis wir alle solchen relevanten Wörter gefunden haben. Wir starten hierbei mit der Menge aller Orakelwörter in O, welche Länge $\leq n$ und Länge $= e(\cdot)$ haben. Beob. dass damit $C \cup D \subseteq O$ und mit O auf Wörtern der Länge $\neq e(\cdot)$ und Wörtern der Länge $\leq n$ übereinstimmt, also auch $L(M_j^{C\cup D}) = \Sigma^*$. Da uns D vorliegt, können wir sogar diese Orakelwerte in M hineincodieren, sodass $M_j'^C(z)$ äquivalent arbeitet. Und da $\mathrm{P}^C = \mathrm{PSPACE}^C$, können wir in FP^O auch einen akzeptierenden Rechenweg von $M_j'^C(x)$ bestimmen.

1 $D \leftarrow \{w \mid w \in O, |w| \leq n, \exists i.|w| = e(i)\}$
2 **repeat**
3 | Sei α ein akzeptierender Rechenweg auf $M_j^{C \cup D}(x)$ und Q die Menge an Orakelfragen
4 | **if** *existiert eine Frage* $q \in Q$ *für die* $q \in O - C$ *aber* $q \notin D$ **then**
5 | | $D \leftarrow D \cup \{q\}$
6 | **else**
7 | | **return** α
8 | **end**
9 **end**

Korrektheit: Beobachte zunächst die Invariante dass $D \subseteq O \cap \{w \mid \exists i.|w| = e(i)\}$. Wie oben argumentiert gilt außerdem, dass $C \cup D \subseteq O$ und mit O auf Wörtern der Länge $\neq e(\cdot)$ und Wörtern der Länge $\leq n$ übereinstimmt. Damit $L(M_j^{C \cup D}) = \Sigma^*$ und insbesondere existiert dann auch ein akzeptierender Rechenweg auf $M_j^{C \cup D}(x)$, und Zeile 3 wohldefiniert.

Terminiert nun der Algorithmus mit einem Rechenweg α, wissen wir auch dass für alle Orakelfragen $q \in Q$ entweder $q \in C$ gilt oder $q \notin O$ oder $q \in O, D$ gilt. Damit stimmt $C \cup D$ mit O auf Q überein, und auch $M_j^O(x)$ akzeptiert mit Rechenweg α.

Laufzeit: Wir zeigen dass der Algorithmus in polynomiell beschränkter deterministischer Zeit (abhängig von $|x|$) relativ zu O arbeitet. Wir wissen, dass für jede Orakelfrage $q \in Q$ gilt, dass $|q| \leq p_j(|x|)$. Zusammen mit o.g. Invariante gilt $D \subseteq O \cap \{w \mid \exists i.|w| = e(i) \leq p_j(|x|)\}$. Nach V2 gilt damit $\ell(D) \leq p_j(|x|) \cdot p_j(|x|) \cdot 2$, denn in den je $\leq p_j(|x|)$ Ebenen der Länge $e(i) \leq p_j(|x|)$ existieren höchstens zwei Wörter der Länge $e(i) \leq p_j(|x|)$. Damit folgt auch unmittelbar, dass der Algorithmus nach höchstens polynomiell vielen Iterationen terminiert.

Zeile 3 kann damit auch in polynomiell beschränkter deterministischer Zeit berechnet werden. Wie oben skizziert kann der Rechenweg in deterministisch polynomieller Zeit abh. von $|x|$ und $\ell(D)$ relativ zu C bestimmt werden. Nach Konstruktion ist leicht zu sehen, dass dieser Rechenweg dann auch relativ zu O bestimmt werden kann. □

2 Orakel mit DisjCoNP und DisjNP und P = UP (O_2)

Sei $e(0) = 2, e(i+1) = 2^{2^{e(i)}}$. (Doppelt exponentiell!) Sei hier $\{H_m\}_{m\in\mathbb{N}}$ eine Familie von paarweise disjunkten, unendlichen Teilmenge von $e(\mathbb{N})$. (Ebenen H_m gehören zur Zeugensprache bzgl. Disj(Co)NP-Maschinenpaar M_a, M_b.) Starte mit PSPACE-vollständiger Menge C welche keine Wörter der Länge $e(\cdot)$ enthält. Definiere folgende Zeugensprachen:

$$A_m^O := \{0^n \mid n \in H_m, \text{für alle } x \in \Sigma^n \text{ gilt } x \in O \to \ x \text{ endet mit } 0\}$$
$$B_m^O := \{0^n \mid n \in H_m, \text{für alle } x \in \Sigma^n \text{ gilt } x \in O \to \ x \text{ endet mit } 1\}$$
$$D_m^O := \{0^n \mid n \in H_m, \text{es existiert ein } x \in \Sigma^n \text{ mit } x \in O \text{ und } x \text{ endet mit } 0\}$$
$$E_m^O := \{0^n \mid n \in H_m, \text{es existiert ein } x \in \Sigma^n \text{ mit } x \in O \text{ und } x \text{ endet mit } 1\}$$
$$\cup\ \overline{\{z \in \Sigma^* \mid |z| = e(i) \text{ für ein } i\}}$$

Fakt: $|O \cap \Sigma^n| \geq 1$ für alle $n \in H_m \implies (A_m^O, B_m^O) \in$ DisjCoNP.
Fakt: $O \cap \Sigma^{n-1}0 \neq \emptyset$ genau dann wenn $O \cap \Sigma^{n-1}1 = \emptyset$ für alle $n \in H_m \implies \overline{D_m^O} = E_m^O$ und $(D_m^O, E_m^O) \in$ DisjNP und $D_m^O \in$ NP $\cap$ coNP.

Idee: erreiche entweder dass M_a, M_b nicht disjunkt ablehnen (Task $\tau^2_{a,b}$), oder dass das Zeugenpaar (A_m, B_m) nicht auf $(L(M_a), L(M_b))$ reduzierbar ist (Task $\tau^2_{a,b,r}$ für Transduktor F_r). Ebenso, für DisjNP, erreiche entweder dass M_a, M_b nicht disjunkt akzeptieren (Task $\tau^3_{a,b}$), oder dass das Zeugenpaar (A_m, B_m) nicht auf $(L(M_a), L(M_b))$ reduzierbar ist (Task $\tau^3_{a,b,r}$ für Transduktor F_r).

Gleichzeitig versuchen wir für möglichst viele M_j erreichen, dass diese nicht kategorisch sind. (Task τ^1_j) Am Ende sind die verbleibenden totalen Maschinen M_j^O sehr speziell, denn sie sind auch für gewisse Teilmengen von O kategorisch. In Kombination mit dem Fakt dass $\mathrm{P}^C = \mathrm{PSPACE}^C$ können wir relevante Wörter in $O - C$ errechnen und so einen akzeptierenden Weg von $M_j^O(x)$ ausgeben – damit entscheiden wir $L(M_j^O)$ und haben also auch P = UP.

Sei wie üblich $t \in \mathcal{T}$ wenn der Definitionsbereich endlich ist, nur die Tasks der Form $\tau^2_{a,b}, \tau^3_{a,b}, \tau^1_j$ enthält, t diese Tasks auf $\mathbb{N}$ abbildet, und injektiv auf dem Support ist.

Ein Orakel $w \in \Sigma^*$ ist t-valide wenn $t \in \mathcal{T}$ und folgendes gilt:

- V1 Wenn $x < |w|$ und $|x| \notin e(\mathbb{N})$, dann gilt $x \in w \iff x \in C$.
  (Orakel w und C stimmen auf Wörtern mit Länge $\neq e(\cdot)$ überein.)
- V2 Wenn $t(\tau^1_j) = 0$, dann existiert ein z sodass $M_j^w(z)$ auf zwei Rechenwegen akzeptiert.
  (M_j nicht kategorisch relativ zum finalen Orakel.)
- V3 Wenn $t(\tau^2_{a,b}) = 0$, dann existiert ein z sodass $M_a^w(z)$ und $M_b^w(z)$ definitiv ablehnen.
  (Wenn $t(\tau^2_{a,b}) = 0$, dann $(\overline{L(M_a)}, \overline{L(M_b)}) \notin$ DisjCoNP relativ zum finalen Orakel.)
- V4 Wenn $0 < t(\tau^2_{a,b}) = m$, dann gilt für alle $n \in H_m$: wenn w für alle Wörter der Länge n definiert ist, dann $|\Sigma^n \cap w| \geq 1$.
  (Wenn $0 < t(\tau^2_{a,b}) = m$, dann $(A_m, B_m) \in$ DisjCoNP.)
- V5 Wenn $t(\tau^3_{a,b}) = 0$, dann existiert ein z sodass $M_a^w(z)$ und $M_b^w(z)$ definitiv akzeptieren.
  (Wenn $t(\tau^3_{a,b}) = 0$, dann $(L(M_a), L(M_b)) \notin$ DisjNP relativ zum finalen Orakel.)
- V6 Wenn $0 < t(\tau^3_{a,b}) = m$, dann gilt für alle $n \in H_m$: wenn w für alle Wörter der Länge n definiert ist, dann entweder $|\Sigma^{n-1}0 \cap w| = 0$ oder $|\Sigma^{n-1}1 \cap w| = 0$ aber nicht beides.
  (Wenn $0 < t(\tau^3_{a,b}) = m$, dann $(D_m, E_m) \in$ DisjNP, $D_m \in$ NP $\cap$ coNP.)

Sei T eine abzählbare Aufzählung der o.g. Tasks sodass $\tau^2_{a,b,r}$ immer nach $\tau^2_{a,b}$, sowie $\tau^3_{a,b,r}$ immer nach $\tau^3_{a,b}$ kommt.

[. . . Üblicher Text zur stufenweisen Erweiterung von w_s und t_s . . .]

Wir definieren nun Stufe $s > 0$, diese startet mit einem $t_{s-1} \in \mathcal{T}$ und eine t_{s-1}-validen Orakel w_{s-1} welche nun den kleinsten Task bearbeitet, welcher noch in T ist. Dieser wird unmittelbar nach der Bearbeitung aus T entfernt. In der Bearbeitung wird das Orakel strikt verlängert.

- τ^1_j: Setze $t' = t_{s-1} \cup \{\tau^1_j \mapsto 0\}$. Existiert ein t'-valides Orakel $v \sqsupsetneq w_{s-1}$, dann setze $t_s := t'$ und $w_s := v$.

  Ansonsten setze $t_s := t_{s-1}$ und setze $w_s := w_{s-1}y$ für geeignetes $y \in \{0,1\}$ sodass w_s auch t_s-valide ist. (Das ist möglich nach Behauptung 2.1.)
- $\tau^2_{a,b}$: Setze $t' = t_{s-1} \cup \{\tau^2_{a,b} \mapsto 0\}$. Existiert ein t'-valides Orakel $v \sqsupsetneq w_{s-1}$, dann setze $t_s := t'$ und $w_s := v$. Entferne außerdem alle Tasks der Form $\tau^2_{a,b,r}$ von T.

  Ansonsten wähle ein hinreichend großes $m \notin \operatorname{img}(t_s)$ sodass w_s kein Wort der Länge $\min H_m$ definiert. Setze $t_s := t_{s-1} \cup \{\tau^2_{a,b} \mapsto m\}$; damit ist w_{s-1} auch t_s-valide. Setze $w_s := w_{s-1}y$ für geeignetes $y \in \{0,1\}$ sodass w_s auch t_s-valide ist. (Das ist möglich nach Behauptung 2.1.)
- $\tau^2_{a,b,r}$: Wir wissen dass $t_{s-1}(\tau^2_{a,b}) = m > 0$. Setze $t_s = t_{s-1}$ und wähle ein t_s-valides Orakel $w_s \sqsupsetneq w_{s-1}$ sodass bezüglich einem $n \in \mathbb{N}$ eine der folgenden Aussagen gilt:
  - $0^n \in A^v_m$ für alle $v \sqsupseteq w_s$ und $M_a(F_r(0^n))$ akzeptiert definitiv relativ zu w_s.
  - $0^n \in B^v_m$ für alle $v \sqsupseteq w_s$ und $M_b(F_r(0^n))$ akzeptiert definitiv relativ zu w_s.

  (Das ist möglich nach Behauptung 2.2.)
- $\tau^3_{a,b}$: Setze $t' = t_{s-1} \cup \{\tau^3_{a,b} \mapsto 0\}$. Existiert ein t'-valides Orakel $v \sqsupsetneq w_{s-1}$, dann setze $t_s := t'$ und $w_s := v$. Entferne außerdem alle Tasks der Form $\tau^3_{a,b,r}$ von T.

  Ansonsten wähle ein hinreichend großes $m \notin \operatorname{img}(t_s)$ sodass w_s kein Wort der Länge $\min H_m$ definiert. Setze $t_s := t_{s-1} \cup \{\tau^3_{a,b} \mapsto m\}$; damit ist w_{s-1} auch t_s-valide. Setze $w_s := w_{s-1}y$ für geeignetes $y \in \{0,1\}$ sodass w_s auch t_s-valide ist. (Das ist möglich nach Behauptung 2.1.)
- $\tau^3_{a,b,r}$: Wir wissen dass $t_{s-1}(\tau^3_{a,b}) = m > 0$. Setze $t_s = t_{s-1}$ und wähle ein t_s-valides Orakel $w_s \sqsupsetneq w_{s-1}$ sodass bezüglich einem $n \in \mathbb{N}$ eine der folgenden Aussagen gilt:
  - $0^n \in D^v_m$ für alle $v \sqsupseteq w_s$ und $M_a(F_r(0^n))$ lehnt definitiv relativ zu w_s ab.
  - $0^n \in E^v_m$ für alle $v \sqsupseteq w_s$ und $M_b(F_r(0^n))$ lehnt definitiv relativ zu w_s ab.

  (Das ist möglich nach Behauptung 2.2.)

Behauptung 2.1. *Für jedes $t \in \mathcal{T}$ und jedes t-valide w existiert ein $b \in \{0,1\}$ sodass wb auch t-valide ist.*

Behauptung 2.2. *Die Bearbeitung eines Tasks $\tau^2_{a,b,r}$ ist möglich: gilt $t_{s-1}(\tau^2_{a,b}) = m > 0$, dann lässt sich w_{s-1} so zu t_{s-1}-validem $u \sqsupsetneq w_{s-1}$ erweitern, dass eine der o.g. Fälle eintritt.*

Skizze. Direkter Beweis. Erweitere durch Beauptung 1 das Orakel w_{s-1} so weit zu u, dass genau alle Wörter der Länge $< n = e(i) \in H_m$ definiert sind, wobei das i hinreichend groß gewählt wird. Sei $u(X), X \subseteq \Sigma^n$ das Orakel was entsteht, wenn die Stufe $e(i)$ mit genau den Wörtern aus X gefüllt wird, bzw. $u(X) = u \cup X \cup C$.

Sei $\hat{s}$ die Stufe, in der $\tau^2_{a,b}$ bearbeitet wurde. Da nach Behauptung 2.1 u auch t_{s-1} valide ist, ist es auch $t_{\hat{s}-1}$-valide. Es ist sogar $u(\emptyset)$ auch $t_{\hat{s}-1}$-valide, denn $t_{\hat{s}-1}(\tau^2_{a,b})$ ist undefiniert.

Sei $y = T_r(0^n)$. Wir wissen, dass eine der Maschinen $M_a(y)$ oder $M_b(y)$ relativ zu $u(\emptyset)$ akzeptieren muss. Andernfalls wäre $u(\emptyset)$ ein geeignetes Orakel zur Zerstörung des Paars M_a, M_b in der Bearbeitung von Task $\tau^2_{a,b}$ und wir hätten $t_{s-1}(\tau^2_{a,b}) = 0$.

Ohne Beschränkung akzeptiert also $M_a(y)^{u(\emptyset)}$ auf einem Rechenweg mit polynomiell vielen Orakelfragen Q. Wähle ein $\alpha \in \Sigma^n - Q$ was mit 0 endet. Dann akzeptiert auch $M_a(y)^{u(\{\alpha\})}$ auf dem gleichen Rechenweg und $0^n \in A^{u(\{\alpha\})}_m$. Es ist leicht zu sehen, dass $u(\{\alpha\})$ auch t_s-valide ist. □

Behauptung 2.3. *Die Bearbeitung eines Tasks $\tau^3_{a,b,r}$ ist möglich: gilt $t_{s-1}(\tau^3_{a,b}) = m > 0$, dann lässt sich w_{s-1} so zu t_{s-1}-validem $u \sqsupsetneq w_{s-1}$ erweitern, dass eine der o.g. Fälle eintritt.*

Damit ist die Konstruktion möglich. Sei $O = \lim_{s\to\infty} w_s$.

Behauptung 2.4. *Kein Paar aus* $\mathrm{DisjCoNP}^O$ *ist* $\leq_{\mathrm{m}}^{\mathrm{pp}}$*-vollständig.*

Behauptung 2.5. *Kein Paar aus* DisjNP^O *ist* $\leq_{\mathrm{m}}^{\mathrm{pp}}$*-hart für* $\mathrm{NP}^O \cap \mathrm{coNP}^O$*. Damit gilt im Übrigen dass keine Sprache aus* $\mathrm{NP}^O \cap \mathrm{coNP}^O$ *auch* $\leq_{\mathrm{m}}^{\mathrm{p}}$*-vollständig ist.*

Wir wollen nun zeigen, dass wir UP-Sprachen in P entscheiden können. Sei im Folgenden M_j eine kategorische Maschine relativ zu O. Um $x \in L(M_j)$ zu entscheiden nutzen wir aus, dass $\mathrm{PSPACE}^C = \mathrm{P}^C$, um so iterativ eine Menge $D \subseteq O$ an Orakelwörtern der Länge $e(\cdot)$ aufbauen, welche für die Berechnung $M_j(x)$ relevant ist, bis wir nach einigen Iterationen alle solchen relevanten Wörter gefunden haben. Wir beschränken uns im Folgenden auf Eingaben hinreichender Länge, sodass es für x ein eindeutiges i gibt, sodass

$$e(i-1) \leq \log(|x|) < e(i), \quad 2p_j(|x|) < 2^{e(i)-1} < e(i+1). \tag{$*$}$$

Wir definieren Folgendes: Eine (U, W, W') *respektierende akzeptierende Berechnung* P *von* M_j *auf Eingabe* x ist ein akzeptierender Rechenweg P von $M_j(x)$ relativ zu einem Orakel $v \subseteq \Sigma^*$, wobei

$$U, W, W' \subseteq \Sigma^{e(i)}, \quad W \subseteq O, \quad W' \subseteq \overline{O}$$

und für v gilt:

1. v ist definiert für genau die Wörter der Länge $\leq p_j(|x|)$.
2. $v(q) = O(q)$ für alle q mit $|q| \neq e(i)$, wobei hier das i diejenige eindeutige Zahl ist für die obigen Ungleichungen $(*)$ bzgl. $|x|$ gelten.
3. $v(q) = 1$ für alle q mit $q \in W$.
4. $v(q) = 0$ für alle q mit $q \in W'$.
5. $v(q) = 1 \implies q \in U$ für alle $q \in \Sigma^{e(i)}$. [v enthält auf Ebene $e(i)$ nur Wörter, die auch in U vorkommen.]

Sei P^{all} die Menge der Orakelfragen auf P, und sei $P^{\mathrm{yes}} = P^{\mathrm{all}} \cap v$, $P^{\mathrm{no}} = P^{\mathrm{all}} \cap \overline{v}$. Beob. dass für festes $U = \Sigma^{e(i)}$ (bzw. $U = \Sigma^{e(i)-1}0$, $U = \Sigma^{e(i)-1}1$) wegen $\mathrm{PSPACE}^C = \mathrm{P}^C$ das Ermitteln einer (U, W, W') respektierenden akzeptierenden Berechnung einfach in Polynomialzeit (abh. von $|x|$ und $\ell(W), \ell(W')$) relativ zu O möglich ist: insbesondere stimmt O mit C auf Wörtern der Länge $\neq e(\cdot)$ überein, und alle anderen Wörter der Länge $e(0), e(1), \ldots, e(i-1)$ können vorab mit Queries an O in Polynomialzeit erfragt werden. Entsprechende Belegungen von v für Wörter der Länge $e(i)$ können z.B. in PSPACE enumeriert werden.

Sei s die Stufe bei der τ_j^1 betrachtet wurde. Zusätzlich zur Einschränkung $(*)$ diskutieren wir ab jetzt nur noch Eingaben, für welche das Orakel w_{s-1} keine Wörter der Länge $e(i)$ definiert. Sei $M = \bigcup\{H_m \mid t_s(\tau_{a,b}^3) = m > 0\}$ eine Menge an Ebenen, welche einer DisjNP-Zeugensprache *in Stufe* s zugewiesen ist. Beob. dass $M \in \mathrm{P}$, denn es sind höchstens endlich viele H_m in der Vereinigung, welche je in P entscheidbar sind.

Wie Nutzen der Menge U erklären?

Behauptung 2.6. *Seien* P_1, P_2 *zwei* (U, W, W') *respektierenden akzeptierenden Berechnungen von* M_j *auf Eingabe* x*. Folgende Aussage gilt jeweils bezüglich dieser beiden Einschränkungen auf* U*:*

- $e(i) \in M$*, sowie* $U = \Sigma^{e(i)-1}0$ *oder* $U = \Sigma^{e(i)-1}1$*,*
- $e(i) \notin M$ *und* $U = \Sigma^{e(i)}$*.*

Wenn P_1^{all} *und* P_2^{all} *beide je eine (nicht notwendigerweise identische) Orakelfrage* q_1, q_2 *der Länge* $e(i)$ *enthalten, welche in* $U - (W \cup W')$ *liegt, dann haben diese zwei Berechnungen eine (identische) Orakelfrage* q *der Länge* $e(i)$ *gemeinsam, welche in* $U - (W \cup W')$ *liegt.*

Skizze. Wir beweisen zunächst den ersten Fall. Angenommen, dies gilt nicht, also sei x sowie $U, W, W' \subseteq \Sigma^{e(i)}$, $W \subseteq O$, $W' \subseteq \overline{O}$ gegeben. Seien außerdem P_1 und P_2 zwei (U, W, W') respektierenden akzeptierenden Berechnungen von M_j auf Eingabe x, welche je eine Orakelfrage der Länge $e(i)$ enthalten, welche nicht in $W \cup W'$ liegt, aber keine Orakelfrage aus $U - (W \cup W')$ gemeinsam haben. Dann sind schon P_1 und P_2 verschieden. Seien ferner v_1, v_2 die zugehörigen Orakel, also für welche $M_j(x)$ akzeptiert und Eigenschaften 1–4 erfüllen.

Wir werden nun ein t_{s-1}-valides Orakel $u \supsetneq w_{s-1}$ konstruieren welches mit v_1 auf P_1^{all} übereinstimmt, und welches mit v_2 auf P_2^{all} übereinstimmt. Außerdem wird es auf Ebene $\Sigma^{e(i)}$ nur Wörter aus U enthalten, woraus wir zeigen können dass u sogar ein geeignetes Orakel zur Zerstörung der UP-Machine in der Bearbeitung von Task τ_j^2 in Stufe s ist, ohne Beschränkungen bzgl. DisjNP-Zeugensprachen zu verletzen. Insbesondere akzeptiert dann auch $M_j^O(x)$ auf den zwei verschiedenen Rechenwegen P_1, P_2, was der Voraussetzung widerspricht.

Sei $Y = (P_1^{\text{yes}} \cup P_2^{\text{yes}}) \cap \Sigma^{e(i)}$, und $N = (P_1^{\text{no}} \cup P_2^{\text{no}}) \cap \Sigma^{e(i)}$. Wir zeigen $Y \cap N = \emptyset$. (Das soll uns helfen nachzuweisen, dass ein geeignetes u existieren kann.) Nehme an es gibt ein $q \in Y \cap N$ der Länge $e(i)$.

- Ist $q \notin U$ dann schon sofort dass $q \notin P^{\text{yes}}, P_2^{\text{yes}}$ was $q \in N$ widerspricht.
- Ist $q \in W$ dann gilt schon sofort dass $q \notin P^{\text{no}}, P_2^{\text{no}}$ was $q \in N$ widerspricht.
- Ist $q \in W'$ dann gilt schon sofort dass $q \notin P^{\text{yes}}, P_2^{\text{yes}}$ was $q \in Y$ widerspricht.
- Andernfalls ist $q \in U - (W \cup W')$, dann gilt $q \in P_1^{\text{yes}} \cap P_2^{\text{no}}$ oder $q \in P_2^{\text{yes}} \cap P_1^{\text{no}}$. In beiden Fällen hätten wir aber, dass P_1 und P_2 eine Orakelfrage der Länge $e(i)$ teilen, welche in $U - (W \cup W')$ liegt. Das widerspricht der urpsrünglichen Annahme.

Es gilt also $Y \cap N = \emptyset$. Wir beobachten außerdem dass $Y \subseteq U$ nach Punkt 5 der Definition gilt. Wähle ein $\alpha \in U - N$. Dieses existiert da $|N| \leq 2p_j(|x|) < 2^{e(i)-1} = |U|$ nach $(*)$. Sei nun u das Orakel was genau alle Wörter der Länge $\leq p_j(|x|)$ definiert sind, und

$$u(z) = \begin{cases} O(z) & \text{falls } |z| \neq e(i) \\ 1 & \text{falls } z = \alpha \\ 1 & \text{falls } z \in Y \\ 0 & \text{sonst,} \end{cases}$$

also wie $O^{\leq p_j(|x|)}$ aufgebaut ist, außer dass die Ebene $e(i)$ mit genau den Wörtern aus Y gefüllt wird. bzw. $u \cap \Sigma^{e(i)} = Y \cup \{\alpha\}$. Es ist leicht zu sehen dass $u \supsetneq w_{s-1}$. Beob. dass

$$u \cap N = \Sigma^{e(i)} \cap u \cap N = (Y \cup \{\alpha\}) \cap N = Y \cap N = \emptyset.$$

Das Orakel u stimmt mit v_1 auf P_1^{all} überein. Sei hierfür $q \in P_1^{\text{all}}$. Ist $|q| \neq e(i)$, dann gilt schon nach Definition $v_1(q) = O(q) = u(q)$. Sei daher im Folgenden $|q| = e(i)$. Ist $q \in P_1^{\text{yes}}$, dann auch $q \in v_1$. Außerdem dann auch $q \in Y$, daher $q \in u$. Ansonsten ist $q \in P_1^{\text{no}}$, dann auch $q \notin v_1$. Außerdem dann auch $q \in N$, daher $q \notin u$ nach obiger Beobachtung.

Auf symmetrische Weise stimmt u mit v_2 auf P_2^{all} überein. Wir zeigen nun dass u auch t_{s-1}-valide ist. Nach obiger Argumentation wäre dann u eine geeignete Erweiterung von w_{s-1} für welche $M_j^O(x)$ nicht mehr kategorisch ist, was der Wahl von $M_j^O(x)$ widerspricht.

Nach Konstruktion ist V1 erfüllt; V2, V3, V5 sind wegen $u \supsetneq w_{s-1}$ erfüllt. Angenommen V3 ist verletzt. Dies kann nur an der Ebene $e(i)$ liegen. Aber dann wäre $e(i) \in H_m$ mit $m = t_{s-1}(\tau_{a,b}^2)$ und $e(i) \notin M$; Widerspruch zur Einschränkung.

Angenommen V5 ist verletzt. Wieder kann das nur an der Ebene $e(i)$ liegen. Aber hier gilt $u \cap \Sigma^{e(i)} = Y \cap \{\alpha\} \subseteq U$ und nach Wahl von U ist damit $|\Sigma^{n-1}0 \cap w| = 0$ oder $|\Sigma^{n-1}1 \cap w| = 0$ aber nicht beides, ist ja $\alpha \in u \cap \Sigma^{e(i)}$.

Wir beweisen jetzt den zweiten Fall. Hier läuft die Konstruktion von u identisch, und wieder gilt dass u mit v_1 auf P_1^{all} übereinstimmt, sowie u mit v_2 auf P_2^{all} übereinstimmt. Nach obiger Argumentation wäre dann u eine geeignete Erweiterung von w_{s-1} für welche $M_j^O(x)$ nicht mehr kategorisch ist, was der Wahl von $M_j^O(x)$ widerspricht.

Nach Konstruktion ist V1 erfüllt; V2, V3, V5 sind wegen $u \supsetneq w_{s-1}$ erfüllt. Angenommen V5 ist verletzt. Dies kann nur an der Ebene $e(i)$ liegen. Aber dann wäre $e(i) \in H_m$ mit $m = t_{s-1}(\tau_{a,b}^3)$ und $e(i) \in M$; Widerspruch zur Einschränkung.

Angenommen V3 ist verletzt. Wieder kann das nur an der Ebene $e(i)$ liegen. Aber hier gilt $u \cap \Sigma^{e(i)} = Y \cap \{\alpha\}$ und nach Wahl von U ist damit $|\Sigma^n \cap w| > 0$, ist ja $\alpha \in u \cap \Sigma^{e(i)}$. □

Behauptung 2.7. P = UP *relativ zu* O.

Skizze. Sei $S \in \mathrm{UP}^O$. Es existiert nach Definition eine Maschine M_j mit $L(M_j)^O = S$. Wir zeigen für hinreichend lange x wie man $x \in S$ in Polyonmialzeit relativ zu O entscheiden kann.

Sei im Folgenden x hinreichend lange wie oben diskutiert, also für dieses $(*)$ mit eindeutigem i gilt, sowie w_{s-1} keine Wörter der Länge $e(i)$ definiert, wobei s die Stufe ist, bei der τ_j^2 betrachtet wurde. Sei wieder $M = \bigcup\{H_m \mid t_s(\tau_{a,b}^3) = m > 0\}$. Für diese Maschine M_j und eine solche Eingabe x gilt dann Behauptung 2.6.

Wir werden diese Eigenschaft ausnutzen und iterativ Mengen W, W' aufbauen, welche für die Berechnung $M_j^O(x)$ relevant ist, bis wir alle solchen relevanten Wörter gefunden haben. Betrachte dafür zunächst folgende Subroutine:

```
1  Function Search(U):
2      assert e(i) ∈ M ⟹ (U = Σ^{e(i)−1}0 ∨ U = Σ^{e(i)−1}1)
3      W ← ∅, W′ ← ∅
4      for k von 0 bis p_j(|x|) + 1 do
5          P ← eine (U, W, W′) respektierende akzeptierende Berechnung P von
              M_j auf x mit |(P^all ∩ U) − (W ∪ W′)| minimal, oder ⊥ falls keine
              existiert
6          if P = ⊥ then
7              return „x ∉ S“
8          end
9          if alle q ∈ P^all mit |q| = e(i) sind in W ∪ W′ then
10             return „x ∈ S“
11         end
12         foreach q ∈ P^all mit |q| = e(i) do
13             if q ∈ O then W ← W ∪ {q}
14             if q ∉ O then W′ ← W′ ∪ {q}
15         end
16     end
17     return „x ∉ S“
```

Es ist leicht zu sehen dass der Algorithmus eine polynomielle Laufzeitschranke einhält. Wir zeigen nun folgende Aussage: Wenn die Assertion in Z. 2 zutrifft dann macht der Algorithmus keinen falsch-positiv-Fehler. Falls zusätzlich $O \cap \Sigma^{e(i)} \subseteq U$, dann macht der Algorithmus keinen falsch-negativ-Fehler.

Wir beobachten die Invariante dass $W \subseteq O \cap \Sigma^{e(i)}$ und $W' \subseteq \overline{O} \cap \Sigma^{e(i)}$. Terminiert also der Algorithmus in Z. 10 mit „$x \in S$“ dann ist auch $x \in S$: Sei v das Orakel der (U, W, W') respektierenden akzeptierenden Berechnung P von $M_j(x)$. Es ist nun leicht zu sehen dass v und O auf P^{all} übereinstimmen. Damit gilt auch dass $M_j^O(x)$ akzeptiert und damit $x \in S$. Der Algorithmus macht also schon mal keinen falsch-positiv-Fehler.

Es verbleibt zu zeigen dass wenn $x \in S$ und $O \cap \Sigma^{e(i)} \subseteq U$ dann der Algorithmus auch in Z. 10 mit „$x \in S$“ terminiert, also keinen falsch-negativ-Fehler macht. Sei hierfür P^* der längste akzeptierende Rechenweg von $M_j^O(x)$. Beob. dass mit der o. g. Invariante sowie der Bedingung $O \cap \Sigma^{e(i)} \subseteq U$ gilt, dass P^* auch immer ein (U, W, W') respektierender akzeptierender Rechenweg ist. Damit ist die Bedingung in Z. 6 nie erfüllt. Wir zeigen nun noch, dass nach $\leq p_j(|x|) + 1$ vielen Iterationen auch die Bedingung in Z. 9 erfüllt ist. Hierfür zeigen wir, dass $|(P^{*\mathrm{all}} \cap U) - (W \cup W')|$ in jeder Iteration um ≥ 1 abnimmt. Da $|P^{*\mathrm{all}}| \leq p_j(|x|)$ ist nach $\leq p_j(|x|) + 1$ vielen Iterationen $|(P^{*\mathrm{all}} \cap U) - (W \cup W')| = 0$. Nach $\leq p_j(|x|) + 1$ vielen Iterationen wird also in Z. 5 eine Berechnung P ausgewählt, bei der alle $q \in P^{\mathrm{all}}$ mit $|q| = e(i)$ in $W \cup W'$ liegen. Dann ist die Bedingung in Z. 10 erfüllt und der Algorithmus terminiert akzeptierend.

Steht der Algorithmus in Z. 12, dann gilt sowohl für das ausgewählte P, als auch für P^* dass beide je eine (nicht notwendigerweise identische) Orakelfrage der Länge $e(i)$ enthalten, welche nicht in $W \cup W'$ liegt. (Andernfalls wäre P in Z. 5 anders ausgewählt worden.)

Sowohl P als auch P^* sind (U, W, W') respektierende akzeptierende Rechenwege. Nachdem die Assertion gilt, ist Behauptung 2.6 anwendbar. Also haben diese zwei Berechnungen eine identische Orakelfrage $q \in P^{\mathrm{all}} \cap P^{*\mathrm{all}} \cap \Sigma^{e(i)}$ gemeinsam, welche in $U - (W \cup W')$ liegt. Diese wird in den Zz. 12–15 irgendwann der Menge $W \cup W'$

hinzugefügt. Damit nimmt auch $|(P^{*\text{all}} \cap U) - (W \cup W')|$ um ≥ 1 ab, wie behauptet.

Betrachte nun folgenden Entscheidungsalgorithmus für $S = L(M_j^O)$.

```
18 if e(i) ∈ M then
19 |  if Search(Σ^{e(i−1)0}) = „x ∈ S“ then
20 |  |  return „x ∈ S“
21 |  else if Search(Σ^{e(i−1)1}) = „x ∈ S“ then
22 |  |  return „x ∈ S“
23 |  else
24 |  |  return „x ∉ S“
25 |  end
26 else
27 |  return Search(Σ^{e(i)})
28 end
```

Es ist leicht zu überprüfen dass in allen Fällen die Subroutine so ausgeführt wird dass die Assertion immer erfüllt ist. Ebenso ist leicht zu sehen, dass dieser Algorithmus in Polynomialzeit läuft.

Wir überprüfen nun Korrektheit in zwei Fällen. Im Fall $e(i) \notin M$ rufen wir die Subroutine mit $U = \Sigma^{e(i)}$ auf, und nach obiger Argumentation macht *Search* sowohl keinen falsch-positiv- also auch keinen falsch-negativ-Fehler. Der zurückgegebene Wert ist also korrekt.

Im Fall $e(i) \in M$ bekommen wir in beiden Aufrufen von *Search* zumindest keinen falsch-positiven Fehler. Wenn also in Zz. 20 oder 22 mit „$x \in S$“ terminiert wird, dann war auch $x \in S$. Fener wissen wir wegen $e(i) \in M$ und V6 dass entweder $O \cap \Sigma^{e(i)} \subseteq \Sigma^{e(i)-1}0$ oder $O \cap \Sigma^{e(i)} \subseteq \Sigma^{e(i)-1}1$. Wenn also $x \in S$, dann macht einer der Aufrufe von *Search* in Zz. 19 oder 21 keinen falsch-negativ-Fehler und die zugehörige If-Bedingung wird positiv ausfallen, und der Algorithmus terminiert mit „$x \in S$“. □

3 Orakelkonstruktion DisjNP, P = UP, und Q

Sei $e(0) = 2, e(i+1) = 2^{2^{e(i)}}$. Sei hier $\{H_m\}_{m\in\mathbb{N}}$ eine Familie von paarweise disjunkten, unendlichen Teilmenge von $e(\mathbb{N})$. (Ebenen H_m gehören zur Zeugensprache bzgl. DisjNP-Maschinenpaar M_a, M_b.) Starte mit PSPACE-vollständiger Menge C welche keine Wörter der Länge $e(\cdot)$ enthält. Definiere folgende Zeugensprachen:

$$A_m^O := \{0^n \mid n \in H_m, \text{existiert } x \in \Sigma^n \text{ mit } x \in O \text{ und } x \text{ endet mit } 0\}$$
$$B_m^O := \{0^n \mid n \in H_m, \text{existiert } x \in \Sigma^n \text{ mit } x \in O \text{ und } x \text{ endet mit } 1\}$$

Fakt: wenn $|O \cap \Sigma^{n-1}0| = 0$ oder $|O \cap \Sigma^{n-1}1| = 0$ für alle $n \in H_m$, dann $(A_m^O, B_m^O) \in \text{DisjNP}^O$.

Idee: erreiche entweder dass M_a, M_b nicht disjunkt akzeptieren (Task $\tau^1_{a,b}$), oder dass das Zeugenpaar (A_m, B_m) nicht auf $(L(M_a), L(M_b))$ reduzierbar ist (Task $\tau^1_{a,b,r}$ für Transduktor F_r).

Gleichzeitig versuchen wir für möglichst viele M_j erreichen, dass diese (a) nicht total sind (Task τ^2_j), und diese (b) nicht kategorisch sind (Task τ^3_j). Am Ende sind die verbleibenden totalen Maschinen M_j^O sehr speziell, denn sie sind auch für gewisse Teilmengen von O total. In Kombination mit dem Fakt dass $\text{P}^C = \text{PSPACE}^C$ können wir relevante Wörter in $O - C$ errechnen und so einen akzeptierenden Weg von $M_j^O(x)$ ausgeben – damit erzielen wir Q bzw. P = UP. Besonders aufgepasst muss hier auf den UP-Entscheidungsalgorithmus: im Korrektheitsbeweis müssen wir sicherstellen, dass höchstens polynomiell viele Wörter in eine Ebene gesetzt werden. Das ist etwas schwieriger, weil üblicherweise im Beweis bis zu $2 \cdot p_j(|x|)$ Wörter eingesetzt werden (Menge Y); und dieses Polynom ist kann nicht von der Eingabelänge allein beschränkt werden da j beliebig.

Sei wie üblich $t \in \mathcal{T}$ wenn der Definitionsbereich endlich ist, nur die Tasks der Form $\tau^1_{a,b}, \tau^2_j$ enthält, t diese Tasks auf $\mathbb{N}$ abbildet, und injektiv auf dem Support ist.

- V1 Wenn $x < |w|$ und $|x| \notin e(\mathbb{N})$, dann gilt $x \in w \iff x \in C$.
  (Orakel w und C stimmen auf Wörtern mit Länge $\neq e(\cdot)$ überein.)
- V2 Für alle $n = e(i)$ gilt $|w \cap \Sigma^n| \leq 2^c$ mit $c = |\{j \mid t(\tau^2_j) = 0\}|$. Ferner definiert w alle Wörter der Länge $e(c)$.
  (Auf den Ebenen der Länge $e(\cdot)$ sind exponeniell so viele Wörter wie Tasks τ^2_j „negativ“ behandelt werden. Wir werden sehen dass mit dieser Eigenschaft das Orakel dünn auf Ebenen der Länge $e(\cdot)$ ist.)
- V3 Wenn $t(\tau^1_j) = 0$, dann existiert ein z sodass $M_j^w(z)$ definitiv ablehnt.
  ($L(M_j) \neq \Sigma^*$ relativ zum finalen Orakel.)
- V4 Wenn $t(\tau^2_j) = 0$, dann existiert ein z sodass $M_j^w(z)$ definitiv auf zwei Rechenwegen akzeptiert.
  (M_j nicht kategorisch relativ zum finalen Orakel.)
- V5 Wenn $t(\tau^3_{a,b}) = 0$, dann existiert ein z sodass $M_a^w(z)$ und $M_b^w(z)$ definitiv akzeptieren.
  (Wenn $t(\tau^3_{a,b}) = 0$, dann $L(M_a) \cap L(M_b) \neq \emptyset$ relativ zum finalen Orakel.)
- V6 Wenn $0 < t(\tau^3_{a,b}) = m$, dann gilt für alle $n \in H_m$ dass $|\Sigma^{n-1}0 \cap w| = 0$ oder $|\Sigma^{n-1}1 \cap w| = 0$.
  (Wenn $0 < t(\tau^3_{a,b}) = m$, dann $(A_m, B_m) \in \text{DisjNP}$.)

Sei T eine abzählbare Aufzählung der o.g. Tasks sodass $\tau^3_{a,b,r}$ immer nach $\tau^3_{a,b}$ kommt.

[. . . Üblicher Text zur stufenweisen Erweiterung von w_s und t_s . . .]

Wir definieren nun Stufe $s > 0$, diese startet mit einem $t_{s-1} \in \mathcal{T}$ und eine t_{s-1}-validen Orakel w_{s-1} welche nun den kleinsten Task bearbeitet, welcher noch in T ist. Dieser wird unmittelbar nach der Bearbeitung aus T entfernt. In der Bearbeitung wird das Orakel strikt verlängert.

- τ^1_j: Setze $t' = t_{s-1} \cup \{\tau^1_j \mapsto 0\}$. Existiert ein t'-valides Orakel $v \supsetneq w_{s-1}$, dann setze $t_s := t'$ und $w_s := v$.

Ansonsten setze $t_s := t_{s-1}$ und setze $w_s := w_{s-1}y$ für geeignetes $y \in \{0,1\}$ sodass w_s auch t_s-valide ist. (Das ist möglich nach Behauptung 3.1.)

- τ_j^2: Setze $t' = t_{s-1} \cup \{\tau_j^2 \mapsto 0\}$. Existiert ein t'-valides Orakel $v \sqsupsetneq w_{s-1}$, dann setze $t_s := t'$ und $w_s := v$.

  Ansonsten setze $t_s := t_{s-1}$ und setze $w_s := w_{s-1}y$ für geeignetes $y \in \{0,1\}$ sodass w_s auch t_s-valide ist. (Das ist möglich nach Behauptung 3.1.)
- $\tau_{a,b}^3$: Setze $t' = t_{s-1} \cup \{\tau_{a,b}^3 \mapsto 0\}$. Existiert ein t'-valides Orakel $v \sqsupsetneq w_{s-1}$, dann setze $t_s := t'$ und $w_s := v$. Entferne außerdem alle Tasks der Form $\tau_{a,b,r}^2$ von T.

  Ansonsten wähle ein hinreichend großes $m \notin \mathrm{img}(t_s)$ sodass w_s kein Wort der Länge $\min H_m$ definiert. Setze $t_s := t_{s-1} \cup \{\tau_{a,b}^3 \mapsto m\}$; damit ist w_{s-1} auch t_s-valide. Setze $w_s := w_{s-1}y$ für geeignetes $y \in \{0,1\}$ sodass w_s auch t_s-valide ist. (Das ist möglich nach Behauptung 3.1.)
- $\tau_{a,b,r}^3$: Wir wissen dass $t_{s-1}(\tau_{a,b}^2) = m > 0$. Setze $t_s = t_{s-1}$ und wähle ein t_s-valides Orakel $w_s \sqsupsetneq w_{s-1}$ sodass bezüglich einem $n \in \mathbb{N}$ eine der folgenden Aussagen gilt:
  - $0^n \in A_m^v$ für alle $v \sqsupseteq w_s$ und $M_a(F_r(0^n))$ lehnt relativ zu w_s definitiv ab.
  - $0^n \in B_m^v$ für alle $v \sqsupseteq w_s$ und $M_b(F_r(0^n))$ lehnt relativ zu w_s definitiv ab.

  (Das ist möglich nach Behauptung 3.2.)

Behauptung 3.1. *Für jedes $t \in \mathcal{T}$ und jedes t-valide w existiert ein $b \in \{0,1\}$ sodass wb auch t-valide ist.*

Behauptung 3.2. *Die Bearbeitung eines Tasks $\tau_{a,b,r}^3$ ist möglich: gilt $t_{s-1}(\tau_{a,b}^3) = m > 0$, dann lässt sich w_{s-1} so zu t_{s-1}-validem $u \sqsupsetneq w_{s-1}$ erweitern, dass eine der o.g. Fälle eintritt.*

Hinweis. Unterschied ist V2. Trotzdem dürfen wir zwei Wörter in die Ebene einer Stufe $e(i)$ einsetzen, und verletzen dabei V2 nicht. □

Damit ist die Konstruktion möglich. Sei $O = \lim_{s\to\infty} w_s$.

Behauptung 3.3. *Kein Paar aus* DisjNP^O *ist* $\leq_{\mathrm{m}}^{\mathrm{pp}}$*-hart für* DisjUP.

Behauptung 3.4. *Sei M_j eine totale Maschine, d.h. $L(M^O) = \Sigma^*$. Es existiert eine Länge n mit folgender Eigenschaft: falls $T \subseteq O$ mit O auf Wörtern der Länge $\neq e(\cdot)$ und Wörtern $\leq n$ übereinstimmt, dann $L(M_j^T) = \Sigma^*$.*

Behauptung 3.5. *Das Orakel O ist dünn auf den Ebenen der Länge $e(i)$. Insbesondere gilt $|O \cap \Sigma^{e(i)}| \leq e(i)$ für alle i.*

Skizze. Sein eine Ebene $e(i)$ beliebig. Sei $c_k = |\{j \mid t_k(\tau_j^2) = 0\}|$. Nachdem in der Folge $c_0, c_1, c_2, \ldots$ die Terme immer um ≤ 1 ansteigen, existiert ein kleinstes s sodass $c_s = i$. Damit hat nach V2 das Orakel $w_s \sqsubsetneq O$ alle Wörter der Länge $c(i)$ definiert. Ferner gilt $|w_s \cap \Sigma^{e(i)}| \leq 2^{c_s}$. Also haben wir $|O \cap \Sigma^{e(i)}| = |w_s \cap \Sigma^{e(i)}| \leq 2^i \leq e(i)$. □

Behauptung 3.6. *Sei M_j eine totale Maschine, d.h. $L(M_j^O) = \Sigma^*$. Dann existiert eine Funktion $g \in \mathrm{FP}^O$ sodass $g(x)$ einen akzeptierenden Rechenweg von $M_j^O(x)$ ausgibt. Damit gilt nach Definition die Hypothese Q relativ zu O.*

Hinweis. Wie im vorigen Abschnitt. Analog gilt für die Menge an erfassten Orakelwörtern D nach V2: $\ell(D) \leq p_j(|x|) \cdot p_j(|x|) \cdot p_j(|x|)$ denn in den je $\leq p_j(|x|)$ Ebenen der Länge $e(i) \leq p_j(|x|)$ existieren nach Behauptung 3.5 höchstens $e(i) \leq p_j(|x|)$ Wörter der Länge $e(i) \leq p_j(|x|)$. Damit folgt auch, dass der Algorithmus nach höchstens polynomiell vielen Iterationen terminiert. □

Um UP-Sprache $S = L(M_j^O)$ in P zu entscheiden, gehen wir wie in vorigem Abschnitt vor. Sei s die Stufe bei der τ_j^1 betrachtet wurde. Sei $c = |\{j \mid t_{s-1}(\tau_j^2) = 0|$. Wir beschränken uns im Folgenden wieder auf Eingaben hinreichender Länge, sodass es für x ein eindeutiges i gibt, sodass

$$e(c+1) \leq e(i), \quad \leq e(i-1) \leq \log(|x|), \quad 2p_j(|x|) < 2^{e(i)-1} < e(i+1). \qquad (*)$$

Zusätzlich zur Einschränkung $(*)$ diskutieren wir ab jetzt nur noch Eingaben, für welche das Orakel w_{s-1} keine Wörter der Länge $e(i)$ definiert. Sei $M = \bigcup\{H_m \mid$

$t_s(\tau^3_{a,b}) = m > 0\}$ eine Menge an Ebenen, welche einer DisjNP-Zeugensprache *in Stufe* s zugewiesen ist.

Wir verfeinern die Definition einer (U, W, W') *respektierende akzeptierende Berechnung* P *von* M_j *auf Eingabe* x, und verlangen zusätzlich

6. $|v \cap \Sigma^{e(i)}| \leq 2^c$.

Behauptung 3.7. *Seien* P_1, P_2 *zwei* (U, W, W') *respektierenden akzeptierenden Berechnungen von* M_j *auf Eingabe* x*. Folgende Aussage gilt jeweils bezüglich dieser beiden Einschränkungen auf* U*:*

- $e(i) \in M$ *und* $U = \Sigma^{e(i)-1}0$ *oder* $U = \Sigma^{e(i)-1}1$*,*
- $e(i) \notin M$ *und* $U = \Sigma^{e(i)}$*.*

Wenn P_1^{all} *und* P_2^{all} *beide je eine (nicht notwendigerweise identische) Orakelfrage* q_1, q_2 *der Länge* $e(i)$ *enthalten, welche in* $U - (W \cup W')$ *liegt, dann haben diese zwei Berechnungen eine (identische) Orakelfrage* q *der Länge* $e(i)$ *gemeinsam, welche nicht in* $W \cup W'$ *liegt.*

Skizze. Wieder beweisen wir zunächst den ersten Fall, der zweite Fall ist noch leichter. Wir konstruieren ein u ähnlich wie im originalen Beweis, verzichten aber auf das zusätzliche Wort α, also sodass $u \cap \Sigma^{e(i)} = Y \subseteq U$.

Es gilt dass u mit v_1 auf P_1^{all} übereinstimmt, sowie u mit v_2 auf P_2^{all} übereinstimmt. Sei $t' = t_{s-1} \cup \{\tau^2_j \mapsto 0\}$. Wir zeigen dass u auch t'-valide ist Damit wäre u dann eine geeignete Erweiterung von w_{s-1} in Bearbeitung von Task τ^2_j, für welche $M_j^O(x)$ nicht mehr kategorisch ist, was der Wahl von $M_j^O(x)$ widerspricht.

Beob. dass $|P_1^{\text{yes}}|, |P_2^{\text{yes}}| \leq 2^c$, damit gilt $|u \cap \Sigma^{e(i)}| = |Y| \leq 2^{c+1}$. Aber nun gilt $c' = |\{j \mid t'(\tau^2_j) = 0\}| = c + 1$, damit $|u \cap \Sigma^{e(i)}| \leq 2^{c'}$. Außerdem definiert u nach Konstruktion alle Wörter der Länge $e(i) \geq e(c+1) = e(c')$ und u erfüllt damit auf jeden Fall V2. Es ist nun auch leicht zu sehen, dass V1, V3, V4, V5, V6 erfüllt sind, daher ist u wie behauptet t'-valide. □

Damit gilt mit gleichem Verfahren

Behauptung 3.8. P = UP *relativ zu* O*.*

4 Orakel mit DisjCoNP und alle Paare aus DisjNP sind P-separierbar (O_3)

Sei $e(0) = 2, e(i+1) = 2^{2^{e(i)}}$. (Doppelt exponentiell!) Sei hier $\{H_m\}_{m\in\mathbb{N}}$ eine Familie von paarweise disjunkten, unendlichen Teilmenge von $e(\mathbb{N})$. (Ebenen H_m gehören zur Zeugensprache bzgl. DisjCoNP-Maschinenpaar M_a, M_b.) Starte mit PSPACE-vollständiger Menge C welche keine Wörter der Länge $e(\cdot)$ enthält. Definiere folgende Zeugensprachen:

$$A_m^O := \{0^n \mid n \in H_m, \text{für alle } x \in \Sigma^n \text{ gilt } x \in O \to \ x \text{ endet mit } 0\}$$
$$B_m^O := \{0^n \mid n \in H_m, \text{für alle } x \in \Sigma^n \text{ gilt } x \in O \to \ x \text{ endet mit } 1\}$$

Fakt: $|O \cap \Sigma^n| \geq 1$ für alle $n \in H_m \implies (A_m^O, B_m^O) \in$ DisjCoNP.

Idee: erreiche entweder dass M_a, M_b nicht disjunkt ablehnen (Task $\tau^2_{a,b}$), oder dass das Zeugenpaar (A_m, B_m) nicht auf $(L(M_a), L(M_b))$ reduzierbar ist (Task $\tau^2_{a,b,r}$ für Transduktor F_r).

Gleichzeitig versuchen wir für möglichst viele Paare M_a, M_b zu erreichen, dass diese nicht disjunkt akzeptieren. (Task $\tau^1_{a,b}$) Am Ende sind die verbleibenden Maschinenpaare M_a^O, M_b^O sehr speziell, denn sie sind auch für gewisse Teilmengen von O kategorisch. In Kombination mit dem Fakt dass $\mathrm{P}^C = \mathrm{PSPACE}^C$ können wir relevante Wörter in $O - C$ errechnen und so $L(M_a^O)$ von $L(M_b^O)$ trennen.

Sei wie üblich $t \in \mathcal{T}$ wenn der Definitionsbereich endlich ist, nur die Tasks der Form $\tau^1_{a,b}, \tau^2_{a,b}$ enthält, t diese Tasks auf $\mathbb{N}$ abbildet, und injektiv auf dem Support ist.

Ein Orakel $w \in \Sigma^*$ ist t-valide wenn $t \in \mathcal{T}$ und folgendes gilt:

- V1 Wenn $x < |w|$ und $|x| \notin e(\mathbb{N})$, dann gilt $x \in w \iff x \in C$.
  (Orakel w und C stimmen auf Wörtern mit Länge $\neq e(\cdot)$ überein.)
- V2 Wenn $t(\tau^1_{a,b}) = 0$, dann existiert ein z sodass $M_a^w(z)$ und $M_b^w(z)$ definitiv akzeptieren.
  (M_a, M_b nicht disjunkt relativ zum finalen Orakel.)
- V3 Wenn $t(\tau^2_{a,b}) = 0$, dann existiert ein z sodass $M_a^w(z)$ und $M_b^w(z)$ definitiv ablehnen.
  (Wenn $t(\tau^2_{a,b}) = 0$, dann $(\overline{L(M_a)}, \overline{L(M_b)}) \notin$ DisjCoNP relativ zum finalen Orakel.)
- V4 Wenn $0 < t(\tau^2_{a,b}) = m$, dann gilt für alle $n \in H_m$: wenn w für alle Wörter der Länge n definiert ist, dann $|\Sigma^n \cap w| \geq 1$.
  (Wenn $0 < t(\tau^2_{a,b}) = m$, dann $(A_m, B_m) \in$ DisjCoNP.)

Sei T eine abzählbare Aufzählung der o.g. Tasks sodass $\tau^2_{a,b,r}$ immer nach $\tau^2_{a,b}$ kommt.

[. . . Üblicher Text zur stufenweisen Erweiterung von w_s und t_s . . .]

Wir definieren nun Stufe $s > 0$, diese startet mit einem $t_{s-1} \in \mathcal{T}$ und eine t_{s-1}-validen Orakel w_{s-1} welche nun den kleinsten Task bearbeitet, welcher noch in T ist. Dieser wird unmittelbar nach der Bearbeitung aus T entfernt. In der Bearbeitung wird das Orakel strikt verlängert.

- $\tau^1_{a,b}$: Setze $t' = t_{s-1} \cup \{\tau^1_{a,b} \mapsto 0\}$. Existiert ein t'-valides Orakel $v \supsetneq w_{s-1}$, dann setze $t_s := t'$ und $w_s := v$.

  Ansonsten setze $t_s := t_{s-1}$ und setze $w_s := w_{s-1}y$ für geeignetes $y \in \{0,1\}$ sodass w_s auch t_s-valide ist. (Das ist möglich nach Behauptung 4.1.)
- $\tau^2_{a,b}$: Setze $t' = t_{s-1} \cup \{\tau^2_{a,b} \mapsto 0\}$. Existiert ein t'-valides Orakel $v \supsetneq w_{s-1}$, dann setze $t_s := t'$ und $w_s := v$. Entferne außerdem alle Tasks der Form $\tau^2_{a,b,r}$ von T.

  Ansonsten wähle ein hinreichend großes $m \notin \mathrm{img}(t_s)$ sodass w_s kein Wort der Länge $\min H_m$ definiert. Setze $t_s := t_{s-1} \cup \{\tau^2_{a,b} \mapsto m\}$; damit ist w_{s-1} auch t_s-valide. Setze $w_s := w_{s-1}y$ für geeignetes $y \in \{0,1\}$ sodass w_s auch t_s-valide ist. (Das ist möglich nach Behauptung 4.1.)
- $\tau^2_{a,b,r}$: Wir wissen dass $t_{s-1}(\tau^2_{a,b}) = m > 0$. Setze $t_s = t_{s-1}$ und wähle ein t_s-valides Orakel $w_s \supsetneq w_{s-1}$ sodass bezüglich einem $n \in \mathbb{N}$ eine der folgenden Aus-

sagen gilt:

- $0^n \in A_m^v$ für alle $v \sqsupseteq w_s$ und $M_a(F_r(0^n))$ akzeptiert definitiv relativ zu w_s.
- $0^n \in B_m^v$ für alle $v \sqsupseteq w_s$ und $M_b(F_r(0^n))$ akzeptiert definitiv relativ zu w_s.

(Das ist möglich nach Behauptung 4.2.)

Behauptung 4.1. *Für jedes $t \in \mathcal{T}$ und jedes t-valide w existiert ein $b \in \{0,1\}$ sodass wb auch t-valide ist.*

Behauptung 4.2. *Die Bearbeitung eines Tasks $\tau_{a,b,r}^2$ ist möglich: gilt $t_{s-1}(\tau_{a,b}^2) = m > 0$, dann lässt sich w_{s-1} so zu t_{s-1}-validem $u \sqsupsetneq w_{s-1}$ erweitern, dass eine der o.g. Fälle eintritt.*

Damit ist die Konstruktion möglich. Sei $O = \lim_{s\to\infty} w_s$.

Behauptung 4.3. *Kein Paar aus* $\mathrm{DisjCoNP}^O$ *ist $\leq_{\mathrm{m}}^{\mathrm{pp}}$-vollständig.*

Wir wollen nun zeigen, dass wir disjunkte Paare von NP-Sprachen in P separieren können. Sei im Folgenden M_a, M_b ein komplementär akzeptierendes Paar an Maschine relativ zu O. Um die Sprachen zu trennen nutzen wir aus, dass $\mathrm{PSPACE}^C = \mathrm{P}^C$, um so iterativ eine Menge $D \subseteq O$ an Orakelwörtern der Länge $e(\cdot)$ aufbauen, welche für die Berechnungen $M_a(x), M_b(x)$ relevant ist, bis wir nach einigen Iterationen alle solchen relevanten Wörter gefunden haben. Wir beschränken uns im Folgenden auf Eingaben hinreichender Länge, sodass es für x ein eindeutiges i gibt, sodass

$$e(i-1) \leq \log(|x|) < e(i), \quad p_a(|x|) + p_b(|x|) < 2^{e(i)} < e(i+1). \tag{$*$}$$

Wir definieren Folgendes: Sei $j \in \{a,b\}$ Eine (W, W') *respektierende akzeptierende Berechnung P von M_j auf Eingabe x* ist ein akzeptierender Rechenweg P von $M_j(x)$ relativ zu einem Orakel $v \subseteq \Sigma^*$, wobei

$$W, W' \subseteq \Sigma^{e(i)}, \quad W \subseteq O, \quad W' \subseteq \overline{O}$$

und für v gilt:

1. v ist definiert für genau die Wörter der Länge $\leq p_j(|x|)$.
2. $v(q) = O(q)$ für alle q mit $|q| \neq e(i)$, wobei hier das i diejenige eindeutige Zahl ist für die obigen Ungleichungen $(*)$ bzgl. $|x|$ gelten.
3. $v(q) = 1$ für alle q mit $q \in W$.
4. $v(q) = 0$ für alle q mit $q \in W'$.

Sei P^{all} die Menge der Orakelfragen auf P, und sei $P^{\mathrm{yes}} = P^{\mathrm{all}} \cap v$, $P^{\mathrm{no}} = P^{\mathrm{all}} \cap \overline{v}$. Beob. dass das Ermitteln einer (W, W') respektierenden akzeptierenden Berechnung einfach in Polynomialzeit (abh. von $|x|$ und $\ell(W), \ell(W')$) relativ zu O möglich ist: insbesondere stimmt O mit C auf Wörtern der Länge $\neq e(\cdot)$ überein, und alle anderen Wörter der Länge $e(0), e(1), \ldots, e(i-1)$ können vorab mit Queries an O in Polynomialzeit erfragt werden. Entsprechende Belegungen von v für Wörter der Länge $e(i)$ können z.B. in PSPACE enumeriert werden.

Sei s die Stufe bei der $\tau_{a,b}^1$ betrachtet wurde. Zusätzlich zur Einschränkung $(*)$ diskutieren wir ab jetzt nur noch Eingaben, für welche das Orakel w_{s-1} keine Wörter der Länge $e(i)$ definiert.

Behauptung 4.4. *Seien P_a, P_b je (W, W') respektierenden akzeptierenden Berechnungen von M_a bzw. M_b auf Eingabe x. Wenn P_a^{all} und P_b^{all} beide je eine (nicht notwendigerweise identische) Orakelfrage q_a, q_b der Länge $e(i)$ enthalten, welche in $\Sigma^{e(i)} - (W \cup W')$ liegt, dann haben diese zwei Berechnungen eine (identische) Orakelfrage q der Länge $e(i)$ gemeinsam, welche nicht in $W \cup W'$ liegt.*

Skizze. Angenommen, dies gilt nicht, also sei x sowie $W, W' \subseteq \Sigma^{e(i)}$, $W \subseteq O$, $W' \subseteq \overline{O}$ gegeben. Seien außerdem P_a und P_b je zwei (W, W') respektierenden akzeptierenden Berechnungen von M_a, M_b auf Eingabe x, welche je eine Orakelfrage der Länge $e(i)$ enthalten, welche nicht in $W \cup W'$ liegt, aber keine Orakelfrage aus $\Sigma^{e(i)} - (W \cup W')$ gemeinsam haben. Dann sind schon P_a und P_b verschieden. Seien ferner v_a, v_b die zugehörigen Orakel, also für welche $M_j(x)$ akzeptiert und Eigenschaften 1–4 erfüllen.

Wir werden nun ein t_{s-1}-valides Orakel $u \sqsupsetneq w_{s-1}$ konstruieren welches mit v_a auf P_b^{all} übereinstimmt, und welches mit v_b auf P_b^{all} übereinstimmt. Außerdem wird es auf

Ebene $\Sigma^{e(i)}$ mindestens ein Wort enthalten, woraus wir zeigen können dass u sogar ein geeignetes Orakel zur Zerstörung dieses DisjNP-Machinenpaars in der Bearbeitung von Task $\tau^1_{a,b}$ in Stufe s ist, ohne Beschränkungen bzgl. DisjCoNP-Zeugensprachen zu verletzen. Insbesondere akzeptiert dann sowohl $M_a^O(x)$ als auch $M_b^O(x)$ was der Voraussetzung widerspricht.

Sei $Y = (P_a^{\text{yes}} \cup P_b^{\text{yes}}) \cap \Sigma^{e(i)}$, und $N = (P_a^{\text{no}} \cup P_b^{\text{no}}) \cap \Sigma^{e(i)}$. Wir zeigen $Y \cap N = \emptyset$. (Das soll uns helfen nachzuweisen, dass ein geeignetes u existieren kann.) Nehme an es gibt ein $q \in Y \cap N$ der Länge $e(i)$.

- Ist $q \in W$ dann gilt schon sofort dass $q \notin P_a^{\text{no}}, P_b^{\text{no}}$ was $q \in N$ widerspricht.
- Ist $q \in W'$ dann gilt schon sofort dass $q \notin P_a^{\text{yes}}, P_b^{\text{yes}}$ was $q \in Y$ widerspricht.
- Andernfalls ist $q \in W \cup W'$, dann gilt $q \in P_a^{\text{yes}} \cap P_b^{\text{no}}$ oder $q \in P_a^{\text{yes}} \cap P_b^{\text{no}}$. In beiden Fällen hätten wir aber, dass P_a und P_b eine Orakelfrage der Länge $e(i)$ teilen, welche in $W \cup W'$ liegt. Das widerspricht der urpsrünglichen Annahme.

Es gilt also $Y \cap N = \emptyset$. Wähle ein $\alpha \in \Sigma^{e(i)} - N$. Dieses existiert da $|N| \leq p_a(|x|) + p_b(|x|) < 2^{e(i)}$ nach $(*)$. Sei nun u das Orakel was genau alle Wörter der Länge $\leq p_j(|x|)$ definiert sind, und

$$u(z) = \begin{cases} O(z) & \text{falls } |z| \neq e(i) \\ 1 & \text{falls } z = \alpha \\ 1 & \text{falls } z \in Y \\ 0 & \text{sonst,} \end{cases}$$

also wie $O^{\leq p_j(|x|)}$ aufgebaut ist, außer dass die Ebene $e(i)$ mit genau den Wörtern aus Y gefüllt wird. bzw. $u \cap \Sigma^{e(i)} = Y \cup \{\alpha\}$. Es ist leicht zu sehen dass $u \supsetneq w_{s-1}$. Beob. dass

$$u \cap N = \Sigma^{e(i)} \cap u \cap N = (Y \cup \{\alpha\}) \cap N = Y \cap N = \emptyset.$$

Das Orakel u stimmt mit v_a auf P_a^{all} überein. Sei hierfür $q \in P_a^{\text{all}}$. Ist $|q| \neq e(i)$, dann gilt schon nach Definition $v_a(q) = O(q) = u(q)$. Sei daher im Folgenden $|q| = e(i)$. Ist $q \in P_a^{\text{yes}}$, dann auch $q \in v_a$. Außerdem dann auch $q \in Y$, daher $q \in u$. Ansonsten ist $q \in P_a^{\text{no}}$, dann auch $q \notin v_a$. Außerdem dann auch $q \in N$, daher $q \notin u$ nach obiger Beobachtung.

Auf symmetrische Weise stimmt u mit v_b auf P_b^{all} überein. Wir zeigen nun dass u auch t_{s-1}-valide ist. Nach obiger Argumentation wäre dann u eine geeignete Erweiterung von w_{s-1} für welche $M_a^O(x)$ und $M_b^O(x)$ akzeptieren, also nicht mehr disjunkt, was der Wahl von M_a, M_b widerspricht.

Nach Konstruktion ist V1 und V2 erfüllt; V3 ist wegen $u \supsetneq w_{s-1}$ erfüllt. Angenommen V4 ist verletzt. Wieder kann das nur an der Ebene $e(i)$ liegen. Aber hier gilt $|u \cap \Sigma^{e(i)}| = |Y \cap \{\alpha\}| \geq 1$. □

Behauptung 4.5. *Zu jedem disjunkten* NP*-Paar* (L_a, L_b) *existiert ein Separator aus* P.

Beweis. Sei $L_a, L_b \in \text{NP}^O, L_a \cap L_b = \emptyset$. Es existiert nach Definition ein Paar an Maschinen M_a, M_b mit $L(M_a^O) = L_a$, $L(M_b^O) = L_b$. Wir zeigen für hinreichend lange x wie man L_1 von L_2 in Polyonmialzeit relativ zu O trennen kann.

Sei im Folgenden x hinreichend lange wie oben diskutiert, also für dieses $(*)$ mit eindeutigem i gilt, sowie w_{s-1} keine Wörter der Länge $e(i)$ definiert, wobei s die Stufe ist, bei der $\tau^1_{a,b}$ betrachtet wurde.

Wir werden diese Eigenschaft ausnutzen und iterativ Mengen W, W' aufbauen, welche für die Berechnungen $M_a^O(x), M_b^O(x)$ relevant sind, bis wir alle solchen relevanten Wörter gefunden haben. Das machen wir je abwechselnd für $M_a^O(x)$ und $M_b^O(x)$. Betrachte dafür folgende Subroutine:

1 $W \leftarrow \emptyset$, $W' \leftarrow \emptyset$
2 **for** k *von* 0 *bis* $\max\{p_a(|x|), p_b(|x|)\} + 1$ **do**
3 $\quad P_a \leftarrow$ eine (W, W') respektierende akzeptierende Berechnung P von M_a auf x mit $|(P^{\text{all}} \cap \Sigma^{e(i)}) - (W \cup W')|$ minimal, oder $\perp$ falls keine existiert
4 $\quad P_b \leftarrow$ eine (W, W') respektierende akzeptierende Berechnung P von M_b auf x mit $|(P^{\text{all}} \cap \Sigma^{e(i)}) - (W \cup W')|$ minimal, oder $\perp$ falls keine existiert
5 $\quad$ **if** $P_a = \perp$ **then return** „$x \in L_b$“
6 $\quad$ **if** $P_b = \perp$ **then return** „$x \in L_a$“
$\quad$ *(ab hier sind P_a, P_b je zwei (W, W') respektierende akzeptierende Berechnungen)*
7 $\quad$ **if** *alle* $q \in P_a^{\text{all}}$ *mit* $|q| = e(i)$ *sind in* $W \cup W'$ **then**
8 $\quad\quad$ **return** „$x \in L_a$“
9 $\quad$ **end**
10 $\quad$ **if** *alle* $q \in P_b^{\text{all}}$ *mit* $|q| = e(i)$ *sind in* $W \cup W'$ **then**
11 $\quad\quad$ **return** „$x \in L_b$“
12 $\quad$ **end**
13 $\quad$ **foreach** $q \in P_a^{\text{all}} \cup P_b^{\text{all}}$ *mit* $|q| = e(i)$ **do**
14 $\quad\quad$ **if** $q \in O$ **then** $W \leftarrow W \cup \{q\}$
15 $\quad\quad$ **if** $q \notin O$ **then** $W' \leftarrow W' \cup \{q\}$
16 $\quad$ **end**
17 **end**
18 **return** „$x \notin L_a \cup L_b$“

Es ist leicht zu sehen dass der Algorithmus eine polynomielle Laufzeitschranke einhält. Wir beobachten die Invariante dass $W \subseteq O \cap \Sigma^{e(i)}$ und $W' \subseteq \overline{O} \cap \Sigma^{e(i)}$.

Wir zeigen zunächst dass der Algorithmus keine falsch-positiven Fehler macht. Für den ersten Fall nehme an dass der Algorithmus mit „$x \in L_a$“ terminiert aber es gilt $x \notin L_a$ und $x \in L_b$. Terminiert der Algorithmus in Z. 6, dann war $P_b = \perp$, was nach obiger Invariante bedeutet dass $M_b^O(x)$ ablehnt (denn sonst existiert immer eine (W, W') respektierende akzeptierende Berechnung); Widerspruch zur Annahme.

Terminiert der Algorithmus in Z. 8, können wir den Widerspruch $x \in L_a$ zeigen: Sei v das Orakel der (W, W') respektierenden akzeptierenden Berechnung P von $M_a(x)$. Es ist nun leicht zu sehen dass v und O auf P_a^{all} übereinstimmen. Damit gilt auch dass $M_a^O(x)$ akzeptiert und damit $x \in L_a$.

Der symmetrische Fall mit L_b läuft analog. Damit macht der Algorithmus also zumindest schon keine falsch-positiven Fehler.

Es verbleibt zu zeigen dass der Algorithus keine falsch-negativen Fehler macht. Wir zeigen dies für den Fall dass $x \in L_a$, der andere Fall $x \in L_b$ läuft analog. Sei hierfür P_a^* der längste akzeptierende Rechenweg von $M_a^O(x)$. Beob. mit obiger Invariante dass P_a^* auch immer ein (W, W') respektierender akzeptierender Rechenweg ist. Nachdem der Algorithmus keine falsch-positven Fehler macht, sind die Bedingungen in Zz. 5 und 10 nie erfüllt.

Wir zeigen nun, dass nach $\leq p_j(|x|) + 1$ vielen Iterationen auch die Bedingung in Z. 7 erfüllt ist. Hierfür zeigen wir, dass $|P_a^{*\text{all}} \cap \Sigma^{e(i)} - (W \cup W')|$ in jeder Iteration um ≥ 1 abnimmt. Da $|P_a^{*\text{all}}| \leq p_j(|x|)$ ist nach $\leq p_j(|x|) + 1$ vielen Iterationen $|P_a^{*\text{all}} \cap \Sigma^{e(i)} - (W \cup W')| = 0$. Nach $\leq p_j(|x|) + 1$ vielen Iterationen wird also in Z. 3 eine Berechnung P_a ausgewählt, bei der alle $q \in P_a^{\text{all}}$ mit $|q| = e(i)$ in $W \cup W'$ liegen. Dann ist die Bedingung in Z. 7 erfüllt und der Algorithmus terminiert akzeptierend.

Steht der Algorithmus in Z. 13, dann gilt sowohl für das ausgewählte P_b, als auch für P_a^* dass beide je eine (nicht notwendigerweise identische) Orakelfrage der Länge $e(i)$ enthalten, welche nicht in $W \cup W'$ liegt. (Andernfalls wäre P_a in Z. 3 anders ausgewählt worden.) Damit ist Behauptung 4.4 anwendbar. Also haben diese zwei Berechnungen eine identische Orakelfrage $q \in P_b^{\text{all}} \cap P_a^{*\text{all}} \cap \Sigma^{e(i)}$ gemeinsam, welche nicht in $W \cup W'$ liegt. Diese wird in den Zz. 13–16 dann auch irgendwann der Menge $W \cup W'$ hinzugefügt. Damit nimmt auch $|P_a^{*\text{all}} \cap \Sigma^{e(i)} - (W \cup W')|$ um ≥ 1 ab, wie behauptet. □

5 Generische Orakelkonstruktionen

Generische Orakel geben uns ein alternatives Framework, um unsere Orakelkonstruktionen zu beschreiben. Wie die sonst von Christian, Titus, Fabian, … (erstmal bei Dose und Glaßer 2019?) konstruierten Orakel setzen generische Orakel eine Form von „existentiell quantifizierter Erweiterung“ um (etwa: „existiert eine endliche Erweiterung dass die Maschine M die Eigenschaft X nicht erfüllt, erweitere genau so, ansonsten gilt für alle Erweiterungen $\neg X$ und hieraus ergibt sich die Möglichkeit, Eigenschaft Y zu erhalten“). Hinzu kommt, dass es oft möglich ist zu zeigen, dass eine bestimmte Eigenschaft (bspw. Kollaps oder Trennung von Klassen) relativ zu *jedem* generischen Orakel gilt. Das macht es einfacher, Orakel zu konstruieren, die mehrere Eigenschaften auf einmal erfüllen. Hinzu kommt, dass – im Gegensatz zur sonst bei uns üblichen stufenweisen Konstruktion – die generischen Orakelkonstruktionen die eigentliche *Konstruktion* im Beweis abstrahieren; die Beweise der Teilaussagen erfolgen ohne Mitdenken der sonst üblichen „zeitlichen Dimension“ einer stufenweisen Konstruktion. Auch fällt das „Buchhalten“ über die sonst übliche Funktion t weg, welche bei der Definition der t-Validität beteiligt war. Die generischen Konstruktionen sind sozusagen gedächtnislos.

Wir bauen hier auf Beschreibungen von Fortnow und Grochow (2011) auf. Diese vereinfachen ein allgemeineres Framework der Generizität von Fenner, Fortnow, Kurtz und Li (2003), auf welches im Folgenden auch Bezug genommen wird. Zur generellen Idee:

> Many oracle constructions proceed by finite extensions: at each stage of the construction, some requirement is to be satisfied (e.g. “the i-th polynomial-time machine does not accept some fixed relativizable language $L(O)$”), and we satisfy it by specifying the oracle on finitely many more strings, leaving those strings we have previously specified untouched. In this paper, a generic oracle is one built by finite extensions which also satisfies Murphy’s law: “anything which can happen will happen”. More prosaically, a generic oracle is built by interleaving all finite extension arguments that are “interleavable”. (Fortnow und Grochow 2011)

Das werden wir im Folgenden spezifizieren. Parallel hierzu erarbeiten wir exemplarisch das klassische Resultat $\mathrm{P} \neq \mathrm{NP}$ relativ zu einem (generischen) Orakel (Baker, Gill und Solovay 1975). Besonders berücksichtigt wurde, dass die Erarbeitung der Ergebnisse zu generischen Orakel ohne größeres Vorwissen erfolgt, und insbesondere sind alle Beweise frei von Topologie. Wie üblich identifizieren wir Wörter aus Σ^* mit Zahlen aus ω und andersherum. Orakel/Mengen beschreiben wir wie üblich auch mittels ihrer (totalen) charakteristischen Funktion $\omega \to \{0,1\}$.

Definition 5.1. (i) Eine *Bedingung* ist eine partielle Funktion von $\omega \to \{0,1\}$ mit endlichem Definitionsbereich.

(ii) Zwei Bedingungen σ, τ nennen wir *konsistent* wenn $\sigma \cup \tau$ auch eine Bedingung ist, d.h. $a \in \mathrm{dom}(\sigma) \cap \mathrm{dom}(\tau) \implies \sigma(a) = \tau(a)$.

(iii) Eine Bedingung τ *erweitert* eine Bedingung σ ($\tau \succeq \sigma$) falls $\mathrm{dom}(\tau) \supseteq \mathrm{dom}(\sigma)$ und $\sigma(a) = \tau(a)$ für alle $a \in \mathrm{dom}(\sigma)$. Wenn $\sigma \neq \tau$ schreiben wir auch $\tau \succ \sigma$. Beob. dass $\preceq$ eine Halbordnung auf den Bedingungen ist.

(iv) Wir überführen diese Notation auch auf Teilmengen von ω welche wir als totale Funktionen von $\omega \to \{0,1\}$ identifizieren. Wir schreiben also $A \succ \sigma$ falls $A(a) = \sigma(a)$ für alle $a \in \mathrm{dom}(\sigma)$.

Die Intuition hinter dem Begriff „erweitert“ ist, dass $\tau \succeq \gamma$ das abschließende Orakel $G \succ \tau$ vollständiger spezifiziert als γ. An geeigneter Stelle verstehen wir eine Bedingung γ auch als Menge $\{x \mid \gamma(x) = 1\}$ und schreiben z.B. $|\gamma \cap \Sigma^n|$ als die Anzahl der Wörter $x \in \Sigma^n$, für welche $\gamma(x) = 1$.

Definition 5.2. Ein *Begriff der Generizität* $\mathcal{G}$ ist eine nichtleere Klasse an Bedingungen mit folgenden Eigenschaften:

(i) (Generizität.) Für alle $\gamma \in \mathcal{G}$, alle $a \in \omega \setminus \mathrm{dom}(\gamma)$ existiert eine Bedingung $\gamma' \in \mathcal{G}$ mit $\gamma' \succ \gamma$ mit $a \in \mathrm{dom}(\gamma')$.

(ii) (Grundlegend.) Sind $\sigma_1, \sigma_2 \in \mathcal{G}$ konsistent, dann ist auch $\sigma_1 \cup \sigma_2 \in \mathcal{G}$.

Die Bedingungen $\gamma \in \mathcal{G}$ nennen wir auch *$\mathcal{G}$-Bedingungen*.

Ein einfacher Begriff der Generizität ist beispielsweise die *Cohen*-Generizität, welche *alle* Bedingungen (also finite partielle Funktionen) in einer Menge zusammenfasst. Dieser Begriff der Generizität ist für unser BGS-Beispiel ausreichend.

Ein $\mathcal{G}$-generisches Orakel ist – intuitiv gesprochen – das Ergebnis einer unendlich fortführenden stufenweisen Erweiterung einer Startbedingung, wobei sukzessiv alle möglichen erdenklichen „Eigenschaften“ abgearbeitet und erfüllt – *„erzwungen“* – werden, welche überhaupt erzwungen werden können. (Im Rahmen der BGS-Konstruktion wäre eine solche Eigenschaft beispielsweise „M_i entscheidet nicht die NP-Zeugensprache $L(O)$“.) Hierzu definieren wir formal, was mit „Eigenschaften“ und „erzwingen“ gemeint ist.

Sei im Folgenden $\mathcal{G}$ ein beliebiger aber fester Begriff von Generizität.

Definition 5.3 (Relativierte Peano-Arithmetik)**.** (i) Sei $\mathcal{L}_{\mathrm{PA}}[X]$ die Erweiterung der Sprache der Peano-Arithmetik $(=, +, \times, S, 0)$, mit einem zusätzlichen unären Prädikatsymbol X. Ohne Beschränkung sind die einzigen logischen Operatoren $\neg, \vee, \exists$. Für $n \in \omega$ bezeichnen wir mit $\overline{n}$ den Term $\underbrace{SS\cdots S}_{n}0$ in $\mathcal{L}_{\mathrm{PA}}[X]$.

(ii) Sei $\mathrm{sent}(\mathcal{L}_{\mathrm{PA}}[X])$ die Menge der Sätze in $\mathcal{L}_{\mathrm{PA}}[X]$.

(iii) Für $A \subseteq \omega$ sei $\omega[A]$ die Erweiterung des Standardmodells der Arithmetik für die Sprache $\mathcal{L}_{\mathrm{PA}}[X]$ in welcher das Prädikat X durch (totale Funktion) A interpretiert wird, das ist, der Satz $X(\overline{n})$ ist wahr genau dann wenn $A(n) = 1$. Für einen Satz $\varphi \in \mathrm{sent}(\mathcal{L}_{\mathrm{PA}}[X])$ meint $\omega[A] \vDash \varphi$ wie üblich, dass die Struktur $\omega[A]$ die Formel φ erfüllt (im Rahmen der üblichen Definition von Wahrheit/Erfüllung).

Die Sätze $\varphi \in \mathrm{sent}(\mathcal{L}_{\mathrm{PA}}[X])$ entsprechen genau unseren oben intuitiven „Eigenschaften“. Wir wollen, dass das abschließend konstruierte generische Orakel G alle für uns relevanten Sätze $\varphi \in \mathrm{sent}(\mathcal{L}_{\mathrm{PA}}[X])$ erfüllt, in dem Sinn dass $\omega[G] \vDash \varphi$. Sei für die BGS-Konstruktion z.B.

$$A_i : (\exists n)[M_i^X(0^n) = 0 \leftrightarrow (\exists x)[|x| = n \wedge X(x)]]$$

eine Familie an Sätzen aus $\varphi \in \mathrm{sent}(\mathcal{L}_{\mathrm{PA}}[X])$. (Nur) in diesem Kontext meinen wir mit $\{M_i\}_{i\in\omega}$ eine vollständige Aufzählung der deterministischen Polymomialzeit-Orakel-Turing-Maschinen. Wir wünschen uns nun vom entsprechenden generischen Orakel G, dass

$$\omega[G] \vDash A_i \text{ für alle } i \in \omega$$

denn dann folgt

$$\omega[G] \nvDash \text{„}\mathrm{P}^X = \mathrm{NP}^X\text{“},$$

oder weniger formell geschrieben, $\mathrm{P} \neq \mathrm{NP}$ relativ zu G.

Zur technischen Umsetzung führen wir den Begriff des Erzwingens bzw. Forcing ein. Diese soll uns als eine „vorläufige Approximation an Wahrheit“ (vgl. Fenner, Fortnow, Kurtz und Li 2003) dienen. Die $\mathcal{G}$-Forcing-Relation $\Vdash$ auf $\mathcal{G} \times \mathrm{sent}(\mathcal{L}_{\mathrm{PA}}[X])$ wird durch eine einfache Rekursion über die Struktur von Formeln definiert, die im Wesentlichen Tarskis Definition von Wahrheit entspricht, außer im Fall der Negation. Im Fall der Negation können wir die Definition grob so verstehen, dass $\neg\varphi$ erzwungen wird, wenn wir φ niemals durch Verfeinerung des Orakels erzwingen können.

Definition 5.4 (Forcing-Relation)**.** Die Variablen γ und τ erstrecken sich über $\mathcal{G}$. Es gilt:

$$\begin{aligned}
\gamma \Vdash \varphi &\iff \varphi \text{ atomar, } X \text{ kommt nicht in } \varphi \text{ vor, und } \omega \vDash \varphi,\\
\gamma \Vdash X(\overline{n}) &\iff (\forall A \succ \gamma).A(n) = 1,\\
\gamma \Vdash \varphi \vee \psi &\iff \gamma \Vdash \varphi \text{ oder } \gamma \Vdash \psi,\\
\gamma \Vdash (\exists x)\varphi &\iff \text{es existiert ein } a \in \omega \text{ sodass } \gamma \Vdash \varphi[x/\overline{a}],\\
\gamma \Vdash \neg\varphi &\iff (\forall \tau \succeq \gamma).\, \tau \nVdash \varphi.
\end{aligned}$$

Es ergeben sich schon aus dieser Definition drei kleine Beobachtungen, welche Forcing mit Wahrheit in Verknüpfung bringen:

Beobachtung 5.5. (i) *Gilt $\gamma \Vdash \varphi$ und $\sigma \succeq \gamma$, dann gilt auch $\sigma \Vdash \varphi$.*

(ii) *Wenn $(\forall A \succ \gamma).\omega[A] \vDash \varphi$, dann gilt $\gamma \Vdash \varphi$.*

(iii) *Wenn $\gamma \Vdash \varphi$ dann gilt $\gamma \Vdash \neg\neg\varphi$.*

Beweis. Jeweils Induktion über Formeln. □

Definition 5.6. Wir schreiben $G \Vdash \varphi$ wenn ein $\gamma \in \mathcal{G}$ existiert mit $G \succ \gamma$ und $\gamma \Vdash \varphi$.

Wir können jetzt die intuitive Definition eines generischen Orakels umsetzen. Ein generisches Orakel erfüllt alle Sätze $\varphi \in \mathrm{sent}(\mathcal{L}_{\mathrm{PA}}[X])$ für welche der der Begriff der Generizität „stark genug“ ist, diese zu erfüllen. Für jeden Satz $\varphi \in \mathrm{sent}(\mathcal{L}_{\mathrm{PA}}[X])$ gilt also intuitiv „Murphy's law“: entweder φ ist durch G erzwungen ($G \Vdash \varphi$), oder φ konnte ohnehin nie in der Konstruktion erzwungen werden ($G \Vdash \neg\varphi$). Nach der Definition müssen wir noch eine Verbindung zwischen Erzwingen und Wahrheit aufbauen.

Definition 5.7. Eine Menge $G \subseteq \omega$ ist *$\mathcal{G}$-generisch* wenn folgende Eigenschaften erfüllt sind:

(i) Für jeden Satz $\varphi \in \mathrm{sent}(\mathcal{L}_{\mathrm{PA}}[X])$ gilt $G \Vdash \varphi \vee \neg\varphi$, es existiert also ein $\gamma \in \mathcal{G}$ mit $G \succ \gamma$ und $\gamma \Vdash \varphi \vee \neg\varphi$.

(ii) Für alle $n \in \omega$ existiert ein $\gamma \in \mathcal{G}$ mit $G \succ \gamma$ und $n \in \mathrm{dom}(\gamma)$.

Lemma 5.8 (Existenz von generischen Mengen)**.** *Für jeden Begriff der Generizität $\mathcal{G}$, jedes $\gamma \in \mathcal{G}$ existiert eine $\mathcal{G}$-generische Menge $G \succ \gamma$. (Da $\mathcal{G}$ nichtleer, existiert also immer eine $\mathcal{G}$-generische Menge.)*

Beweis. Sei $\{\varphi_i\}_{i\in\omega}$ eine Aufzählung aller Sätze in $\mathcal{L}_{\mathrm{PA}}[X]$. Starte mit $\gamma_{-1} = \gamma$. Für jedes $i \geq 0$, gegeben γ_{i-1}, wähle ein $\sigma \in \mathcal{G}$ mit $\sigma \succeq \gamma_{i-1}$ und sodass $\sigma \Vdash \varphi_i \vee \neg\varphi_i$. Dieses σ existiert: angenommen kein $\sigma \in \mathcal{G}$, $\sigma \succeq \gamma_{i-1}$ existiert mit $\sigma \Vdash \varphi_i$. Dann gilt nach Definition $\gamma_{i-1} \Vdash \neg\varphi_i$; setze $\sigma = \gamma_{i-1}$.

Falls $i \notin \mathrm{dom}(\sigma)$, setze $\gamma_i \in \mathcal{G}$ mit $\gamma_i \succ \sigma$ und $i \in \mathrm{dom}(\gamma_i)$. Diese existiert nach 5.2(i). Andernfalls, setze $\gamma_i = \sigma$. In beiden Fällen haben wir $\gamma_i \Vdash \varphi_i \vee \neg\varphi_i$.

Wir definieren nun

$$G(n) = \gamma_n(n).$$

Wir haben immer $n \in \mathrm{dom}(\gamma_n)$ (sogar $0, 1, \ldots, n \in \mathrm{dom}(\gamma_n)$) und damit ist G wohldefiniert. Ferner gilt $G \succ \gamma_i$ für alle i: angenommen $G \not\succ \gamma_i$, dann existiert ein $k \in \mathrm{dom}(\gamma_i)$ und $\gamma_i(k) \neq G(k) = \gamma_k(k)$. Gleichzeitig ist aber $k \in \mathrm{dom}(\gamma_k)$ und entweder $\gamma_i \preceq \gamma_k$ oder $\gamma_k \preceq \gamma_i$. In beiden Fällen sind die beiden Bedingungen konsistent, und damit gilt $\gamma_i(k) = \gamma_k(k)$; Widerspruch zu oben. Also gilt 5.7(i).

Damit gilt auch 5.7(ii): Für beliebiges $n \in \omega$ existiert $\gamma_n \in \mathcal{G}$ mit $G \succ \gamma_n$ und $n \in \mathrm{dom}(\gamma_n)$. □

Lemma 5.9 (Erzwingen ist Wahrheit; Fenner, Fortnow, Kurtz und Li 2003)**.** *Sei G eine $\mathcal{G}$-generische Menge. Für alle $\varphi \in \mathrm{sent}(\mathcal{L}_{\mathrm{PA}}[X])$ gilt: $G \Vdash \varphi$ genau dann wenn $\omega[G] \vDash \varphi$.*

Beweis. Induktion über die Struktur der Formeln. Variablen γ, γ', τ gehen über $\mathcal{G}$. Variable A geht über $\{0,1\}^\omega$.

- Lemma klar wenn φ atomar ist und X nicht in φ vorkommt.
- Falls $\varphi = X(\overline{n})$ dann haben wir

$$\begin{aligned} G \Vdash X(\overline{n}) &\iff (\exists\gamma, \gamma \prec G).\, \gamma \Vdash X(\overline{n}) \\ &\iff (\exists\gamma, \gamma \prec G)(\forall A, A \succ \gamma),\, A(n) = 1 \\ &\iff G(n) = 1 \\ &\iff \omega[G] \vDash X(\overline{n}), \end{aligned}$$

  wobei die ersten zwei Äquivalenzen aus Definition folgen. Rückrichtung dritter Äquivalenz folgt aus der Existenz eines $\gamma \prec G$ mit $n \in \mathrm{dom}(\gamma)$ nach 5.7(ii).
- Lemma klar für Disjunktionen und Existenzquantor: Induktionsannahme ver-

wenden.

- Für Negationen gilt:

$$\begin{aligned} G \nVdash \neg\varphi &\implies (\forall \gamma', \gamma' \prec G).\, \gamma' \nVdash \neg\varphi \\ &\implies (\exists \gamma, \gamma \prec G).\, \gamma \Vdash \varphi \\ &\implies G \Vdash \varphi \implies \omega[G] \vDash \varphi. \end{aligned}$$

Die zweite Implikation gilt, denn für ein $\gamma \in \mathcal{G}, \gamma \prec G$ gilt nach Definition 5.7(i) dass $\gamma \Vdash \varphi \vee \neg\varphi$. Wenn also nach Voraussetzung für alle solche γ' schon $\gamma' \nVdash \neg\varphi$, dann muss $\gamma \Vdash \varphi$.

Für die andere Richtung gilt:

$$\begin{aligned} G \Vdash \neg\varphi &\implies (\exists \gamma, \gamma \prec G).\, \gamma \Vdash \neg\varphi \\ &\implies (\forall \gamma', \gamma' \prec G).\, \gamma' \nVdash \varphi \\ &\implies G \nVdash \varphi \implies \omega[G] \nvDash \varphi. \end{aligned}$$

Um die zweite Implikation zu sehen, nimm an dass eine $\mathcal{G}$-Bedingung $\gamma' \prec G$ existiert mit $\gamma' \Vdash \varphi$. Dann gilt $\gamma, \gamma' \prec G$ und die $\mathcal{G}$-Bedingungen γ, γ' sind konsistent. Nach Definition 5.2(ii) ist also $\tau = \gamma \cup \gamma'$ ist auch eine $\mathcal{G}$-Bedingung und es gilt sogar $\tau \prec G$. Da $\tau \succeq \gamma'$ und $\gamma' \Vdash \varphi$ haben wir nach Beobachtung 5.5(i) auch $\tau \Vdash \varphi$. Genauso haben wir dann aber auch $\tau \succeq \gamma$ und damit $\tau \Vdash \neg\varphi$. Aus Letzterem folgt insbesondere $\tau \nVdash \varphi$; Widerspruch zu Ersterem. Letzte Implikation ist Induktionsannahme. □

Die bisherigen Ergebnisse sind in dieser Form nicht sonderlich nützlich. (Wir wissen erst mal nur dass $G \Vdash \varphi \vee \neg\varphi$, also nach vorigem Satz $\omega[G] \vDash \varphi \vee \neg\varphi$ – eine Trivialität.) Es ist erst mal unklar, ob für gegebenen Satz φ die Konstruktion $G \Vdash \varphi$ oder $G \Vdash \neg\varphi$ erzwungen hat, also für welchen Satz φ der gewählte Begriff der Generizität „stark genug“ war, um φ zu erzwingen. Konkret für die BGS-Konstruktion: haben wir $G \Vdash A_i$ für alle $i \in \omega$?

Wir könnten jetzt die Konstruktion aus Satz 5.8 präzisieren. (Beim genauen Beobachten stellen wir z.B. fest dass $\gamma_i \Vdash \varphi_i$ genau dann wenn wir die vorhergehende Bedingung γ_{i-1} entsprechend so erweitern konnten um φ_i zu erzwingen.) Stattdessen können wir aber auch auf Eigenschaften unserer bisherigen Definitionen aufbauen, um folgende „extensionale“ Charakterisierung zu formulieren, für welche Sätze φ auch $G \Vdash \varphi$ (und damit $\omega[G] \vDash \varphi$) gilt. Damit abstrahieren wir von der konkreten Konstruktion.

Lemma 5.10. *Für alle $\gamma \in \mathcal{G}$ und alle $\varphi \in \mathrm{sent}(\mathcal{L}_{\mathrm{PA}}[X])$ gilt:*

$$\gamma \Vdash \neg\varphi \iff (\forall \mathcal{G}\text{-generischen } G \succ \gamma).\, \omega[G] \nvDash \varphi.$$

In anderen Worten: $\gamma \Vdash \neg\varphi$ genau dann wenn φ unter allen generischen Erweiterungen von γ als falsch interpretiert wird.

Beweis. Sei $\gamma \Vdash \neg\varphi$. Das bedeutet dass für alle $\mathcal{G}$-Bedingungen τ mit $\tau \succeq \gamma$ auch $\tau \nVdash \varphi$ gilt. Nimm jetzt an dass eine $\mathcal{G}$-generische Menge $G \succ \gamma$ existiert mit $\omega[G] \vDash \varphi$. Nach Lemma 5.9 gilt $G \Vdash \varphi$, nach Definition existiert ein $\sigma \prec G$ mit $\sigma \Vdash \varphi$.

Nun sind γ und σ konsistent, gilt ja $\sigma, \gamma \prec G$. Nach Definition 5.2(ii) ist also auch $\tau' = \gamma \cup \sigma \in \mathcal{G}$ und wir haben $\tau' \succeq \gamma$, nach Beobachtung 5.5(i) also $\tau' \Vdash \varphi$; Widerspruch zu oben.

Für die andere Richtung sei $\gamma \nVdash \neg\varphi$. Dann existiert eine Bedingung τ mit $\tau \succeq \gamma$ und $\tau \Vdash \varphi$. Nach Lemma 5.8 existiert eine $\mathcal{G}$-generische Menge $G \succ \tau$. Nach Lemma 5.9 gilt $\omega[G] \vDash \varphi$. □

Beachte dass $\omega[G] \nvDash \neg\varphi$ genau dann wenn $\omega[G] \vDash \varphi$. (Vgl. hierzu die Definition $\omega \vDash \neg\varphi$ genau dann wenn nicht $\omega \vDash \varphi$.) Wir haben dadurch

Korollar 5.11. *Für alle $\gamma \in \mathcal{G}$ und alle $\varphi \in \mathrm{sent}(\mathcal{L}_{\mathrm{PA}}[X])$ gilt: $\gamma \Vdash \neg\neg\varphi$ genau dann wenn $\omega[G] \vDash \varphi$ für alle $\mathcal{G}$-generischen $G \succ \gamma$.*

Wir formulieren nur die konkrete formale Übersetzung von „stark genug um φ zu erzwingen“: Für einen festen Begriff von Generizität $\mathcal{G}$ und einen festen Satz $\varphi \in \mathrm{sent}(\mathcal{L}_{\mathrm{PA}}[X])$ sagen wir dass $\mathcal{G}$ *stark genug ist, um φ zu erzwingen* wenn für jede $\mathcal{G}$-Bedingung γ ein eine weitere $\mathcal{G}$-Bedingung τ existiert mit $\tau \succeq \gamma$ und $\tau \Vdash \varphi$. Wir sehen,

dass diese Aussage sogar äquivalent zur Aussage „$\gamma \Vdash \neg\neg\varphi$ für alle $\mathcal{G}$-Bedingungen γ“ ist.

Beobachtung 5.12. $(\forall \gamma \in \mathcal{G}).\gamma \Vdash \neg\neg\varphi \iff$ *$\mathcal{G}$ ist stark genug, um φ zu erzwingen.*

Korollar 5.13. *Für alle $\varphi \in \mathrm{sent}(\mathcal{L}_{\mathrm{PA}}[X])$ gilt: Ist $\mathcal{G}$ stark genug um φ zu erzwingen, also zur Erinnerung*

$$(\forall \gamma \in \mathcal{G})(\exists \tau \in \mathcal{G}, \gamma \preceq \tau).\, \tau \Vdash \varphi, \text{ bzw. } (\forall \gamma \in \mathcal{G}).\, \gamma \Vdash \neg\neg\varphi,$$

dann ist $\omega[G] \vDash \varphi$ für alle $\mathcal{G}$-generischen Mengen G, d.h. unter allen $\mathcal{G}$-generischen Mengen wird der Satz φ als wahr interpretiert.

Beweis. Sei G eine beliebige $\mathcal{G}$-generische Menge. Es existiert immer eine Bedingung $\sigma \in \mathcal{G}$ mit $G \succ \sigma$; das folgt schon aus Definition 5.7(ii). Nach vorigem Korollar 5.11 gilt dann $\omega[G] \vDash \varphi$. □

In unserem BGS-Beispiel sehen wir, dass die Cohen-Generizität stark genug ist, um alle A_i zu erzwingen. Zur Erinnerung:

$$A_i : (\exists n)[M_i^X(0^n) = 0 \leftrightarrow (\exists x)[|x| = n \wedge X(x)]]$$

Sei hierfür im Folgenden $\mathcal{G}$ der Begriff der Cohen-Generizität (d.h. alle endlichen partiellen Funktionen). Sei nun $\gamma \in \mathcal{G}$ beliebig. Wir müssen nun ein $\tau \in \mathcal{G}$ angeben, welches γ erweitert und $\tau \Vdash A_i$. Zunächst wählen wir ein $n \in \omega$ sodass γ keine Wörter der Länge n definiert, und sodass $2^n > p_i(n)$. Definiere $A \succ \gamma$ mit

$$A(x) = \begin{cases} \gamma(x) & \text{wenn } x \in \mathrm{dom}(\gamma) \\ 0 & \text{sonst.} \end{cases}$$

Die Berechnung $M_i^A(0^n)$ ist definiert. Tatsächlich können wir nun eine partielle Funktion $\alpha \prec A$ finden, sodass $\mathrm{dom}(\alpha)$ genau mit Orakelfragen dieses Rechenwegs übereinstimmt. Dann ist leicht zu sehen: für jedes Orakel $B \succ \alpha$ gilt $M_i^B(0^n) = M_i^A(0^n)$. Im Übrigen ist $\alpha, \gamma \prec A$ und damit sind α und γ kompatibel.

Ist nun $M_i^A(0^n) = 1$, dann setze

$$\tau(x) = \begin{cases} \gamma(x) & \text{für } x \in \mathrm{dom}(\gamma) \\ \alpha(x) & \text{für } x \in \mathrm{dom}(\alpha) \\ 0 & \text{für } x \in \Sigma^n \\ \text{undef.} & \text{sonst.} \end{cases}$$

Dann ist τ offensichtlich eine Cohen-Bedingung, und es gilt für alle $B \succ \tau$ dass auch $B \succ \alpha$, also auch $\omega[B] \vDash M_i^B(0^n) = 1$. Trotzdem gilt $\omega[B] \nvDash (\exists x)[|x| = n \wedge X(x)]$. Wir haben also

$$(\forall B \succ \tau)\omega[G] \vDash A_i$$

und nach Beobachtung 5.5(ii) haben wir $\tau \Vdash A_i$.

Symmetrisch falls $M_i^A(0^n) = 0$. Sei $a \in \Sigma^n \setminus \mathrm{dom}(\alpha)$. Dieses existiert da $|\mathrm{dom}(\alpha)| \leq p_i(n) < 2^n = |\Sigma^n|$. Setze nun

$$\tau(x) = \begin{cases} \gamma(x) & \text{für } x \in \mathrm{dom}(\gamma) \\ \alpha(x) & \text{für } x \in \mathrm{dom}(\alpha) \\ 1 & \text{für } x = a \\ 0 & \text{für } x \in \Sigma^n,\, x \neq a \\ \text{undef.} & \text{sonst.} \end{cases}$$

Wieder ist τ eine Cohen-Bedingung, und es gilt für alle $B \succ \tau$ dass $\omega[B] \vDash M_i^B(0^n) = 0$. Trotzdem gilt $\omega[B] \vDash (\exists x)[|x| = n \wedge X(x)]$ und damit

$$(\forall B \succ \tau)\omega[B] \vDash A_i, \text{ und damit nach 5.5(ii) } \tau \Vdash A_i.$$

Wir wissen also nun, dass Cohen-Generizität stark genug ist, um alle A_i zu erzwingen. Damit gilt nach Korollar 5.13 schon mal das $\omega[G] \vDash A_i$ für alle Cohen-generische G, alle $i \in \omega$.

Sei nun ein G ein beliebiges Cohen-generisches Orakel G. Wir zeigen nun dass

$\mathrm{P}^G \neq \mathrm{NP}^G$. Angenommen $\mathrm{P}^G = \mathrm{NP}^G$, dann wird die Menge

$$L(G) = \{0^n \mid n \in \omega, \text{ex. } x \in \Sigma^n \text{ mit } x \in G\} \in \mathrm{NP}^G$$

auch in P^G entschieden. Dann existiert eine deterministische Polyzeitmaschine M_i mit $L(M_i^G) = L(G)$. Gleichzeitig haben wir $\omega[G] \vDash A_i$, also existiert in $n \in \omega$ sodass $M_i^G(0^n)$ ablehnt genau dann wenn ein Wort der Länge n in G liegt, bzw. $0^n \in L(G)$. Damit $L(M_i^G) \neq L(G)$; Widerspruch zur Annahme und es gilt $\mathrm{P}^G \neq \mathrm{NP}^G$. (Oder eben dass kein solches generisches G existieren kann – das aber ist ausgeschlossen nach Lemma 5.8.)

5.1 Relativierte Orakelkonstruktion

Die oben beschriebenen Definitionen und Ergebnisse bezüglich Forcing können einfach relativiert werden. Sei $B \subseteq \omega$. Forcing und Wahrheit relativ zu B definieren wir wie im unrelativierten Fall, aber erweitern die Sprache $\mathcal{L}_{\mathrm{PA}}[X]$ und das Standardmodell ω um ein zweiteres unäres Prädikat B zu $\mathcal{L}_{\mathrm{PA}}[B, X]$ bzw. ω^B mit Prädikat B interpretiert über Menge B. Dann können wir z.B. über „$\mathcal{G}$-generisch relativ zu B“ sprechen. Insbesondere überträgt sich das Hauptresultat vom vorigen Abschnitt: Ist $\mathcal{G}$ stark genug um den Satz $\varphi \in \mathrm{sent}(\mathcal{L}_{\mathrm{PA}}[B, X])$ zu erzwingen, dann ist $\omega^B[G] \vDash \varphi$ für alle $\mathcal{G}$-generischen Mengen G (welche auch existieren).

Alle folgenden Orakelkonstruktionen und Resultate, sowie die vorige BGS-Konstruktion, relativieren auf ein beliebiges zweites $B \subseteq \omega$. Mit dieser Erkenntnis lassen sich leichter komplexere Orakelkonstruktionen definieren, welche im Wesentlichen durch Kombination mehrerer Orakel entstehen. Ein Beispiel ist hier die klassische Konstruktion von Baker, Gill und Solovay eines Orakels relativ zu dem $\mathrm{P} = \mathrm{NP} \cap \mathrm{coNP} \neq \mathrm{NP}$: hierbei wird ein PSPACE-vollständiges Orakel B und ein zweites Orakel A kombiniert; in der Originalfassung zu $A \cup B$. (Das Orakel A trennt P von NP, ist aber so „einfach“ dass ein Maschinenpaar aus NP ∩ coNP eine relevante Portion von A entscheiden kann und dann mittels PSPACE-vollständigem B abfragen kann, ob entsprechende Eingabe akzeptiert wird.) Wir werden im Folgenden etwas Ähnliches mittels *Rerelativierungen* umsetzen.

Wir erweitern die Definition einer Orakel-Turing-Maschine M^A auf eine *2-Orakel-Turing-Maschine* $M^{A,B}$, welche anstelle eines Fragezustands einen zusätzlichen zweiten ausgezeichneten Fragezustand bekommt. (Erster Fragezustand fragt $q \in A$ ab, zweiter fragt $q \in B$ ab.) Aussagen können also nicht nur auf *ein* Orakel A relativiert werden, sondern auch auf *zwei* Orakel (A, B). Mit „Relativierung von Orakelkonstruktionen“ meine ich beispielsweise:

> Hat UP^A keine vollständige Menge, dann hat auch $\mathrm{UP}^{A,B}$ keine vollständige Menge (Klasse definiert über 2-Orakel-Turing-Maschinen), und dann hat auch $\mathrm{UP}^{A \oplus B}$ keine vollständige Menge.

Methodisch laufen die Orakelkonstruktionen durch Blackbox-Simulationen ab. Deshalb relativieren diese Konstruktionen: haben wir beispielsweise die relativierte Aussage

$$\omega[A] \vDash \text{„}\mathrm{UP}^X \text{ hat keine vollständige Menge“}$$

dann gilt auch die *Re*-Relativierung

$$\omega^B[A] \vDash \text{„}\mathrm{UP}^{X,B} \text{ hat keine vollständige Menge“}.$$

Eine auf zwei Orakel relativierte Klasse, wie z.B. $\mathrm{UP}^{A,B}$, kann nun mittels des Join-Operators $\oplus$ definiert als

$$A \oplus B = \{0x \mid x \in A\} \cup \{1x \mid x \in B\}$$

auch mittels *einem* kombinierten Orakel $A \oplus B$ erfasst werden. Wir haben z.B.

$$\mathrm{UP}^{A,B} = \mathrm{UP}^{A \oplus B},$$

da jede „unambigous“ 2-Orakel-Turing-Maschinen relativ zu (A, B) durch eine „unambigous“ 1-Orakel-Turing-Maschine relativ zu $A \oplus B$ effektiv simuliert werden kann, und umgekehrt. In unserem Beispiel hatten wir dass $\mathrm{UP}^{A,B}$ keine vollständige Menge hat, also hat auch $\mathrm{UP}^{A \oplus B}$ keine vollständige Menge.

5.2 Anwendung: Orakel relativ zu diesem UP und Q gilt

In diesem Abschnitt wollen wir das Orakel O_1 mit dem bisher präsentierten Framework rekonstruieren, und zeigen dass relativ zu einem generischen Orakel (bezüglich einem Begriff von Generizität welcher im Folgenden definiert wird) keine Menge vollständig für UP ist, und die Aussage Q gilt. Zur Erinnerung: Q sagt aus dass für alle (Poly-)Maschinen M mit $L(M) = \Sigma^*$ eine Funktion $g \in \mathrm{FP}$ existiert sodass $g(x)$ ein akzeptierender Rechenweg von $M(x)$ ist. In diesem Kontext meinen wir mit $\{M_i\}_{i\in\omega}$ eine vollständige Aufzählung der nichtdeterministischen Polymomialzeit-Orakel-Turing-Maschinen.

Wir nennen eine Berechnung $M_i^\sigma(x)$ *relativ zu einer Bedingung* σ *definiert*, wenn M_i auf Eingabe x hält, hierbei sämtliche Orakelfragen in $\mathrm{dom}(\sigma)$ liegen und dabei entsprechend σ beantwortet werden. Entsprechend können wir Aussagen wie „$M_i^\sigma(x)$ akzeptiert auf zwei Rechenwegen“ formulieren. Klar ist, dass sich solche Aussagen relativ zu σ auf Orakel $A \succ \sigma$ übertragen, d.h. wir hätten z.B. aus obiger Aussage dass auch $M_i^A(x)$ auf zwei Rechenwegen akzeptiert, für alle $A \succ \sigma$, und zwar immer mit den gleichen Orakelfragen $Q \subseteq \mathrm{dom}(\sigma)$.

In der klassischen Konstruktion haben wir mit einem PSPACE-vollständigen Orakel A gestartet, dieses mit weiteren Wörtern B augmentiert, sodass relativ zu $A \cup B$ einerseits UP, andererseits war B so einfach, dass eine totale Maschine M relevante Portionen von B rekonstruieren kann. Dann kann mit dem PSPACE-vollständigen A auch ein akzeptierender Rechenweg von $M(x)$ bestimmt werden.

Wir gehen im Folgenden einen ähnlichen Weg. Zunächst konstruieren wir ein spezielles generisches G für welches UP gilt. Das Orakel G ist dann auch wieder so einfach, dass $G \oplus A$ mit PSPACE-vollständigem A das Rekonstruieren der Rechenwege erlaubt, bzw. Aussage Q gilt, aber gleichzeitig noch UP relativ $G \oplus A$.

Definiere die Funktion tower mit

$$\mathrm{tower}(0) = 2, \quad \mathrm{tower}(k+1) = 2^{2^{\mathrm{tower}(k)}}.$$

Definiere polynomialzeit-berechenbare und -invertierbare Familie $\{H_i\}_{i\in\omega}$,

$$H_i \subseteq \{\mathrm{tower}(k) \mid k \in \omega\},$$

wobei alle Mengen der Familie paarweise disjunkt sind. Wir nennen eine Länge $n \in \omega$ *erlaubt* wenn ein $k \in \omega$ existiert mit $n = \mathrm{tower}(k)$, ansonsten *verboten*. Wir nennen eine Bedingung (bzw. Menge) $\sigma \subseteq \omega$ *lückenhaft* wenn für alle x mit $\sigma(x) = 1$ auch x eine erlaubte Länge hat. Definiere folgende Familie an Zeugensprachen $\{L_i(A)\}_{i,j\in\omega}$ abhängig von $A \subseteq \omega$:

$$L_i(A) = \{0^n \mid n \in H_{i,j},\ \text{es existiert ein } x \in \Sigma^n \text{ mit } x \in A\}$$

Wir haben $L_i(A) \in \mathrm{UP}^A$ wenn kein $n \in H_{i,j}$ existiert dass $x, y \in A$ für zwei verschiedene $x, y \in \Sigma^n$.

Das folgende Lemma setzt den zweiten, „algorithmischen“ Teil der Orakelkonstruktion um. Wie in der Literatur zu Forcing üblich, formulieren wir die Aussage mit Annahme P = PSPACE (und sprechen erst mal nicht von einem PSPACE-vollständigen Orakel), weil die Aussage ohnehin auf ein zweites Orakel relativiert.

Lemma 5.14 (vgl. Fortnow und Rogers 2002)**.** *Angenommen* P = PSPACE *und sei S ein dünnes und lückenhaftes Orakel. Nimm auch an dass für alle Maschinen M_j mit $L(M_j^S) = \Sigma^*$ gilt: für hinreichend langes z wird auch $M_j^T(z)$ akzeptieren, und das für jedes $T \subseteq S$ welches mit S auf allen Wörtern der Länge $< \mathrm{tower}(k)$ übereinstimmt, wobei $k \in \omega$ maximal sodass $\mathrm{tower}(k) \leq p_j(|z|)$. (In anderen Worten: S und T stimmen bis zur (exklusive) höchsten abfragbaren Stufe erlaubter Länge überein.)*

Dann gilt Q relativ zu S: zu jeder Maschine M_j mit $L(M_j^S) = \Sigma^*$ existiert ein $g \in \mathrm{FP}^S$ sodass, für alle x, die Ausgabe $f(x)$ ein akzeptierender Rechenweg von $M_j(x)$ ist.

Beweis. Sei M_j mit $L(M_j^S) = \Sigma^*$ gegeben, und sei nun n die oben spezifizierte Längenschranke.

Sei $z \in \Sigma^*$ eine Eingabe, und $k \in \omega$ die wie oben spezifizierte größte Zahl sodass $\mathrm{tower}(k) \leq p_j(|z|)$. Wir können k aus z effizient (in ab von $|z|$) berechnen und können daher ohne Beschränkung diese Zahl als dem Algorithmus gegeben ansehen. Wir werden im Folgenden nur hinreichend lange Eingaben z betrachten für welche $|z| \geq n$

und

$$\text{tower}(k-1) \leq \log|z| \leq \text{tower}(k).$$

Wir werden die in der Behauptung vorausgesetzten Eigenschaft ausnutzen und und iterativ eine Menge $T \subseteq S$ an Orakelwörtern aufbauen, welche für die Berechnung $M_j^S(x)$ relevant ist, bis wir alle solche relevanten Wörter gefunden haben. Wir starten hierbei mit der Menge $T_0 = S \cap \Sigma^{<\text{tower}(k)}$. (Wir zeigen später dass diese in P^S berechnet werden kann.)

Da uns T vorliegt, können wir sogar diese Orakelwerte in M_j hinein codieren, sodass $M_j'(x)$ äquivalent arbeitet, aber ohne Orakelfragen auskommt. Und da P = PSPACE können wir in FP auch einen akzeptierenden Rechenweg von $M_j'(x)$ bestimmen.

```
1  T_0 ← ⋃_{i=0}^{k-1} S ∩ Σ^{tower(i)}
2  T ← T_0
3  repeat
4  |  Sei α ein akzeptierender Rechenweg auf M_j^T(x) und Q die Menge an
   |   Orakelfragen
5  |  if existiert eine Frage q ∈ Q für die q ∈ S aber q ∉ T then
6  |  |  T ← T ∪ {q}
7  |  else
8  |  |  return α
9  |  end
10 end
```

Korrektheit: Es ist klar dass der in Zeile 1 berechnete Term tatsächlich $T_0 = S \cap \Sigma^{<\text{tower}(k)}$ entspricht, da die ausreicht die erlaubten Längen abzufragen. Beobachte nun die Invariante dass $T_0 \subseteq T \subseteq S$; damit existiert dann auch ein akzeptierender Rechenweg auf $M_j^T(x)$ nach Voraussetzung, und Zeile 4 ist wohldefiniert.

Terminiert nun der Algorithmus mit einem Rechenweg α, wissen wir auch dass für alle Orakelfragen $q \in Q$ entweder $q \in T$ gilt oder $q \notin S$ gilt. Zusammen mit der aus der Teilmengen-Invariante folgt, dass T mit S auf Q übereinstimmt, und auch $M_j^O(x)$ akzeptiert mit Rechenweg α.

Laufzeit: Wir zeigen dass der Algorithmus in polynomiell beschränkter deterministischer Zeit (abhängig von $|x|$) arbeitet, ohne Beschränkung aber nur für Eingaben die entsprechend der obigen Einschränkungen lang genug sind. Wie bereits argumentiert können wir k als gegeben verstehen. Zeile 1 macht für jede der $O(|z|)$ Teilmengen $S \cap \Sigma^{\text{tower}(i)}$ höchstens $O(|z|)$ viele Queries der Länge $O(|x|)$ denn wir haben ja $\text{tower}(i) \leq \log|z|$.

Wir zeigen, dass der Algorithmus nach höchstens polynomiell vielen Iterationen terminiert. Für jede Orakelfrage $q \in Q$ gilt, dass $|q| \leq p_j(|x|)$. Zusammen mit o.g. Invariante gilt $T \subseteq S \cap \Sigma^{\leq p_j(|x|)}$. Nun aber ist S dünn, also kann T nur polynomiell viele Elemente (abh. von $|z|$) enthalten.

Zeile 4 kann damit auch in polynomiell beschränkter deterministischer Zeit berechnet werden. Wie oben skizziert kann die Berechnung in deterministisch polynomieller Zeit abh. von $|z|$ und $\ell(T)$ simuliert werden. □

Wir definieren nun unseren Begriff der Generizität:

Definition 5.15. Eine Bedingung γ ist eine **UPC**-Bedingung, falls

(i) $\text{dom}(\gamma) = \Sigma^{\leq n}$ für ein $n \in \omega$, und

(ii) für alle $m \leq n$ gilt: $|\gamma \cap \Sigma^m| \leq 2$ falls $m \in H_i$ für ein $i \in \omega$, sonst $|\gamma \cap \Sigma^m| = 0$.

(iii) für alle $i \in \omega$ gilt: wenn ein $m \in H_i, m \leq n$ existiert mit $|\gamma \cap \Sigma^m| = 2$, dann existiert eine Eingabe y sodass $M_i^\gamma(y)$ auf zwei Rechenwegen akzeptiert. Dabei stellt die Berechnung nur Fragen der Länge $\leq 2^m$.

Der Begriff der Generizität **UPC** ist nun die Menge aller **UPC**-Bedingungen.

Es ist leicht zu sehen, dass **UPC** tatsächlich ein Begriff der Generizität im Sinne der Definition 5.2 ist. Die Eigenschaften von **UPC**-Bedingungen übertragen sich auf **UPC**-generische Orakel (routinierte Anwendung von Definition 5.2(i)):

Eigenschaft 5.16. *Sei G ein* **UPC***-generisches Orakel.*

(ii) *für alle $m \in \omega$ gilt: $|G \cap \Sigma^m| \leq 2$ falls $m \in H_i$ für ein $i \in \omega$, sonst $|G \cap \Sigma^m| = 0$.*

(iii) *für alle $i \in \omega$ gilt: wenn ein $m \in H_i$ existiert mit $|G \cap \Sigma^m| = 2$, dann existiert eine Eingabe y sodass $M_i^G(y)$ auf zwei Rechenwegen akzeptiert. Dabei stellt die Berechnung nur Fragen der Länge $\leq 2^m$.*

Wir wollen nun folgende Aussagen erzwingen:

$$R_{i,r}\colon \text{„}M_i^X \text{ akz. ein } x \text{ auf zwei Rechenwegen“} \vee \neg\text{„}L_i(X) \leq_{\mathrm{m}}^{\mathrm{pp}} L(M_i^X) \text{ via } T_r^X\text{“}.$$

Hierfür zeigen wir, dass unser Begriff der Generizität stark genug ist, um alle $R_{i,r}$ zu erzwingen, also $\gamma \Vdash \neg\neg R_{i,r}$ für alle $\gamma \in \mathbf{UPC}$. Dann sind wir auch schon mit dem UP-Teil unserer Orakelkonstruktion fertig. Beachte wie alle folgenden Aussagen relativieren.

Behauptung 5.17. *Angenommen $\gamma \Vdash \neg\neg R_{i,r}$ für alle $\gamma \in \mathbf{UPC}$. Dann existiert keine vollständige Menge für* UP *relativ zu jedem* **UPC***-generischem Orakel G.*

Beweis. Sei G ein beliebiges **UPC**-generisches Orakel. Angenommen es existiert eine vollständige Menge für UP^G, welche durch N_i^G entschieden wird. Insbesondere wird N_i^G keine Eingabe auf zwei Rechenwegen akzeptieren.

Wir haben $L_i(G) \not\leq_{\mathrm{m}}^{\mathrm{pp}} L(N_i^G)$; wir zielen auf einen Widerspruch und nehmen an, es existiert eine Reduktionsfunktion T_r^G. Es gibt ein $\gamma \in \mathbf{UPC}$ mit $G \succ \gamma$ (z.B. via Definition 5.7(ii)); nach Voraussetzung gilt $\gamma \Vdash \neg\neg R_{i,r}$. Nach Korollar 5.11 gilt also insbesondere für unser $G \succ \gamma$ dass $\omega[G] \vDash \varphi_{i,r}$. Da nach Annahme aber $\omega[G] \nvDash$ „M_i^X akz. ein x auf zwei Rechenwegen“ haben wir

$$\omega[G] \vDash \neg\text{„}L_i(X) \leq_{\mathrm{m}}^{\mathrm{pp}} L(M_i^X) \text{ via } T_r^X\text{“}.$$

Das bedeutet aber genau dass T_r^G eben nicht die Reduktion von $L_i(G)$ auf $L(M_i^G)$ leistet; Widerspruch wie gewünscht.

Wir zeigen nun dass $L_i(G) \in \mathrm{UP}^G$. Dann sind wir auch schon fertig, denn dann ist $L(M_i^G)$ nicht vollständig. Angenommen $L_i(G) \notin \mathrm{UP}^G$. Dann existiert ein $m \in H_i$ und $|G \cap \Sigma^m| = 2$. Dann gilt mit Eigenschaft 5.16(iii) sofort, dass auch $M_i^G(y)$ auf zwei Rechenwegen akzeptiert. Widerspruch zur Wahl von M_i. □

Wir zeigen nun wie angekündigt, dass unser Begriff der Generizität stark genug ist, alle $R_{i,r}$ zu erzwingen.

Behauptung 5.18. *Sei $\gamma \in \mathbf{UPC}$, $i, r \in \omega$ beliebig. Es gilt $\gamma \Vdash \neg\neg R_{i,r}$.*

Beweis. Sei $\gamma \in \mathbf{UPC}$ beliebig. Wir zeigen dass ein $\tau \in \mathbf{UPC}, \tau \succeq \gamma$ existiert mit $\tau \Vdash \varphi$; das ist ausreichend um $\gamma \Vdash \neg\neg R_{i,r}$ zu zeigen. Im Wesentlichen implementieren wir genau die selbe Idee wie in der sonst üblichen Stufenkonstruktion: entweder es existiert eine „kompatibele“ Erweiterung $\tau \succeq \gamma$ sodass M_i auf zwei Rechenwegen akeptiert. Ansonsten ist M_i „inhärent UP“; gilt nun die Reduktion muss aber M_i auch gleichzeitig sensitiv gegenüber jedem Wort in einer Ebene von Wörtern gleicher Länge sein. Das lässt sich ausnutzen um zu zeigen dass M_i auf zwei Rechenwegen akeptieren muss, was Wahl von M_i widerspricht.

Wähle ein $n \in H_i$ hinreichend groß sodass γ kein Wort der Länge n definiert, und sodass $2^n > p_i(p_r(n))$. Definiere nun für $S \subseteq \Sigma^n$ die Bedingung σ_S mit

$$\sigma_S(x) = \begin{cases} \gamma(x) & \text{falls } x \in \mathrm{dom}(\gamma), \\ S(x) & \text{falls } |x| = n, \\ 0 & \text{falls } n < |x| \leq 2^n, \\ \text{undef.} & \text{sonst.} \end{cases}$$

Wir haben $\sigma_S \succeq \gamma$. Es gilt $\sigma_\emptyset \in \mathbf{UPC}$ und $\sigma_{\{a\}} \in \mathbf{UPC}$ für alle $a \in \Sigma^n$: (i) Offenbar $\mathrm{dom}(\sigma_S) = \Sigma^{\leq p_i(p_r(n))}$. (ii) Wenn $\sigma_S(x) = 1, x \notin \mathrm{dom}(\gamma)$ dann nach Definition $x \in S$ und damit $|x| = n \in H_i$. Auch (iii) ist nur verletzt wenn verschiedene $a, b \in \mathrm{dom}(\sigma_S) \setminus \mathrm{dom}(\gamma)$ existieren mit $\sigma_S(a) = \sigma_S(b) = 1$, dann aber $a, b \in S$ aber wir haben $S = \emptyset$ oder $S = \{a\}$.

Der weitere Beweis erfolgt nun in Fallunterscheidung. Nehme für den ersten Fall an dass

$$\sigma_\emptyset \Vdash \neg\text{„}L_i(X) \leq_{\mathrm{m}}^{\mathrm{pp}} L(M_i^X) \text{ via } T_r^X\text{“}$$

oder für ein $a \in \Sigma^n$

$$\sigma_{\{a\}} \Vdash \neg„L_i(X) \leq_{\mathrm{m}}^{\mathrm{pp}} L(M_i^X) \text{ via } T_r^X“.$$

Dann sind wir fertig: für eines dieser σ_S gilt $\sigma_S \Vdash R_{i,r}$ und setze $\tau = \sigma_S$.

Für den anderen Fall gilt

$$\sigma_\emptyset \nVdash \neg„L_i(X) \leq_{\mathrm{m}}^{\mathrm{pp}} L(M_i^X) \text{ via } T_r^X“ \text{ und } \sigma_{\{a\}} \nVdash \neg„L_i(X) \leq_{\mathrm{m}}^{\mathrm{pp}} L(M_i^X) \text{ via } T_r^X“$$

für alle $a \in \Sigma^n$. Wir wollen hieraus ableiten dass ein $\tau \succeq \gamma$ existiert sodass

$$\tau \Vdash (\exists z)[„M_i^X(z) \text{ akz. auf zwei Rechenwegen“}] \text{ und damit } \tau \Vdash R_{i,r}.$$

Nach Annahme und Lemma 8 existiert ein $A \succ \sigma_\emptyset \succeq \gamma$ mit $\omega[A] \vDash „L_i(X) \leq_{\mathrm{m}}^{\mathrm{pp}} L(M_i^X) \text{ via } T_r^X“$. Da $0^n \notin L_i(A)$ gilt also $N_i^A(T_r^A(0^n)) = 0$. Also existiert ein kleinstes $\alpha \prec A$ sodass auch $N_i^\alpha(T_r^\alpha(0^n)) = 0$ definiert (fixiere die Orakelfragen); in anderen Worten

$$N_i^B(T_r^B(0^n)) = 0 \text{ für alle } B \succ \alpha. \tag{1}$$

Beobachte dass α und $\sigma_\emptyset$ (und damit auch γ) kompatibel sind. Insbesondere gilt $\max(\mathrm{dom}(\alpha)) \leq p_i(p_r(n))$ und damit $\mathrm{dom}(\alpha) \subseteq 2^{p_i(p_r(n))} = \mathrm{dom}(\sigma_\emptyset)$ also auch $\alpha \preceq \sigma_\emptyset$.

Auf gleiche Weise sehen wir, für je ein $a \in \Sigma^n$, ein kleinstes β_a existiert sodass $M_i(T_r(0^n)) = 1$ relativ zu β_a (β_a fixiert alle Orakelfragen), also wieder

$$N_i^B(T_r^B(0^n)) = 1 \text{ für alle } B \succ \beta_a \text{ mit Orakelfragen } \mathrm{dom}(\beta_a). \tag{2}$$

Wieder gilt $\mathrm{dom}(\beta_a) \subseteq 2^{p_i(p_r(n))}$, und $\beta_a \preceq \sigma_{\{a\}}$ und sogar $|\mathrm{dom}(\beta_a)| \leq p_i(p_r(n))$.

Ein kombinatorisches Standardargument zeigt aus Letzterem dass nun zwei unterschiedliche $a, b \in \Sigma^n$ existieren mit $a \notin \mathrm{dom}(\beta_b)$, $b \notin \mathrm{dom}(\beta_a)$. Fixiere diese zwei a, b. Wir zeigen nun dass $\sigma_{\{a,b\}} \succeq \beta_a$ und $\sigma_{\{a,b\}} \succeq \beta_b$. Für ersteres erinnern wir uns daran dass $\beta_a \preceq \sigma_{\{a\}}$, sowie

$$\mathrm{dom}(\beta_a) \subseteq \mathrm{dom}(\sigma_{\{a\}}) = \mathrm{dom}(\sigma_{\{a,b\}}).$$

Dass $\sigma_{\{a,b\}}$ also den Definitionsbereich von β_a erweitert haben wir damit also gezeigt. Wir müssen nun noch zeigen, dass für $z \in \mathrm{dom}(\beta_a)$ die Funktionen β_a und $\sigma_{\{a,b\}}$ gleiche Bilder haben: wir haben

$$\beta_a(z) = \sigma_{\{a\}}(z) = \sigma_{\{a,b\}}(z),$$

erste Gleichung Kompatibilität, zweite Gleichung klar aus Definition da $z \neq b$ nach Wahl von $z \in \mathrm{dom}(\beta_a)$ und Wahl von β_a mit $b \notin \beta_a$. Zweite Aussage $\sigma_{\{a,b\}} \succeq \beta_b$ geht analog.

Wenden wir nun (2) je einmal mit β_a und mit β_b an sehen wir, dass für alle $C \succ \sigma_{\{a,b\}}$ die Berechnung $N_i^C(T_r^C(0^n)) = 1$ auf zwei Rechenwegen akzeptiert, je eine mit Orakelfragen $\mathrm{dom}(\beta_a)$ und eine mit Orakelfragen $\mathrm{dom}(\beta_b)$. Wir zeigen später dass diese zwei Rechenwege nicht gleich sind. Damit haben wir dann

$$(\forall C \succ \sigma_{\{a,b\}}). \quad \omega[C] \vDash (\exists z)[„M_i^X(z) \text{ akz. auf zwei Rechenwegen“}],$$

(der implizite $\exists$-Quantor ist erfüllt für Eingabe $z = T_r^X(0^n)$) nach Beobachtung 5.5(ii) also

$$\sigma_{\{a,b\}} \Vdash (\exists z)[„M_i^X(z) \text{ akz. auf zwei Rechenwegen“}]$$

und damit ist mit $\tau = \sigma_{\{a,b\}}$ auch $\tau \Vdash R_{i,r}$ wie gewünscht.

Wir zeigen nun dass (für jedes festes C) die beiden oberen Rechenwege nicht gleich sind. Wir erreichen das, in dem wir $a \in \mathrm{dom}(\beta_a)$ zeigen, denn nach Wahl gilt $a \notin \mathrm{dom}(\beta_b)$ und die zwei Rechenwege stellen unterschiedliche Orakelfragen.

Um ersteres nun zu zeigen nimm an dass auch $a \notin \mathrm{dom}(\beta_a)$. Unter dieser Annahme sind β_a und α kompatibel: Wir erinnern uns dass

$$\alpha \preceq \sigma_\emptyset, \quad \beta_a \preceq \sigma_{\{a\}}.$$

Sei $z \in \mathrm{dom}(\alpha) \cap \mathrm{dom}(\beta_a)$. Es gilt insbesondere $z \neq a$ und damit

$$\alpha(z) = \sigma_\emptyset(z) = \sigma_{\{a\}} = \beta_a(z),$$

wobei zweite Gleichung klar aus Definition von σ da $z \neq a$. Wähle nun ein beliebiges

Orakel $B \succ \alpha \cup \beta_a$. Wir haben nun

$$M_i^B(T_r^B(0^n)) = 0 \text{ nach (1), und } M_i^B(T_r^B(0^n)) = 1 \text{ nach (2).}$$

Widerspruch, also gilt $a \notin \mathrm{dom}(\beta_a)$ und die obigen zwei Rechenwege können nicht gleich sein, wie gewünscht. □

Jedes **UPC**-generischem Orakel G erfüllt die in Lemma 5.14 genannte Voraussetzung. Im Kontext von generischen Orakelkonstruktionen wäre das eine Eigenschaft, die sich sozusagen kostenlos aus der Konstruktion ergibt, ohne sie explizit in den eigentlichen Konstruktionsprozess zu benennen.

Behauptung 5.19. *Sei G ein* **UPC***-generischem Orakel. Das Orakel G ist dünn, es ist lückenhaft, und für alle M_j mit $L(M_j^G) = \Sigma^*$ gilt: für hinreichend lange z wird auch $M_j^T(z)$ akzeptieren, und das für jedes $T \subseteq G$ welches mit G auf allen Wörtern der Länge $< \mathrm{tower}(k)$ übereinstimmt, wobei $k \in \omega$ maximal sodass $\mathrm{tower}(k) \leq p_j(|z|)$.*

Beweis. Dünnheit und Lückenhaftigkeit folgen direkt aus Eigenschaft 5.16(ii). Wir argumentieren nun für die zweite Teilaussage. Sei M_j mit $L(M_j^G) = \Sigma^*$ eine feste Maschine. Betrachte Aussage Q_j mit

$$Q_j : (\forall x)\text{„}M_j^X(x) \text{ akzeptiert“}.$$

Es gilt nach Lemma 5.7(i) dass ein $\gamma \in \mathbf{UPC}$ existiert mit $G \succ \gamma$ und $\gamma \Vdash \neg Q_j \vee \neg\neg Q_j$.

Intuitiv haben wir hier nun eine Situation ganz ähnlich zur der sonst üblichen Stufenkonstruktion: entweder $\gamma \Vdash \neg Q_j$ und M_j^G akzeptiert nicht alle Eingaben, und wir sind fertig. Ansonsten $\gamma \Vdash \neg\neg M_j$ und M_j ist „inhärent total“; die Totatlität von M_j kann durch keine „gültige“ Erweiterung $\tau \succeq \gamma$ zerstört werden. Würde jetzt also für geeignetes T die Maschine $M_j^T(z)$ ablehnen, können wir T zurechtstucken und eine Bedingung $\tau \succeq \gamma$ konstruieren, welche die Totalität von M_j zerstört. Das widerspricht dann aber $\gamma \Vdash \neg\neg Q_j$. Diese Idee werden wir im Folgenden konkretisieren.

Wir erinnern uns dass entweder $\gamma \Vdash \neg Q_j$ oder $\gamma \Vdash \neg\neg Q_j$. Im ersteren Fall gilt nach Lemma 5.10 insbesondere für unser vorliegendes **UPC**-generisches G die Aussage $\omega[G] \nvDash Q_j$. Also lehnt $M_j^G(x)$ für eine geeignete Eingabe x ab; das widerspricht der oben genannten Wahl von M_j.

Es gilt also $\gamma \Vdash \neg\neg Q_j$. Dann aber haben wir nach Korollar 5.11 die Aussage dass

$$(\forall \textbf{UPC}\text{-generischen } G' \succ \gamma). \quad \omega[G'] \vDash (\forall x)\text{„}M_j^X(x) \text{ akzeptiert“}. \tag{3}$$

Da γ eine **UPC**-Bedingung ist existiert ein $n \in \omega$ sodass $\mathrm{dom}(\gamma) = \Sigma^{\leq n}$. Fixiere dieses n. Wir betrachten nun nur noch Eingaben z, welche hinreichend lang sind sodass $\mathrm{tower}(k-1) \geq n$ (wobei wie in der Aussage der Behauptung das k maximal ist sodass $\mathrm{tower}(k) \leq p_j(|z|)$).

Sei nun ein solches z gegeben und entsprechendes $T \subseteq G$ gegeben. Wir wollen nun zeigen dass $M_j^T(z)$ akzeptiert. Hierfür zielen wir auf einen Widerspruch ab und nehmen an, dass $M_j^T(z)$ ablehnt. Dann existiert auch ein minimales α sodass $M_j^\alpha(z)$ ablehnt, insb. mit $\mathrm{dom}(\alpha) \subseteq \Sigma^{\leq p_j(|z|)}$. Im Folgenden wollen wir nun ein **UPC**-generisches $G' \succ \gamma, \alpha$ konstruieren, denn dann lehnt $M_j^{G'}(z)$ ab und wir erhalten den Widerspruch zu (3).

Sei $\tau = T \cap \Sigma^{<\mathrm{tower}(k+1)}$ die Einschränkung der totalen Funktion G auf die Wörter der Länge $< \mathrm{tower}(k+1)$. Es ist unmittelbar klar dass $\alpha \preceq \tau$. (Alle Wörter in $\mathrm{dom}(\alpha)$ haben Länge $\leq p_j(|z|)$ und diese sind $< \mathrm{tower}(k+1)$, denn andernfalls $\mathrm{tower}(k+1) \leq p_j(|z|)$ was der Wahl von k widerspricht.)

Nach Definition von T stimmt τ mit G auf allen Wörtern der Länge $< \mathrm{tower}(k)$ überein. Damit ist klar dass $\gamma \preceq \tau$. (Für alle Wörter $x \in \mathrm{dom}(\gamma)$ gilt $|x| \leq n \leq \mathrm{tower}(k-1) < \mathrm{tower}(k)$, also stimmen S und T auf x überein und es gilt $\gamma(x) = S(x) = T(x) = \tau(x)$.)

Wir wollen nun zeigen dass τ eine **UPC**-generische Bedingung ist, dann existiert nach Lemma 5.8 auch ein **UPC**-generisches Orakel $G' \succ \tau \succeq \gamma, \alpha$ und $M_j^{G'}(z)$ lehnt ab und wir haben den Widerspruch zu (3) wie gewünscht. Hierfür gehen wir die Eigenschaften der Definition 5.15 durch: (i) klar, ist ja $\mathrm{dom}(\tau) = \Sigma^{<\mathrm{tower}(k+1)}$. (ii) ebenso klar, denn für alle $m < \mathrm{tower}(k+1)$ gilt $|\tau \cap \Sigma^m| = |T \cap \Sigma^m| \leq |G \cap \Sigma^m|$ und wende Eigenschaft 5.16(ii) an.

Für (iii), betrachte ein $i \in \omega$ und $m \in H_i$ mit $m < \text{tower}(k+1)$. Damit ist $m = \text{tower}(k')$ für ein $k' \leq k$. Sei $|\tau \cap \Sigma^m| = 2$. Dann gilt auch $|G \cap \Sigma^m| = 2$ (τ stimmt mit T auf Wörtern auf Σ^m überein und $T \subseteq G$). Nach Eigenschaft 5.16(iii) existiert also eine Eingabe y sodass ein $\beta \prec G$ existiert, wo $M_i^\beta(y)$ mit zwei Rechenwegen akzeptiert, und $\text{dom}(\beta) \subseteq \Sigma^{\leq 2^m}$.

Wir zeigen nun $\beta \preceq \tau$, denn dann akzeptiert auch $M_i^\tau(y)$ auf zwei Rechenwegen und wir haben die gewünschte Eigenschaft der **UPC**-Definition 5.15(iii) gezeigt. Klar ist schon mal dass $\text{dom}(\beta) \subseteq \text{dom}(\tau)$. (Für alle $x \in \text{dom}(\beta)$ gilt $|x| \leq 2^m = 2^{\text{tower}(k')} < \text{tower}(k+1)$ also $x \in \text{dom}(\tau)$.)

Gilt $k' < k$ gilt für alle $x \in \text{dom}(\beta)$ auch $|x| \leq 2^m = 2^{\text{tower}(k')} < \text{tower}(k)$, und da je β und τ mit G auf Wörtern der Länge $< \text{tower}(k)$ übereinstimmen, haben wir sofort dass β und τ kompatibel sind; in diesem Fall $\beta \preceq \tau$ gezeigt.

Gilt $k' = k$ dann gilt wie oben nach Konstruktion $2 = |T \cap \Sigma^m| = |G \cap \Sigma^m|$ und da $T \subseteq G$ stimmen T und G auf Wörtern der Länge $m = \text{tower}(k)$ überein. Zusammen mit Voraussetzung ergibt sich, dass dann T und G auf Wörtern der Länge $< \text{tower}(k+1)$ übereinstimmen. Also gilt $\tau \prec G$ und nach Wahl $\beta \prec G$; wir haben dass β und τ kompatibel sind und haben $\beta \preceq \tau$ gezeigt.

Das schließt den Beweis ab. □

Nun können wir abschließend alles zusammenführen.

Satz 5.20. *Es existiert ein Orakel relativ zu diesem* UP *keine vollständigen Mengen hat, aber auch Aussage Q gilt.*

Insbesondere gilt das für alle Orakel der Form $G \oplus A$ wobei G ein **UPC***-generisches Orakel ist, und A ein* PSPACE*-vollständiges Orakel.*

Beweis. Klar ist, dass solche Orakel G und A existieren. Kombiniert man Behauptung 5.18 und 5.17 ergibt sich, dass UP relativ zu G keine vollständige Menge hat. Nachdem die Behauptungen relativieren gilt auch, dass UP relativ zu $G \oplus A$ keine vollständige Menge hat, und das für alle Orakel A.

Kombiniert man Behauptung 5.19 und Lemma 5.14 und rerelativiert man auf A ergibt sich, dass die Aussage Q relativ zu $G \oplus A$ gilt, *unter Voraussetzung* dass $\text{P}^A = \text{PSPACE}^A$. Aber diese Voraussetzung ist durch ein PSPACE-vollständiges A ja genau erfüllt. □

Literaturverzeichnis

Baker, Theodore, John Gill und Robert Solovay. 1975. »Relativizations of the P=? NP Question«. *SIAM Journal on Computing* 4 (4): 431–442. https://doi.org/10.1137/0204037.

Dose, Titus, und Christian Glaßer. 2019. *NP-Completeness, Proof Systems, and Disjoint NP-Pairs.* Technical Report 2019-050. Electronic Colloquium on Computational Complexity (ECCC). https://eccc.weizmann.ac.il/report/2019/050/.

Fenner, Stephen, Lance Fortnow, Stuart A. Kurtz und Lide Li. 2003. »An oracle builder's toolkit«. *Information and Computation* 182 (2): 95–136. https://doi.org/10.1016/S0890-5401(03)00018-X.

Fortnow, Lance, und Joshua A. Grochow. 2011. »Complexity classes of equivalence problems revisited«. *Information and Computation* 209 (4): 748–763. https://doi.org/10.1016/j.ic.2011.01.006.

Fortnow, Lance, und John D. Rogers. 2002. »Separability and one-way functions«. *Computational Complexity* 11 (3): 137–157. https://doi.org/10.1007/s00037-002-0173-4.