Sei $A \in \mathrm{NP}$ mit folgender Padding-Eigenschaft: es existiert eine Menge $B \in \mathrm{P}$ sodass $A = \{x \mid \exists y, |y| \leq p(|x|), (x,y) \in B\}$ und für alle $n \in \mathbb{N}$ gilt

$$(x,y) \in B \iff (pad(x, 1^n), y) \in B.$$

wobei $pad \in \mathrm{FP}$ und verlängernd ist. Die folgenden Aussagen sind äquivalent:

(A1) Für alle NPTM N mit $L(N) = A$ lassen sich akzepierende Rechenwege von N in Zertifikate umrechnen: es existiert eine Funktion $f \in \mathrm{FP}$ sodass

$$N(x) \text{ akz. mit RW } \alpha \implies (x, f(x,\alpha)) \in B.$$

(A2) Das Standard-Beweissystem std_B bzgl. B ist p-optimal. Dieses ist

$$std_B(w) = \begin{cases} x & \text{wenn } w = (x,y) \text{ und } (x,y) \in B \\ \top & \text{sonst} \end{cases}.$$

*

Diese Padding-Eigenschaft erfüllen folgende Mengen:

- $\mathrm{SAT} = \{\varphi \mid \exists y.\varphi(y) = 1\}$, Padding durch Anhängen von Tautologien. Hier ist std_B das Standard-Beweissystem für SAT.
- Das kanonische vollständige Problem

  $$K = \{(N, x, 1^i) \mid \exists \alpha, |\alpha| \leq i, N(x) \text{ akz. mit RW } \alpha\},$$

  Padding durch „Verlängern“ von N mit nicht erreichbaren Zuständen.

  Hier lässt sich (A1) auch wie folgt interpretieren: jede universelle NP-Maschine U mit $L(U) = K$ ist „transparent“ in dem Sinn dass ein akzeptierender Rechenweg α von $U(N, x, 1^i)$ sich effizient in einen akzeptierenden Rechenweg β für $N(x)$ umrechnen lassen kann.
- Jede Menge $A \in \mathrm{NP}$, die im klassischen Sinn paddable ist, also für die eine Funktion g mit $g, g^{-1} \in \mathrm{FP}$ existiert sodass $x \in A \iff g(x,z) \in A$.

*

Sei $A \in \mathrm{NP}$ mit folgender Eigenschaft von Vollständigkeit: es existiert eine Menge $B \in \mathrm{P}$ sodass $A = \{x \mid \exists y, |y| \leq p(|x|), (x,y) \in B\}$ und für alle Mengen $A' \in \mathrm{NP}$, $A' = \{x \mid \exists y, |y| \leq p'(|x|), (x,y) \in B'\}$ existieren zwei Funktionen $r, r^{-1}, t \in \mathrm{FP}$ sodass

$$x \in A' \iff r(x) \in A, \quad (r(x), z) \in B \implies (x, t(x,z)) \in B'.$$

($A' \leq_m^p A$ via invertierbarem r, Funktion t bildet Zertifikate für $r(x) \in A$ auf Zertifikate für $x \in A'$ ab. Vgl. Reduktionsbegriff unter TFNP-Problemen. Vgl. Levin-Reduktionsbegriff.) Folgende Aussagen sind äquivalent:

(A1) Für alle NPTM N mit $L(N) = A$ lassen sich akzepierende Rechenwege von N in Zertifikate umrechnen: es existiert eine Funktion $f \in \mathrm{FP}$ sodass

$$N(x) \text{ akz. mit RW } \alpha \implies (x, f(x,\alpha)) \in B.$$

(Q) $\mathrm{NPMV}_t \subseteq_c \mathrm{FP}$.

*

Zumindest SAT und K erfüllen sowohl die Padding-Eigenschaft, als auch die Vollständigkeits-Eigenschaft. Deren Standard-Beweissysteme sind p-optimal genau dann wenn Q gilt.

Die genannte Padding-Eigenschaft und Vollständigkeits-Eigenschaft werden von Mengen erfüllt, welche Levin-vollständig und paddable (im klassichen Sinn) sind.

Für Relation R ist zulässig wenn $R(x,y) \implies |y| \leq p(|x|)$. Für zulässige Relationen R ist $L(R) = \{x \mid \exists y.R(x,y)\} \in \text{NP}$.

Eine Relation R ist Levin-vollständig falls diese zulässig ist, und für jede zulässige Relation Q gilt: es existieren FP-Funktionen f und g mit

$$x \in L(Q) \iff f(x) \in L(R), \quad R(f(x),z) \implies Q(x,g(x,z)).$$

Die folgenden Aussagen sind äquivalent:

(1) Das Standardbeweissystem *sat* für SAT ist p-optimal
[wahrscheinlich falsch da wahrscheinlich SAT]

(2) Für eine Levin-vollständige, paddable Relation R ist das Standardbeweissystem std_R p-optimal. Zur Erinnerung:

$$std_R(w) = \begin{cases} x & \text{wenn } w = (x,y) \text{ und } R(x,y) \\ \top & \text{sonst} \end{cases}.$$

(3) Für *alle* Levin-vollständigen, paddable Relationen R ist das Standardbeweissystem std_R p-optimal

(4) $\text{NPMV}_t \subseteq_c \text{FP}$, i.e., Q
[wahrscheinlich falsch da wahrscheinlich $\text{P} \neq \text{NP} \cap \text{coNP}$]

(5) Jedes optimale Beweissystem ist p-optimal

(6) Für jede Relation gilt:
R ist Levin-vollständig $\iff$ $L(R)$ ist many-one-vollständig.
[„empirisch“ wahr]

Orakelkonstruktion DisjNP, UP, und 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 $(A_m^O, B_m^O) \in \text{DisjNP}^O$.
Wenn $|O \cap \Sigma^n| \leq 1$ für alle $n \in H_m$, dann $C_m^O \in \text{UP}^O$.

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 $\text{P}^C = \text{PSPACE}^C$ können wir relevante Wörter in $O - C$ errechnen und so einen akzeptierenden Weg von $M_j^O(x)$ ausgeben – damit erzielen wir Q.

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 $(A_m, B_m) \in \text{DisjNP}$.)
- 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 $L(M_a) \notin \text{UP}$ 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 $C_m \in \text{UP}$.)

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.)

- $\tau^2_{a,b}$: 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 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 Aussagen gilt:
  - $0^n \in A^v_m$ für alle $v \supseteq w_s$ und $M_a(F_r(0^n))$ lehnt relativ zu w_s definitiv ab.
  - $0^n \in B^v_m$ für alle $v \supseteq w_s$ und $M_b(F_r(0^n))$ lehnt relativ zu w_s definitiv ab.

  (Das ist möglich nach Behauptung 2.)

- τ^3_a: Setze $t' = t_{s-1} \cup \{\tau^3_{a,b} \mapsto 0\}$. 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 $\tau^3_{a,b,r}$ 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 $t_s := t_{s-1} \cup \{\tau^3_{a,b} \mapsto m\}$; 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.)

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

  (Das ist möglich nach Behauptung 3.)

Behauptung 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. *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 $u \supsetneq w_{s-1}$ erweitern, dass eine der o.g. Fälle eintritt.*

Skizze. Widerspruchsbeweis. Erweitere w_{s-1} mit Behauptung 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 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 $u \supsetneq w_{s-1}$ 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 $0^n \in C^{u(\emptyset)}_m$ 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 $0^n \in C^{u(\{\alpha\})}_m$ und daher akzeptiert $M_a(F_r(0^n))$ definitiv relativ zu

$u(\{\alpha\})$.

- für ungerades $\beta \in \Sigma^n$ gilt $0^n \in C_m^{u(\{\beta\})}$ 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 $u(\{\alpha\})$. 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 4. *Kein Paar aus* DisjNP^O *ist* $\leq_{\mathrm{m}}^{\mathrm{pp}}$*-hart für* DisjUP.

Behauptung 5. *Keine Menge aus* UP^O *ist* $\leq_{\mathrm{m}}^{\mathrm{p}}$*-vollständig.*

Behauptung 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 $< n$ übereinstimmt, aber für ein Wort z lehnt $M_j^T(x)$ ab. Sei $t' = t_{s-1} \cup \{\tau_j^1 \mapsto 0\}$ und sei $v = T \cap \Sigma^{\leq p_j(|x|)}$. 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 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 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 $C \cup D$ 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 ← {w | w ∈ O, |w| ≤ n, ∃i.|w| = e(i)}
2 repeat
3 |   Sei α ein akzeptierender Rechenweg auf M_j^{C∪D}(x) und Q die Menge an Orakelfragen
4 |   if existiert eine Frage q ∈ Q für die q ∈ O − C aber q ∉ D then
5 |   |   D ← D ∪ {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 $C \cup D$ 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. □

Orakel mit DisjCoNP und DisjNP und P = UP

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.)

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 $v \supsetneq w_{s-1}$, 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.)

- $\tau^2_{a,b}$: 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 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 Aussagen gilt:
  - $0^n \in A^v_m$ für alle $v \supseteq w_s$ und $M_a(F_r(0^n))$ akzeptiert definitiv relativ zu w_s.
  - $0^n \in B^v_m$ für alle $v \supseteq w_s$ und $M_b(F_r(0^n))$ akzeptiert definitiv relativ zu w_s.

  (Das ist möglich nach Behauptung 2.)

- $\tau^3_{a,b}$: Setze $t' = t_{s-1} \cup \{\tau^3_{a,b} \mapsto 0\}$. 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 $\tau^3_{a,b,r}$ 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 $t_s := t_{s-1} \cup \{\tau^3_{a,b} \mapsto m\}$; 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.)

- $\tau^3_{a,b,r}$: 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 $w_s \supsetneq w_{s-1}$ sodass bezüglich einem $n \in \mathbb{N}$ eine der folgenden Aussagen gilt:
  - $0^n \in D^v_m$ für alle $v \supseteq w_s$ und $M_a(F_r(0^n))$ lehnt definitiv relativ zu w_s ab.
  - $0^n \in E^v_m$ für alle $v \supseteq w_s$ und $M_b(F_r(0^n))$ lehnt definitiv relativ zu w_s ab.

  (Das ist möglich nach Behauptung 2.)

Behauptung 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. *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 $u \supsetneq w_{s-1}$ 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 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 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 $u \supsetneq w_{s-1}$ erweitern, dass eine der o.g. Fälle eintritt.*

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

Behauptung 4. *Kein Paar aus* $\mathrm{DisjCoNP}^O$ *ist* $\leq^{\mathrm{pp}}_{\mathrm{m}}$*-vollständig.*

Behauptung 5. *Kein Paar aus* DisjNP^O *ist* $\leq^{\mathrm{pp}}_{\mathrm{m}}$*-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* $\leq^{\mathrm{p}}_{\mathrm{m}}$*-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 $P^{\text{yes}} = P^{\text{all}} \cap v$, $P^{\text{no}} = P^{\text{all}} \cap \overline{v}$. Beob. dass für festes $U = \Sigma^{e(i)}$ (bzw. $U = \Sigma^{e(i)-1}0$, $U = \Sigma^{e(i)-1}1$) wegen $\text{PSPACE}^C = \text{P}^C$ 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 $M \in \text{P}$, denn es sind höchstens endlich viele H_m in der Vereinigung, welche je in P entscheidbar sind.

[TODO was soll die Menge U?]

Behauptung 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*:*

- $e(i) \in M$ *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. 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 $\Sigma^{e(i)} - (W \cup W')$ 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 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 7. P = UP *relativ zu* O.

Skizze. Sei $S \in \text{UP}^O$. 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 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 ∩ Σ^{e(i)} − (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 $|P^{*\text{all}} \cap \Sigma^{e(i)} - (W \cup W')|$ in jeder Iteration um ≥ 1 abnimmt. Da $|P^{*\text{all}}| \leq p_j(|x|)$ ist nach $\leq p_j(|x|) + 1$ vielen Iterationen $|P^{*\text{all}} \cap \Sigma^{e(i)} - (W \cup W')| = 0$. Nach $\leq p_j(|x|) + 1$ vielen Iterationen wird also in Z. 5 eine Berechnung P ausgewählt, bei der alle $q \in P^{\text{all}}$ 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 4 anwendbar. Also haben diese zwei Berechnungen eine identische Orakelfrage $q \in P^{\text{all}} \cap P^{*\text{all}} \cap \Sigma^{e(i)}$ 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 $|P^{*\text{all}} \cap \Sigma^{e(i)} - (W \cup W')|$ 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 „$x \in S$“. □

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 1.)
- τ^2_j: Setze $t' = t_{s-1} \cup \{\tau^2_j \mapsto 0\}$. 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.)

- $\tau^3_{a,b}$: Setze $t' = t_{s-1} \cup \{\tau^3_{a,b} \mapsto 0\}$. 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 $t_s := t_{s-1} \cup \{\tau^3_{a,b} \mapsto m\}$; 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.)
- $\tau^3_{a,b,r}$: 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 Aussagen gilt:
  - $0^n \in A^v_m$ für alle $v \supseteq w_s$ und $M_a(F_r(0^n))$ lehnt relativ zu w_s definitiv ab.
  - $0^n \in B^v_m$ für alle $v \supseteq w_s$ und $M_b(F_r(0^n))$ lehnt relativ zu w_s definitiv ab.

  (Das ist möglich nach Behauptung 2.)

Behauptung 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. *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 $u \supsetneq w_{s-1}$ 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. *Kein Paar aus* DisjNP^O *ist* $\leq^{\mathrm{pp}}_{\mathrm{m}}$*-hart für* DisjUP.

Behauptung 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 $L(M^T_j) = \Sigma^*$.*

Behauptung 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 $c_k = |\{j \mid t_k(\tau^2_j) = 0\}|$. 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 $w_s \subsetneq O$ 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 6. *Sei M_j eine totale Maschine, d.h. $L(M^O_j) = \Sigma^*$. Dann existiert eine Funktion $g \in \mathrm{FP}^O$ sodass $g(x)$ einen akzeptierenden Rechenweg von $M^O_j(x)$ 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 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 $S = L(M^O_j)$ in P zu entscheiden, gehen wir wie in vorigem Abschnitt vor. Sei s die Stufe bei der τ^1_j betrachtet wurde. Sei $c = |\{j \mid t_{s-1}(\tau^2_j) = 0|$. Wir beschränken uns im Folgenden wieder auf Eingaben hinreichender Länge, sodass es für x ein eindeutiges i gibt, sodass

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

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 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 $t' = t_{s-1} \cup \{\tau_j^2 \mapsto 0\}$. Wir zeigen dass u auch t'-valide ist Damit wäre u dann eine geeignete Erweiterung von w_{s-1} in Bearbeitung von Task τ_j^2, 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 $c' = |\{j \mid t'(\tau_j^2) = 0\}| = c + 1$, 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 8. P = UP *relativ zu O.*

Orakel mit DisjCoNP, CON^N und alle Paare aus DisjNP sind P-separierbar

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:

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

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

Definiere die Orakel-ergänzte aussagenlogische Formel

$\theta_n =$ „$\neg \mathtt{query}_n(x_1, \ldots, x_n)$“.

Die Forme θ_n ist polynomiell groß abh. von n, und $\theta_n \in \mathrm{TAUT}^O \iff O \cap \Sigma^n = \emptyset$. Definiere das Beweissystem

$$g_m(x) = \begin{cases} \theta_n & \text{wenn } x = 0^n \text{ und } n \in H_m \\ \top & \text{sonst} \end{cases}.$$

Damit ist $\mathrm{img}(g_m) \subseteq \mathrm{TAUT}^O \iff \forall n \in H_m . O \cap \Sigma^n = \emptyset$.

TODO

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). Erreiche dass T_r kein Beweissystem für TAUT ist (Task τ^3_r), oder falls das nicht möglich ist, das zugehörige Zeugen-Beweissystem g_m tatsächlich ein Beweissystem für eine Teilmenge von TAUT^O ist.

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 $\tau^1_{a,b}, \tau^2_{a,b}, \tau^3_i$ 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.)
- V5 Wenn $t(\tau^3_r) = 0$, dann existiert ein z sodass definitiv $T_r^w(z) = y$ und $y \notin \mathrm{TAUT}^w$.
  (T_r kein Beweissystem für TAUT relativ zum finalen Orakel.)
- V6 Wenn $0 < t(\tau^3_r) = m$, dann gilt für alle $n \in H_m$: $|\Sigma^n \cap w| = 0$.
  (Wenn $0 < t(\tau^3_r) = m$, dann $\mathrm{img}\, g_m \subseteq \mathrm{TAUT}^w$.)

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 $v \sqsupsetneq w_{s-1}$, dann setze $t_s \coloneqq t'$ und $w_s \coloneqq v$.

  Ansonsten setze $t_s \coloneqq t_{s-1}$ und setze $w_s \coloneqq w_{s-1}y$ für geeignetes $y \in \{0,1\}$ sodass w_s auch t_s-valide ist. (Das ist möglich nach Behauptung 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 $t_s \coloneqq t'$ und $w_s \coloneqq v$. 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 $t_s \coloneqq t_{s-1} \cup \{\tau^2_{a,b} \mapsto m\}$; damit ist w_{s-1} auch t_s-valide. Setze $w_s \coloneqq w_{s-1}y$ für geeignetes $y \in \{0,1\}$ sodass w_s auch t_s-valide ist. (Das ist möglich nach Behauptung 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.)
- τ^3_r: Setze $t' = t_{s-1} \cup \{\tau^3_r \mapsto 0\}$. Existiert ein t'-valides Orakel $v \sqsupsetneq w_{s-1}$, dann setze $t_s \coloneqq t'$ und $w_s \coloneqq v$.

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

Behauptung 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. *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.*

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

Behauptung 3. *Kein Paar aus* $\mathrm{DisjCoNP}^O$ *ist $\leq^{\mathrm{pp}}_{\mathrm{m}}$-vollständig.*

Behauptung 4. *Es existiert kein optimales Beweissystem für* TAUT^O.

Beweis. Angenommen es existiert ein optimales Beweissytem T^O_r für TAUT^O. Sei s die Stufe in der τ^3_r bearbeitet wurde. Sei $n_0 = |w_s|$. Wir haben $t_{s'}(\tau^3_r) = m > 0$ für alle $s' \geq s$, da andernfalls nach V5 ein z existiert mit $T^O_r(z) = T^w_r(z) \notin \mathrm{TAUT}^O$; Widerspruch zur Wahl als Beweissystem für TAUT^O.

Ferner ist g_m ein Beweissystem mit $\operatorname{img}(g_m) \subseteq \mathrm{TAUT}^O$, denn wenn immer $g_m(0^n) = \theta_n$, dann war $n \in H_m$ und nach V6 ist $\Sigma^n \cap O = \emptyset$ und daher $\theta_n \in \mathrm{TAUT}^O$. Nach Wahl von T_r als optimal existiert daher ein Polynom p sodass

$$g_m(0^n) = \theta_n \implies \exists z \in \Sigma^{\leq p(z)}.\, T^O_r(z) = \theta_n.$$

Wähle nun ein $n \in H_m$ sodass (a) $n > n_0$, und (b) $|\theta_n|^{\log|\theta_n|} > p(n)$ und

$$t_r(|\theta_n|^{\log|\theta_n|}) < 2^n. \tag{c}$$

Nachdem $g_m(0^n) = \theta_n$ existert also ein T_r-Beweis z mit $|z| \leq p(z)$ für θ_n, bzw. $T^O_r(z) = \theta_n$. Ohne Beschränkung ist t_r monoton und wir haben

$$t_r(|z|) \leq t_r(p(z)) \leq t_r(|\theta_n|^{\log|\theta_n|}) < 2^n.$$

Also existiert ein Wort $q \in \Sigma^n$ welches sicher nicht von der Berechnung $T^O_r(z)$ gestellt wurde.

Wir können also durch schrittweises Erweitern von w_{s-1} ein finites Orakel w' konstruieren, welches alle Wörter der Länge $\leq t_r(|z|)$ definiert, mit

$$w'(z) = \begin{cases} O(z) & \text{falls } |z| \neq n \\ 1 & \text{falls } z = q \\ 0 & \text{sonst.} \end{cases}$$

Dieses Orakel stimmt mit t_{s-1} überein, ist damit t_{s-1}-valide, und es wäre $\theta_n \notin \mathrm{TAUT}^{w'}$ und $T_r^{w'}(z) = \theta_n$. Damit wäre w' eine dann eine geeignete Erweiterung von w_{s-1} in Bearbeitung von Task τ_k^3, für welche T_k^O kein Beweissystem für TAUT mehr wäre. Widerspruch zur Wahl von T_k. □

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. Sei $M = \bigcup\{H_m \mid t_s(\tau_r^3) = m > 0\}$ eine Menge an Ebenen, welche einem TAUT-Beweissystem *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.

Behauptung 5. *Sei $e(i) \notin M$. 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 $u \supsetneq w_{s-1}$ 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 $|u \cap \Sigma^{e(i)}| = |Y \cap \{\alpha\}| \geq 1$.

Angenommen V6 ist verletzt. Wieder kann das nur an der Ebene $e(i)$ liegen. Aber dann wäre $e(i) \in H_{m'}$ für m', r mit $0 < m' = t_{s-1}(\tau^3_r)$. Damit ist $e(i) \in M$, was der Voraussetzung der Behauptung widerspricht. □

Behauptung 6. *Zu jedem disjunkten* NP*-Paar* (L_a, L_b) *existiert ein Separator aus* P*.*

Skizze. 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. Sei wieder $M = \bigcup\{H_m \mid t_s(\tau^3_{a,b}) = m > 0\}$.

Im einfachen Fall ist $e(i) \in M$. Dann wissen wir nach V6 dass $\Sigma^{e(i)} \cap O = \emptyset$. Wie oben skizziert können alle anderen Wörter der Länge $e(0), \ldots, e(i-1)$ in Polynomialzeit erfragt werden. Wir haben damit alle relevanten Wörter von $O \setminus C$ erfasst und können mittels einem trivialen PSPACE-Algorithmus entscheiden ob $M_a^O(x)$ oder $M_b^O(x)$ akzeptiert.

Wir betrachten im Folgenden also nur den schwierigen Fall mit $e(i) \notin M$. Hier gilt die Behauptung 5. 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  assert e(i) ∉ M
2  W ← ∅, W' ← ∅
3  for k von 0 bis max{p_a(|x|), p_b(|x|)} + 1 do
4  |  P_a ← eine (W, W') respektierende akzeptierende Berechnung P von M_a auf x mit
   |    |P^all ∩ Σ^e(i) − (W ∪ W')| minimal, oder ⊥ falls keine existiert
5  |  P_b ← eine (W, W') respektierende akzeptierende Berechnung P von M_b auf x mit
   |    |P^all ∩ Σ^e(i) − (W ∪ W')| minimal, oder ⊥ falls keine existiert
6  |  if P_a = ⊥ then return „x ∈ L_b“
7  |  if P_b = ⊥ then return „x ∈ L_a“
   |  (ab hier sind P_a, P_b je zwei (W, W') respektierende akzeptierende Berechnungen)
8  |  if alle q ∈ P_a^all mit |q| = e(i) sind in W ∪ W' then
9  |  |  return „x ∈ L_a“
10 |  end
11 |  if alle q ∈ P_b^all mit |q| = e(i) sind in W ∪ W' then
12 |  |  return „x ∈ L_b“
13 |  end
14 |  foreach q ∈ P_a^all ∪ P_b^all mit |q| = e(i) do
15 |  |  if q ∈ O then W ← W ∪ {q}
16 |  |  if q ∉ O then W' ← W' ∪ {q}
17 |  end
18 end
19 return „x ∉ L_a ∪ 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 $P_b = \bot$, 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 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. □

Gegeben:

- Betrachte Wortmenge X mit n Elementen.
- Alle Maschinen $m \in \mathcal{M}$ wollen ein für sie geeignetes Wort $x \in X_m \subseteq X$ zugewiesen bekommen.
- Die Maschinen in $\mathcal{M}$ bilden Paare: für jedes $m \in \mathcal{M}$ gibt es ein $\overline{m} \in M$ mit $X_m \cup X_{\overline{m}} = X$. Damit gilt $|X_m| + |X_{\overline{m}}| = n$.
- Es gibt p viele Paare, also $|\mathcal{M}| = 2p$.
- Außerdem sind die beiden Mengen ungefähr gleich groß: es gibt ein festes q sodass $q < |X_m| < n - q$ für alle m gilt.
- Gewisse Wörter stehen gegenseitig in Konflikt: Wenn $\{a, b\} \in \mathcal{K}$, dann stehen $a, b \in X$ in Konflikt.
- Jedes Wort x steht zu höchstens r anderen Wörtern in Konflikt.

Gesucht: Menge $X' \subseteq X$ sodass keine zwei Wörter in X' in Konflikt stehen, sowie für jede $m \in \mathcal{M}$ gilt $X_m \cap X' \neq \emptyset$.
Idee Konstruktion:

- Konstruiere einen Graphen G.
- Zeichne für jede Maschine $m \in \mathcal{M}$ einen Cluster G_m. Dieser Cluster hat die Knoten $(m, x_1), (m, x_2), ...$ wobei $x_i \in X_m$, und keine Kanten.
- Verbinde Knoten über die Clusters hinaus: Wenn (m, x) ein Knoten von G_m, und (m', x') ein Knoten von $G_{m'}$ ist mit $m \neq m'$, und $\{x, x'\} \notin \mathcal{K}$, dann verbinde diese zwei Knoten.
- Existiert eine Clique mit $|\mathcal{M}|$ Knoten in G, dann existiert auch die gesuchte Teilmenge X'.
- Der Graph hat folgende Anzahl an Knoten:

$$|V(G)| = \sum_{\text{Paar } m, \overline{m} \text{ aus } \mathcal{M}} |V(K_m)| + |V(K_{\overline{m}})| = \sum_{\text{Paar } m, \overline{m} \text{ aus } \mathcal{M}} |X_m| + |X_{\overline{m}}| = pn.$$

- Die Anzahl der Kanten können wir folgendermaßen abschätzen: Sei (m, x) ein Knoten. Dieses x ist zu mindestens $n - r$ Wörtern nicht in Konflikt. Sei y ein solches Wort sodass x und y nicht in Konflikt stehen.. Für jedes Paar $m', \overline{m'} \in \mathcal{M}$ mit $m \notin \{m', \overline{m'}\}$ ist nun $x' \in X_m$ oder $x' \in X_{\overline{m}}$.

  Also ist (m, x) entweder zu (m', y) oder $(\overline{m'}, y)$ inzident in G. Es gibt $p - 1$ viele solche Paare, also ist $d((m, x)) \geq (p - 1)(n - r)$.

  Nach Handschlagssatz also

$$|E| = \sum_{(m,x) \in V} d((m, x)) \geq |V(G)|(p - 1)(n - r)/2 = \frac{(pn)^2 - pn^2 - p^2rn}{2}.$$

- Wir zeigen nun $|E| > (1 - 1/(2p))(pn)^2/2$. Nach dem Satz von Turán existiert dann eine Clique mit $2p$ vielen Knoten. Es gilt

$$\begin{aligned}
&\implies 1/p - r/n < 1/(2p) \\
&\implies 1 - 1/p - r/n > 1 - 1/(2p) \\
&\implies (1 - 1/p - r/n)\,(pn)^2 > (1 - 1/(2p))(pn)^2 \\
&\implies (pn)^2 - pn^2 - p^2rn > (1 - 1/(2p))(pn)^2 \\
&\implies \frac{(pn)^2 - pn^2 - p^2rn}{2} > (1 - 1/(2p))(pn)^2/2 \\
&\implies |E| > (1 - 1/(2p))(pn)^2/2
\end{aligned}$$

	DisjNP	DisjCoNP	UP	$\neg Q$	oracle
0	$-$	$(-)$	$-$	$-$	BGS
1	$-$	$-$	$-$	$+$	Dingel
2	$-$	$(-)$	$+$	$-$	at least as hard as UP $\not\Rightarrow$ DisjNP
3	$-$	$-$	$+$	$+$	
4	$-$	$+$	$-$	$-$	inconsistent
5	$-$	$+$	$-$	$(+)$	Khaniki
6	$-$	$+$	$+$	$-$	inconsistent
7	$-$	$+$	$+$	$(+)$	at least as hard as UP $\not\Rightarrow$ DisjNP
8	$+$	$(-)$	$-$	$-$	Sec. 3
9	$+$	$-$	$-$	$+$	
10	$+$	$(-)$	$+$	$-$	O1
11	$+$	$-$	$+$	$+$	Dose20a
12	$+$	$+$	$-$	$-$	inconsistent
13	$+$	$+$	$-$	$(+)$	O2
14	$+$	$+$	$+$	$-$	inconsistent
15	$+$	$+$	$+$	$(+)$	EEG

Orakel mit SAT und $\neg Q'$

Wir verstehen die relativierte Version von SAT als die Nicht-Existenz eines p-optimalen Beweissystems zur Menge SAT^O. Das ist die Menge der erfüllbaren aussagenlogischen Formeln, welche zusätzlich die Prädikate $\texttt{query}_m(a_1, a_2, \dots, a_m)$ verdenden darf. Interpretation: $\texttt{query}_m(a_1, a_2, \dots, a_m)$ evaluiert zu 1 genau dann wenn $a_1 a_2 \cdots a_m \in O$.

Nach Dingel wissen wir:

Lemma 1. *Sei $f, g \in \mathrm{FP}^O$ mit $g(\Sigma^*) \subseteq f(\Sigma^*)$. Wird g nicht von f p-simuliert, also*

$$\forall h \in \mathrm{FP}^O \, \exists x \in \Sigma^*. \, f(h(x)) \neq g(x),$$

dann ist f nicht p-optimal für $f(\Sigma^*)$.

Sei $e(0) = 2, e(i+1) = 2^{e(i)}$. (Doppelt exponentiell!) Sei hier $\{H_m\}_{m \in \mathbb{N}}$ eine Familie von paarweise disjunkten, unendlichen Teilmengen von $e(\mathbb{N})$. (Ebenen H_m gehören zum Zeugen-Beweissytem g_m)

Idee: wir erreichen, dass alle Funktionen $T_j \in \mathrm{FP}$ entweder kein Beweissystem für relativiertes SAT sind, oder dass diese nicht das folgende Zeugen-Beweisystem g_m p-simulieren (wobei m von T_j abhängig ist). Wir setzen

$$g_m(x) = \begin{cases} \text{„}\texttt{query}_n(a_1, \dots, a_n)\text{“} & \text{falls } x = 0^n \text{ und } n \in H_m \\ \text{„}a_1 \vee \neg a_1\text{“} & \text{sonst.} \end{cases}$$

Fakt: gilt $O \cap \Sigma^n \neq \emptyset$ für alle $n \in H_m$, dann ist $g_m(\Sigma^*) \subseteq \mathrm{SAT}^O$.

Wollen wir also erreichen, dass T_j mit $T_j(\Sigma^*) = \mathrm{SAT}^O$ nicht p-optimal ist, genügt es für alle Kandidaten T_r an Übersetzungsfunktionen zu sichern, dass $T_j(T_r(0^n)) \neq g_m(0^n)$ für geeignetes $n \in H_m$, und gleichzeitig $O \cap \Sigma^n \neq \emptyset$ für alle $n \in H_m$.

Die Aussage $\neg Q'$ ist äquivalent zur Aussage, dass alle Paare aus DisjCoNP auch P-separierbar sind. Versuche daher in der Konstruktion zunächst möglichst Paare an coNP-Maschinen zu zerstören, i.e. dass beide Maschinen ablehnen. Ist das nicht möglich, werden wir durch eine Codierung das Akzeptanzverhalten dieses Paars in das Orakel eintragen.

Für die Kodierung definieren wir injektiv den Codewort $c(a, b, x) := 0^a 1 0^b 1 0^l 1 0^p 1 x$ mit $p = p_a(|x|) + p_b(|x|)$, $l \in \mathbb{N}$ minimal, sodass $l \geq 7/8|c(a, b, x)|$ und $c(a, b, x)$ ungerade Länge hat. Auf diese Weise enthält ein Codewort das Wort x als Information und ist auf ausreichende Länge aufgefüllt. Wir bezeichnen jedes Wort der Form $c(\cdot, \cdot, \cdot)$ als *Codewort*. Hierdurch werden die folgenden Eigenschaften garantiert:

Behauptung 2. *Für alle $a, b \in \mathbb{N}$, $x \in \Sigma^*$ gilt folgendes:*

(1) $|c(a, b, x)| \notin H_m$ für jedes m.

(2) Für feste a, b ist die Funktion $x \mapsto c(a, b, x)$ polynomialzeit-berechenbar und polynomialzeit-invertierbar bezüglich $|x|$.

(3) Relativ zu einem beliebigen Orakel ist die Ausführungszeit von $M_a(x)$ und $M_b(x)$ beide auf $< |c(a, b, x)|/2$ beschränkt.

(4) Für jedes partielle Orakel $w \in \Sigma^*$ gilt, wenn $c(a, b, x) \leq |w|$, dann sind $M_a^w(x)$ und $M_b^w(x)$ definit.

Für ein DisjCoNP-Paar $(\overline{L(M_a)}, \overline{L(M_b)})$ setzen wir $c(a, b, x) \in O$ genau dann wenn $M_a(x)$ akzeptiert. Entsprechend ist $c(a, b, c) \notin O$ genau dann wenn $M_b(x)$ akzeptiert. Das ist ausreichend, um dieses Paar zu P-separieren, denn $S = \{x \mid c(a, b, x) \in O\}$ ist ein Separator. Es gilt $\overline{L(M_b)} \subseteq S$: für $x \in \overline{L(M_b)}$ muss gelten dass $M_a(x)$ akzeptiert (denn ansonsten ist das Paar nicht disjunkt), und nach Codierung ist $c(a, b, x) \in O$ und daher $x \in S$.

Ebenso gilt $\overline{L(M_a)} \subseteq \overline{S}$: äquivalent zu $S \subseteq L(M_a)$, was nach Konstruktion sofort klar ist.

Der Zerstörung eines Beweissystems T_j ordnen wir Task τ_j^1 zu, der Diagonalisierung gegen Übersetzungsfunktion T_r ordnen wir $\tau_{j,r}^1$ zu. Der Zerstörung eines DisjCoNP-Paares $(\overline{L(M_a)}, \overline{L(M_b)})$ ordnen wir den Task $\tau_{a,b}^2$ zu.

Sei wie üblich $t \in \mathcal{T}$ wenn der Definitionsbereich endlich ist, nur die Tasks der Form $\tau^1_n, \tau^2_{a,b}$ 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 $t(\tau^1_j) = 0$, dann existiert ein z sodass $T^w_j(z) \notin \mathrm{SAT}^w$ und das definitiv.
(T_j kein Beweissystem für SAT relativ zum finalen Orakel.)

V2 Wenn $t(\tau^2_{a,b}) = 0$, dann existiert ein z sodass $M^w_a(z)$ und $M^w_b(z)$ definitiv ablehnen.
(Wenn $t(\tau^2_{a,b}) = 0$, dann $(\overline{L(M_a)}, \overline{L(M_b)}) \notin$ DisjCoNP relativ zum finalen Orakel.)

V3 Wenn $0 < t(\tau^1_j) = m$, dann gilt für alle $n \in H_m$ dass $\Sigma^n \cap w \neq \emptyset$.
(Wenn $0 < t(\tau^1_j) = m$, dann ist $g_m \subseteq \mathrm{SAT}^w$.)

V4 Wenn $0 < t(\tau^2_{a,b}) \leq c(a,b,x) < |w|$, dann ist $M_a(x)^w$ definit. Die Berechnung akzeptiert wenn $c(a,b,x) \in w$, und lehnt ab wenn $c(a,b,x) \notin w$.
(Wenn $t(\tau^1_j) > 0$, dann codieren wir, von Wort $t(\tau^1_j)$ an, die Menge $L(M_a)$ in das Orakel.)

Sei T eine abzählbare Aufzählung der o.g. Tasks sodass $\tau^1_{j,r}$ immer nach τ^1_j kommt. 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 wähle ein hinreichend großes $m \notin \mathrm{img}(t_{s-1})$ sodass w_{s-1} kein Wort der Länge $\min H_m$ definiert. Setze $t_s := t_{s-1} \cup \{\tau^1_j \mapsto m\}$; 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.)

  (Bedeutung: falls möglich, zerstöre das Beweissystem für SAT. Ist das nicht möglich, ordne diesem Beweissystem eine Menge an Stufen an, in der wir die Diagonalisierung für $g_m \not\leq^p T_j$ durchführen können.)

- $\tau^2_{a,b}$: 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 setze $t_s := t_{s-1} \cup \{\tau^2_{a,b} \mapsto n\}$ wobei $n > |w_{s-1}|, \max\{\mathrm{img}(t_{s-1}\}$. Damit ist t_s injektiv auf dem Support, und w_{s-1} ist 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.)

  (Bedeutung: falls möglich, mach dass M_a und M_b beide ablehnen. Ist das nicht möglich, verlange ab diesem Punkt dass die Menge $L(M_a)$ in das Orakel hineincodiert wird.)

- $\tau^1_{j,r}$: Wir wissen dass $t_{s-1}(\tau^1_j) = m > 0$. Setze $t_s = t_{s-1}$ und wähle ein t_s-valides Orakel $w_s \supsetneq w_{s-1}$ sodass bezüglich einem $n \in H_m$ die Folgende Aussage gilt: $z \in w_s \cap \Sigma^n$ und $T_j(T_r(0^n)) \neq g_m(0^n)$ (=„`query`$(a_1, \ldots a_n)$“) (Das ist möglich nach Behauptung 3.)

Behauptung 3. *Für jedes $t \in \mathcal{T}$ und jedes t-valide w existiert ein $y \in \{0,1\}$ sodass wb auch t-valide ist. Insbesonderte gilt folgendes: Sei $s \in \mathbb{N}$, $(w_0, t_0), \ldots, (w_s, t_s)$ definiert und sei $w \in \Sigma^n$ ein t_s-valides Orakel mit $w \sqsupseteq w_s$ und setze $z = |w|$. dann existiert ein $y \in \{0,1\}$ sodass wy auch t_s-valide ist.*

(i) Wenn $z = c(a,b,x)$ und $0 < t_s(\tau^1_j) \leq z$, dann ist $w1$ auch t_s-valide wenn $M^w_a(x)$ akzeptiert, und $w0$ ist t_s-valide wenn $M^w_b(x)$ akzeptiert.

(ii) Wenn ein $\tau^2_{a,b}$ existiert mit $0 < t_s(\tau^2_{a,b}) = m$ und ein $n \in H_m$ existiert sodass $|z| = n$, sowie $|z+1| > n$, dann ist $w1$ ein t_s-valides Orakel.

(iii) In allen anderen Fällen ist $w0$ und $w1$ ein t_s-valides Orakel.

Behauptung 4. *Die Bearbeitung eines Tasks $\tau^1_{j,r}$ ist möglich. Sei $t_{s-1}(\tau^1_j) = m > 0$. dann existiertein t_{s-1}-valides $w \supsetneq w_{s-1}$ und ein $n \in H_m$, sodass*

- *$w \cap \Sigma^n \neq \emptyset$ (und daher $g_m(0^n) \in \mathrm{SAT}^u$), und*
- *$T^w_j(T^w_r(0^n)) \neq g_m(0^n)$.*

Im Folgenden werden wir diese Behauptung direkt beweisen, indem wir ein solches w explizit konstruieren. Hierfür fixieren wir j, r und m. Sei $\hat{s} < s-1$ die Stufe, in welcher der Task τ_j^1 bearbeitet wurde. Sei

$$\gamma(n) = p_j(p_r(n)) + p_r(n)$$

die Laufzeit der Berechnug $T_j^u(T_r^u(z))$ mit Eingabe z der Länge na. Beobachte, dass wenn Q die Menge an Orakelfragen einer Berechnung $T_j^u(T_r^u(0^n))$ (relativ zu einem beliebigen Orakel) ist, dann ist $\ell(Q) \leq \gamma(n)$.

Sei im Folgenden $n \in H_m$ das kleinste n sodass w_{s-1} keine Wörter der Länge n definiert, und

... $(*)$

Wir können nun mittels Behauptung 2 das Orakel w_{s-1} nun so lange zu u erweitern bis u genau alle Wörter dr Länge $< n$ definiert. Dieses Orakel u ist t_{s-1}-valide.

Für $X \subseteq \Sigma^n$ definieren wir $u(X) \supsetneq u$, das für alle Wörter der Länge $\leq \gamma(n)$ definiert ist und sodass $u(X) \cap \Sigma^n = X$. Wieder assoziieren wir mit $u(X)$ einen Abhängigkeitsgraphen $G(X)$, wie schon in z.B. EEG eingesetzt. Für $X \neq \emptyset$ ist das klar, da $u(X)$ dann ja auch ein t_{s-1}-valides Orakel ist. (Insbesondere ist V3 nicht verletzt). Der Abhängigkeitsgraph hat in diesem Sinne nur „korrekte“ Codewörter.

Für $X = \emptyset$ ist das nicht klar, da $u(\emptyset)$ eben nicht t_{s-1}-valide ist, ist ja gerade V3 verletzt. Die „korrekten“ Codewörter können wir wie oben behanden. Für die „inkorrekten“ Codewörter $c(a,b,x)$ (bei der sowohl M_a als auch M_b ablehnen) werden wir die Zugehörigkeit zu $u(\emptyset)$ durch ein „Majoritätskriterium“ festlegen, aber werden im Abhängigkeitsgraph keine ausgehenden Kanten einzeichnen. Die Definition lässt zu, dass das Orakel zumindest $t_{\hat{s}}$-valide ist.

Vor der konkreten Definition hier das ungefähre Vorgehen: Relativ zu dem pathologischen $u(\emptyset)$ wird $T_j(T(r(0^n))$ die Ausgabe y berechnen, wobei $y \neq g_m(0^n)$. (Ansonsten hätte τ_j^1 den Transduktor T_j als Beweissystem ausgeschlossen.) Sei nun Q die Menge an Orakelfragen auf dieser obigen Berechnung. Betrachte nun die transitive Menge an Orakelfragen $Q^+ = R_{G(\emptyset)}(Q)$.

Am liebsten würden wir jetzt ein $\alpha \in \Sigma^n$, was nicht in Q^+ liegt, und dann hoffen dass auch $T_j(T(r(0^n))$ relativ zu $u(\{\alpha\})$ die Ausgabe y berechnet. Das klappt tatsächlich, falls Q^+ keine inkorrekten Codewörter bzgl. $u(\emptyset)$ enthält. Dann sind wir sofort fertig.

Im anderen Fall stimmt das leider nicht: die inkorrekten Codewörter in Q^+ haben nicht im Mechanismus des Abhängigkeitsgraphs mitgespielt. Das bedeutet, $u(\emptyset)$ und $u(\{\alpha\})$ stimmen im Allgemeinen nicht auf den inkorrekten Codewörtern in Q^+ überein.

Seien $z_1, \dots, z_k$ die inkorrekten Codewörter aus Q^+. Als Alternative werden wir Wörter $\beta_1, \dots, \beta_k \in \Sigma^n - Q^+$ mit folgenden Eigenschaften finden:

(A1) Für alle $1 \leq i \leq k$ gilt $z_i \in u(\emptyset)$ genau dann wenn $z_i \in u(\{\beta_i\})$, i.e. $u(\emptyset)$ und $u(\{\beta_i\})$ stimmen auf z_i überein.

(A2) Die $\beta_1, \dots$ kommen sich nicht in die Quere. Sei für $1 \leq l \leq k$ die Menge $Q_i^+ = R^+_{G(\{\beta_i\})}(z_i)$ die Orakelwörter, von welcher die Zugehörigkeit von z_i zu $u(\{\beta_i\})$ abhängig ist. Beobachte dass bzgl. $u(\{\beta_i\})$ das Codewort β_i korrekt ist. Die Aussage ist nun, dass $\beta_{i'} \notin Q_i^+$ für alle $i' \neq i$.

Betrachte nun $u(\{\beta_1, \dots, \beta_k\})$. Nach A2 folgt, dass $u(\{\beta_i\})$ und $u(\{\beta_1, \dots, \beta_k\})$ auf Q_i^+ übereinstimmen, und damit auf z_i. Nach A1 folgt dann, dass $u(\{\beta_1, \dots, \beta_k\})$ mit $u(\emptyset)$ auf z_i übereinstimmt. Nachdem das für alle i gilt, stimmt $u(\{\beta_1, \dots, \beta_k\})$ mit $u(\emptyset)$ auf allen inkorrekten Codewörtern überein. Die korrekten Codewörter stimmen schon deshalb überein, weil ja $\beta_1, \dots \notin Q^+$.

Abschließend stimmt also $w = u(\{\beta_1, \dots, \beta_k\})$ mit $u(\emptyset)$ auf Q^+ überein, und damit läuft die Rechnung $T_j(T(r(0^n))$ identisch auf Orakel w. Die Berechnung wird die Ausgabe $y \neq g_m(0^n)$ ausgeben. Außerdem ist $w \cap \Sigma^n = \{\beta_1, \dots\} \neq \emptyset$ und wir haben beide Aussagen der Behauptung erreicht.

Definition of $u(X)$, $G(X)$: Sei $X \subseteq \Sigma^n$. Wir konstruieren $u(X)$ und $G(X) = (V, E)$ induktiv. Fixiere die Knotenmenge $V = \Sigma^{\leq \gamma(n)}$. Basisklauseln:

(1) Für $z \in \Sigma^{<n}$, setze $z \in u(X)$ genau dann wenn $z \in u$.

(2) Für $z \in \Sigma^n$, setze $z \in u(X)$ genau dann wenn $z \in X$.

Induktive Klauseln: Sei $z \in \Sigma^{\leq \gamma(n)}$, $|z| > n$ und $u(X)$ für Wörter $< z$ definiert.

(3) Wenn $z = c(a,b,x)$ für geeignete a, b, x mit $0 < t_s(\tau^1_{a,b}) \leq z$, und *mindestens eine* der Be-

rechnungen $M_a^{u(X)}(x)$ oder $M_b^{u(X)}(x)$ akzeptiert, gehe wie folgt vor:

Markiere den Knoten z als *korrektes Codewort*. Wenn $M_a^{u(X)}(x)$ akzeptiert, dann definiere $z \in u(X)$. Setze $(z, q) \in E$ für alle Orakelfragen q auf dem linkesten akzeptierenden Pfad von $M_a^{u(X)}(x)$.

Anderenfalls akzeptiert $M_b^{u(X)}(x)$, und definiere dann $z \notin u(X)$. Setze $(z, q) \in E$ für alle Orakelfragen q auf dem linkesten akzeptierenden Pfad von $M_b^{u(X)}(x)$.

(4) Anderenfalls, wenn $z = c(a, b, x)$ für geeignete a, b, x mit $0 < t_s(\tau_{a,b}^1) \leq z$, und *keine* der Berechnugen $M^{u(X)}a(x)$ oder $Mb^{u(X)}(x)$ akzeptiert, gehe wie folgt vor:

Markiere den Knoten z als *inkorrektes Codewort*. Definiere $Q_{\text{in}}^z := \{\xi \in \Sigma^n \mid z \in u(\xi)\}$, $Q_{\text{out}}^z := \{\xi \in \Sigma^n \mid z \notin u(\xi)\}$.

Wenn $|Q_{\text{out}}^z| \leq |Q_{\text{in}}^z|$, definiere $z \in u(X)$. Anderenfalls, definiere $z \notin u(X)$. Es werden insbesondere keine Kanten hinzugefügt.

(5) Anderenfalls, $z \notin u(X)$.

Extremale Klausel: (6) Es sind keine weiteren Kanten in E.

Wiederholen wir die folgenden Definitionen, die in Klausel (4) verwendet wurde:

$$Q_{\text{in}}^z := \xi \in \Sigma^n \mid z \in u(\xi), \quad Q_{\text{out}}^z := \xi \in \Sigma^n \mid z \notin u(\xi).$$

Beob. dass $Q_{\text{in}}^z, Q_{\text{out}}^z$ die Menge Σ^n partitioniert.

Wir stellen einige Behauptungen bezüglich $u(X), G(X)$ auf.

Behauptung 5. *(1) Wann immer $X \neq \emptyset$, ist $u(X)$ wohldefiniert, ist t_{s-1}-gültig und $G(X)$ enthält keine inkorrekten Codewörter. (Kann jedoch Wörter der Form $c(\cdot, \cdot, \cdot)$ enthalten, die nicht als korrekte Codewörter markiert sind.)*

(2) Für jedes beliebige $X \subseteq \Sigma^n$ ist $u(X)$ für alle Wörter der Länge $\leq \gamma(n)$ wohldefiniert, $u(X) \cap \Sigma^n = X$ und $u(X) \sqsupsetneq u \sqsupseteq w_{s-1}$.

(3) Wann immer $s' < \hat{s}$, ist $u(\emptyset)$ auch $t_{s'}$-gültig.

(4) Sei $z = c(a, b, x)$ ein korrektes Codewort in $G(X)$. Die folgenden Aussagen sind äquivalent: (a) $z \in u(X)$, (b) $M_a^{u(X)}$ akzeptiert, (c) $M_a^{u(X)}$ akzeptiert oder $M_b^{u(X)}$ lehnt ab.

(5) $G(X)$ bildet einen gerichteten azyklischen Graphen (der nicht unbedingt verbunden ist). Insbesondere gilt für jede gerichtete Kante vom Knoten a nach b, dass $a > b$.

Behauptung 6. *Sei $X \subseteq \Sigma^n$ und $Q \subseteq \Sigma^{\leq \gamma(n)}$. Dann gilt $\ell(R_{G(X)}(Q)) \leq 2 \cdot \ell(Q)$.*

Orakel mit NP = coNP und SAT und alle Paare aus DisjNP sind P-separierbar

Wir verstehen die relativierte Version von SAT als die Nicht-Existenz eines p-optimalen Beweissystems zur Menge SAT^O. Das ist die Menge der erfüllbaren aussagenlogischen Formeln, welche zusätzlich die Prädikate $\texttt{query}_m(a_1, a_2, \dots, a_m)$ verdenden darf. Interpretation: $\texttt{query}_m(a_1, a_2, \dots, a_m)$ evaluiert zu 1 genau dann wenn $a_1 a_2 \cdots a_m \in O$.

Nach Dingel wissen wir:

Lemma 1. *Sei $f, g \in \mathrm{FP}^O$ mit $g(\Sigma^*) \subseteq f(\Sigma^*)$. Wird g nicht von f p-simuliert, also*

$$\forall h \in \mathrm{FP}^O \, \exists x \in \Sigma^*. \, f(h(x)) \neq g(x),$$

dann ist f nicht p-optimal für $f(\Sigma^*)$.

Sei $e(0) = 2, e(i+1) = 2^{e(i)}$. (Doppelt exponentiell!) Starte mit PSPACE-vollständiger Menge C welche keine Wörter der Länge $e(\cdot)$ enthält.

Sei hier $h \colon \mathbb{N} \times \mathbb{N} \to e(\mathbb{N})$ eine injektive Funktion mit folgender Eigenschaft:

$$h(j, r) = e(i) \implies t_j(t_r(e(i))) < 2^{e(i)}.$$

(Wörter der Länge $h(j, r)$ ist die Ebene zugehörig zum Zeugen-Beweissytem g_j und Übersetzungsfunktion T_r.) Diese soll in Polynomialzeit berechenbar und invertierbar sein. Zur Einfachheit schreiben wir $H_j = \{h(j, r) \mid r \in \mathbb{N}\}$.

Idee für SAT: wir erreichen, dass alle Funktionen $T_j \in \mathrm{FP}$ entweder kein Beweissystem für relativiertes SAT sind, oder dass diese nicht das folgende Zeugen-Beweisystem g_j p-simulieren (wobei m von T_j abhängig ist). Wir setzen

$$\varphi_n = \text{„}\texttt{query}_n(a_1, \dots, a_n)\text{“}$$

und damit ist

$$\varphi_n \in \mathrm{SAT}^O \iff \exists w \in O \cap \Sigma^n.$$

Sei dann

$$g_j(x) = \begin{cases} \varphi_n & \text{falls } x = 0^n \text{ und } n = h(j, r) \\ \text{„}a_1 \vee \neg a_1\text{“} & \text{sonst.} \end{cases}$$

Fakt: gilt $O \cap \Sigma^n \neq \emptyset$ für alle $n \in H_j$, dann ist $g_j(\Sigma^*) \subseteq \mathrm{SAT}^O$.

Wollen wir also erreichen, dass das („inhärente“) Beweissystem T_j mit $T_j(\Sigma^*) = \mathrm{SAT}^O$ nicht p-optimal ist, genügt es für alle Kandidaten T_r an Übersetzungsfunktionen zu sichern, dass $T_j(T_r(0^n)) \neq g_j(0^n)$ für geeignetes $n = h(j, r)$, und gleichzeitig $O \cap \Sigma^n \neq \emptyset$ für fast alle $n \in H_j$.

Wir können dieses Vorgehen sogar noch abschwächen: für jede *Funktion* (und nicht Beweissystem) T_j und Übersetzungsfunktion T_r bestimmen wir in *jeder* Stufe $n = h(j, r)$ die Ausgabe $y = T_j(T_r(0^n))$ (relativ zu einem Orakel w mit $w \cap \Sigma^n = \emptyset$.

- Falls $y = \varphi_n$ lassen wir die Stufe n leer, und T_j ist dann sicher kein Beweissystem.
- Falls $y \neq \varphi_n$ setzen wir (mindestens) ein Wort $\alpha \in \Sigma^n$ in die Stufe n ein welches nicht erfragt wurde. Dann hat T_r nicht den g_j-Beweis für φ_n in einen T_j-Beweis übersetzt.

Dieses Vorgehen hat den Vorteil, dass die Entscheidung $O \cap \Sigma^{e(i)} \stackrel{?}{=} \emptyset$ in P^O liegt: Bestimme j, r sodass $e(i) = n = h(j, r)$. Teste ob $T_j(T_r(0^n)) = \varphi_n$, aber beantworte hierbei Orakelfragen der Länge n negativ, alle anderen leitest Du weiter.

Idee für P-Separierung von DisjNP**:** Es reicht im Wesentlichen aus, so viele DisjNP-Paare wie möglich zu zerstören. Um nun ein Paar $(L(M_a^O), L(M_b^O))$ zu P-separieren reicht es aus, den Inhalt einer relevanten Stufe $O \cap \Sigma^{e(i)}$ zu bestimmen. Wie oben können wir schon in P^O erkennen, ob $O \cap \Sigma^{e(i)} = \emptyset$. Falls ja sind wir schon fertig.

Falls nein, können wir über C können wir akzeptierende Berechnungen eines Paars $(L(M_a^O), L(M_b^O))$ bestimmen und iterativ uns die relevanten Stufen $e(\cdot)$ rekonstruieren. Wir

machen hierbei in jeder Iteration Fortschritt, weil sonst das Orakel O so hätte konstruiert werden können, dass sowohl M_a^O als auch M_b^O eine Eingabe akzeptieren.

Konstruktion 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: wenn $n \notin \operatorname{img} h$, dann gilt $w \cap \Sigma^n = \emptyset$.

V3 Wenn $T_j^w(T_r^w(0^n))$ definit ist, dann gilt

$$w \cap \Sigma^n = \emptyset \iff T_j^w(T_r^w(0^n)) = \varphi_n.$$

V4 Für alle $n = e(i)$ gilt: wenn $i \notin \operatorname{img} t$, dann gilt $|w \cap \Sigma^n| \leq 1$.

V5 Wenn $t(\tau_{a,b}^1) = i$, dann existiert ein $z \in \Sigma^*$ und je ein definitiv akzeptierender Rechenweg von $M_a^w(z)$ mit Orakelfragen Q_a, bzw. von $M_b^w(z)$ mit Orakelfragen Q_b. Außerdem existieren $\alpha \in Q_a$, $\beta \in Q_b$ und ein γ sodass $w \cap \Sigma^{e(i)} = \{\alpha, \beta, \gamma\}$.

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

Überlegungen zu Genericity

Definition 1.
- *Eine* Bedingung *ist eine partielle Funktion von* $\omega \to \{0,1\}$ *mit endlichem Definitionsbereich.*
- *Zwei Bedingungen* σ, τ *nennen wir* konsistent *wenn* $\sigma \cup \tau$ *auch eine Bedingung ist, d.h.* $a \in \mathrm{dom}(\sigma) \cup \mathrm{dom}(\tau) \implies \sigma(a) = \tau(a)$.
- *Eine Bedingung* τ erweitert *eine Bedingung* σ *(*$\tau \succeq \sigma$*) falls* $\mathrm{dom}(\sigma) \subseteq \mathrm{dom}(\tau)$ *und* $\sigma(a) = \tau(a)$ *für alle* $a \in \mathrm{dom}(\sigma)$*. Wenn* $\sigma \neq \tau$ *schreiben wir auch* $\tau \succ \sigma$.
- *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* $a \in \mathrm{dom}(\sigma)$.

Definition 2. *Ein* Begriff der Generizität $\mathcal{G}$ *ist eine nichtleere Klasse an Bedingungen mit folgenden Eigenschaften:*

(1) (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* $a \in \mathrm{dom}(\gamma')$.

(2) (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.

Sei im Folgenden $\mathcal{G}$ ein beliebiger aber fester Begriff von Generizität. Im Folgenden ist $\mathcal{L}_{\mathrm{PA}}[X]$ die Sprache der Peano-Arithmetik ($=$, $+$, $\times$, S, 0), mit einem zusätzlichen unären Prädikatsymbol X. Sei $\mathrm{sent}(\mathcal{L}_{\mathrm{PA}}[X])$ die Menge der Sätze in $\mathcal{L}_{\mathrm{PA}}[X]$. Ohne Beschränkung sind die einzigen logischen Operationen $\neg, \vee, \exists$. Für $n \in \omega$ bezeichnen wir mit $\overline{n}$ den Term $\underbrace{SS\cdots S}_{n}0$ in $\mathcal{L}_{\mathrm{PA}}[X]$.

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 Wahrheitsdefinition entspricht, außer im Fall der Negation. Eine Bedingung τ *erweitert* eine Bedingung γ, wenn $\tau \succ \gamma$. Die Intuition hinter dem Begriff „erweitert“ ist, dass τ das Orakel vollständiger spezifiziert als γ. Somit besagt die Klausel (5) grob, dass $\neg\varphi$ erzwungen wird, wenn wir φ niemals erzwingen können, indem wir unsere Annäherung an das Orakel verfeinern.

Definition 3 (Forcing-Relation)**.** *Die Variablen* γ *und* τ *erstrecken sich über* $\mathcal{G}$*. Es gilt:*

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

Definition 4. *Wir schreiben* $G \Vdash \varphi$ *wenn ein* $\gamma \in \mathcal{G}$ *existiert mit* $G \succ \gamma$ *und* $\gamma \Vdash \varphi$.

Definition 5. *Eine Menge* $G \subseteq \omega$ *ist* $\mathcal{G}$-generisch *wenn folgende Eigenschaften erfüllt sind:*

(1) Für alle $\sigma_1, \sigma_2 \in \mathcal{G}$ *gilt: wenn* $G \succ \sigma_1, \sigma_2$ *(also insb. auch konsistent), dann existiert auch eine Bedingung* $\tau \in \mathcal{G}$ *sodass* $G \succ \tau \succeq \sigma_1 \cup \sigma_2$.

(2) 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* $\gamma \Vdash \varphi \vee \neg\varphi$.

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

Lemma 6 (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 2(1). Andernfalls, setze $\gamma_i = \sigma$.

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 A wohldefiniert. Ferner gilt $A \succ \gamma_i$ für alle i: angenommen $A \not\succ \gamma_i$, 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 gilt $\gamma_i(k) = \gamma_k(k)$; Widerspruch zu oben. Also gilt (2).

Damit gilt auch (3): Für beliebiges $n \in \omega$ existiert $\gamma_n \in \mathcal{G}$ mit $G \succ \gamma_n$ und $n \in \mathrm{dom}(\gamma_n)$. Die Eigenschaft (1) folgt unmittelbar aus Grundlegendheit bzw. 2(2). □

Lemma 7 (Erzwingen ist Wahrheit)**.** *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 $\{0,1\}^{\mathbb{N}}$.

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

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

  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 4(3).
- Lemma klar für Disjunktionen und Existenzquantor: Induktionsannahme verwenden.
- 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 4(2) dass $\gamma \Vdash \varphi \vee \neg\varphi$. Wenn also nach Voraussetzung für alle solche γ' schon $\gamma' \nVdash \neg\varphi$, dann muss $\gamma \Vdash \varphi$.

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

  Um die zweite Implikation zu sehen, nimm ann dass eine $\mathcal{G}$-Bedingung $\gamma' \prec G$ existiert mit $\gamma' \Vdash \varphi$. Dann gilt $\gamma, \gamma' \prec G$ und nach Definition 4(1) existiert eine $\mathcal{G}$-Bedingung τ mit $\gamma \cup \gamma' \preceq \tau \prec G$. Dann haben wir aber $\tau \Vdash \varphi$ und $\tau \Vdash \neg\varphi$. Aus Letzterm folgt insbesondere $\tau \nVdash \varphi$; Widerspruch zu Ersterem. Letzte Implikation ist Induktionsannahme. □

Unter dieser Definition von Forcing gilt insbesondere:

- $\gamma \Vdash \varphi \implies \gamma \Vdash \neg\neg\varphi$ (*"strong forcing implies weak forcing"*): Induktion über Formeln.
- $[(\forall A \succ \gamma).\omega[A] \vDash \varphi] \implies \gamma \Vdash \varphi$: TODO

Lemma 8. *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 Erweiterungenen 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 6 gilt $G \Vdash \varphi$, nach Definition existiert ein $\sigma \prec G$ mit $\sigma \Vdash \varphi$.

Nun sind γ und σ konsistent: gilt $a \in \mathrm{dom}(\gamma) \cap \mathrm{dom}(\sigma)$ dann gilt $\gamma(a) = G(a) = \sigma(a)$. Nach Definition 2(2) ist also auch $\tau' = \gamma \cup \sigma \in \mathcal{G}$ und wir haben $\tau' \succeq \gamma$ und $\tau' \Vdash \varphi$; 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 existiert eine $\mathcal{G}$-generische Menge $G \succ \tau$. Nach Lemma 6 gilt $\omega[G] \vDash \varphi$. □

Beachte dass $\omega[G] \nvDash \neg\varphi$ genau dann wenn $\omega[G] \vDash \varphi$. Wir haben dadurch

Korollar 9. *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 $G \succ \gamma$.*

Korollar 10. *Für alle $\varphi \in \mathrm{sent}(\mathcal{L}_{\mathrm{PA}}[X])$ gilt: Ist die Menge an $\mathcal{G}$-Bedingungen, welche φ erzwingen, dicht, also*

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

dann gilt auch $\gamma \Vdash \neg\neg\varphi$ für alle $\gamma \in \mathcal{G}$. Insbesondere wird unter allen $\mathcal{G}$-generischen Mengen der Satz φ als wahr interpretiert.

Beweis. Sei $\gamma \in \mathcal{G}$ beliebig. Wir haben nach Voraussetzung dass

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

Für die zweite Aussage sei G eine beliebige $\mathcal{G}$-generische Menge. Es existiert eine Bedingung $\sigma \in \mathcal{G}$ mit $G \succ \sigma$; das folgt schon aus Definition 4(2) für einen beliebigen Satz, z.B. $\varphi = X(\overline{0})$. Wie bereits gezeigt $\sigma \Vdash \neg\neg\varphi$ und nach vorigem Lemma gilt dann $\omega[G] \vDash \varphi$. □

Beispiel: BGS-Konstruktion

Es ist unmittelbar klar, dass die Menge

$$L(G) = \{0^n \mid \exists x \in G, |x| = n\} \in \mathrm{NP}^G$$

liegt. Wir zeigen nun, wie sich durch Forcing ein Orakel G angeben lassen kann, sodss $L(G)$ nicht in P^G liegt. Dafür werden wir als Begriff der Generizität die *Cohen*-Generizität (im Folgenden $\mathcal{G}$) verwenden, welche aus allen partiellen Funktionen mit endlichem Definitionsbereich besteht.

Wir zeigen nun, dass für jedes

$$\varphi_i : \neg\text{„}L(M_i^X) = L(X)\text{“}.$$

die jeweilige Menge, welche φ_i erzwingt, dicht ist. (Maschine M_i^X ist eine deterministische Poly-OTM.) Nach 9 sind wir dann fertig und es existiert ein $\mathcal{G}$-generische Orakel mit $\mathrm{P}^G \neq \mathrm{NP}^G$.

Sei daher $\gamma \in \mathcal{G}$. Wir müssen zeigen dass ein $\tau \succeq \gamma$ existiert dass in $\mathcal{G}$ liegt und $\tau \Vdash \varphi_i$. Zunächst wählen wir ein $n \in \omega$ sodass γ keine Wörter der Länge n definiert, und sodass $2^n > p_i(n)$.

Nun wollen wir ein σ angeben sodass $M_i^\sigma(0^n)$ definiert ist – hier ist $M_i^\sigma(0^n)$ so definiert dass Queries q mit $\sigma(q)$ beantwortet werden, und falls $q \notin \mathrm{dom}(\sigma)$, dann ist $M_i^\sigma(0^n)$ nicht definiert.

Das lösen wir, in dem wir, wenn immer ein noch nicht definierter Query q gestellt wird, $\sigma(q) = 0$ setzen. Präziser: starte mit $\sigma_0 = \gamma$. Setze für $k > 0$

$$\sigma_k = \begin{cases} \sigma_{k-1} \cup \{q \mapsto 0\} & \text{falls } M_i^{\sigma_{k-1}}(0^n) \text{ nicht definiert und } q \notin \mathrm{dom}(\sigma_{k-1}) \text{ der erste undef. Query auf dem Rechenweg ist} \\ \sigma_{k-1} & \text{sonst.} \end{cases}$$

Sei $\sigma = \sigma_N$ mit $N = t_i(n)$. Hierfür ist $M_i^\sigma(0^n)$ definiert, existieren ja höchstens N viele Queries auf jedem Rechenweg.

Ist nun $M_i^\sigma(0^n)$ akzeptierend, dann setze

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

Dann ist $\tau \in \mathcal{G}$ (endliche Definitionsmenge), und für alle $A \succ \tau$ haben wir $\omega[A] \nvDash$ „$L(M_i^X) = L(X)$“: Denn $\omega[A] \vDash$ „$M_i^X(0^n)$ akz.“ (A und γ stimmen auf den Orakelfragen überein) und $\omega[A] \vDash$ „$0^n \notin L(X)$“ ($A(x) = \gamma(x) = 0$ für alle $x \in \Sigma^n$).

Also haben wir nach Lemma 8 dass $\tau \Vdash \neg$„$L(M_i^X) = L(X)$“ bzw. $\tau \Vdash \varphi_i$ wie gewünscht.

Symmetrisch falls $M_i^\sigma(0^n)$ ablehnt. Sei $a \in \Sigma^*$ mit $a \notin \mathrm{dom}(\sigma)$ (existiert da $2^n > t_i(n)$), und setze

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

Wir haben für alle $A \succ \tau$ dass $\omega[A] \vDash$ „$M_i^X(0^n)$ abl.“ und $\omega[A] \vDash$ „$0^n \in L(X)$“ ($A(a) = \gamma(a) = 1$ für $a \in \Sigma^n$). Wiedernach Lemma 8 dann $\tau \Vdash \varphi_i$ wie gewünscht.

Relativierung von Forcing

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.

Außerdem können zwei unäre Prädikate in arithmetischen Formel in ein einziges Prädikat überführt werden, mittels des Join-Operators $\oplus$:

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

Gegeben Formel φ über $\mathcal{L}_{\mathrm{PA}}[X, B]$ können wir effektiv eine Formel ψ über $\mathcal{L}_{\mathrm{PA}}[X]$ finden sodass für alle $A, B \in \omega$ gilt:

$$\omega^B[A] \vDash \varphi \text{ genau dann wenn } \omega[A \oplus B] \vDash \psi.$$

TODO wie geht der Trick mit Re-Relativierung bzgl. QBF?

Beispiel: keine UP-vollständige Menge und $\neg Q$

Eine generische Konstruktion, welche vollständige Elemente einer Klasse ausschließt, ist komplizierter. Hier liegt ein Abhängigkeitsverhältnis zwischen Maschinen und den Zeugensprachen vor. (Wenn Sprache in UP, dann auch entsprechende Zeugensprache in UP.) Aber auch hier können wir analog zu dem sonst üblichen Vorgehen von Dose/Glaßer/Egidy fahren und „falls möglich, zerstöre Maschine, ansonsten sichere dass die entsprechende Zeugensprache in UP und schließe Reduktionen aus“.

Definiere $\mathrm{tower}(1) = 2$, $\mathrm{tower}(k+1) = 2^{2^{\mathrm{tower}(k)}}$. Definiere polynomialzeitberechenbare und -invertierbare Familie $\{H_i\}_{i \in \omega}$ wobei

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

und alle Mengen der Familie sind paarweise disjunkt. 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 $x, y \in \Sigma^n$.

Definiere nun folgenden Begriff der Generizität **UPC**: Eine Bedingung γ ist in **UPC** genau dann wenn

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

(2) wenn $\gamma(x) = 1$, dann $|x| = \mathrm{tower}(k)$ für ein $k \in \omega$, und

(3) für alle $i \in \omega$ gilt: wenn ein $n \in H_i$ existiert mit $a, b \in \Sigma^n$, $a \neq b$, $\gamma(a) = \gamma(b) = 1$, dann existiert eine Eingabe z sodass $N_i^\sigma(z)$ auf zwei Rechenwegen akzeptiert.

TODO zeigen dass das ein Begriff der Generizität ist! Insbesondere ist ϵ, die nirgends definierte Funktion, eine **UPC**-Bedingung.

Betrachte folgende Aussage:

$$\varphi_{i,r}\colon \neg„N_i^X \text{ ist eindeutig}“ \vee \neg„L_i(X) \leq_{\mathrm{m}}^{\mathrm{pp}} L(N_i^X) \text{ via } T_r^X“.$$

Wir wollen nun zeigen dass die Menge an **UPC**-Bedingungen welche $\varphi_{i,r}$ erzwingen jeweils dicht sind, also $\gamma \Vdash \neg\neg\varphi_{i,r}$ für alle $\gamma \in \mathbf{UPC}$. Dann sind wir auch schon fertig:

Behauptung 11. *Angenommen $\gamma \Vdash \neg\neg\varphi_{i,r}$ 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 ist N_i^G eindeutig akzeptierend.

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 gilt $\epsilon \in \mathbf{UPC}$ und nach Voraussetzung gilt $\epsilon \Vdash \neg\neg\varphi_{i,r}$. Nach Korollar 9 gilt also insbesondere für unser $G \succ \epsilon$ dass $\omega[G] \vDash \varphi_{i,r}$. Da nach Annahme aber $\omega[G] \nvDash \neg„N_i^X$ ist eindeutig“ haben wir

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

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

Wir zeigen nun dass $L_i(G) \in \mathrm{UP}^G$. Dann sind wir auch schon fertig, denn dann ist $L(N_i^X)$ nicht vollständig. Angenommen $L_i(G) \notin \mathrm{UP}^G$. Dann existiert ein $n \in H_i$ und zwei verschiedene $a, b \in \Sigma^n$ mit $a, b \in G$. Nach Definition 5(3) existiert ein $\sigma \in \mathbf{UPC}$ mit $G \succ \sigma$ und $a, b \in \mathrm{dom}(\sigma)$. Insbesondere gilt $\sigma(a) = \sigma(b) = 1$ (ansonsten $G \not\succ \sigma$). Also gilt nach Definition [UPC](3) dass eine Eingabe z existiert sodass $N_i^\sigma(z)$ auf zwei Rechenwegen akzeptiert. Also gilt auch für $G \succ \sigma$ dass $N_i^G(z)$ auf zwei Rechenwegen akzeptiert. Dann ist aber N_i relativ zu G nicht mehr eindeutig akzeptierend; Widerspruch zur Wahl von N_i. □

Wir zeigen nun wie angekündigt die Dichtheit:

Behauptung 12. *Sei $\gamma \in \mathbf{UPC}$, $i, r \in \omega$ beliebig. Es gilt $\gamma \Vdash \neg\neg\varphi_{i,r}$.*

Beweis. Sei $\gamma \in \mathbf{UPC}$ beliebig. Wir zeigen dass ein $\tau \in \mathbf{UPC}, \tau \succeq \gamma$ existiert mit $\tau \Vdash \varphi$. Nach Korollar 10 ist das ausreichend um $\gamma \Vdash \neg\neg\varphi_{i,r}$ zu zeigen.

Wähle ein $n \in H_i$ sodass γ kein Wort der Länge n definiert. Definiere nun für $S \subseteq \Sigma^n$ die Bedingung σ_S mit

$$\sigma_S(x) = \begin{cases} \gamma(x) & \text{falls } x \in \mathrm{dom}(\gamma), \\ 1 & \text{falls } x \notin \mathrm{dom}(\gamma), x \in S, \\ 0 & \text{falls } x \notin \mathrm{dom}(\gamma), x \notin S, |x| \leq p_i(p_r(n)), \\ \text{undef.} & \text{sonst.} \end{cases}$$

Wir haben $\sigma_S \succeq \gamma$. Es gilt $\sigma_\emptyset \in \mathbf{UPC}$ und $\sigma_{\{a\}} \in \mathbf{UPC}$ für alle $a \in \Sigma^n$: (1) Offenbar $\mathrm{dom}(\sigma_S) = \Sigma^{\leq p_i(p_r(n))}$. (2) Wenn $\sigma_S(x) = 1, x \notin \mathrm{dom}(\gamma)$ dann nach Definition $x \in S$, $|x| = n \in H_i$ also $n = \mathrm{tower}(k)$ für ein k. Auch (3) 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

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

oder für ein $a \in \Sigma^n$

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

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

Für den anderen Fall gilt

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

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

$$\tau \Vdash \neg\text{„}N_i^X \text{ ist eindeutig“ und damit } \tau \Vdash \varphi_{i,r}.$$

Nach Annahme und Lemma 8 existiert ein $A \succ \sigma_\emptyset \succeq \gamma$ mit $\omega[A] \vDash \text{„}L_i(X) \leq_{\mathrm{m}}^{\mathrm{pp}} L(N_i^X)$ 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 $N_i(T_r(0^n)) = 1$ relativ zu β_a (β_a fixiert alle Orakelfragen), also wieder

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

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

Ein kombinatorisches Standardargument zeigt aus Leterem 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 Kompatiblität, 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.

Auf gleiche Weise sehen wir $\sigma_{\{a,b\}} \succeq \beta_b$. Damit gilt für alle $C \succ \sigma_{\{a,b\}}$ dass $N_i^C(T_r^C(0^n)) = 1$, und es existieren auf dieser Rechnung zwei akzeptierende Rechenwege, je eine mit Orakelfragen $\mathrm{dom}(\beta_a)$ und eine mit Orakelfragen $\mathrm{dom}(\beta_b)$. Wir zeigen gleich dass diese zwei Rechenwege nicht gleich sind. Damit haben wir dann

$$(\forall C \succ \sigma_{\{a,b\}}).\, \omega[C] \nvDash \text{„}N_i^X \text{ ist eindeutig“},$$

(der implizite $\forall$-Quantor nicht erfüllt für Eingabe $y = T_r^X(0^n)$) nach Korollar 9 also

$$\sigma_{\{a,b\}}) \Vdash \neg\text{„}N_i^X \text{ ist eindeutig“}$$

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

Wir zeigen nun dass (für jedes 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 unterscheidliche 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

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

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