§2 Advanced Computability

- WHILE programs
- UTM Theorem
- Normalform Theorem
- SMN Theorem / Currying (Schönfinkeling)
- Fixedpoint Theorem, Quines
- Rice-Myhill-Shapiro
- Oracle WHILE programs, Higher Halting Problem
- Arithmetic Hierarchy: semantially & syntactically
- Post\`s Problem / Friedberg&Muchnik\`s Proof

WHILE Programs

Syntax in Backus—Naur Form:

body better modify x_j

$P := (\, x_j := 0 \mid x_j := x_i \pm 1 \mid P \,;\, P \mid$
$\text{LOOP } x_j \text{ DO } P \text{ END} \mid \text{WHILE } x_j \text{ DO } P \text{ END}\,)$

Semantics: loop executed as long as $x_j \neq 0$

Observation: a) To every LOOP program P there is an equivalent WHILE program P' with*out* LOOPs.

b) As opposed to LOOP programs, WHILE programs have *un*decidable Halting Problem.

Rado's Corollary: WHILE programs do **not** admit a bound $t(P,n)$ such that P on input $\underline{x} \in \mathbb{N}^k$ either at most $t(P,\|\underline{x}\|_1)$ steps or runs indefinitely.

First UTM Theorem

UTM-Theorem: There exists a LOOP program U' that, given $\langle P\rangle\in\mathbb{N}$ and $\langle x_1,\ldots,x_k\rangle\in\mathbb{N}$ and $N\in\mathbb{N}$, simulates P on input $(x_1,\ldots,x_k)$ for N steps.

Proof (Sketch): Use one variable y for $\langle x_1,\ldots,x_k\rangle$, and z to store the current program counter of P:

Switch/case $\langle P\rangle[z]$ of:

„x_j:=0" : $\langle x_1,\ldots x_j,\ldots,x_k\rangle := \langle x_1,\ldots 0,\ldots,x_k\rangle$; $z{:=}z+1$

„x_j:=x_i+1" : $\langle x_1,\ldots x_j,\ldots x_k\rangle := \langle x_1,\ldots x_i+1\ldots x_k\rangle$; $z{:=}z+1$

„WHILE x_j DO" : if $x_j=0$ then $z:=1+\#$of corresponding END

„END" : $z :=$ line# of corresponding WHILE

Definition: Let $\langle P\rangle\in\mathbb{N}$ denote the encoding of WHILE program P (e.g. as ascii sequence).

Normalform Theorem

UTM-Theorem: There exists a LOOP program U' that, given $\langle P\rangle\in\mathbb{N}$ and $\langle x_1,\ldots,x_k\rangle\in\mathbb{N}$ and $N\in\mathbb{N}$, simulates P on input $(x_1,\ldots,x_k)$ for N steps.

Normalform-Thm: To every WHILE program P there exists an equivalent one P' containing only one WHILE command (and several LOOPs).

Normalform Theorem 2: Decision problem $L\subseteq\mathbb{N}$ is semi-decidable (by a WHILE program) iff $L = \{\, x\in\mathbb{N} : \exists y\colon \langle x,y\rangle\in V \,\}$ for some decidable $V\subseteq\mathbb{N}$

SMN Theorem: Currying

Definition: Let $C = \langle P \rangle \in \mathbb{N}$ denote the encoding of WHILE program P , $P = \rangle C \langle$ its inverse/decoding.

Type conversion **example**

$f(x,y) = \sin(x) \cdot e^y$

SMN-Theorem: There exists a WHILE program that, given $\langle P \rangle \in \mathbb{N}$ and $x \in \mathbb{N}$, returns $\langle P(x,\cdot\,) \rangle$, where $P(x,\cdot\,)(\underline{y}) :\equiv P(x,\underline{y})$

UTM-Theorem: There is a WHILE program that, given $\langle P \rangle \in \mathbb{N}$, returns $\langle Q \rangle \in \mathbb{N}$ with $Q(x,\underline{y}) = \rangle P(x) \langle\, (\underline{y})$

WHILE program that, given $\langle P \rangle, \langle Q \rangle$, returns $\langle Q \circ P \rangle$

Fixedpoint Theorem and Quines

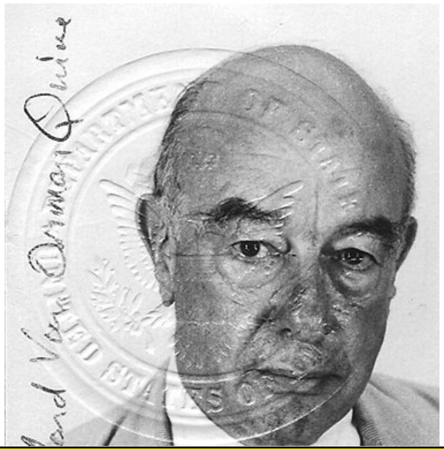
Def: For partial functions $f,g:\subseteq\mathbb{N}\to\mathbb{N}$ write $f\equiv g$ to mean $\mathrm{dom}(f)=\mathrm{dom}(g)$ and $\forall x\in\mathrm{dom}: f(x)=g(x)$.

$\rangle\langle P\rangle\langle = P$ $\langle\rangle C\langle\rangle \equiv C$ $x\equiv y \;:\Leftrightarrow\; \rangle x\langle \equiv \rangle y\langle$

Theorem: Every total computable function $\varphi:\mathbb{N}\to\mathbb{N}$ has a „semantic fixedpoint", i.e. $x\in\mathbb{N}$ s.t. $\varphi(x) \equiv x$.

Proof: Let $\varphi(x)=\varphi\circ\psi'(\langle\varphi\circ\psi\rangle)$, where $\psi(y):=\rangle y\langle\,(y)$ partial!

$\psi'(y) := \langle\, z \to \rangle\psi(y)\langle\,(z)\,\rangle$ <u>semantic extension</u> of $\psi(y)$



Application (Quines):

Let $\mathcal{A}=\mathcal{A}(p,y)$ be a program.

Consider "fixedpoint" P of $\varphi(p) := \langle\mathcal{A}(p,\cdot\,)\rangle$.

$\mathbb{N}\ni\langle P\rangle$ = code of program P, $\rangle C\langle$ = program with code $C\in\mathbb{N}$

Rice's Theorem

Un/Decidable?: a) syntactical correctness ✓

b) $\{ \langle P \rangle : \langle P \rangle$ is ≤1000 characters long $\}$ ✓

c) $\{ \langle P \rangle : P$ makes ≤1000 steps (on input ε) $\}$ ✓

d) $\{ \langle P \rangle : P$ terminates (on input ε) $\} = H$ ↯

e) $\{ \langle P \rangle : L(P) \neq \emptyset \} = N$ ↯

where $L(P) \subseteq \mathbb{N}$ denote the set semi-decided by P.

Theorem (Rice-Myhill-Shapiro): Fix $S \subseteq 2^{\mathbb{N}}$.

Suppose $L^- \notin S$ and $L^+ \in S$ are semi-decidable.

Then $\mathcal{L}(S) := \{ \langle P \rangle : L(P) \in S \}$ is <u>un</u>decidable.

Rice‘s Theorem

„Any non-trivial <u>semantic</u> *property of a given program is undecidable*"

Proof: First suppose $\varnothing\in S$. Given P, decide "$\langle P\rangle\in H$" so:

- Construct from P a program Q which
  - first performs P (and doesn't terminate if P doesn't)
  - then invokes the program semi-deciding L^-.
- Q semi-decides $\varnothing\in S$ if $\langle P\rangle\notin H$ and $L^-\notin S$ else.
- Case $\varnothing\notin S$: Let Q first perform P, then semi-decide $L^+\in S$

Theorem (Rice-Myhill-Shapiro): Fix $S\subseteq 2^{\mathbb{N}}$.

Suppose $L^-\notin S$ and $L^+\in S$ are semi-decidable.

Then $\mathcal{L}(S) := \{\ \langle P\rangle : L(P)\in S\ \}$ is <u>un</u>decidable.

Oracle WHILE programs

$P^{\varphi} := (x_j := 0 \mid x_j := x_i \pm 1 \mid P ; P \mid x_j := \varphi(x_i) \mid$
$\text{LOOP } x_j \text{ DO } P \text{ END} \mid \text{WHILE } x_j \text{ DO } P \text{ END})$

Fix some *arbitrary* total $\varphi:\mathbb{N}\to\mathbb{N}$

Examples:

- $\varphi := \chi_{\mathbb{P}}$ characteristic function of Primality Probl.
- $\varphi := \chi_H$ characteristic function of Halting Problem
- $\varphi := \chi_T$ characteristic function of Totality Problem

$\chi_{\mathbb{P}} \preccurlyeq \chi_H \equiv \chi_{\bar{H}} \preccurlyeq \chi_T$ (cmp. set cardinalities…)

For $\psi,\varphi:\mathbb{N}\to\mathbb{N}$ write $\psi\preccurlyeq\varphi$ if there is a WHILE program with oracle φ computing ψ.

a) φ computable $\Rightarrow$ so ψ

b) $\psi\preccurlyeq\varphi\preccurlyeq\chi \Rightarrow \psi \preccurlyeq \chi$

Higher Halting Problems

$P^\varphi := (x_j := 0 \mid x_j := x_i \pm 1 \mid P ; P \mid x_j := \varphi(x_i) \mid$
$\text{LOOP } x_j \text{ DO } P \text{ END} \mid \text{WHILE } x_j \text{ DO } P \text{ END})$

Fix some *arbitrary* total $\varphi:\mathbb{N}\to\mathbb{N}$

$H^L := \{ \langle P\rangle\in\mathbb{N} : P^L$ terminates (on input ε) $\}$

Lemma: a) H^L is <u>semi</u>-decidable with oracle L

b) but not decidable with oracle L: $H^L \not\preccurlyeq L$

Hierarchy: $\ldots H^{H^H} \not\preccurlyeq H^H \not\preccurlyeq H$

For $\psi,\varphi:\mathbb{N}\to\mathbb{N}$ write $\psi\preccurlyeq\varphi$ if there is a WHILE program with oracle φ computing ψ.

a) φ computable $\Rightarrow$ so ψ

Identify L with χ_L, $L\subseteq\mathbb{N}$

Semantic Arithmetic Hierarchy

M. Ziegler

$P^{\varphi} := (x_j := 0 \mid x_j := x_i \pm 1 \mid P ; P \mid x_j$

LOOP x_j DO P END | WHILE x_j D

Fix some *arbitrary* t

Def: $\Delta_1 = \Sigma_0 = \Pi_0 =$ decidable

Δ_{k+1} = decidable$^{\Sigma_k}$ = decidable$^{\Pi_k}$

Σ_{k+1} = semi-decidable$^{\Sigma_k}$

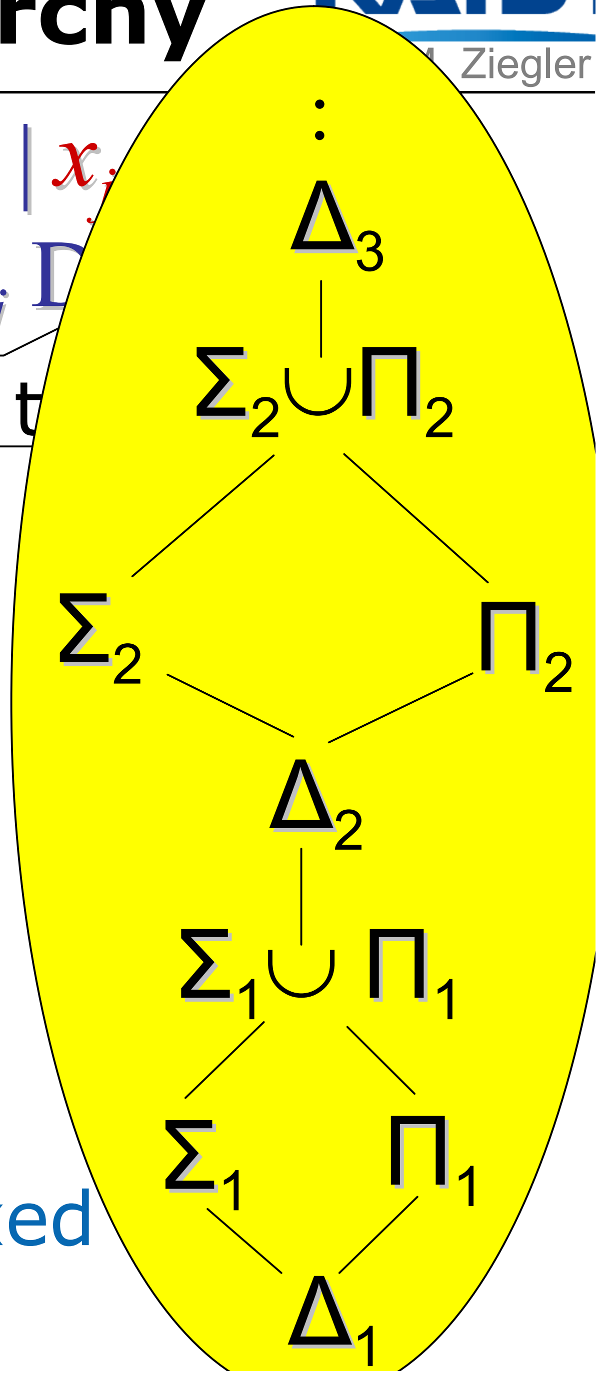
Π_{k+1} = co-semi-decidable$^{\Sigma_k}$

Lemma: a) Δ_k = co-Δ_k

b) $\Delta_k = \Sigma_k \cap \Pi_k$

c) $\Sigma_k \cup \Pi_k \subseteq \Delta_{k+1}$

any single fixed oracle $L \in \Pi_k$



Syntactic Arithmetic Hierarchy

$P^{\varphi} := (x_j := 0 \mid x_j := x_i \pm 1 \mid P ; P \mid x_j$

$\text{LOOP } x_j \text{ DO } P \text{ END} \mid \text{WHILE } x_j \text{ D}$

Fix some *arbitrary* t

Def: $\Delta_1 = \Sigma_0 = \Pi_0$ = decidable

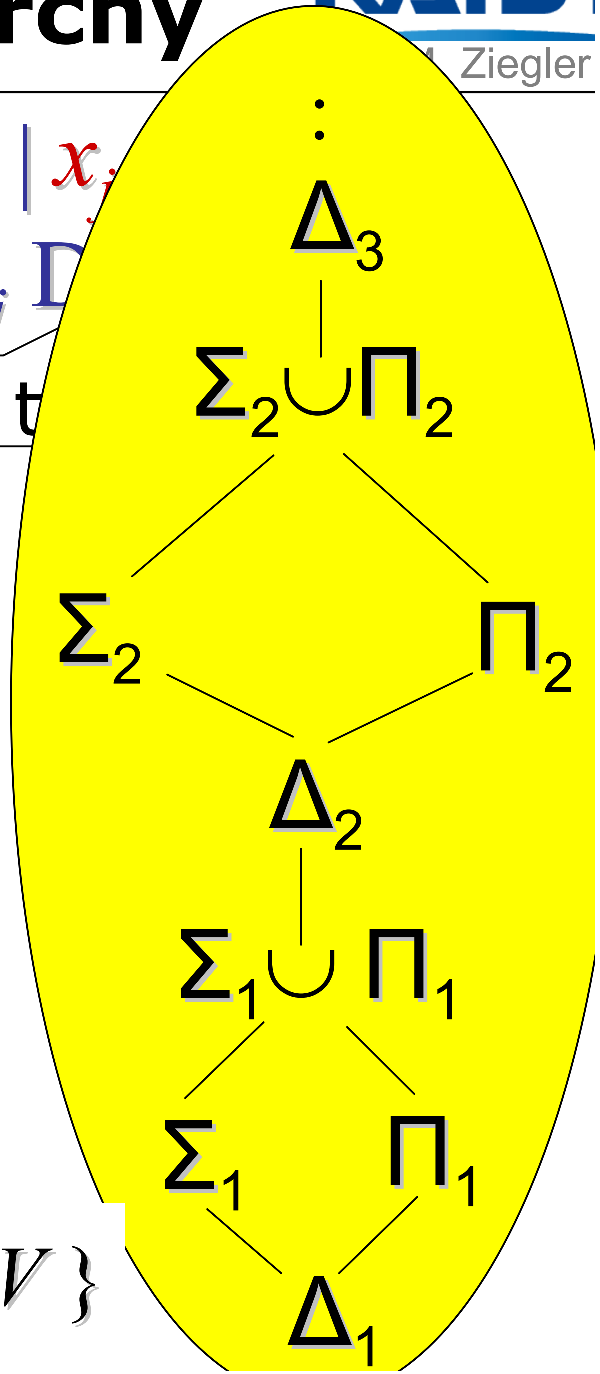
Δ_{k+1} = decidable$^{\Sigma_k}$ = decidable$^{\Pi_k}$

Σ_{k+1} = semi-decidable$^{\Sigma_k}$

Normalform: $L \in \Sigma_4$

iff, for some decidable $V \subseteq \mathbb{N}$,

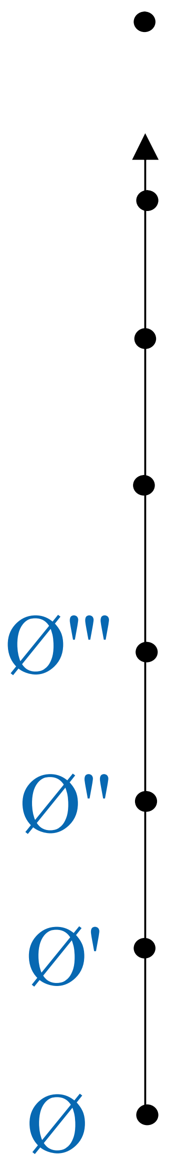
$L = \{ x \in \mathbb{N} : \exists y \, \forall z \, \exists u \, \forall v : \langle x,y,z,u,v \rangle \in V \}$



Post's Question

- Decidable problems
- *Un*decidable N s.t. H is decidable by P^N
- Strictly "more" undecidable than H: $T \equiv H^H$
- Emil Post'44: a) Anything between H, H^H?
- b) Are there *in*comparable problems?
- That is, do there exist
  - semi-decidable problems A, B s.t.
  - A is not decidable with oracle B
  - nor is B decidable with oracle A.
- Answered 1956/57 by Friedberg&Muchnik

Partially Ordered Degrees



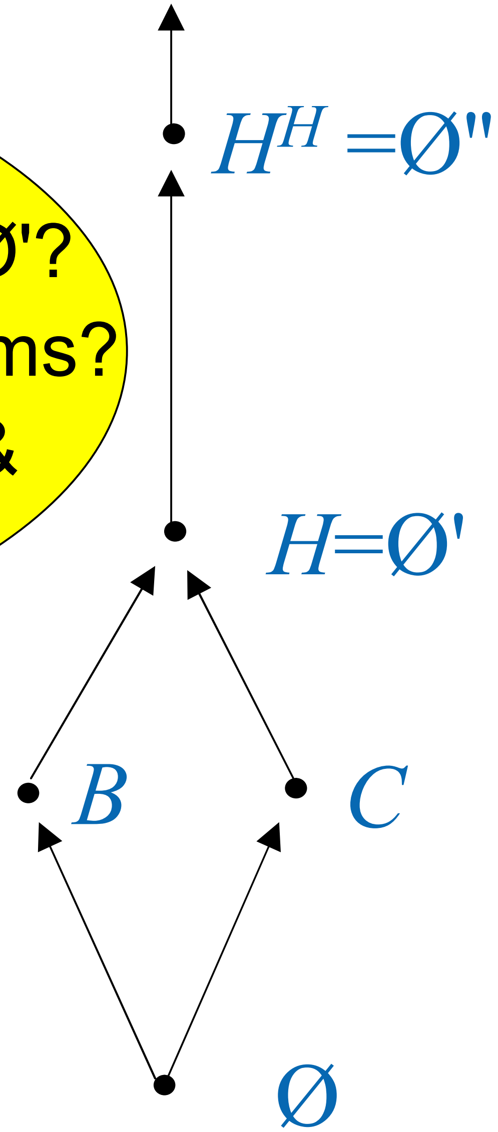
Emil Post 1944:

a) Is anything in between Ø and Ø'?

b) Are there incomparable problems?

Answered 1956/57 by Friedberg & Muchnik: such B,C exist!

For $\psi,\varphi:\mathbb{N}\to\mathbb{N}$ write $\psi \preccurlyeq \varphi$ if there is a WHILE program with oracle φ computing ψ.

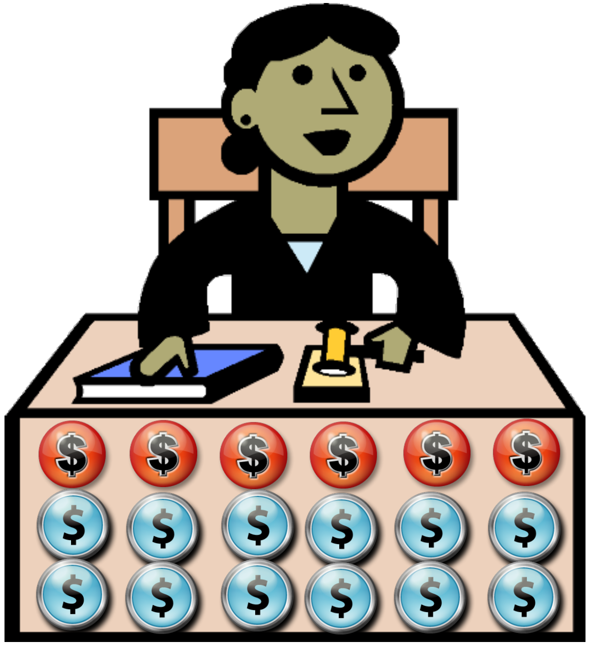


Priority Diagonalization: Trading with the Devil

You have countably many coins

- Devil takes *one* of them
- and gives you *two new* ones,
- Then repeat.

How many coins do you own *ultimately* ?



......

......

......

NONE!

Courtesy of Joel D. Hamkins

Advanced Diagonalization

Proof idea: „Construct" semi-deciable $A,B \subseteq \mathbb{N}$ s.t.

- To each P exists $x[P]$ s.t $x \in A \Leftrightarrow P^B(x)$ terminates
- To each Q exists $y[Q]$ st $y \in B \Leftrightarrow Q^A(y)$ terminates

$D = \{ \langle P \rangle \mid P(\langle P \rangle)$ does not terminate $\}$

$\mathbb{N} \setminus A$ is not *semi*-decidable with oracle B,
and $\mathbb{N} \setminus B$ is not *semi*-decidable with oracle A.

Theorem (Friedberg,Muchnik'57): There exist (undecidable but) semi-decidable $A,B \subseteq \mathbb{N}$ s.t. A is *un*decidable with oracle B, and vice versa.

Two Incomparable Problems

Proof idea: „Construct" semi-deciable $A,B \subseteq \mathbb{N}$ s.t.

- To each P exists $x[P]$ s.t $x \in A \Leftrightarrow P^B(x)$ terminates
- To each Q exists $y[Q]$ st $y \in B \Leftrightarrow Q^A(y)$ terminates

---

- „Thought experiment": Start with $x,y:=0$, $A,B:=\emptyset$.
- Enumerate all oracle WHILE programs $P^?,Q^?$.
- If P^B accepts x, set $A:=A\cup\{x\}$; else keep A.
- If Q^A accepts y, set $B:=B\cup\{y\}$; else keep
- Let $x := x+1$, $y := y+1$

But oracles A,B change throughout construction, might *later* violate witness conditions

Two Incomparable Problems

Proof idea: „Construct“ semi-decidable $A,B\subseteq\mathbb{N}$ s.t.

- To each P exists $x[P]$ s.t $x\in A \Leftrightarrow P^B(x)$ terminates
- To each Q exists $y[Q]$ st $y\in B \Leftrightarrow Q^A(y)$ terminates

---

- „Thought experiment“: Start with $x,y:=0$, $A,B:=\emptyset$.
- Enumerate all oracle WHILE programs $P^?,Q^?$.
- If P^B accepts x, set $A:=A\cup\{x\}$; else keep A.
- If Q^A accepts y, set $B:=B\cup\{y\}$; else keep B.
- $x := \max\{ x$, largest query by $Q^A(y)$ $\} +1$

  $y := \max\{ y$, largest query by $P^B(x)$ $\} +1$

But oracles A,B change throughout construction, might *later* violate witness conditions

Finite Injury Priority Proof

Proof idea: „Construct" semi-deciable $A,B\subseteq\mathbb{N}$ s.t.

- To each P exists $x[P]$ s.t $x\in A \Leftrightarrow P^B(x)$ terminates

Idea: Maintain 2 finite lists of 'candidate' witnesses.
E.g. (P_1,x_1) , (P_2,x_2) , (P_3,x_3) for A; (Q_1,y_1) , (Q_2,y_2) for B.
Call (P,x) **active** if 'simulation' of $P^B(x)$ is still running.

For each N:=0,1,2,…

- Add (N,x) to list. For **active** (P,a), increasing in P:
- If P^B accepts a within $\leq N$ steps, set $A:=A\cup\{a\}$
- y:=1+max{ y , largest oracle query by P^B on a }
- Mark (P,a) **inactive**. For all (Q,b) with $Q>P$ do
  - replace (Q,b) with $(Q,y{+}{+})$ marked **active**.
- Add (N,y) to list. For **active** (Q,b), increasing in Q:
- If Q^A accepts b within $\leq N$ steps, …

Finite Injury Priority Technique

Witness $y[P]$ for "$y \in B \Leftrightarrow Q^A(y)$ stops" changes (injury)

- but only finitely often:
- namely when some $P<Q$ terminates (priority)
- and, once settled, *maintains* witness condition!
- Both A,B are *enumerated*, hence semi-decidable.

For each $N:=0,1,2,\ldots$

- Add (N,x) to list. For **active** (P,a), increasing in P:
- If P^B accepts a within $\leq N$ steps, set $A:=A \cup \{a\}$
- $y:=1+\max\{\, y$, largest oracle query by P^B on a $\}$
- Mark (P,a) **inactive**. For all (Q,b) with $Q>P$ do
  - replace (Q,b) with $(Q,y{+}{+})$ marked **active**.
- Add (N,y) to list. For **active** (Q,b), increasing in Q:
- If Q^A accepts b within $\leq N$ steps, …

§2 Advanced Computability

- WHILE programs
- UTM Theorem
- Normalform Theorem
- SMN Theorem / Currying (Schönfinkeling)
- Fixedpoint Theorem, Quines
- Rice-Myhill-Shapiro
- Oracle WHILE programs, Higher Halting Problem
- Arithmetic Hierarchy: semantially & syntactically
- Post‵s Problem / Friedberg&Muchnik‵s Proof