Th 6.2: The busy beaver function, $\sigma(n)$, is eventually larger than any total, recursive function.

Th. 6.3: There is a *Universal Turing Machine* U such that,

$$U(P(n,m)) \quad = \quad M_n(m)$$

Thm. 6.4: (Unsolvability of Halting Problem) Let,

HALT $=$ $\{P(n,m) \mid$ TM $M_n(m)$ eventually halts$\}$

Then, HALT is r.e. but not recursive.

Listing of all r.e. sets: $W_0, W_1, W_2, \cdots$

$$W_i \quad = \quad \{n \mid M_i(n) = 1\}$$

Cor. 6.6: Let,

$$K = \{n \mid M_n(n) = 1\} \quad = \quad \{n \mid U(P(n,n)) = 1\}$$
$$= \{n \mid n \in W_n\}$$

Then,

$K \in$ **r.e.** $-$ **Recursive** .

Notation: $M_n(x)\downarrow$ means that TM M_n **converges** on input x, i.e.,

$$M_n(x)\downarrow \quad \Leftrightarrow \quad M_n(x) \in \mathbf{N} \quad \Leftrightarrow \quad M_n(x) \neq \nearrow$$

Fundamental Th. of r.e. Sets: Let $S \subseteq \mathbf{N}$. T.F.A.E.

1. S is the domain of a partial, recursive function, i.e.,

$$(\exists n)(S \ = \ \text{dom}(M_n(\cdot)) \ = \ \{x \in \mathbf{N} \mid M_n(x)\downarrow\})$$

2. $S = \emptyset$ or S is the range of a total, recursive function, i.e., $S = \emptyset$ or S = range($M_n(\cdot)$) = $M_n(\mathbf{N})$, for some total, recursive function $M_n(\cdot)$.

3. S is the range of a partial, recursive function, i.e.,

$$S \ = \ M_n(\mathbf{N}), \quad \text{for some } n \in \mathbf{N} \ .$$

4. S is r.e., i.e., $S = W_n$, for some $n \in \mathbf{N}$

Proof: (Please learn this proof!)

$(1 \Rightarrow 2)$: Assume (1), $S = \{x \mid M_n(x)\downarrow\}$.

case 1: $S = \emptyset$. Thus S satisfies (2).

case 2: $S \neq \emptyset$. let $a_0 \in S$.

From M_n compute M_r, which on input z does the following:

1. $x := L(z);\ y := R(z)$ // i.e., $z = P(x, y)$
2. run $M_n(x)$ for y steps
3. **if** it halts **then return**(x)
4. **else return**(a_0)

Claim: $S = M_r(\mathbf{N}) = \{M_r(x) \mid x \in \mathbf{N}\}$.

$M_r(\mathbf{N}) \subseteq S$

$M_r(\mathbf{N}) \supseteq S$

Suppose $x \in S$.

Thus $M_n(x)$ converges in some number y of steps.

Therefore, $M_r(P(x, y)) = x$.

Non-computable step in construction: no way to tell if we are in case 1 or case 2.

$(2) \Rightarrow (3)$: Assume **(2).** If $S = \emptyset$ then $S = M_0(\mathbf{N})$ where M_0 is a Turing machine that halts on no inputs.

Otherwise, $S = M_n(\mathbf{N})$, i.e., S is the range of the partial, recursive function $M_n(\cdot)$.

Note: Even though $M_n(\cdot)$ is total, it is still considered a "partial, recursive function". However, of course, $M_n(\cdot)$ is not "strictly partial".

$(3) \Rightarrow (4)$: Assume **(3),** $S = M_n(\mathbf{N})$.

From M_n construct M_d, which on input x does the following:

1. **for** $i := 1$ to ∞ $\{$
2. run $M_n(0), M_n(1), \ldots, M_n(i)$ for i steps each.
3. **if** any of these output x, **then return**(1)$\}$

above construction called **dove-tailing**

Claim: $M_d(\cdot) = p_S(\cdot)$.

If $x \in S$, then $x \in$ range($M_n(\cdot)$)

$M_n(j) = x$, computation takes k steps, for some j, k

Thus, at round $i = \max(j, k)$, $M_d(x)$ will halt and output "1".

If $x \notin S$, then $M_d(x)$ will never halt.

Thus, $S = W_d = \{x \mid M_d(x) = 1\}$.

$(4) \Rightarrow (1)$: Assume (4), and thus $S = W_n$.

$$S \quad = \quad \{i \mid M_n(i) = 1\}$$

From M_n, construct M_d, which on input x does the following:

1. run $M_n(x)$
2. **if** $(M_n(x) = 1)$ **then return**(1)
3. **else** run forever

$$S \quad = \quad \{x \mid M_d(x)\downarrow\}$$

Thus, $S \ = \ \mathrm{dom}(M_d(\cdot)) \ = \ \{x \mid M_d(x)\downarrow\}$. ♠

Definition 7.1 We say that S is *reducible* to T $(S \leq T)$ iff $\exists$ total, recursive f : **N** $\rightarrow$ **N**,

$$(\forall w \in \mathbf{N}) \quad (w \in S) \qquad \Leftrightarrow \qquad (f(w) \in T)$$

♠

Note: Later we require $f \in F(\mathbf{DSPACE}[\log n])$.

$$A_{0,17} \quad = \quad \{n \mid M_n(0) = 17\}$$

Claim: $K \leq A_{0,17}$.

Proof: Define $f(n)$ as follows:

$M_{f(n)}$ =	erase input; write n	M_n	if 1 then write 17 else loop

$$n \in K \;\Leftrightarrow\; M_n(n) = 1 \;\Leftrightarrow\; M_{f(n)}(0) = 17 \;\Leftrightarrow\; f(n) \in A_{0,17}$$

♠

Fundamental Th. of Reductions: If $S \leq T$, then,

1. If T is **r.e.**, then S is **r.e.**.
2. If T is co-**r.e.**, then S is co-**r.e.**.
3. If T is **Recursive**, then S is **Recursive**.

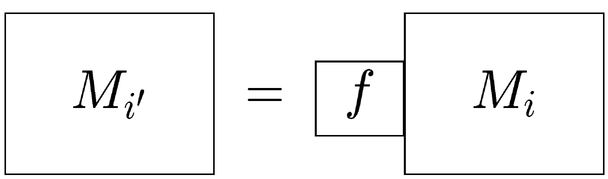
Moral: Suppose $S \leq T$. Then,

- If T is easy, then so is S.
- If S is hard, then so is T.

Proof: Let $f : S \leq T$, i.e., $(\forall x)(x \in S \;\Leftrightarrow\; f(x) \in T)$

1. Suppose $T \;=\; W_i \;=\; \{x \mid M_i(x) = 1\}$.
   From M_i compute the TM $M_{i'}$ which on input x does the following:

   (a) compute $f(x)$
   (b) run $M_i(f(x))$



$$(x \in S) \;\Leftrightarrow\; (f(x) \in T) \;\Leftrightarrow\; (M_i(f(x)) = 1) \;\Leftrightarrow\; (M_{i'}(x) = 1$$

Therefore, $S = W_{i'}$, and S is r.e. as desired.

$$f : S \leq T, \qquad \text{i.e.,} \qquad (\forall x)(x \in S \;\Leftrightarrow\; f(x) \in T)$$

2. **Note:** $S \leq T \quad \Leftrightarrow \quad \overline{S} \leq \overline{T}$.

$T \in$ co-**r.e.** $\overline{T} \in$ **r.e.** $\overline{S} \in$ **r.e.** $S \in$ co-**r.e.**

3. $T \in$ **Recursive** $\quad \Rightarrow \quad$ ($T \in$ **r.e.** $\wedge$ $T \in$ co-**r.e.**) $\quad \Rightarrow$

($S \in$ **r.e.** $\wedge$ $S \in$ co-**r.e.**) $\quad \Rightarrow \quad$ $S \in$ **Recursive**

♠

Definition 7.2 Let $C \subseteq \mathbf{N}$. C is r.e.-*complete* iff

1. $C \in$ **r.e.**, and
2. $(\forall A \in$ **r.e.**$) \quad (A \leq C)$

Intuition: C is a "hardest" r.e. set. In the "$\leq$" ordering, it is above all other r.e. sets. ♠

Theorem 7.3 *K is r.e. complete.*

Proof: Let $A \in$ **r.e.**, i.e., $A = W_i$ for some i.

Wanted: $$(\forall n)(n \in A \quad \Leftrightarrow \quad f(n) \in K)$$

Define the recursive function f which on input n computes the following TM:

$M_{f(n)} =$	Erase input	Write n	M_i

$$n \in A \Leftrightarrow M_i(n) = 1 \quad \Leftrightarrow \quad (\forall x) M_{f(n)}(x) = 1$$

$$\Leftrightarrow M_{f(n)}(f(n)) = 1 \quad \Leftrightarrow \quad f(n) \in K$$

♠

Proposition 7.4 *Suppose that C is r.e.-complete and the following hold:*

1. *$S \in$ **r.e.**, and*
2. *$C \leq S$*

then S is r.e.-complete.

Proof: Show: $(\forall A \in \textbf{r.e.})(A \leq S)$

Know: $(\forall A \in \textbf{r.e.})(A \leq C)$

Follows by transitivity of $\leq$: $A \leq C \leq S$. ♠

Corollary 7.5 *$A_{0,17}$ is r.e.-complete.*
Every r.e.-complete set is r.e. and not recursive.

$$\text{HALT} \quad = \quad \{P(n,m) \mid \text{TM } M_n(m) \text{ eventually halts}\}$$

Proposition 7.6 HALT *is r.e.-complete.*

Proof: We have already seen that HALT is r.e. It thus suffices to show that $K \leq$ HALT.

We want to build a total, recursive f such that for all $w \in \mathbf{N}$,

$$w \in K \quad \Leftrightarrow \quad f(w) \in \text{HALT}$$

$$M_w(w) = 1 \quad \Leftrightarrow \quad M_{L(f(w))}(R(f(w))) \text{ halts}$$

That is, we want,

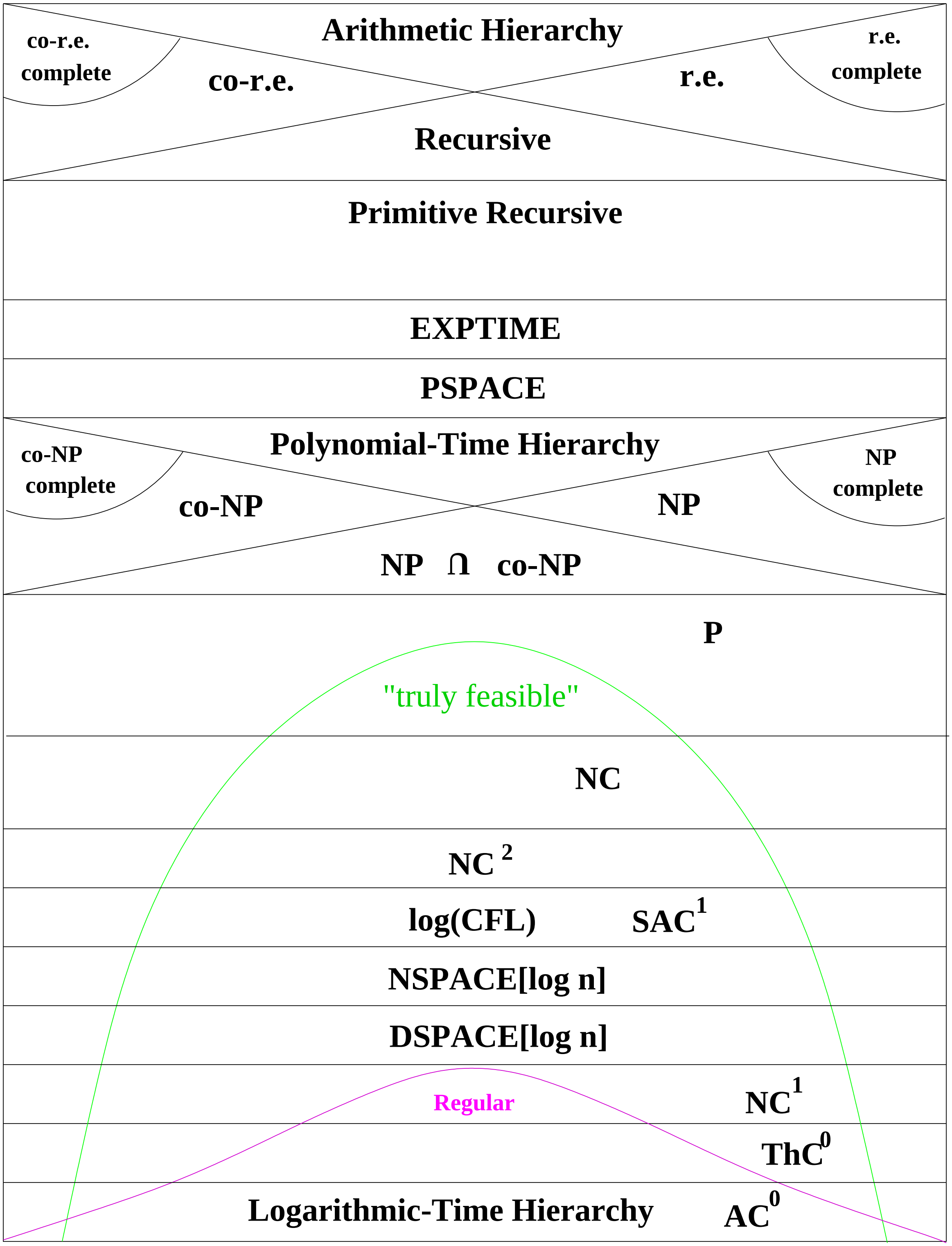
$$M_w(w) = 1 \quad \Leftrightarrow \quad M_\ell(r) \text{ halts}, \quad \text{where } f(w) = P(\ell, r)$$

Given w, let, $M_{\ell(w)} \ =$

Erase input	Write w	M_w	if 1 then halt else diverge

Letting $f(w) = P(\ell(w), 0)$, we have that

$$M_w(w) = 1 \quad \Leftrightarrow \quad M_{\ell(w)}(0) \text{ halts} \quad \Leftrightarrow \quad f(w) \in \text{HALT}♠$$


Arithmetic Hierarchy
co-r.e. complete
co-r.e.
r.e.
r.e. complete
Recursive
Primitive Recursive
EXPTIME
PSPACE
Polynomial-Time Hierarchy
co-NP complete
co-NP
NP
NP complete
NP ∩ co-NP
P
"truly feasible"
NC
NC 2
log(CFL)
SAC1
NSPACE[log n]
DSPACE[log n]
Regular
NC1
ThC0
Logarithmic-Time Hierarchy
AC0