Recall From Last Time

Busy Beaver Function, $\sigma(n)$, grows faster than any total, recursive function.

Universal Turing Machine, $U(P(n,m)) \;=\; M_n(m)$.

$\text{HALT} \;=\; \{P(n,m) \mid M_n(m) \text{ halts}\} \;\in\; \textbf{r.e.} - \textbf{Recursive}$

Listing of all r.e. sets: $W_0, W_1, W_2, \cdots$ W_i is the i^{th} r.e. set

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

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

$$=\; \{n \mid n \in W_n\} \quad \in \quad \textbf{r.e.} - \textbf{Recursive}$$

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 Theorem 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., for some $n \in \mathbf{N}$, $S = \left\{x \in \mathbf{N} \;\middle|\; M_n(x)\!\downarrow\right\}$

2. $S = \emptyset$ or S is the range of a total, recursive function,

   i.e., for some total, recursive $M_m(\cdot)$, $S = \emptyset$ or $S = M_m(\mathbf{N})$

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

   i.e., for some $r \in \mathbf{N}$, $S = M_r(\mathbf{N})$

4. S is r.e.,

   i.e., for some $t \in \mathbf{N}$, $S = W_t$,

$$S = \{x \mid M_n(x)\!\downarrow\} \quad \Rightarrow \quad S = \emptyset \ \vee \ \exists m(S = M_m(\mathbf{N}))$$

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

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

Build TM M_m, 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 converges **then return(**x**)**
4. **else return(**a_0**)**

Claim: $S = M_m(\mathbf{N})$: $M_m(\mathbf{N}) \subseteq S$ $\surd$

$M_m(\mathbf{N}) \supseteq S$: Suppose $x \in S$.

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

Therefore, $M_m(P(x, y)) = x$. $\surd$

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

$$S = \emptyset \text{ or } S = M_m(\mathbf{N}) \quad \Rightarrow \quad \exists r(S \;=\; M_r(\mathbf{N}))$$

If $S = \emptyset$ then $S = M_0(\mathbf{N})$ where M_0 is a Turing machine that halts on no inputs. $r := 0$

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

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

$$S = M_r(\mathbf{N}) \quad \Rightarrow \quad \exists t (S = W_t)$$

Construct TM M_t, which on input x does the following:

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

above construction called **dove-tailing**

Claim: $M_r(\mathbf{N}) = \mathcal{L}(M_t)$.
Suppose $x \in M_r(\mathbf{N})$, i.e., $M_r(j) = x$, for some j, computation takes k steps, for some k

At round $i = \max(j, k)$, $M_t(x)$ will halt and output "1". √

Suppose $x \notin M_r(\mathbf{N})$, then $M_t(x)$ will never halt. √

$$S = W_t \quad \Rightarrow \quad \exists n\left(S \;=\; \left\{x \in \mathbf{N} \;\middle|\; M_n(x)\!\downarrow\right\}\right)$$

Construct TM M_n, which on input x does the following:

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

Recall that, $\quad S \quad = \quad W_t \quad = \quad \mathcal{L}(M_t)$

Thus, $\quad S \;=\; \mathrm{dom}(M_n(\cdot)) \;=\; \left\{x \;\middle|\; M_n(x)\!\downarrow\right\}$. $\square$

Please learn the above proof: it's constructions are central to recursive function theory.

Reductions

Def. We say that S is **reducible** to T $(S \leq T)$ iff there exists total, recursive f, $\forall w \in \mathbf{N}\ (w \in S \ \Leftrightarrow\ f(w) \in T)$.

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

Claim: $K \leq A_{0,17} \ = \ \{n \mid M_n(0) = 17\}$

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

$M_{f(n)} \ =$

erase input; write n	M_n	if 1 then write 17 else loop

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

Note: f inputs program n and outputs program $f(n)$.

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

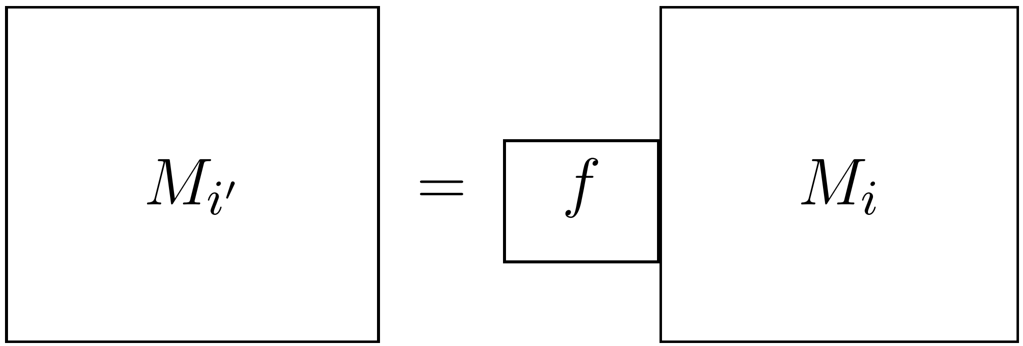
1. If T is **r.e.**, then S is **r.e.**.
2. If T is co-**r.e.**, then S is co-**r.e.**.
3. If T is **Recursive**, then S is **Recursive**.

Proof:

$S \le T \;\wedge\; T \in \textbf{r.e.} \quad \Rightarrow \quad S \in \textbf{r.e.}$

Let $f : S \le T$, i.e., $\forall x(x \in S \;\Leftrightarrow\; f(x) \in T)$, $\qquad T = W_i$.

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.

$$f : S \leq T, \ \textbf{i.e.} \quad \forall x (x \in S \Leftrightarrow f(x) \in T)$$

Observation: $f : S \leq T \quad \Leftrightarrow \quad f : \overline{S} \leq \overline{T}$.

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

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

Fundamental Th. of Reductions: If $S \le T$, 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 $S \le T$. Then,

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

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

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

Intuition: C is a "hardest" r.e. set.

Th: K is r.e. complete.

Proof: We already know that K is r.e.

Let A be an arbitrary r.e. set, i.e., $A = W_i$ for some i.

Wanted: total recursive f, s.t.: $\forall n(n \in A \Leftrightarrow f(n) \in K)$

Define total, recursive f which on input n computes:

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

$M_{f(n)}$ ignores its input and instead runs $M_i(n)$.

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

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

Prop: Suppose C is r.e.-complete and:

1. $S \in \mathbf{r.e.}$, and
2. $C \leq S$

then S is r.e.-complete.

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

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

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

Cor: $A_{0,17}$ is r.e.-complete.

Every r.e.-complete set is r.e. and not recursive.

Prop: $\mathrm{HALT} = \{P(n,m) \mid M_n(m) \text{ halts}\}$ is r.e.-complete.

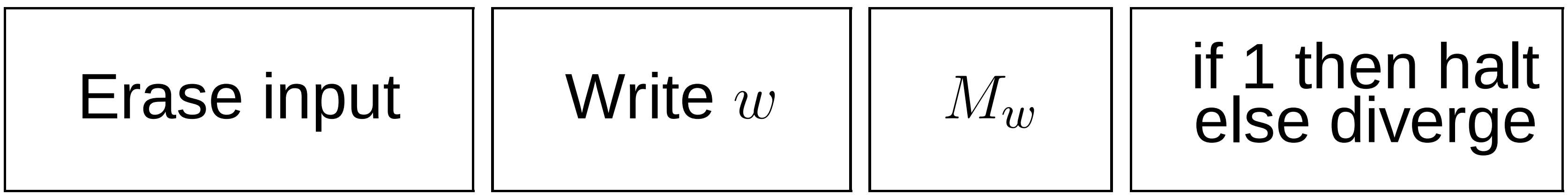
Proof: Already seen HALT is r.e.. Must show $K \leq \mathrm{HALT}$.

Wanted: total, recursive f s.t. $\forall w(w \in K \;\Leftrightarrow\; f(w) \in \mathrm{HALT})$

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

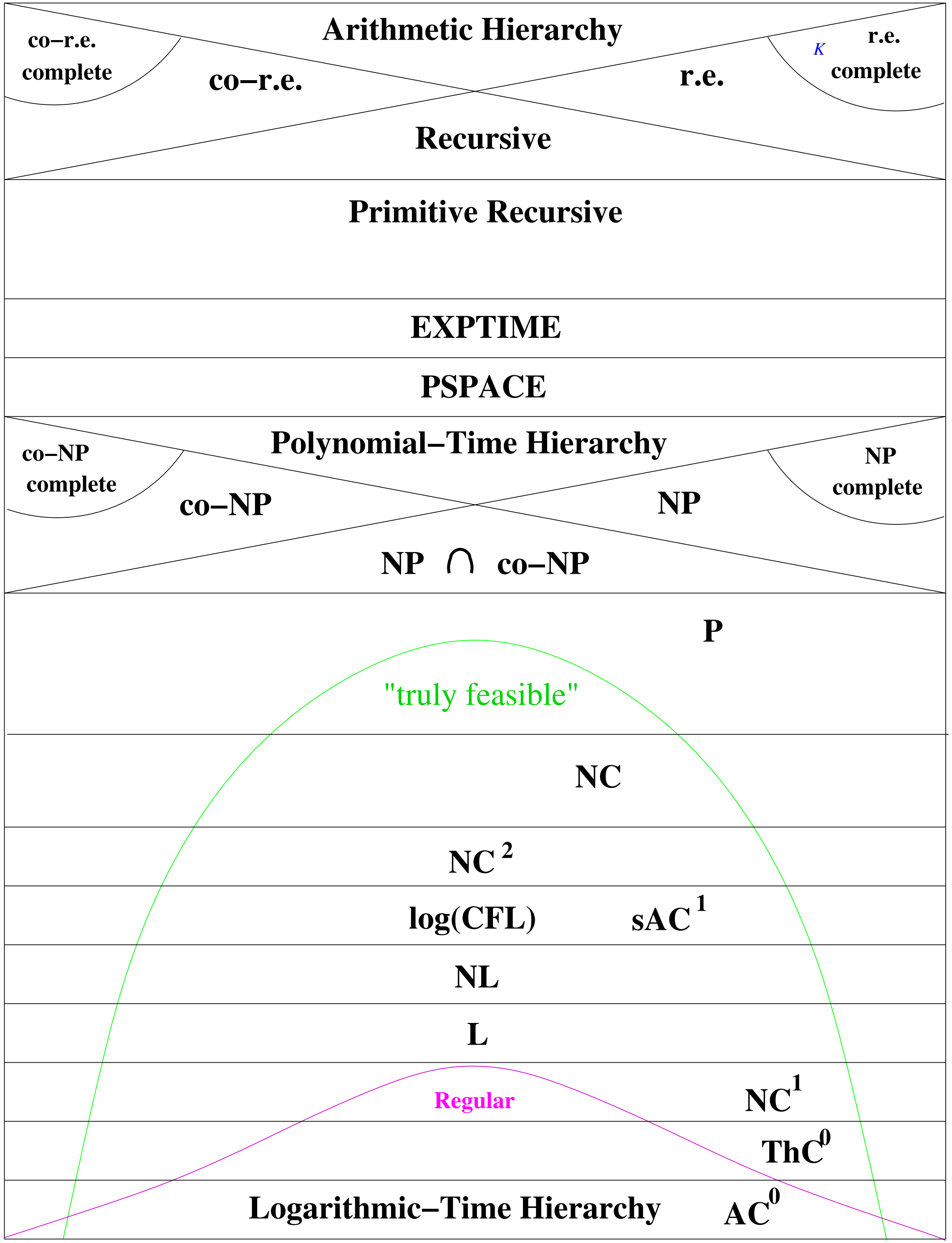
$$M_w(w) = 1 \qquad \Leftrightarrow \qquad M_\ell(r) \text{ halts,} \quad \text{where } f(w) = P(\ell, r)$$

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



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

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


Arithmetic Hierarchy
co–r.e. complete
co–r.e.
r.e.
K
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)
sAC 1
NL
L
Regular
NC1
ThC0
Logarithmic–Time Hierarchy
AC0