Def: **DTIME**, **NTIME**, **DSPACE**, measured on **Multi-tape Turing Machines**.

Th: $\textbf{DTIME}[t(n)] \subseteq \text{RAM-TIME}[t(n)] \subseteq \textbf{DTIME}[(t(n))^3]$

$$\begin{array}{lclcl}
\textbf{L} & \equiv & \textbf{DSPACE}[\log n] & & \\
\textbf{P} & \equiv & \textbf{DTIME}[n^{O(1)}] & \equiv & \bigcup_{i=1}^{\infty} \textbf{DTIME}[n^i] \\
\textbf{NP} & \equiv & \textbf{NTIME}[n^{O(1)}] & \equiv & \bigcup_{i=1}^{\infty} \textbf{NTIME}[n^i] \\
\textbf{PSPACE} & \equiv & \textbf{DSPACE}[n^{O(1)}] & \equiv & \bigcup_{i=1}^{\infty} \textbf{DSPACE}[n^i]
\end{array}$$

Th: For $t(n) \geq n, s(n) \geq \log n$,

$$\textbf{DTIME}[t(n)] \subseteq \textbf{NTIME}[t(n)] \subseteq \textbf{DSPACE}[t(n)]$$

$$\textbf{DSPACE}[s(n)] \subseteq \textbf{DTIME}[2^{O(s(n))}]$$

Cor: $\textbf{L} \subseteq \textbf{P} \subseteq \textbf{NP} \subseteq \textbf{PSPACE}$

Definition 6.1 The *busy beaver function* $\sigma(n)$ is the maximum number of one's that an n state TM with alphabet $\Sigma = \{0, 1\}$ can leave on its tape and halt when started on the all 0 tape. (To fit our definitions, note that "0" is now the "blank character".) ♠

Note that $\sigma(n)$ is well defined:

There are only finitely many n-state TMs, with $\Sigma = \{0, 1\}$.

Some finite subset, F_n, of these eventually halt on input 0.

Some element of F_n prints the max # of 1's = $\sigma(n)$.

	q_1	q_2	q_3
0	$q_2, 1, \rightarrow$	$q_3, 0, \rightarrow$	$q_3, 1, \leftarrow$
1	$h, 1, -$	$q_2, 1, \rightarrow$	$q_1, 1, \leftarrow$

$\sigma(3) \geq 6$

q_1	0	$\boxed{0}$	0	0	0	0	0
q_2	0	1	$\boxed{0}$	0	0	0	0
q_3	0	1	0	$\boxed{0}$	0	0	0
q_3	0	1	$\boxed{0}$	1	0	0	0
q_3	0	$\boxed{1}$	1	1	0	0	0
q_1	$\boxed{0}$	1	1	1	0	0	0
q_2	1	$\boxed{1}$	1	1	0	0	0
q_2	1	1	$\boxed{1}$	1	0	0	0
q_2	1	1	1	$\boxed{1}$	0	0	0
q_2	1	1	1	1	$\boxed{0}$	0	0
q_3	1	1	1	1	0	$\boxed{0}$	0
q_3	1	1	1	1	$\boxed{0}$	1	0
q_3	1	1	1	$\boxed{1}$	1	1	0
q_1	1	1	$\boxed{1}$	1	1	1	0
h	1	1	$\boxed{1}$	1	1	1	0

How quickly does $\sigma(n)$ grow as n gets large?

$$
\begin{array}{lcl}
\text{Is } \sigma(n) \in & O(n^2) & ? \\
& O(n^3) & ? \\
& O(2^n) & ? \\
& O(n!) & ? \\
& O(2^{2^n}) & ? \\
& O(\exp^*(n)) & ? \\
& O(\exp^*(\exp^*(n))) & ?
\end{array}
$$

$$
\exp^*(n) \quad = \quad \left. 2^{2^{2^{\cdot^{\cdot^{\cdot^{2}}}}}} \right\} n
$$

States	**Max # of 1's**	**Lower Bound for $\sigma(n)$**
3	$\sigma(3)$	6
4	$\sigma(4)$	13
5	$\sigma(5)$	≥ 4098
6	$\sigma(6)$	$> 10^{865}$

See the web pages of Penousal Machado (`eden.dei.uc.pt`) and Heiner Marxen (`www.drb.insel.de/ heiner/BB`) for more on this problem and its variants.

Theorem 6.2 *Let $f : \mathbf{N} \rightarrow \mathbf{N}$ be a total, recursive function.*

$$\lim_{n\rightarrow\infty}\left(\frac{f(n)}{\sigma(n)}\right) \quad = \quad 0$$

That is, $f(n) = o(\sigma(n))$.

Proof:

$$g(n) \quad = \quad n \cdot \left(1 + \sum_{i=0}^{n} f(i)\right)$$

Note:

$$\lim_{n\rightarrow\infty}\left(\frac{f(n)}{g(n)}\right) \quad = \quad 0$$

We will show for all sufficiently large n,

$$\sigma(n) \quad \geq \quad g(n)$$

$g(n)$ is computed by a k-state TM for some k.

For any n, define the TM

$$C_n \;=\; \underbrace{\boxed{\text{print} \quad n}}_{\lceil \log n \rceil}\underbrace{\boxed{\text{compute } g}}_{k}\underbrace{\boxed{\begin{array}{c}\text{binary}\\ \text{to unary}\end{array}}}_{17}$$

C_n has $\lceil \log n \rceil + k + 17$ states.

C_n prints $g(n)$ 1's.

Once n is big enough that $n \geq \lceil \log n \rceil + k + 17$,

$$\sigma(n) \;\geq\; \sigma(\lceil \log n \rceil + k + 17) \;\geq\; g(n)$$

$\spadesuit$

A Pairing Function

On HW#2, we define a pairing function:

$$P : \mathbf{N} \times \mathbf{N} \underset{onto}{\overset{1:1}{\rightarrow}} \mathbf{N}$$

$$\begin{aligned} P(L(w), R(w)) &= w \\ L(P(i,j)) &= i \\ R(P(i,j)) &= j \end{aligned}$$

We can use the pairing function to think of a natural number as a pair of natural numbers.

Thus, the input to a Turing machine is a single binary string which may be thought of as a natural number, a pair of natural numbers, a triple of natural numbers, and so forth. (Later we will worry about the complexity of the pairing and string-conversion functions – do you think they are in **L**)?

Turing machines can be encoded as **character strings** which can be encoded as **binary strings** which can be encoded as **natural numbers**.

TM_n	1	2	3	4
0	$1, 0, \rightarrow$	$3, \sqcup, \rightarrow$	$0, 0, -$	$0, 0, -$
1	$1, 1, \rightarrow$	$4, \sqcup, \rightarrow$	$0, 1, -$	$0, 1, -$
$\sqcup$	$2, \sqcup, \leftarrow$	$0, \sqcup, -$	$1, 0, \leftarrow$	$1, 1, \leftarrow$
$\triangleright$	$1, \triangleright, \rightarrow$	$0, \triangleright, -$	$0, \triangleright, -$	$0, \triangleright, -$

ASCII: $1, 0, \rightarrow; 1, 1, \rightarrow; 2, \sqcup, \leftarrow; 1, \triangleright, \rightarrow; ; \ \cdots \ 0, \triangleright, -$

$\{0, 1\}^\star$: w

N : n

There is a simple, countable listing of all TM's:

$$M_0, M_1, M_2, \cdots$$

Theorem 6.3 *There is a* Universal Turing Machine U *such that,*

$$U(\langle n, m\rangle) \quad = \quad M_n(m)$$

Proof: Given $\langle n, m\rangle$, compute n and m. n is a binary string encoding the state table of TM M_n. We can simulate M_n on input m by keeping track of its state, its tape, and looking at its state table, n, at each simulated step. ♠

Let's look at $L(U)$, the set of numbers $P(n, m)$ such that the Turing machine M_n eventually halts on input n. We'll call this language HALT. The existence of U proves that HALT is r.e., and we'll now show it's not recursive.

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

Theorem 6.4 (Unsolvability of the Halting Problem) HALT *is r.e. but not recursive.*

Proof:

$$\begin{aligned} \text{HALT} &= \{w \mid U(w) \text{ eventually halts}\} \\ &= \{w \mid U'(w) = 1\} \end{aligned}$$

$$U' = \boxed{\quad U \quad}\boxed{\text{erase tape}}\boxed{\text{print 1}}$$

Suppose HALT were recursive. Then $\sigma(n)$ would be a total recursive function: Cycle through all n-state TMs, M_i, and if $P(i,0) \in$ HALT, then count the number of 1's in $M_i(0)$. Return the maximum of these. But $\sigma(n)$ *isn't* total recursive, so we have a contradiction.

$\spadesuit$

Listing All r.e. Sets

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

The set of all r.e. sets = $W_0, W_1, W_2, \cdots$

n	0	1	2	3	4	5	6	7	8	$\cdots$	W_n
0	$\boxed{0}$	0	0	0	0	0	0	0	0	$\cdots$	W_0
1	1	$\boxed{1}$	1	1	1	1	1	1	1	$\cdots$	W_1
2	1	0	$\boxed{1}$	0	1	0	1	0	1	$\cdots$	W_2
3	0	1	0	$\boxed{1}$	0	1	0	1	0	$\cdots$	W_3
4	1	0	0	0	$\boxed{0}$	0	0	0	0	$\cdots$	W_4
5	0	1	1	0	1	$\boxed{0}$	0	0	1	$\cdots$	W_5
6	1	0	0	1	0	0	$\boxed{1}$	0	0	$\cdots$	W_6
7	1	1	0	0	0	0	0	$\boxed{0}$	0	$\cdots$	W_7
8	0	1	0	0	0	0	0	0	$\boxed{0}$	$\cdots$	W_8
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\cdots$	$\vdots$
	0	1	1	1	0	0	1	0	0	$\cdots$	K
	1	0	0	0	1	1	0	1	1	$\cdots$	$\overline{K}$

$$
\begin{aligned}
K &= \{n \mid M_n(n) = 1\} \\
&= \{n \mid U(P(n,n)) = 1\} \\
&= \{n \mid n \in W_n\}
\end{aligned}
$$

Theorem 6.5 $\overline{K}$ *is not r.e.*

Proof: $$\overline{K} \quad = \quad \{n \mid n \notin W_n\}$$

Suppose $\overline{K}$ were r.e. Then for some c,

$$\overline{K} \quad = \quad W_c \quad = \quad \{n \mid M_c(n) = 1\}$$

$$c \in K \;\Leftrightarrow\; M_c(c) = 1 \;\Leftrightarrow\; c \in W_c \;\Leftrightarrow\; c \in \overline{K}$$

♠

Corollary 6.6 $K \in$ **r.e.** $-$ **Recursive**