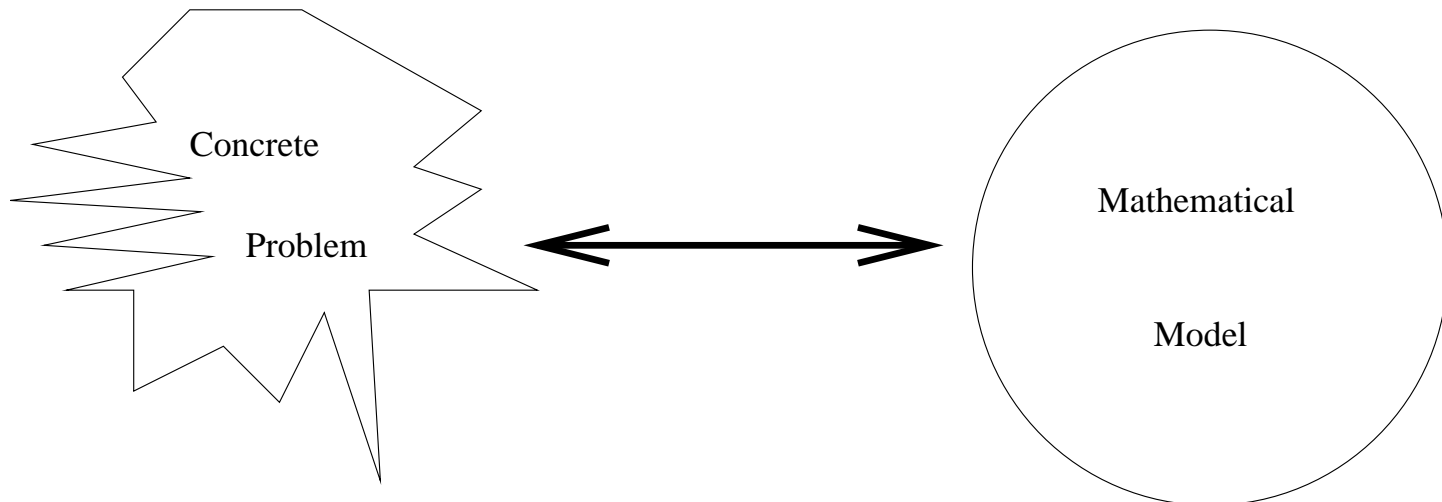


# Summary & Conclusions

In-depth introduction to main models, concepts of theory of computation:

- **Computability:** what can be computed in principle
- **Logic:** how can we express our requirements and verify our conclusions
- **Complexity:** what can be computed in practice



# Formal Models of Computation:

- FA  $\cong$  Regular Expression
- PDA  $\cong$  CFG
- TM  $\cong$  Recursive Function  $\cong$  Boolean Circuits . . .
- logical formula

# Regular Sets

**Kleene's Theorem:** Let  $A \subseteq \Sigma^*$  be any language. Then the following are equivalent:

1.  $A = \mathcal{L}(D)$ , for some DFA  $D$ .
2.  $A = \mathcal{L}(N)$ , for some NFA  $N$  wo  $\epsilon$  transitions
3.  $A = \mathcal{L}(N)$ , for some NFA  $N$ .
4.  $A = \mathcal{L}(e)$ , for some regular expression  $e$ .

**Myhill-Nerode Theorem:** The language  $A$  is regular iff  $\sim_A$  has a finite number of equivalence classes. Furthermore, this number of equivalence classes is equal to the number of states in the minimum-state DFA that accepts  $A$ .

**Pumping Lemma for Regular Sets:** Let  $D = (Q, \Sigma, \delta, q_0, F)$  be a DFA. Let  $n = |Q|$ . Let  $w \in \mathcal{L}(D)$  s.t.  $|w| \geq n$ . Then  $\exists x, y, z \in \Sigma^*$  s.t. the following all hold:

- $xyz = w$
- $|xy| \leq n$
- $|y| > 0$ , and
- $\forall k \geq 0 (xy^kz \in \mathcal{L}(D))$

# CFL's

**Closure Theorem for Context Free Languages:** Let  $A, B \subseteq \Sigma^*$  be context-free languages, let  $R \subseteq \Sigma^*$  be a regular language, and let  $h : \Sigma^* \rightarrow \Gamma^*$  and  $g : \Gamma^* \rightarrow \Sigma^*$  be homomorphisms. Then the following languages are context-free:

1.  $A \cup B$
2.  $AB$
3.  $A \cap R$
4.  $h(A)$
5.  $g^{-1}(A)$

**CFL Pumping Lemma:** Let  $A$  be a CFL. Then there is a constant  $n$ , depending only on  $A$  such that if  $z \in A$  and  $|z| \geq n$ , then there exist strings  $u, v, w, x, y$  such that  $z = uvwxy$ , and,

- $|vx| \geq 1$ ,
- $|vwx| \leq n$ , and
- for all  $k \in \mathbf{N}$ ,  $uv^kwx^ky \in A$

# Recursive Sets

A (partial) function,  $f$ , is **recursive** iff it is computed by some TM  $M_i$ , i.e.,  $M_i(\cdot) = f(\cdot)$ .

Let  $S \subseteq \{0, 1\}^*$  or  $S \subseteq \mathbf{N}$ .  $S$  is a **recursive set** iff the function  $\chi_S$  is a (total) recursive function,

$$\chi_S(x) = \begin{cases} 1 & \text{if } x \in S \\ 0 & \text{otherwise} \end{cases}$$

$S$  is a **recursively enumerable set** ( $S$  is **r.e.**) iff the function  $p_S$  is a (partial) recursive function,

$$p_S(x) = \begin{cases} 1 & \text{if } x \in S \\ \nearrow & \text{otherwise} \end{cases}$$

**Thm:**      **Recursive**      =      **r.e.**  $\cap$  **co-r.e.**

A set,  $S \subseteq \mathbf{N}$ , is

- **r.e.**      if every  $a \in S$  has a proof that  $a \in S$ .
- **co-r.e.**      if every  $a \notin S$  has a proof that  $a \notin S$ .
- **Recursive**      if for every  $a \in \mathbf{N}$  one of  $a \in S$  or  $a \notin S$  is provable.

**Def:** the **primitive recursive functions** is the smallest class of functions that

- contains the **Initial functions**:  $\zeta$ ,  $\sigma$ , and  $\eta$ , and
- is closed under **Composition**, and
- is closed under **Primitive Recursion**

**Prop:** The set of **primitive recursive functions** is equal to the set of **Bloop functions**.

**Def:** the **Gödel recursive functions** is the smallest class of functions that

- contains the **Initial functions**, and
- is closed under **Composition**, and
- is closed under **Primitive Recursion**, and
- is closed under **Unbounded Minimalization**

**Thm:** [Kleene]  $\text{COMP}(n, x, c, y)$  is a primitive recursive predicate.

**Thm:** The **Gödel recursive functions** are identical to the **partial recursive functions**.

# Cantor's Thm: $\wp(\mathbf{N})$ is not countable!

**Proof:** Suppose it were.

Let  $f : \mathbf{N} \xrightarrow[\text{onto}]{1:1} \wp(\mathbf{N})$ .

Define the diagonal set,  $D = \{n \mid n \notin f(n)\}$ .

Thus  $D = f(k)$  for some  $k \in \mathbf{N}$ .

$$k \in D \iff k \notin f(k) \iff k \notin D$$

$$\Rightarrow \Leftarrow$$

Therefore,  $\wp(\mathbf{N})$  is not countable!



$n$	0	1	2	3	4	5	6	7	8	...	$k$	$f(n)$
0	<b>0</b>	0	0	0	0	0	0	0	0	...	0	$f(0)$
1	1	<b>1</b>	1	1	1	1	1	1	1	...	1	$f(1)$
2	1	0	<b>1</b>	0	1	0	1	0	1	...		$f(2)$
3	0	1	0	<b>1</b>	0	1	0	1	0	...		$f(3)$
4	1	0	0	0	<b>0</b>	0	0	0	0	...	0	$f(4)$
5	0	1	1	0	1	<b>0</b>	0	0	1	...		$f(5)$
6	1	0	0	1	0	0	<b>1</b>	0	0	...		$f(6)$
7	1	1	0	0	0	0	0	<b>0</b>	0	...	0	$f(7)$
8	0	1	0	0	0	0	0	0	<b>0</b>	...	0	$f(8)$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	...		$\vdots$
$k$	1	0	0	0	1	1	0	1	1	...	<b>?</b>	$D \neq f(k)$

# Uses of Diagonalization

$$K = \{n \mid M_n(n) = 1\}$$

**Thm:**  $\overline{K}$  is not r.e.

**Thm:** Primitive Recursive  $\subsetneq$  Recursive

**Ladner's Thm:** If  $\mathbf{P} \neq \mathbf{NP}$  then there exists a problem  $C$  in  $\mathbf{NP} - \mathbf{P}$  that is not  $\mathbf{NP}$  complete.

# Hierarchy Theorems

**If**  $f(n)$  is a  $\mathcal{C}$ -constructible function;

$\mathcal{C}$  is **DSPACE**, **NSPACE**, **DTIME**, or **NTIME**; and,

if  $g(n)$  is sufficiently smaller than  $f(n)$

**Then**  $\mathcal{C}[g(n)]$  is strictly contained in  $\mathcal{C}[f(n)]$ .

$g(n)$  **sufficiently smaller** than  $f(n)$  means:

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

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

$\mathcal{C} =$  **DSPACE, NSPACE, NTIME**

$\mathcal{C} =$  **DTIME**

**Thm:** The busy beaver function is eventually larger than any total, recursive function.

**Th:** Let  $S \subseteq \mathbf{N}$ . T.F.A.E.

1.  $S$  is the domain of a partial, recursive function.
2.  $S = \emptyset$  or  $S$  is the range of a total, recursive function.
3.  $S$  is the range of a partial, recursive function.
4.  $S = W_n$ , some  $n = 0, 1, 2, \dots$  where

$$W_n = \mathcal{L}(M_n) = \{m \mid M_n(m) = 1\}$$

## Logic: definitions of **Formula**, **Structure** and **Truth**

**Proof rule:** (M.P.,  $\rightarrow$  elim) From  $\varphi$ ,  $\varphi \rightarrow \psi$ , conclude  $\psi$ .

**Axioms:** all generalizations of the following:

0	Tautologies on at most three boolean variables
1a	$t = t$
1b	$(t_1 = t'_1 \wedge \dots \wedge t_k = t'_k) \rightarrow f(t_1, \dots, t_k) = f(t'_1, \dots, t'_k)$
1c	$(t_1 = t'_1 \wedge \dots \wedge t_k = t'_k) \rightarrow R(t_1, \dots, t_k) \rightarrow R(t'_1, \dots, t'_k)$
2	$\forall x (\varphi) \rightarrow \varphi[x \leftarrow t]$
3	$\varphi \rightarrow \forall x (\varphi)$ , $x$ not free in $\varphi$
4	$\forall x (\varphi \rightarrow \psi) \rightarrow (\forall x (\varphi) \rightarrow \forall x (\psi))$

**Soundness Thm:** If  $\vdash \varphi$  then  $\models \varphi$ .

$$\text{FO-THEOREMS} \subseteq \text{FO-VALID}$$

**Completeness Thm:** If  $\models \varphi$  then  $\vdash \varphi$ .

$$\text{FO-THEOREMS} \supseteq \text{FO-VALID}$$

**Cor:**  $\vdash = \models$ ;  $\text{FO-THEOREMS} = \text{FO-VALID}$

**Compactness Thm:** If every finite subset of  $\Gamma$  has a model, then  $\Gamma$  has a model.

**Gödel's Incompleteness Thm:**  $\text{Theory}(\mathbf{N})$  is not r.e. and thus not axiomatizable.

# Complexity Classes

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

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

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

**Savitch's Theorem:** For  $s(n) \geq \log n$ ,

$$\mathbf{NSPACE}[s(n)] \subseteq \mathbf{ATIME}[(s(n))^2] \subseteq \mathbf{DSPACE}[(s(n))^2]$$

**Immerman-Szelepcsényi Thm:** For  $s(n) \geq \log n$ ,

$$\mathbf{NSPACE}[s(n)] = \mathbf{co-NSPACE}[s(n)]$$

# Reductions

**Def:**  $A$  is **reducible** to  $B$ , ( $A \leq B$ ), iff  $\exists f \in F(L)$  such that,

$$\forall w \in \Sigma^* \quad (w \in A \Leftrightarrow f(w) \in B)$$

The question whether  $w \in A$  is reduced to the question whether  $f(w) \in B$ :

- If  $f(w) \in B$  then  $w \in A$ , and the answer is “yes”
- If  $f(w) \notin B$  then  $w \notin A$ , and the answer is “no”

The reduction function involves minimal computation, i.e.,  $f \in F(L)$ , or even weaker, e.g.,  $f \in \mathbf{FO}$ .

The problem,  $B$ , that we are reducing to does all the hard work.

# Fundamental Theorem of Reductions

**Thm:** Let  $\mathcal{C}$  be one of the following complexity classes:  
**L, NL, P, NP, CO-NP, PSPACE, EXPTIME,**  
**Primitive Recursive, Recursive, r.e., CO-r.e..**

Suppose  $A \leq B$ .      If  $B \in \mathcal{C}$     Then  $A \in \mathcal{C}$ .

All the above complexity classes are **closed under reductions.**

**Lower Bounds:** If  $A$  is hard then  $B$  is hard.

**Upper Bounds:** If  $B$  is easy then  $A$  is easy.

# Complete Problems

Complexity Class	Complete Problems
NL	REACH, 2-SAT
P	CVP, MCVP, HORN-SAT, AREACH, CFL-EMPTY
NP	TSP, SAT, 3-SAT, 3-COLOR, CLIQUE, SUBSET SUM, KNAPSACK
PSPACE	QSAT, GEOGRAPHY, SUCCINCT REACH, NFA- $\Sigma^*$
r.e.	K, HALT, FO-VALID
co-r.e.	$\bar{K}$ , $\Sigma^*$ CFL, EMPTY, FO-SAT

# Descriptive Complexity

**Arithmetic Hierarchy** = **FO(N)**

**r.e.** = **FO $\exists$ (N)**

**co-r.e.** = **FO $\forall$ (N)**

**PH** = **SO**

**NP** = **SO $\exists$**

**P** = **SO-Horn**

**AC<sup>0</sup>** = **CRAM[1]** = **LH** = **FO**

One can understand the complexity of a problem via the richness of a logical language that is needed to describe the problem.

# Alternation

**Thm:** For  $s(n) \geq \log n$ , and for  $t(n) \geq n$ ,

$$\bigcup_{k=1}^{\infty} \mathbf{ATIME}[(t(n))^k] = \bigcup_{k=1}^{\infty} \mathbf{DSPACE}[(t(n))^k]$$

$$\mathbf{ASPACE}[s(n)] = \bigcup_{k=1}^{\infty} \mathbf{DTIME}[k^{s(n)}]$$

**Cor:**

$$\mathbf{ASPACE}[\log n] = \mathbf{P}$$

$$\mathbf{ATIME}[n^{O(1)}] = \mathbf{PSPACE}$$

$$\mathbf{ASPACE}[n^{O(1)}] = \mathbf{EXPTIME}$$

# Circuit Complexity

depth = parallel time

width = hardware

number of gates = computational work = sequential time

**Thm:** For all  $i$ ,  $\text{CRAM}[(\log n)^i] = \text{AC}^i$

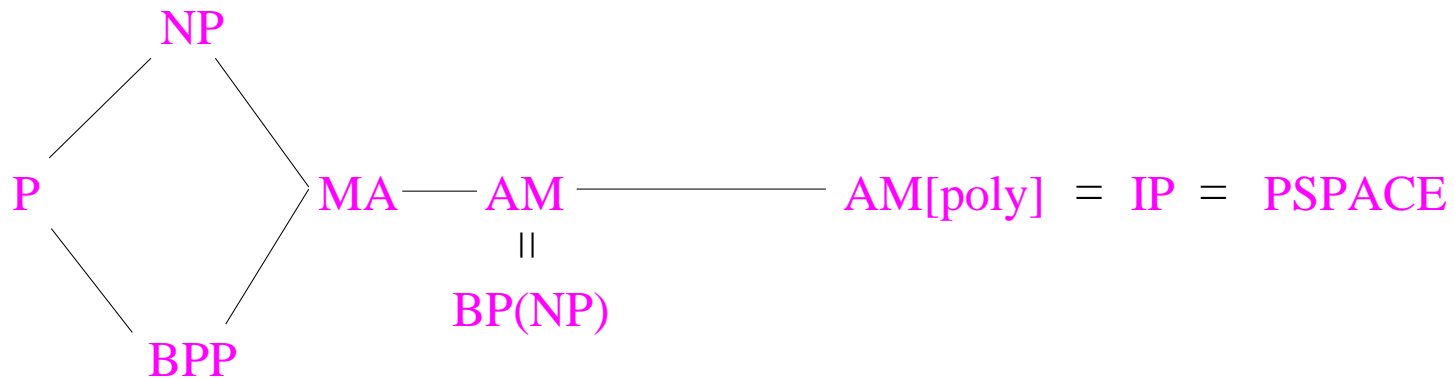
$$\begin{array}{cccccccccccc} \text{AC}^0 & \subseteq & \text{ThC}^0 & \subseteq & \text{NC}^1 & \subseteq & \mathbf{L} & \subseteq & \mathbf{NL} & \subseteq & \text{sAC}^1 & \subseteq & \text{AC}^1 \\ \text{AC}^1 & \subseteq & \text{ThC}^1 & \subseteq & \text{NC}^2 & & & & & \subseteq & \text{sAC}^2 & \subseteq & \text{AC}^2 \\ \text{AC}^2 & \subseteq & \text{ThC}^2 & \subseteq & \text{NC}^3 & & & & & \subseteq & \text{sAC}^3 & \subseteq & \text{AC}^3 \\ \vdots & \subseteq & \vdots & \subseteq & \vdots & & & & & \subseteq & \vdots & \subseteq & \vdots \\ & \subseteq & \text{NC} & \subseteq & \mathbf{P} & \subseteq & \mathbf{NP} & \subseteq & \mathbf{PH} & \subseteq & \mathbf{PSPACE} & & \end{array}$$

**Thm:** PRIME, FACTORING  $\in$  NP  $\cap$  co-NP.

**Thm:** PRIME  $\in$  BPP in fact, P

**Thm:** UREACH  $\in$  BPL in fact, L

## Interactive Proofs



**Thm:** PCP[log n, 1] = NP

# Where's the Catch?

*Why are the following so hard to prove?*

- **$P \neq NP$**
- **$P \neq PSPACE$**
- **$ThC^0 \neq NP$**
- **$BPP = P$**

We do know a lot about computation. Reductions and complete problems are a key tool. So is the equivalence of apparently different models and methods. Yet much remains unknown.

