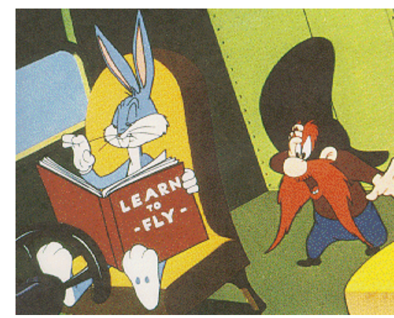
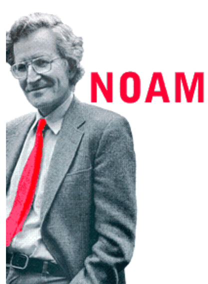
All good things…must come to a q_{ACC} state

Course Highlights





Chapter 0: Highlights

- Sets, strings, and languages
  - Operations on strings/languages (concat $\circ$, *, $\cup$, -, etc)
  - Complement of L = Σ^* - L
  - Lexicographic ordering of strings in Σ^*
- Proof techniques
- Pigeonhole principle

- Dovetailing and Diagonalization
  - Countably infinite and uncountable sets

Regular Languages: Highlights

- DFAs and NFAs
  - Equivalance
- Regular languages and their properties
- Regular expressions and GNFAs
  - Equivalence with NFAs/DFAs
- The Pumping lemma

Da Pumpin' Lemma

(adapted from a poem by Harry Mairson)

Hear it on the new album:
Dig dat funky DFA

Any regulah language *L* has a magic numba *p*
And any long-enuff word *s* in *L* has da followin' propa'ty:
Amongst its first *p* symbols issa segment u can find
Whoz repetition or omission leaves *s* amongst its kind.

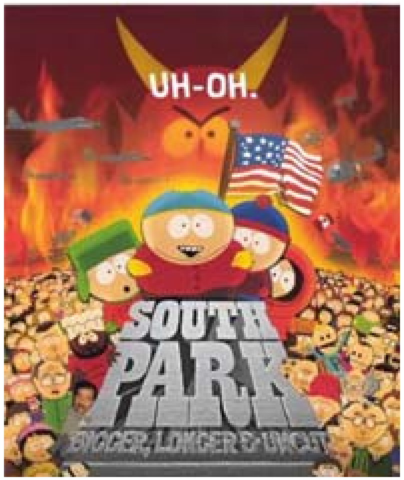
So if ya find a lango *L* which fails dis acid test,
And some long word ya pump becomes distinct from all da rest,
By contradixion ya have shown *L* is not
A regular homie, resilient to da pumpin' u've wrought.

But if, on da otha' hand, *s* stays within its *L*,
Then eitha *L* is regulah, or else ya chose not well.
For *s* is *xyz*, where *y* is not empty,
And *y* must come befo' da $p+1^{th}$ symbol u see.

Based on: http://www.cs.brandeis.edu/~mairson/poems/node1.html

Context Free Languages: Highlights

- Context Free Grammars: CFG $G = (V, \Sigma, R, S)$
  - Ambiguity
- Closure properties of Context-Free languages
  - Closed under $\cup$, concat, * *but not* $\cap$ *or complementation*
- Pushdown Automata: PDA $P = (Q, \Sigma, \Gamma, \delta, q_0, F)$
- CFGs and PDAs are equivalent in computational power
- Return of the Pumping Lemma
  - Property obeyed by all CFLs
  - Used to show languages are not CFLs



Turing Machines

- TM $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{ACC}, q_{REJ})$
  - *Configurations* of a TM capture its computation
- A language is Turing-recognizable if there is a TM M such that $L(M) = L$
  - For all strings in L, M halts in state q_{ACC}
  - For strings not in L, M may either halt in q_{REJ} or loop forever
- A language is decidable if there is a "decider" TM M such that $L(M) = L$
  - For all strings in L, M halts in state q_{ACC}
  - For all strings not in L, M halts in state q_{REJ}
- Implementation and high level description of TMs

The Church of Turing

Revelations 101: The Church-Turing Thesis

- Varieties of TMs: Multi-tape, multi-headed TMs, Nondeterministic TMs (NTMs), enumerator TMs etc.
  - All are equivalent to standard TM
- <u>Church-Turing Thesis (not a theorem!)</u>: Any formal definition of "algorithms" or "programs" is equivalent to Turing machines

To be or not to be decidable…

- Any problem can be cast as a language membership problem
  - Does DFA D accept input w?
  - Equivalent to:
    Is $\langle D,w \rangle$ in $A_{DFA} = \{\langle D,w \rangle \mid D \text{ is a DFA that accepts input } w\}$?
- Decidable problems are those that can be solved by algorithms (decider TMs): A_{DFA}, A_{NFA} , A_{REX} , $A_{empty\text{-}DFA}$, A_{CFG} , $A_{empty\text{-}CFG}$ etc.
- Many problems are undecidable
  - A_{TM}: Turing-recognizable but not decidable (Proof by diagonalization)
- Can also use the concept of *reducibility* to show undecidability
- Some problems are not even recognizable

Reducibility and Unrecognizability

Reducibility and Unrecognizability

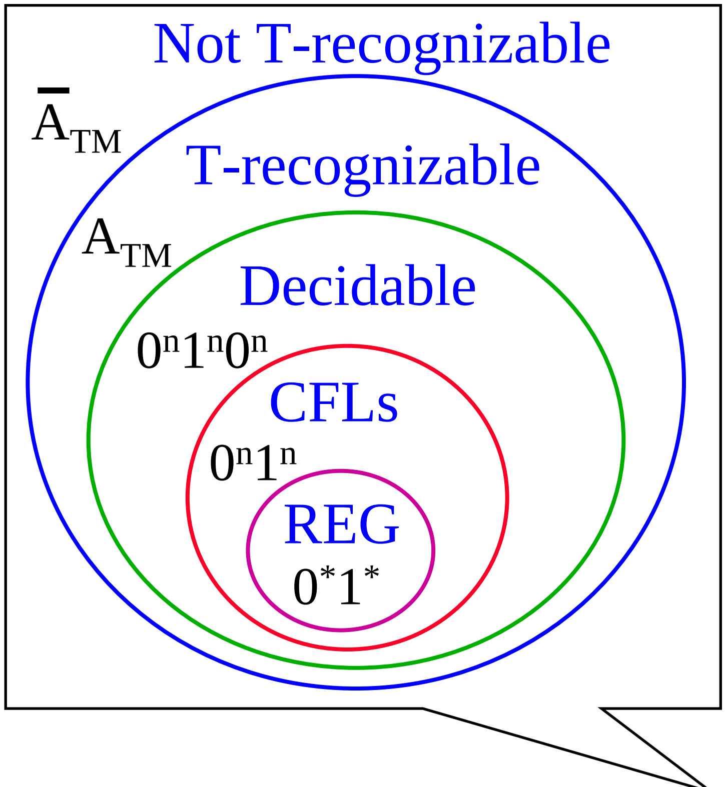
- To show a new problem A is undecidable, reduce A_{TM} (or some other undecidable problem) to A
  - Use a decider for A as a *subroutine* to decide the undecidable problem (and get a contradiction)
  - E.g. Halting problem = "Does a program halt for an input or go into an infinite loop?"
  - Can show Halting problem is undecidable by reducing A_{TM} to $A_H = \{ \langle M,w \rangle \mid \text{TM } M \text{ halts on input } w \}$
  - Similarly for $E_{TM} = \{ \langle M \rangle \mid M \text{ is a TM and } L(M) = \varnothing \}$
- A is decidable iff A and $\overline{A}$ are both Turing-recognizable
  - Corollary: $\overline{A_{TM}}$ and $\overline{A_H}$ are <u>not Turing-recognizable</u>

The Chomsky Hierarchy of Languages

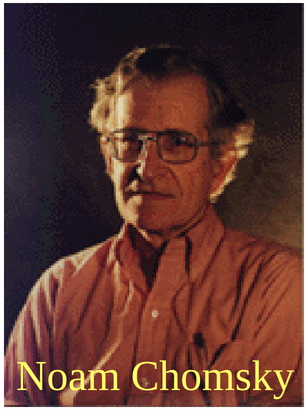
Increasing generality →

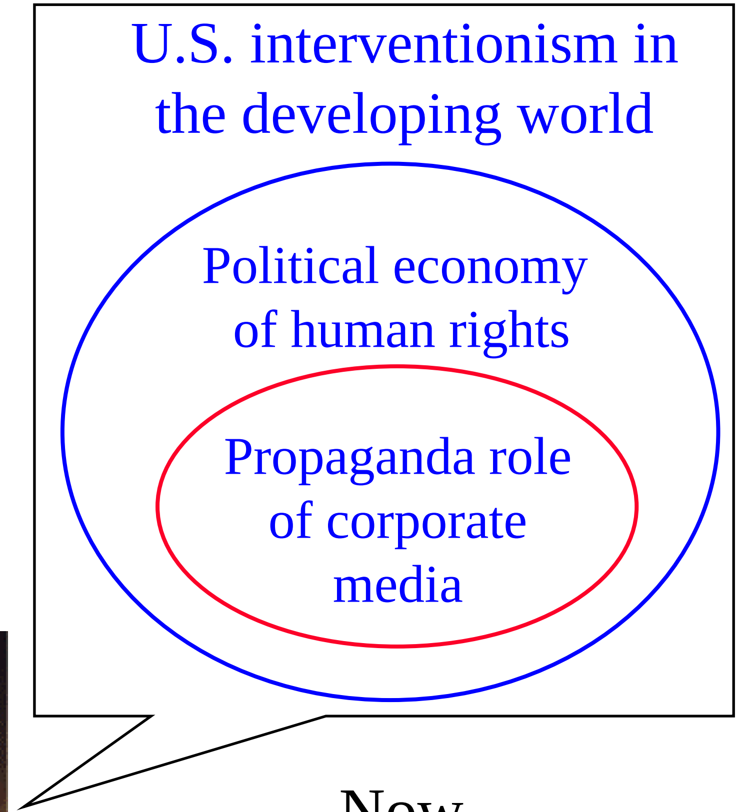
Language	Regular	Context-Free	Decidable	Turing-Recognizable
Computational Models	DFA, NFA, RegExp	PDA, CFG	Deciders – TMs that halt for all inputs	TMs that may loop for strings not in language
Examples	$(0 \cup 1)^*11$	$\{0^n 1^n \mid n \geq 0\}$, $\{ww^R \mid w \in \{0,1\}^*\}$	$\{0^n 1^n 0^n \mid n \geq 0\}$, A_{DFA}, A_{CFG}	A_{TM}, A_H, E_{TM}

The Chomsky Hierarchy – Then & Now…



Then (1950s)

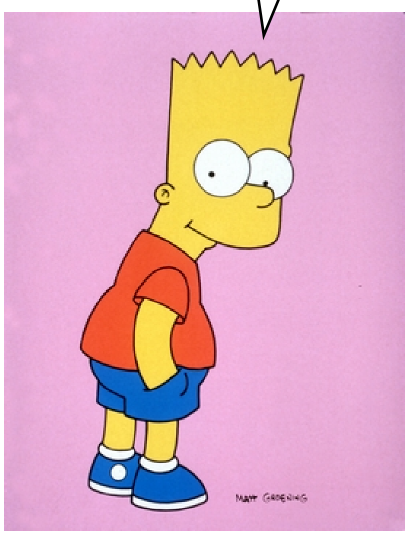




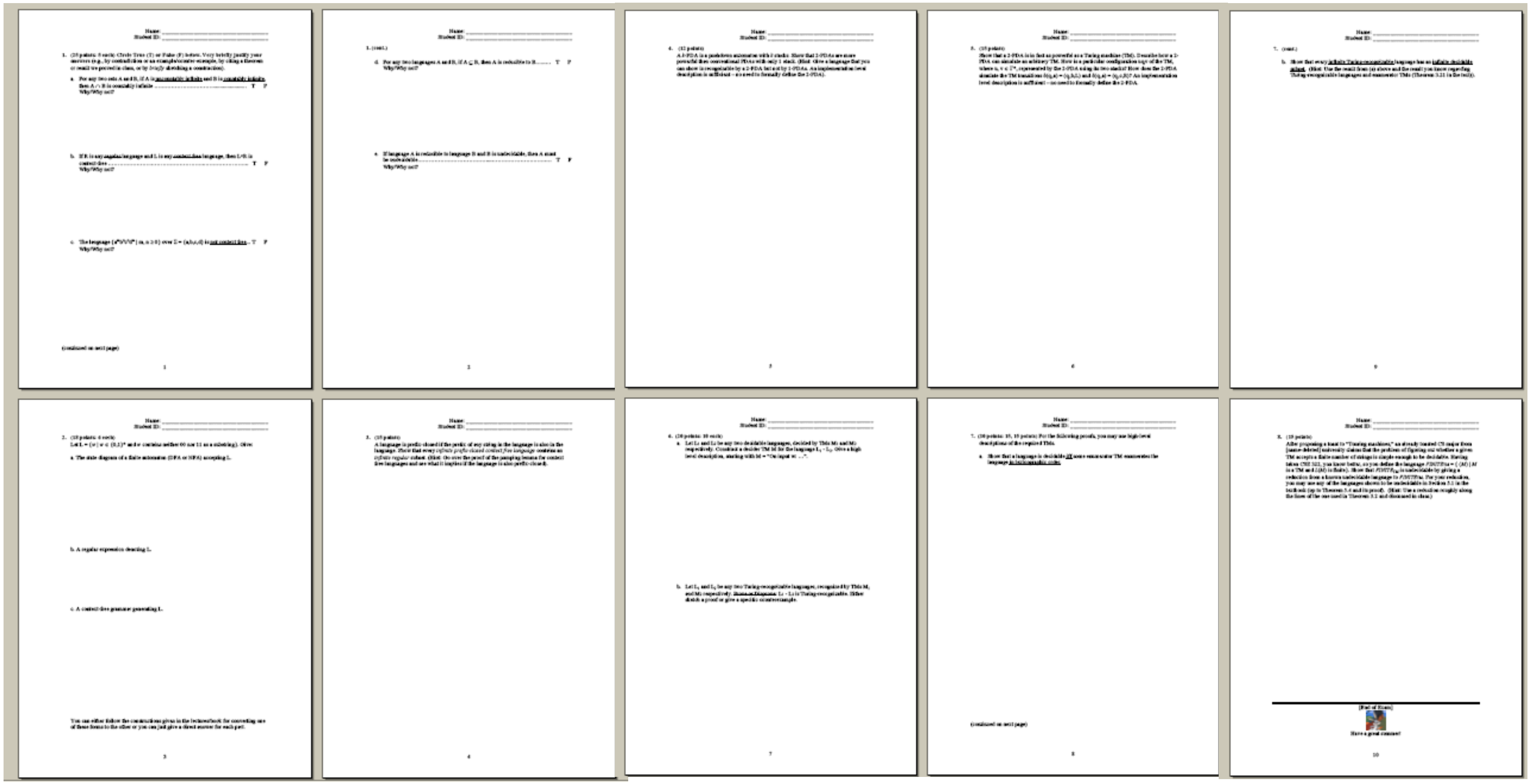
Now

The Final Exam is 2 slides away…

This space for rent



The Final Exam



On class website after class today

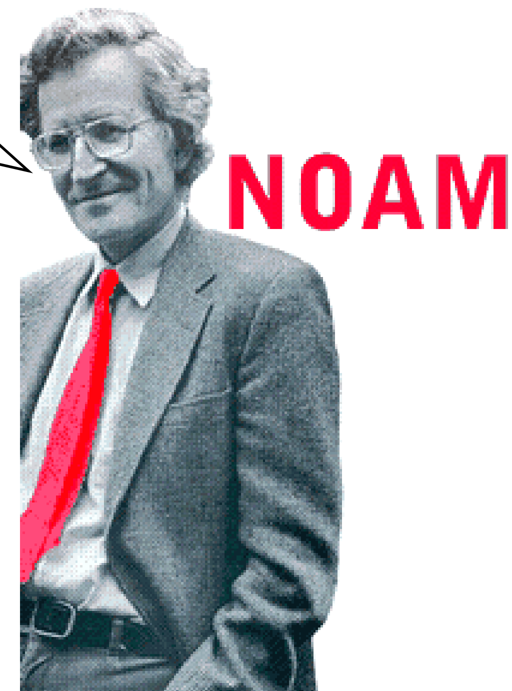
Solutions to the Final Exam

On class website on Tuesday

Stay cool with da pumpin' lemma!

The final exam is decidable!

I believe the world's problems are politically decidable.



I believe my next movie will be unrecognizable.