CS 273

5/1/2007

May Day!

Last Day!

- CYK algorithm for parsing
- Review of Big Ideas
- Looking forward
- Evaluations

Given a CFG G in CNF form

and a $\omega \in \Sigma^*$

is $\omega \in L(G)$?

Ex: C program

is it a valid program?

G – grammar for C

G : in CNF form

$w \in \Sigma^*$

$|w| = n$ $\quad$ G is fixed

CYK -- Cocke-Younger-Kasami

$O(n^3)$ time algorithm

$O(g\, n^3)$

↑ size of grammar

Valiant reduced this problem to boolean matrix multiplication

To multiply two $n \times n$ matrices there is an $O(n^{2.3\ldots})$ time algorithm.

Given G, w

is $w \in L(G)$

$w \in L(G) \iff S \overset{*}{\Rightarrow} w$

$S \Rightarrow AB \overset{*}{\Rightarrow} w$

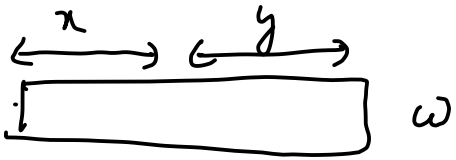
$A \overset{*}{\Rightarrow} x$

$B \overset{*}{\Rightarrow} y$

$w = xy$

$|x| < |w|$

$|y| < |w|$



to check $S \overset{*}{\Rightarrow} w$

need to know how substrings of w are derived by other variables

algorithmic idea

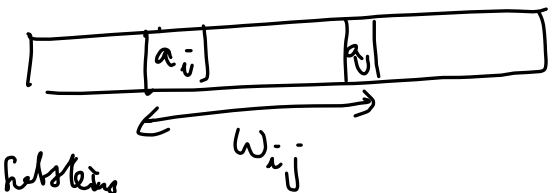
— dynamic programming

a way to make recursive algorithms efficient by re-using work. (via storing)

$w = a_1 a_2 a_3 \cdots a_n$

$a_i \in \Sigma \quad i = 1, 2, \ldots, n$

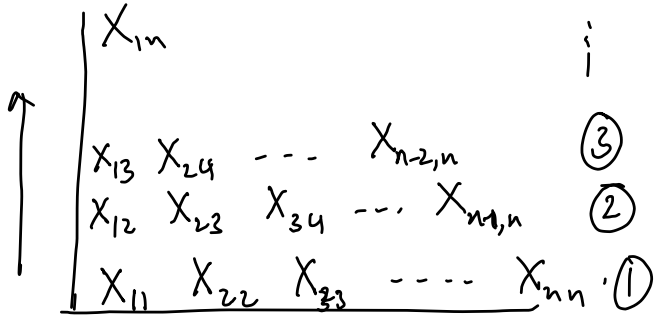
$w_{ij} = a_i a_{i+1} \cdots a_j$



$\binom{n}{2} = \frac{n(n-1)}{2}$ subsets

$X_{ij} = \{ A \in V \mid A \overset{*}{\Rightarrow} w_{ij} \}$

$X_{ii} = \{ A \mid A \overset{*}{\Rightarrow} a_i \}$



$$w \in L(G) \iff S \overset{*}{\Rightarrow} w$$

$$\iff S \in X_{1n}$$

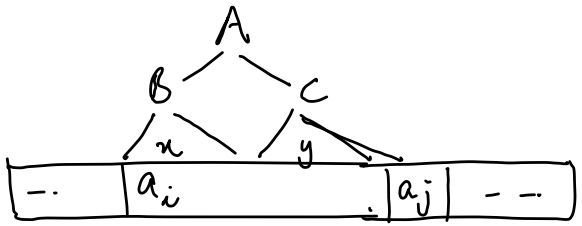
$$X_{ij} = \{ A \mid A \overset{*}{\Rightarrow} w_{ij} \}$$

$A \in X_{ij}$ iff $\exists$ k, B, C (s.t. $j-i \geq 1$)

① $A \to BC$ is a rule in G

② $B \in X_{ik}$

③ $C \in X_{k+1,j}$



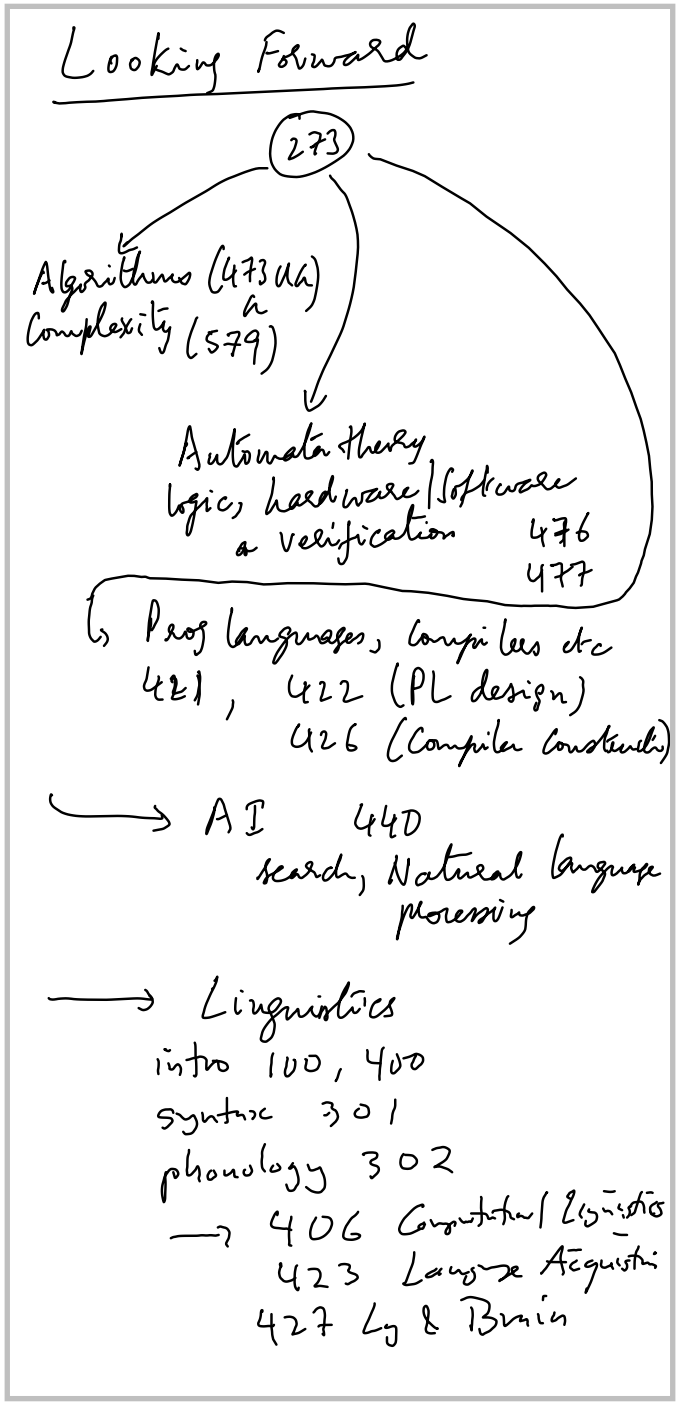
Algorithm

1. for $i = 1$ to n
   Compute X_{ii} using base case

2. for $l = 1$ to $n-1$ do // increasing substring length
   for $i = 1$ to $n-l$ do
   $j = i + l$
   $X_{ij} = \phi$
   for $k = i$ to $j-1$ do
   $X_{ij} = X_{ij} \cup \{ A \mid$
   $A \to BC$ is a rule
   $B \in X_{ik}$
   $C \in X_{k+1,j} \}$

Review of Big Ideas

1. Computation is a formal object that can/should be reasoned about
2. Non-determinism (a mathematical construct) (P vs NP)
3. Equivalence of different models via simulation (Specification vs Verification logic vs computation)
4. Formal defn of algorithm as a TM
5. Consequences of Alg $\Leftrightarrow$ TM
   - undecidability of useful tasks
   - formal way to reason about computational complexity
6. Reductions


Looking Forward
273
Algorithms (473 UA)
&
Complexity (579)
Automata theory,
logic, hardware/software
& verification 476
477
Prog languages, compilers etc
421, 422 (PL design)
426 (Compiler construction)
AI 440
search, Natural language
processing
Linguistics
intro 100, 400
syntax 301
phonology 302
406 Computational Linguistics
423 Language Acquisition
427 Lg & Brain