Theory of Computation

Introduction

Barath Raghavan
CS 361 Fall 2009
Williams College
What is COMPUTATION?
Input

Program

Output
Input

Human Genome (3 billion base pairs)
Input
Human Genome (3 billion base pairs)

Program
Find the PARK7 gene sequence
(Parkinson disease autosomal recessive, early onset 7)
Input
Human Genome (3 billion base pairs)

Program
Find the PARK7 gene sequence
(Parkinson disease autosomal recessive, early onset 7)

Output
1 (found) / 0 (not found)
How might you write such a program?
Suppose I want to login to a system:
donuts.
The system needs to accept or reject my password.
Usual approach:
Store $s = \text{hash}($\text{realpw}\text{)}$. Check $\text{hash}($\text{pw}\text{)} = s$. 
Consider this code:
/* returns NULL terminated string of arbitrary length */
char * hash(char * r);

/* pw and s are NULL terminated */
void func(char * pw, char * s) {
    while (1) {
        if (strcmp(pw, s) == 0) {
            printf("pw accepted");
        }
        pw = hash(pw);
    }
}
Input

char * pw, char * s, code for func()
Program

Determine whether func(pw, s) prints “pw accepted”
Input

char * pw, char * s, code for func()

Program

Determine whether func(pw, s) prints “pw accepted”

Output

1 (prints “pw accepted”) / 0 (otherwise)
How might you write such a program?
Some programs are easy, some are hard, some are impossible.
1. Computation without storage.
   (Finite automata, regular expressions)

2. Computation with storage.
   (Pushdown automata, context-free grammars)

3. Limits of computation.
   (Turing machines, undecidability)

   (Complexity classes, reductions)
Assessment

40%: Homework
40%: Quizzes
15%: Final Exam
5%: Class Participation
Homework

8 assignments
100 points each

$N$ hours late: $2^{N/6}$ point penalty

LaTeX typeset: 10% bonus
Quizzes

4 quizzes
Self-administered
Assigned in class, due in 24 hours
No late turn-ins
Final Exam

Take home, self-administered
Comprehensive
Assigned on 12/12, due on 12/15
No late turn-ins
Class Participation

Keep the discussion going.
Misc

Office hours: Monday, Tuesday, Thursday 2:30-4.
Skipping assessments: give me 2 weeks notice.
Homework schedule: mostly weekly.
Lecture slides: on webpage after each lecture.
Recommended reading: given each lecture.
Reading: Chapter 0.
Questions!
We need formal specifications.
Specify what this looks like → Input → Program → Output
Symbols compose an alphabet

\[ \Sigma = \{0, 1\} \]

\[ \Sigma = \{a, b, c, d, e, f\} \]
An alphabet is used to make strings

$s_1 = 1001010$

$s_2 = \text{feed}$
String operations

\[ s = 1001 \]
\[ ss = 100111001 \]
\[ s^2 = 100111001 \]
\[ |s| = 4 \quad |\varepsilon| = 0 \]
\[ 0001^R = 1000 \]
A set of strings makes a language

\[ L_1 = \{01, 10, 011\} \]

\[ L_2 = \{\text{cad}, \text{feed}\} \]
A set of languages makes a language class

\[ L_1 = \{0, 000, \ldots\} \]

\[ L_2 = \{00, 00000, \ldots\} \]

\[ C = \{L_1, L_2\} \]
Language definitions can be colloquial

\[ L_3 = \{ss \mid s \text{ has more 0s than 1s} \} \]
Simple model of computation: the automaton.
Consider an elevator.
If we end in state 3, the automaton $A$ accepts. Otherwise it rejects.
Language B of all button sequences that end on the 3rd floor.
Automaton $A$ recognizes $L(A) = B$

“the language of $A$”
Claim:
In any set of $n$ hats, all are the same color.

“Proof” by induction:
Base case: If $n = 1$, all hats are the same color.
Inductive case: Assume that the claim holds for all $n \leq k$. Examine any set of $n = k+1$ hats. Remove one hat from the set; the remaining set has $k$ hats and thus is of a single color. Replace the hat and remove a different hat from the set; the resulting set has $k$ hats and thus is of a single color. The two sets overlap, thus they must be of the same color. Therefore, all $k+1$ hats are the same color.  "\[\square\]"