MA/CSSE 474

Theory of Computation

Course Intro
Finish T = S proof from Day 1
More about strings and languages

Today's Agenda

- Roll call
- Student questions
- Introductions and Course overview
- Overview of yesterday's proof
  - I placed online a "straight-line" write-up of the proof in detail, without the "here is how a proof works" commentary that was in the slides. See Session 1 resources on schedule page
- Responses to Reading Quiz 1 (0 4 8 12)
- Languages and Strings
- (if time) Operations on Languages

Introductions

- Roll Call
  - If I mispronounce your name, or you want to be called by a nickname or different name but did not list that yesterday, let me know.
  - I have had most of you in class, but for some of you it has been a long time.
- Graders: 8 of them! See schedule page, day 1
- Instructor: Claude Anderson: F-210, x8331
- Random Note: I often put more in my PowerPoint slides for a day than I expect we can actually cover that day, "just in case".

Instructor Professional Background

See optional video on Moodle for some personal background

- **Formal Education:**
  - BS Caltech, Mathematics 1975
  - Ph.D. Illinois, Mathematics 1981
  - MS Indiana, Computer Science 1987
- **Teaching:**
  - TA at Illinois, Indiana 1975-1981, 1986-87
  - Wilkes College (now Wilkes University) 1981-88
  - RHIT 1988 –??
- **Major Consulting Gigs:**
  - Pennsylvania Funeral Directors Assn 1983-88
  - Navistar International 1994-95
  - Beckman Coulter 1996-98
  - ANGEL Learning 2005-2008
- **Theory of Computation history**

What do we Study in Theory of Computation?

- Larger issues, such as
  - What can be computed, and what cannot?
  - What problems are tractable?
  - What are reasonable mathematical models of computation?

Applications of the Theory

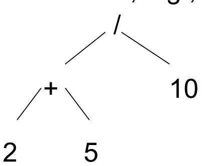
- Finite State Machines (FSMs) for parity checkers, vending machines, communication protocols, and building security devices.
- Interactive games as nondeterministic FSMs.
- Programming languages, compilers, and context-free grammars.
- Natural languages are mostly context-free. Speech understanding systems use probabilistic FSMs.
- Computational biology: DNA and proteins are strings.
- The undecidability of a simple security model.
- Artificial intelligence: the undecidability of first-order logic.

Some Language-related Problems

int alpha, beta;
alpha = 3;
beta = (2 + 5) / 10;

(1) **Lexical analysis**: Scan the program and break it up into variable names, numbers, operators, punctuation, etc.

(2) **Parsing**: Create a tree that corresponds to the sequence of operations that should be executed, e.g.,



(3) **Optimization**: Realize that we can skip the first assignment since the value is never used, and that we can pre-compute the arithmetic expression, since it contains only constants.

(4) **Termination**: Decide whether the program is guaranteed to halt.

(5) **Interpretation**: Figure out what (if anything) useful it does.

A Framework for Analyzing Problems

We need a single framework in which we can analyze a very diverse set of problems.

The framework we will use is

Language Recognition

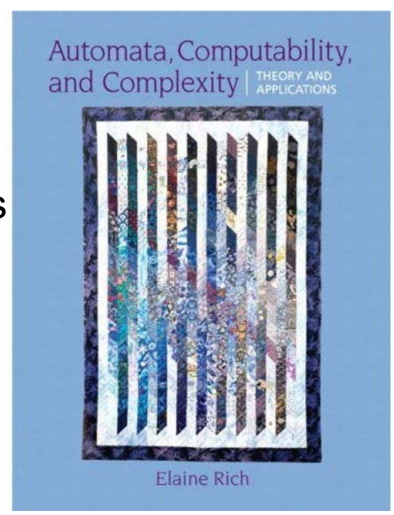
Most interesting problems can be restated as language recognition problems.

What we will focus on in 474

- Definitions
- Theorems
- Examples
- Proofs
- A few applications, but mostly theory

Textbook

- Thorough
- Literate
- Large (and larger!)
- Theory and Applications
- We'll focus more on theory; applications are there for you to see
- The book is online and free



Online Materials Locations

- On the Schedule page – public stuff
  - Reading, HW, topics, resources,
  - Suggestion: bookmark schedule page
- On Moodle – personal stuff
  - surveys, solutions, grades
- On piazza.com:
  - Discussion forums and announcements
- csse474-staff@rose-hulman.edu
- Many things are under construction and subject to change, especially the course schedule.

My most time-consuming courses (for students)

This is my perception, not a scientific study!

- 220 (object-oriented)
- 473 (design and analysis of algorithms)
- 280 (web programming)
- 304 (PLC)
- 404 (Compilers)
- 474 (Theory of Computation)
- 230 (Data Structures & Algorithms)

The learning outcomes include a lot of difficult material. Most of you will need a lot practice in order to understand it.

Questions about course policies and procedures?

- From Syllabus?
- Schedule page?
- Things said in class yesterday?
- Attendance?
- Early Days? (There are no late days)
- How to find my office hours for a given day?
- Anything else?

Prove S⊆T by induction on |w| :

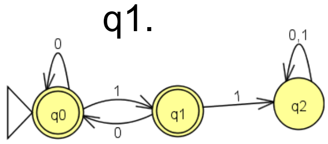
More general statement that we will prove:

Both of the following statements are true:

1. If δ(q0, w) = q0, then w does not end in 1 and w has no pair of consecutive 1's.
2. If δ(q0, w) = q1, w ends in 1 and w has no pair of consecutive 1's.

Can you see that (1) and (2) imply S⊆T?

- **Base case:** |w| = 0; i.e., w = ε.
  - (1) holds since ϵ has no 1's at all.
  - (2) holds ***vacuously***, since δ(q0, ϵ) is not q1.

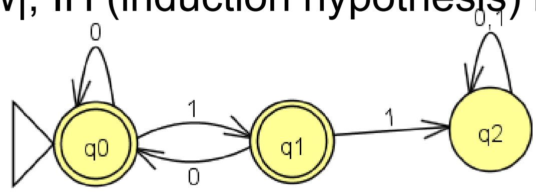


Important logic rule:
If the "if" part of any "if..then" statement is false, the whole statement is true.

15

Inductive Step for S ⊆ T

- Let |w| be ≥ 1, and assume (1) and (2) are true for all strings shorter than w.
- Because w is not empty, we can write w = ua, where *a* is the last symbol of w, and u is the string that precedes that last a.
- Since |u| < |w|, IH (induction hypothesis) is true for u.

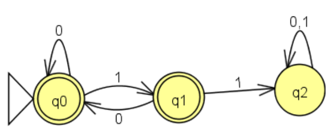


Reminder: What we are proving by induction:

1. If δ(q0, w) = q0, then w has no consecutive 1's and does not end in 1.
2. If δ(q0, w) = q1, then w has no consecutive 1's and ends in 1.

Inductive Step: S ⊆ T (2)

- Need to prove (1) and (2) for w = ua, assuming that they are true for u.
- (1) for w is: If δ(q0, w) = q0, then w has no consecutive 1's and does not end in 1. **Show it:**
- Since δ(q0, w) = q0, δ(q0, u) must be q0 or q1, and *a* must be 0 (look at the DFSM).
- By the IH, u has no 11's. The *a* is a 0.
- Thus, w has no 11's and does not end in 1.

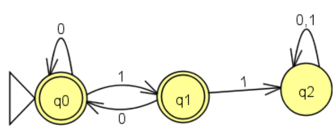


1. If δ(q0, w) = q0, then w has no consecutive 1's and does not end in 1.
2. If δ(q0, w) = q1, then w has no consecutive 1's and ends in 1.

17

Inductive Step : S ⊆ T (3)

- Now, prove (2) for w = ua: If δ(q0, w) = q1, then w has no 11's and ends in 1.
- Since δ(q0, w) = q1, δ(q0, u) must be q0, and *a* must be 1 (look at the DFSM).
- By the IH, u has no 11's and does not end in 1.
- Thus, w has no 11's and ends in 1.



1. If δ(q0, w) = q0, then w has no consecutive 1's and does not end in 1.
2. If δ(q0, w) = q1, then w has no consecutive 1's and ends in 1.

18

Part B: T ⊆ S

- Now, we must prove: if w has no 11's, then w is accepted by M

X (w has no 11's)

Y (w is accepted by M)

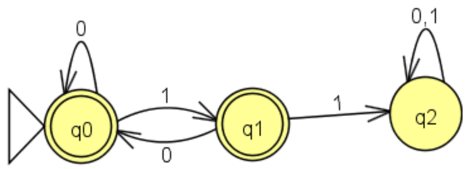
- *Contrapositive* : If w is *not* accepted by M then w has 11 as a substring.

Key idea: contrapositive of "if X then Y" is the equivalent statement "if not Y then not X."

19

Using the Contrapositive

- ***Contrapositive*** : If w is *not* accepted by M then w has 11 as a substring.
- **Base case** is again vacuously true.
- Because there is a unique transition from every state on every input symbol, each w gets the DFSM to exactly one state.
- The only way w can not be accepted is if it takes the DFSM M to q2. How can this happen?

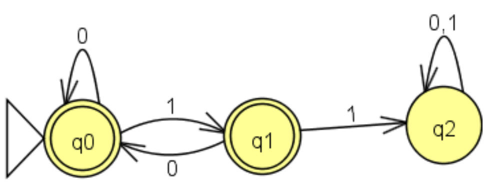


20

Using the Contrapositive – (2)

Looking at the DFSM, there are two possibilities: (recall that w=ua)

1. δ(q0,u) = q1 and a is 1. We proved earlier that if δ(q0,u) = q1, then u ends in 1. Thus w ends in 11.
2. δ(q0,u) = q2. In this case, the IH says that u contains 11 as a substring. So does w=ua.



21

Your 474 HW induction proofs

- Can be slightly less detailed
  - Many of the details above were about how the proof process works in general, rather than about the proof itself.
  - *You* can assume that the reader knows the proof *techniques*.
- You must always make it clear what the IH is, and where you apply it.
  - When in doubt about whether to include a detail, include it!
- Well-constructed proofs often contain more words than symbols.

This Proof as a 474 HW Problem

- An example of how I would write up this proof if it was a 474 HW problem will be linked from the schedule page this afternoon.
- You do not need to copy it exactly in your proofs, but it gives an idea of the kinds of things to include or not include.
- Also, I will post another version of the slides that includes the parts that I wrote on the board today.

Responses to Reading Quiz 1

Responses to Reading Quiz 1

- **From #4:** $\wp(\emptyset) = \{ \emptyset \}$ (not $\wp(\emptyset) = \emptyset$)
  What is $\wp(\wp(\emptyset))$?
- **From #4:** $\{a, b\} \times \{1, 2, 3\} \times \emptyset = \emptyset$
- **#10:** (representing {1, 4, 9, 16, 25, 36, …}
  in the form: $\{x \in A : P(x)\}$
  $\{x \in \mathbb{N} : x>0 \land \exists y \in \mathbb{N}\ (y*y = x)\}$
  Why not $\{x \in \mathbb{N} : x>0 \land \mathrm{sqrt}(x) \in \mathbb{N}\}$?
- **From #15:** $\forall x \in \mathbb{N}\ (\exists y \in \mathbb{N}\ (y < x))$.
  Why is this **not** satisfiable? (e. g. by x=3, y=2)

Responses to Reading Quiz 1

#16: Let $\mathbb{N}$ be the set of nonnegative integers. Let *A* be the set of nonnegative integers *x* such that $x \equiv_3 0$.
Show that $|\mathbb{N}| = |A|$.

Define a function $f : \mathbb{N} \rightarrow A$ by f(n) = 3n.

f is one-to-one: if f(n) = f(m), then 3n = 3m, so m=n.

f is onto: Let $k \in A$. Then k = 3m for some $m \in \mathbb{N}$. So k = f(m).

Responses to Reading Quiz 1

#19: Prove by induction: $\forall n>0\ (n! \geq 2^{n-1})$.
Why is the following "proof" of the induction step shaky at best, perhaps wrong?

$(n+1)! \geq 2^n$	*what we're trying to show*
$(n+1)n! \geq 2(2^{n-1})$	*definitions of ! And exponents*
$(n+1) \geq 2$	induction hypothesis ($n! \geq 2^{n-1}$)

Since n is at least 1, this statement is true, therefore $(n+1)! \geq 2^n$ *is true.*