COMP2022|2922
Models of Computation

USS, Recap, Exam Prep

Sasha Rubin

November 3, 2022

THE UNIVERSITY OF SYDNEY

USS

Student feedback matters. It is used to improve the course!

- based on USS and tutor feedback from 2020/2021, this year I:
  1. shortened the tutorials
  2. connected the lectures to the tutorials more
  3. made expectations for the course clearer
  4. created a stricter timeline for returning assignments to students
- USS is anonymous and confidential.
- Usually it is students that were unhappy with some part of the course that respond. This is good! But we also need to know what went **well** so that we don't make uneccessary changes to the course and so that credit goes where credit is due.
- Also, the lecturers talk to each other, so your feedback now could benefit you in another course.

Looking Back

We covered a lot of ground!

- Formal Languages and Regular Expressions
- Regular Languages
- Turing Machines
- Propositional Logic
- Predicate Logic
- Context-free Languages

Important ideas

1. Models can be used to specify:
   - Regular expressions specify tokens.
   - Grammars specify valid programs.
   - Logical formulas specify database queries.
2. Models can be used for computation:
   - Automata recognise strings corresponding to tokens.
   - The Satisfiability problem can be used to solve other problems.
3. Specification and Computation are often related!
   - RE = DFA

General skills

1. How to write and apply recursive definitions to do with computational problems.
2. How to model computational problems using models of computation (such as REs, DFAs, NFAs, CFGs, TMs).
3. How to reason about computational problems and models of computation.

What follows is a non exhaustive list of specific notions/skills that you should have mastered.

Regular languages

1. Know all definitions (REs, DFAs, NFAs, etc.) and the connections between these notions.
2. Show that a given language is regular (by building RE, DFA, NFA for it).
3. Describe the language of a given RE/DFA/NFA.
4. Prove that a given language is not regular.
5. (Transform RE to NFA with epsilon transitions, to NFA without epsilon transitions, to DFA, to RE.)
6. Reason about regular languages, including closure properties.

Universal models of computation

1. Know all definitions (Turing machine, Turing-decidable, Turing-recognisable, etc.) and the connections between these notions as well as to weaker models of computations (DFAs, NFAs, CFGs, etc).
2. Describe the language recognised by a given TM.
3. Create a TM for recognising or deciding a given language.
4. Reason about Turing-decidable/recognisable languages, including closure properties.

Propositional logic

1. Know all definitions (propositional formula, subformula, logically equivalence, satisfiability, validity, NNF, CNF, etc.) and the connections betweeen these notions.
2. Express propositional logic formulas in English and vice-versa.
3. Recognise and apply shorthands for propositional logic.
4. Prove that two formulas are logically equivalent.
5. Check if a formula is valid.
6. Put a formula in NNF, CNF.
7. Construct proofs in natural deduction.
8. Analyse algorithms for propositional logic, e.g., for testing satisfiability, validity, logical equivalence, for putting formulas into NNF, CNF.

Predicate logic

1. Know all definitions (predicate, variable, quantifiers, etc.)
2. Express predicate logic formulas in English and vice-versa.
3. Be able to evaluate if a sentence is true.
4. Be able to apply laws of equivalence, e.g., to put formulas into NNF, PNF.
5. Construct proofs in natural deduction.

Context-free languages

1. Know all definitions (CFGs, generate, yield, CNF, etc.) and the connections between these notions.
2. Create a CFG for a given CFL, and explain each rule.
3. Describe the language of a given CFG.
4. Show that a given CFG is ambiguous.
5. Argue that a given CFG is not ambiguous.
6. Convert a CFG to CNF.
7. Apply the CYK algorithm to a grammar in CNF and an input string.
8. Reason about context-free languages, including closure properties.

Learning outcomes

LO1. Knowledge of basic discrete mathematics and proof techniques.

LO2. Understand the basic models of computation, including their relationships with each other, their strengths and weaknesses.

LO3. Understand Propositional and Predicate Logic, including their relationship with each other.

LO4. Have obtained practical experience in designing formal machines and grammars for computing formal languages.

LO5. Have obtained practical experience in using Logic for formal reasoning about computation, including as a specification language for computation.

Overview of exam

- 3 hour writing plus 10 minutes reading
- Five sections, total of 50 points
- Worth 50% of your overall COMP2-22 grade
- Final exam has a 40% barrier to pass the course

Overview of exam

Five sections

1. Regular languages
2. Turing recognisable/decidable languages
3. Propositional logic
4. Predicate logic
5. Context-free languages

Questions in each section are roughly in increasing order of difficulty.

Open Book Exam

- Always write in your own words.
- Can refer to slides, tutorial solutions, assignment solutions, books used in the unit (making a 2-page summary is highly recommended)
- Can't share anything about the exam with anyone else
- Never copy text verbatim from anywhere, including the slides (few points and potential academic dishonesty). If you do refer to anything from the permitted material, write it in your own words.

Submission Format

- Type your answers and submit it as an assignment in Canvas (there is a special canvas page just for this exam).
  - Handwritten/scanned answers will not be accepted.
    - The only exception is diagrams.
    - But, no scanned paragraphs of handwritten text in diagrams allowed
  - Your pdf must be scannable by turnitin
- Start your submission with your student ID (in the top-left corner). Don't include your name.
- Do not contact teaching staff during the exam.
  - If you think you found a mistake in a question, state what you found and try answer as best you can.

Technical difficulties

- If you experience technical issues uploading the exam and can prove you did not leave the uploading to the very last minute, you should apply for Special Consideration.
- You must attach student declaration that explains the technical issue you experienced.

What is examinable?

- Everything from the lectures, the referenced sections of the textbooks, the tutorials, the quizzes, the assignments.
- In general if it happened during this unit, you are expected to know about it!
- Exceptions to this rule:
  - when explicitly labeled as non-examinable.
- Focus on the things we put most emphasis on, as seen in lectures, tutorials and assignments.

Exam technique

- Read all questions to see which ones you can answer quickly and accurately
- Plan how you will allocate time
- Start with easy problems and move to harder ones
- Write clearly and efficiently
  - No need for fancy style
- Figures take a lot of time to draw
  - Describing in text is faster
  - E.g., for an automaton, list the states and transitions, or give the transition table.

Pragmatic Advice

- Before the exam
  - Do the practice exams (on Ed)!
  - There is a Quiz (number 13) on GS that closes before the exam and tests your knowledge of exam logistics.
  - Practice submitting a file in Canvas
- On exam day
  - Be alone in your room to avoid distractions
  - Let housemates know when your exam is to avoid distractions
  - Bring water
  - Have clothing in layers
  - Breathe, Relax

Good Luck !