CS 245: Logic and Computation

Carmen Bruni

Fall 2018

Based on slides by Jonathan Buss, Lila Kari and Anna Lubiw with thanks to B. Bonakdarpour, D. Maftuleac, A. Gao, C. Roberts, R. Trefler, and P. Van Beek

Introduction

On your index card, write:

- Your actual name
- Your preferred name (and tips for pronunciation!)
- Something interesting about you
- Someone famous that shares the same name (first or last) or someone who shares the same birthday as you.

About Me

Carmen Bruni, DC 3119, cbruni@uwaterloo.ca

Websites:

- `https://learn.uwaterloo.ca/`
- `www.student.cs.uwaterloo.ca/~cs245`
- `https://piazza.com/`
- `https://cs.uwaterloo.ca/~cbruni/Math135Resources/index.php` (Review first 4 weeks!)
- `https://cs.uwaterloo.ca/~cbruni/CS245Resources/index.php` (My slides will be here)

Office Hours: Mondays 10:00-11:00am and Tuesdays 10:00-11:20am in my office.

Grading Scheme

- 25% Assignments (At most 3 doctor's notes will be accepted!)
- 30% Midterm
- 45% Final

Clickers

Please bring your clicker to class. It won't count towards grades but will help break up the class and help me to see where you are in your understanding.

What CS 245 is about...

First-order logic (predicate logic)

- a formal and precise language that we can use to model and to reason about real world scenarios.
- fundamental to many disciplines and has many applications in Computer Science.

and its applications:

- (Program verification) Prove whether a piece of code meets its formal specification.
- (Limitations of computers) Prove that some problems cannot be solved using computer algorithms.

A roadmap of this course

Four units: (1) Propositional Logic, (2) Predicate logic, (3) Program verification, (4) Undecidability.

Essential questions:

- (Modeling) How do we use first-order logic to model real world phenomenon?
- (Syntax) How do we formulate well-formed formulas in this language?
- (Semantics) How do we interpret the formulas written in this language?
- (Reasoning) How do we perform reasoning and inference using this language?
- (Program verification) How do we make sure that a piece of code does what it's supposed to do?
- (Undecidability) Are there problems that cannot be solved by computer algorithms?

The schedule of lectures is on the course website.

Logic from two perspectives

You will learn about first-order logic from two perspectives: a logician and a practitioner.

A practitioner cares about:

- How can I use logic to model specific things?
- How do I determine whether two formulas are logically equivalent?
- How do I prove that a conclusion logically follows a set of premises?

A logician cares about:

- Does every well-formed formula have a unique interpretation?
- What is the smallest set of connectives that is sufficient to formulate all possible formulas?
- Is everything that I can prove true? Can I prove everything that is true?

Is this course only about logic?

Not quite! We also want you to develop these skills:

- Thinking and communicating precisely and clearly
- Problem solving
- Creative thinking
- Critical thinking
- ...

How should I study and collaborate?

- Focus on the learning goals.
- Read with questions in mind. After reading, summarize. After studying, test yourself. (Study strategies)
- Spend time struggling before looking at the solutions/asking for help. (Dealing with failures)
- Academic integrity

Learning goals

By the end of the lecture, you should be able to

- Give a high-level description of logic.
- Give examples of applications of logic in computer science.
- Define propositions.
- Classify English sentences into propositions and non-propositions.
- Give multiple translations of English sentences with ambiguity.
- Translate English sentences with no ambiguity into compound propositions.

What is logic?
Why do we study logic in computer science?

What is logic?

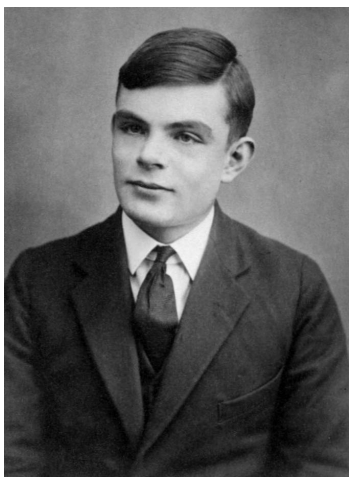
Logic is the science of reasoning

- Etymology: *Logykos* (Greek) - pertaining to reasoning
- Logic = The science of reasoning, proof, thinking, or inference
- Logic = The systematic study of the form of arguments

Why study logic?

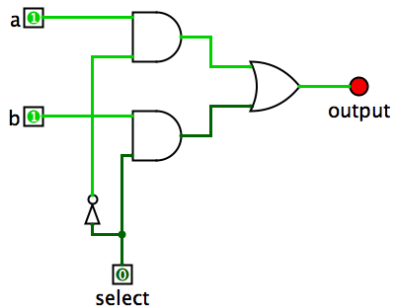
- Logic is fun! Think about logical puzzles.
- Logic improves one's ability to think analytically and to communicate precisely.
- Logic has many applications in Computer Science.

"I expect that digital computing machines will eventually stimulate a considerable interest in symbolic logic . . . . The language in which one communicates with these machines . . . forms a sort of symbolic logic." (Alan Turing, 1947)

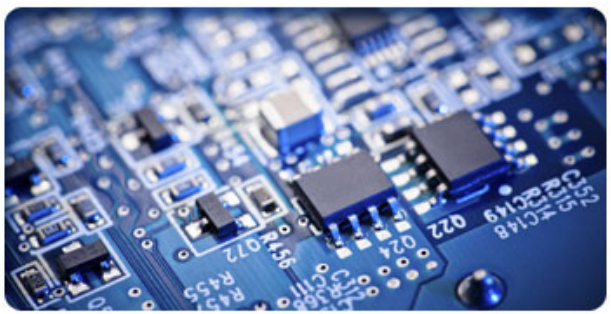


Applications of logic to computer science

- Circuit design



- An electronic computer consists of digital circuits, formed out of logic gates.



Applications of logic to computer science

- Logic programming
- PROLOG (PROgramming with LOGic) has roots in first-order logic.
- Theorem proving, expert systems, and natural language processing
- On February 14-16, 2011, IBM Watson won the Jeopardy Man vs. Machine Challenge by defeating two former grand champions, Ken Jennings and Brad Rutter. Watson relied on natural language processing technology to analyze the vast amount of unstructured text (encyclopedias, dictionaries, news articles, etc.). Prolog was the ideal language for this due to its simplicity and expressiveness.

Applications of logic to computer science

- Expert systems (knowledge base + inference engine)
- Medical diagnosis systems: The user describes their symptoms to the computer as they would to a doctor and the computer returns a medical diagnosis.
- MYCIN: acceptable therapy in about 69% of cases, which was better than the performance of infectious disease experts who were judged using the same criteria.
- MISTRAL: an expert system for the monitoring of dam safety

Applications of logic to computer science

- Databases
- Core of modern database systems (e.g. SQL) uses first-order logic
- Efficient query evaluation based on relational algebra

Applications of logic to computer science

- Formal verification
- Designs are increasingly complex, and errors are costly and dangerous.
- Hardware example: Intel's Pentium FDIV bug (1994) cost them half a billion dollars. Intel is now heavily invested in formal verification.
- Software examples: Death of cancer patients who received severe overdoses of radiation due to a software error in the Therac 25 (1985 – 1987)

An analysis of arguments

- Logic is the analysis of arguments.
- An argument is a set of statements, namely one or several premises and a conclusion, usually connected by "therefore".
- An argument here isn't a quarrel or fight. Rather it is the verbal expression of a reasoning process.
- Consider the following argument:

  Every pig can fly.
  Aristotle is a pig.

  Aristotle can fly.

The argument is valid. It is impossible for all the premises to be true but the conclusion to be false.

The form of arguments

- Logic studies forms of reasoning.
- The content doesn't matter. We want the reasoning process to apply to any subject - water purity, mathematics, cooking, nuclear physics, ethics, or whatever.
- Consider the following two arguments:

Every pig can fly.
Aristotle is a pig.

Aristotle can fly.

All humans are mortal.
Socrates is human.

Socrates is mortal.

A logician considers these two arguments to be the same because they have the same form.

Propositional logic: An informal introduction

Propositions

A proposition is a declarative sentence that is either true or false.

A proposition can never be both true and false.

Key ideas about propositions

- Many English sentences are not propositions, for example, commands, questions, paradoxes, and non-sensical sentences. We cannot use propositional logic to model these sentences.
- To be a proposition, the sentence must have sufficient information for us to determine whether it's true or false. For example, it cannot have an unspecified variable x.
- A sentence can be a proposition even if nobody knows whether it's true or false, e.g. the twin prime conjecture.

Examples of propositions

- The sum of 3 and 5 is 8.
- The sum of 3 and 5 is 35.
- $-1 \geq 5$.
- Program p terminates, on any input. (p is a known program.)
- If the input to program p is a non-negative integer x, then p outputs the square of x. (p is a known program, and x is a known integer.)
- Every even number greater than 2 is the sum of two prime numbers.

Examples of non-propositions

- Question: Where shall we go to eat?
- Command: Please pass the salt.
- Sentence fragment: The dogs in the park
- Non-sensical: Green ideas sleep furiously.
- Paradox: This sentence is false.

We cannot use propositional logic to model and reason about these sentences.

Modeling relationships with logical connectives

Connective	***Meaning***	***Some alternative expressions***
Negation $(\neg p)$	not p	p does not hold; p is false; it is not the case that p
Conjunction $(p \wedge q)$	p and q	p but q; not only p but q; p while q; p despite q; p yet q; p although q
Disjunction $(p \vee q)$	p or q	p or q or both; p and/or q; p unless q
Conditional $(p \rightarrow q)$	if p then q	p implies q; q if p; p only if q; q when p; p is sufficient for q; q is necessary for p
Biconditional $(p \leftrightarrow q)$	p if and only if q (p iff q)	p is equivalent to q; p exactly if q; p is necessary and sufficient for q

Translating from English to Propositional Logic

Translate the following sentences to propositional logic formulas.

1. She is clever but not hard working.
2. I will eat an apple or an orange but not both.
3. If he does not study hard, then he will fail.
4. If it is sunny tomorrow, then I will play golf, provided that I do not feel stressed.
5. He will not fail only if he studies hard.

English can be ambiguous.

Give multiple translations of the following sentences into propositional logic. Are these translations logically equivalent?

1. Pigs can fly and the grass is red or the sky is blue.

2. He will fail unless he studies hard.