CSC165 Mathematical Expression and Reasoning for Computer Science
Winter 2015

Department of Computer Science
University of Toronto

Lisa Jing Yan

General Info

- **Dr. Lisa Jing Yan** (L0201: MWF 2-3 pm)

Email: lyan at cs.toronto.edu

Office hours: Wednesday 3-5 in BA4261.

- **Bahar Aameri** (L0101: MWF 11-noon)

Email: bahar at cs.toronto.edu

Office hours: Friday 12:30-1:30 & 3:30-5 in BA4261

Lisa: Monday afternoon, Wednesday morning & afternoon sessions.

Bahar: Monday morning, Friday morning & afternoon sessions.

Course Info

- Course web page:

  www.cdf.toronto.edu/~csc165h/winter

- Course info sheet (important):

  www.cdf.toronto.edu/~csc165h/winter/165infosheet.pdf

- Course Notes:

  www.cdf.toronto.edu/~csc165h/winter/165notes.pdf

What’s 165 about?

- Expressing in mathematical terms: precisely
- Learn:
  - Describe a problem using Logical notations
  - Techniques of how to prove (or disprove) a statement
  - Analyzing algorithm efficiency and complexity

Example

- How could we translate this sentence into a logical expression?

"You can access the Internet from campus only if you are a computer science major or your are not a freshman."

$$\mathbf{A \rightarrow (C \vee \neg F)}$$

“Another” math problem...

Prove that:

$$(A \Rightarrow B) \Rightarrow C \;\Leftrightarrow\; (A \land \neg B) \lor C$$

Example

Let A, B, and C be three statements.

The statement "A being true implying B being true implies C being true" is true if and only if either A is true and B is false or C is true.

Is this true?

Course Evaluation

Item	Due Date	Weight (%)
A1	Jan. 30, 11:59pm, Friday	10
A2	Mar. 06, 11:59pm, Friday	10
A3	Apr. 02, 11:59pm, Thursday	10
Quizzes	W 2,3,4,6,8,9,11,12	10
Test 1	Feb. 03 / Feb. 05	10
Test 2	Mar. 10 / Mar. 12	10
Final Exam	Apr. TBD	40

Must obtain 40% on the final exam to pass the course!

Assignments

- Assignments may be submitted in groups of up to two students.
  - You may choose your group-mate from students in the other section.
  - You may change your group-mate for each assignment.
- Each group must submit a single PDF file on MarkUs:
  - Submissions must be typed (LATEX is strongly recommended)
- Recommendation: work on all questions individually! Then discuss your answers with your group-mate and prepare the final submission together.

Policy - 1

- Late Submission: One time 24-hour grace period with no penalty (per person).
- Remark requests for assignments must be submitted through MarkUs within one week of receiving the assignment back.

Policy - 2

- Discussion board (Piazza):
  - http://piazza.com/utoronto.ca/winter2015/csc165
  - Use your UTOR email (mail.utoronto.ca) to sign up.
  - The discussion board will be monitored by TAs and instructors.
  - Don't discuss assignment solutions until 24h after the due dates.
- Use email only for personal issues such as requesting special considerations.

How to do well in 165

- Check the course web page regularly
- Read course information sheet
- Spend 8-10 hours/week on the course:
  - 3 hours in lectures
  - 2 hours in tutorial
  - 3-5 hours reviewing slides and course notes, working on assignments.
- Need more practice?

  http://www.cdf.toronto.edu/~csc165h/winter/165PraQueSoln.pdf

By the end of the course

- With and through programs / with developers:
  - communicating precisely:
    - Knowing and saying what you mean.
    - Understanding what others say and mean.
  - Analyzing arguments, programs
- Foundation course for your university career whenever you need to read and understand technical material, course textbooks, assignment specifications, etc.

Who should take CSC165?

- If you do:
  - memorize math
  - have trouble explaining what you're doing in technical work
  - have trouble understanding word problems
- If you don’t:
  - like reading math books to learn new math
  - Enjoy talking about abstract x and y as much as concrete examples are given for x and y

Why CS needs math

- Artificial Intelligence:
  - Mathematical Logic, Set Theory, Probability
- Cryptography:
  - Number Theory, Field Theory
- Algorithm:
  - Combinatorics, Set Theory
- Programming Languages:
  - Mathematical Logic, Set Theory
- Databases:
  - Mathematical Logic, Set Theory
- Networking:
  - Graph Theory, Statistics

Chapter 1 Introduction

Ambiguity

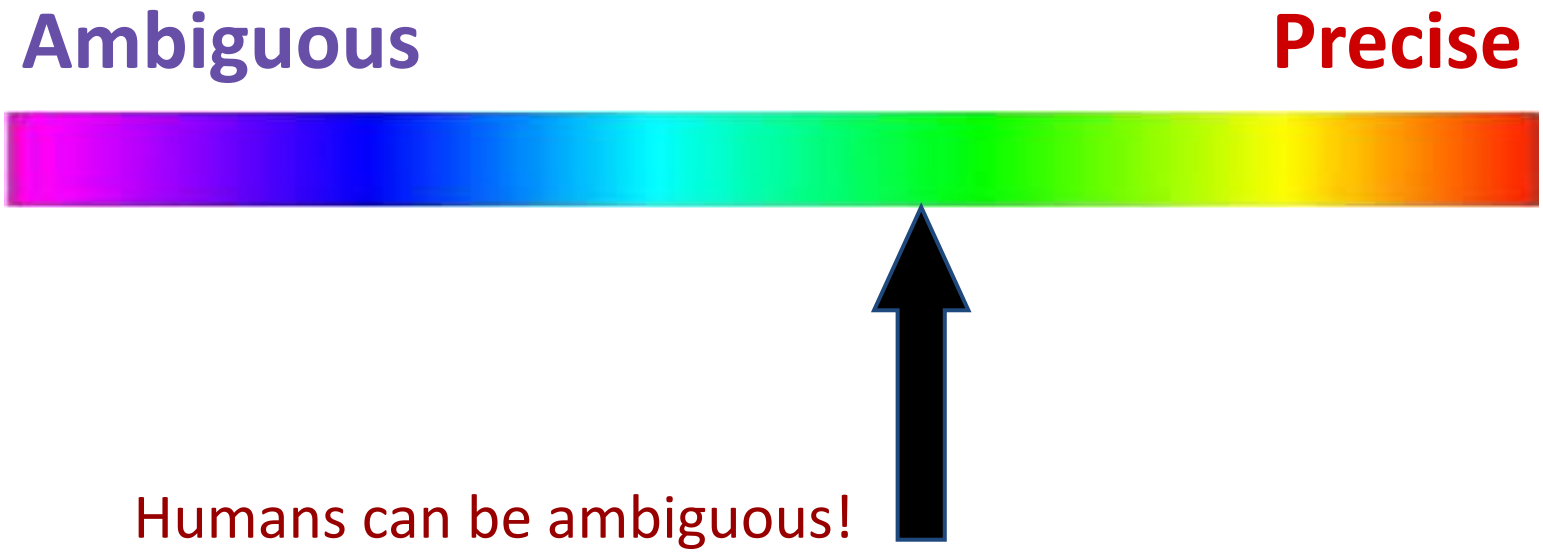
A woman asks her husband to peel half the potatoes and put them on to boil, and then leaves the house.

Ambiguity

A woman asks her husband to peel half the potatoes and put them on to boil, and then leaves the house.

Here's what she finds when she is back!

Natural language precise?



Ambiguity

When you use a natural language (English, Chinese) you can make it as precise or ambiguous as you need.

- For some purposes (jokes, gossip) rich ambiguity is essential.
- For other purposes (getting instructions on heart surgery) precision is essential.

Ambiguity

When you use a natural language (English, Chinese) you can make it as precise or ambiguous as you need.

- For some purposes (jokes, gossip) rich ambiguity is essential.
- For other purposes (getting instructions on heart surgery) precision is essential.

We're all equipped to work in both modes. Work out the double meanings of these headlines:

- Two sisters reunite after 18 years at checkout counter
- Iraqi head seeks arms

Precision

How to be precise?

➔ Restrict the meanings of words.

Being a profession means learning the vocabulary (words with restricted meanings) in the filed.

➔ i.e., for mathematicians:

- continuous, open, closed
- group, ring, field
- for all, for each: $\forall$
- there is (exists): $\exists$

Balance

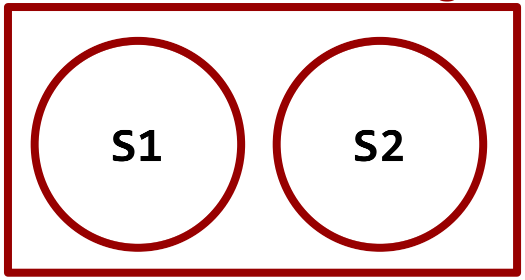
- computers are precise: in identical environments they execute identical instructions identically
- humans are as precise as necessary, and different human audiences require different levels of precision
- difficult job is finding the right level of precision:
  - too much precision introduces unbearable tedium;
  - too little introduces unavoidable ambiguity.

Computer language => human language

S1 and S2 are two sets

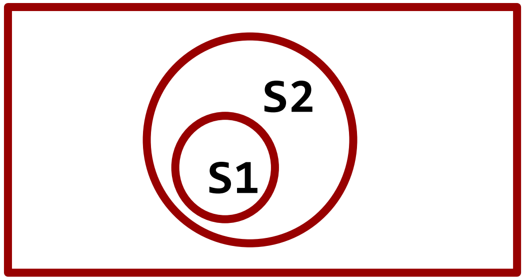
```
def q1(S1, S2):
    '''Return whether S1 and S2 have NO
       intersection
    '''
    for x in S1:
        if x in S2 : return False
    return True
```

Venn Diagram



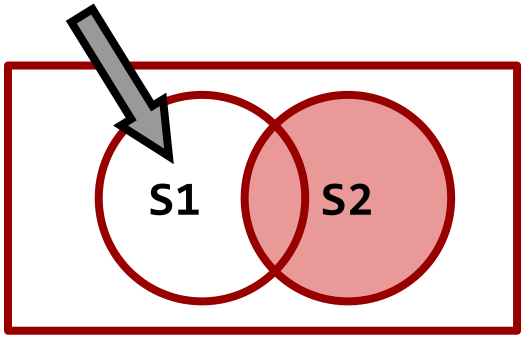
Computer language => human language

```
def q2(S1, S2):
    '''Return whether all elements in S1
       are in S2
    '''
    for x in S1:
        if x not in S2 : return False
    return True
```



Computer language => human language

```
def q3(S1, S2):
    '''Return whether there exists an element
       in S1 which is not in S2
    '''
    for x in S1:
        if x not in S2 : return True
    return False
```



Verify

Check your comments for q1-q3 in various ways (checking isn't proving, but it increases our confidence or reveals flaws).

- try out particular values for S1 and S2; see whether the results are consistent with your comments. Check “corner” values, e.g. when one or both lists are empty.
- Draw a Venn diagram.

Problem Solving: Polya's approach

4 Steps:

- Understand the problem
- Plan a solution
- Carry out your plan
- Review any solution you achieve

1.Understand the problem

- Know what is required;
- Know what is given:
  - usually the given information is necessary for solving the problem.
  - BUT, sometimes completing a proof needs information that are not given but have been proved or are well-known to be true.
- Re-state the problem in your own words;
- Might help to draw some diagrams.

2. Plan solution(s)

- If you have seen something similar, you may be able to use its result or its method.
- Work backwards: assume you have solved the problem and deduce the next-to-last step.
- Try solving simpler versions of the problem.

3. Carry out your plan

See whether your plan for a solution leads somewhere:

- You may need to repeat (parts of) the earlier steps.
- If you are stuck, identify *exactly* what information/assumptions you require that are missing and find a way to achieve them.

4. Review your solution

Look back on any pieces of puzzle you solve:

- Verify that your solution is correct and convincing.
- Extend the solution to new problems.

Example: Streetcar Drama

On a streetcar, you overhear the following conversation:

A: Haven't seen you in a long time! How old are your three kids now?

B: The product of their ages (in years) is 36.

A: That doesn't really answer my question.

B: Well, the sum of their ages is -- [fire engine goes by]

A: Hmm… Still, that doesn't tell me how old they are.

B: Well, the eldest plays piano.

A: Okay, I see, so their ages are -- [you have to get off the streetcar]

Restate the problem

→ The product of their ages is 36:

$x \cdot y \cdot z = 36$ **(Given)**

→ Knowing the sum of their ages does not lead to a unique solution:

$x+y+z$ is not unique among all possible combinations **(Deduced from the given information)**

→ There is an eldest one:

$x > y$ and $x > z$ **(well-known to be true)**

So, what is the value of x, y and z?

A plan for solution

- $x \cdot y \cdot z = 36$,
- $x + y + z$ is not unique among all possible combinations,
- $x > y$ and $x > z$.

Possible Answers:

36; 1; 1
18; 2; 1
12; 3; 1
9; 4; 1
9; 2; 2
6; 6; 1
6; 3; 2
4; 3; 3

A plan for solution

- $x \cdot y \cdot z = 36$,
- $x + y + z$ is not unique among all possible combinations,
- $x > y$ and $x > z$.

Possible Answers:

36; 1; 1 where $36 + 1 + 1 = 38$
18; 2; 1 where $18 + 2 + 1 = 21$
12; 3; 1 where $12 + 3 + 1 = 16$
9; 4; 1 where $9 + 4 + 1 = 14$
9; 2; 2 where $9 + 2 + 2 = 13$
6; 6; 1 where $6 + 6 + 1 = 13$
6; 3; 2 where $6 + 3 + 2 = 11$
4; 3; 3 where $4 + 3 + 3 = 10$

A plan for solution

- $x \cdot y \cdot z = 36$,
- $x + y + z$ is not unique among all possible combinations,
- $x > y$ and $x > z$.

Possible Answers:

36; 1; 1 where $36 + 1 + 1 = 38$
18; 2; 1 where $18 + 2 + 1 = 21$
12; 3; 1 where $12 + 3 + 1 = 16$
9; 4; 1 where $9 + 4 + 1 = 14$
9; 2; 2 where $9 + 2 + 2 = 13$
6; 6; 1 where $6 + 6 + 1 = 13$
6; 3; 2 where $6 + 3 + 2 = 11$
4; 3; 3 where $4 + 3 + 3 = 10$

A plan for solution

- $x \cdot y \cdot z = 36$,
- $x + y + z$ is not unique among all possible combinations,
- $x > y$ and $x > z$.

The Answer of the three kids' ages is:

9; 2; 2

Exercises

Court Case

At a murder trial, four witnesses give the following testimony.

Alice: If either Bob or Carol is innocent, then so am I.
Bob: Alice is guilty, and either Carol or Dan is guilty.
Carol: If Bob is innocent, then Dan is guilty.
Dan: If Bob is guilty, then Carol is innocent; however, Bob is innocent.

Questions

(a) Is the testimony consistent, *i.e.*, is it possible that everyone is telling the truth?

(b) If every innocent (not guilty) person tells the truth and every guilty person lies, determine (if possible) who is guilty and who is innocent.

Answer (a)

(a) Is the testimony consistent, *i.e.*, is it possible that everyone is telling the truth?

Solution:

- First, we define some abbreviations:

  A: "Alice is innocent"
  B: "Bob is innocent"
  C: "Carol is innocent"
  D: "Dan is innocent"

- Then, we can rewrite the statements symbolically (with the understanding that "guilty = ¬ innocent"):

  Alice: $(B \vee C) \rightarrow A$
  Bob: $\neg A \wedge (\neg C \vee \neg D)$
  Carol: $B \rightarrow \neg D$
  Dan: $(\neg B \rightarrow C) \wedge B$

Answer (a)

Now, let us assume that everyone's testimony is true. We will see that this leads to a contradiction, which means that someone must be lying.

Bob's testimony means that Alice is guilty. Then, the contrapositive of Alice's testimony means that both Bob and Carol are guilty. But this contradicts Dan's testimony that Bob is innocent.

Answer (b)

(b) If every innocent (not guilty) person tells the truth and every guilty person lies, determine (if possible) who is guilty and who is innocent.

Solution:

Using the same notation as above, there are many ways of determining who is innocent and who is guilty, including trial-and-error. One possibility is that Alice and Carol are guilty while Bob and Dan are innocent. Then, Alice's statement is false (because B is true but A is false), Bob's statement is true (because A is false and C is false), Carol's statement is false (because B is true and D is true), and Dan's statement is true (because B is true), as required.