CSC165H, Mathematical expression and reasoning for computer science week 1

19th May 2005

Gary Baumgartner and Danny Heap
heap@cs.toronto.edu
SF4306A
416-978-5899
http://www.cs.toronto.edu/~heap/165/S2005/index.shtml

What's csc165 about?

In addition to hacking, computer scientists have to be able to understand program specifications, APIs, and their workmate's code. They also have to be able to write clear, concise documentation for others.

In our course you'll work on:

Expressing yourself clearly (using English and mathematical expression).

Understanding technical documents and logical expressions.

Deriving conclusions from logical arguments, several proof techniques.

Analyzing program efficiency.

Binary notation, why it's used and how to manipulate it. Toward the end of the course you'll learn about how real numbers are represented in computers with floating-point data types.

In this course we care about communicating precisely:

Knowing and saying what you mean

Understanding what others say and mean

We want this course to help you during your university career whenever you need to read and understand technical material (course textbooks, assignment specifications).

Who needs csc165?

You need this course if you do:

memorize math

have trouble explaining what you are doing in a mathematical or technical question

HAVE trouble understanding word problems

YOU need this course if you DON'T:

- LIKE reading math textbooks to learn new math
- ENJOY talking about abstract x and y just as much as when concrete examples are given for x and y.
- HAVE a credit for CSC238 in your academic history, or intend to take CSC240.

Why does CS need mathematical expressions and reasoning?

We all enjoy hacking, that is designing and implementing interesting algorithms on computers. Perhaps not all of us associate this with the sort of abstract thinking and manipulation of symbols associated with mathematics. However there is a useful two-way contamination between mathematics and computer science. Here are some examples of branches of mathematics that taint particular branches of Computer Science:

COMPUTER GRAPHICS use multi-variable calculus, projective geometry, linear algebra, physics-based modelling

NUMERICAL ANALYSIS uses multivariable calculus and linear algebra

CRYPTOGRAPHY uses number theory, field theory

NETWORKING uses graph theory, statistics

ALGORITHMS use combinatorics, probability, set theory

DATABASES use set theory, logic

AI uses set theory, probability, logic

PROGRAMMING LANGUAGES use set theory, logic

How to do well in csc165

CHECK the course web page frequently.

UNDERSTAND the course information sheet. This is the document that we are committed to live by in this course.

GET in the habit of asking questions and contributing to the answers.

SPEND time on this course. The model that we instructors assume is that you work 10 hours per week (2 in lecture, 1 in tutorial, 7 reviewing and working on assignments) on this course. Any material that's new to you will require time for you to really acquire it and use it on other courses.

DON'T plagiarize. Passing off someone else's work as your own is an academic offense. Always give generous and complete credit when you consult other sources (books, web pages, other students).

Human versus technical communication

Natural languages (English, Chinese, Arabic, for example) are rich and full of potential ambiguity. In many cases humans speaking these languages share a lot of history, context, and assumptions that remove or reduce the ambiguity. If we don't share (or choose to momentarily forget) the history and context, there is a rich source of humour in double-meanings created by natural languages, for example consider these headlines (http://www.departments.bucknell.edu/linguistics/semhead.htm)

Prostitutes appeal to Pope

Iraqi head seeks arms

Police begin campaign to run down jay-walkers

Death may cause loneliness, feelings of isolation

Two sisters reunite after 18 years at checkout counter.[1]

Computers are notorious for lacking a sense of humour, and we communicate with them using extremely constrained languages, programming languages. In programming languages, expressions aren't expected to be ambiguous.

Human technical communication about computing must be similarly constrained. We have to assume less common history and context is shared with the other humans participating in technical communication, and misplaced assumptions can result in catastrophe. We aim for increased PRECISION, that is a smaller tolerance for ambiguity. We will use MATHEMATICAL LOGIC, a precise language, as a form of communication in this course.

Mathematicians share a common dialect to talk precisely about precise concepts in their work (e.g. "differentiable functions are continuous"). Often ordinary words ("continuous") are used with restricted, or special, meanings. The same word may have different technical meaning in different mathematical contexts, for example GROUP may mean one thing in group theory, another in combinatorial design theory.

Since technical language is used between human beings, some degree of ambiguity is tolerated, and probably necessary. For example, an audience of Java programmers would not object to the subtle shift in the meaning used for "a" in the following fragments:[2]

```
/** Sorts a in ascending order */
public void sort(int[] a) ...
```

versus

```
// sets a to 1
a = 1;
```

Since another human is reading our comments, this potential for double meaning is benign. A computer reading our comments would, of course, be unforgiving. However, Java programmers have to assume familiarity with programming from their audience, to avoiding driving others crazy by writing long comments (and being driven crazy by long comments written by others).

In this course we can't assume the necessary context to always conclude that you know what you're saying, so you'll have to demonstrate it explicitly. On the other hand, you will learn to understand somewhat imprecise statements that can be made precise from the context.

Universal quantification

Consider the following table that associates employees with properties:

Employee	Gender	Salary
Al	male	60,000
Betty	female	500
Carlos	male	40,000
Doug	male	30,000
Ellen	female	50,000
Flo	female	20,000

Claims about individual objects can be evaluated immediately (Al is male, Flo makes 20,000). However the tabular form allows claims about the entire database to be considered. Consider:

> Every employee makes less than 70,000.

Is this claim true? So long as we restrict our universe to the six employees, the answer is[3]

When a claim is made about all the objects (in this context, humans are objects!) being considered (i.e., in our "universe"), this is called UNIVERSAL QUANTIFICATION. The meaning is that we make explicit the logical quantity (quantify) every member of a class or universe. English being the slippery object it is allows several ways to say the same thing:

> Each employee makes less than 70,000.
>
> All employees make less than 70,000.
>
> Employees make less than 70,000.

Our universe (AKA "domain") is the given set of six employees. When we say every, we mean every. This is not always true in English, for example "Every day I have homework," probably doesn't consider the days preceding your birth or after your death. Now consider

> Each employee makes at least 10,000.

Is this claim true? How do you know?[4] A single counter-example is sufficient to refute a universally-quantified claim. What about the following claim:

> All female employees make less than 55,000.

Is this claim true? Restrict the domain and check each case. [5] What about

> Every employee that earns less than 55,000 is female?[6]

How about this claim:

> Every male employee makes less than 55,000.

It worked for females.[7] Notice a pattern. To disprove a universally-quantified statement you need just one counter-example. To prove one you need to consider every element in a domain. A universally-quantified statement of the form

> Every P is a Q

(a universally-quantified implication, as we shall see) needs a single counter-example to disprove, and consideration of an entire domain to prove.

Properties and sets

Let's look at that table again.

Employee	Gender	Salary
Al	male	60,000
Betty	female	500
Carlos	male	40,000
Doug	male	30,000
Ellen	female	50,000
Flo	female	20,000

Saying that Al is male is equivalent to saying Al belongs to the set of males. Symbolically we might write $Al \in M$ or $M(Al)$. It's useful and natural to interchange the ideas of properties and sets. If we denote the set of employees as E, the set of female employees as F, the set of male employees as M, and the set of employees who earn less than 55,000 as L, then we have a notation for concisely (and precisely) evaluating claims such as $M(Flo)$,[8] or $L(Carlos)$.[9] So far the notation doesn't seem to have achieved much, but how about

> Everything in F is also in L. In other notation, $F \subseteq L$.

So our universally-quantified claim that all females make less than 55,000 turns in to a claim about subsets. We already have some intuition about subsets, so lets put it to work by drawing a Venn diagram (see implication.ps on our web page).

Make sure you are solid on the meaning of "subset." Is a set always a subset of itself?[10] Is the empty set (the set with no elements) a subset of any set?[11] Now consider the claim

> Everything in M is also in L. This is equivalent to $M \subseteq L$.

If we try to verify this claim using the Venn diagram approach, the circle representing M stubbornly resists being included in L, due to the counter-example Al.

Implications

Consider a claim of the form

> IF an employee is male, THEN he makes less than 55,000.

This is called an IMPLICATION. It says that being male IMPLIES making less than 55,000.[12] This is universal quantification in disguise. Since logical implication borrows the English word "if," we need to reject some of the common English uses of "if" that we don't mean. In logic "if...then" tells you nothing about causality. "If it rained yesterday, then the sun rose today," is a true implication, but the (possible) rain didn't cause the (certain) rising. When my mother told me "if you eat your vegetables, then you can have dessert," she also meant "otherwise you'll get no dessert." Although in ordinary English we sometimes use "if ... then" to mean "if and only if ... then," in logic we use the more constrained meaning. We want "If P then Q" to mean "Every P is a Q."

We can also model this using a programming language. A claim such as $A(x)$ can be modelled by a method "of" that returns true exactly when $x \in A$. The implication "$A(x)$ implies $B(x)$" can be modelled by a method that returns true exactly when either $A(x)$ and $B(x)$ are both true, or $A(x)$ is false. Universal quantification can be modelled by iterating over the elements of a domain. Look at the .java files on the website, and try experimenting with them in a Java runtime environment.

Notes

[1]The word "appeal" in the first headline has two meanings, so one interpretation is that the Pope is fond of prostitutes, and another is that prostitutes have asked the Pope for something. The words "head" and "arms" each have two meanings, so one interpretation is that the body part above some Iraqi person's shoulders is looking for the appendages below their shoulders, and another is that the most senior Iraqi is looking for weapons. The phrase "run down" can mean either hitting with a car or looking for. In the fourth headline, it's not clear to whom death causes loneliness and isolation: the dead person or their survivors. In the fifth headline it's not clear whether the checkout line was moving REALLY slowly, or that the checkout counter was just the location of their reunion.

[2]In the first fragment "a" means "the object referred to by the value in a." In the second "a" means "the variable a."

[3]Yes, by verifying the claim for each employee.

[4]Betty makes 5,000, which is well-known to be less than 10,000.

[5]Restrict to females, and each one make less than 55,000.

[6]False. Doug and Carlos are counterexamples.

[7]But it is false for males. Al is a counter-example.

[8]False, check the table.

[9]True, check the table.

[10]Yes, since it includes only elements of itself. Don't confuse SUBSET with PROPER SUBSET.

[11]Yes, indeed it is a subset of every set. The reason is that it contains no elements that aren't in any other set.

[12]An untrue implication in the universe we're considering, due to the counter-example Al.