CSCI 2824 - CU Boulder, 2019 Summer

Lecture 10: Set theory

18 June 2019

Lecturer: J. Marcus Hughes

Content is borrowed from Susanna Epp's Discrete Mathematics with Applications,
Rosens's Discrete Mathematics and its Applications,
Bettina and Thomas Richmond's A Discrete Transition to Advanced Mathematics, and Andrew Altomare's notes.

We defined a ton of stuff yesterday. Let's quickly review it. Then, we will do some proofs!

1 Subset relationships

These are some simple ideas that are quick to prove.

Exercise
Prove the inclusion of intersection: that for all sets A and B, we have $A \cap B \subset A$ and $A \cap B \subset B$.

Exercise
PRove the inclusion in union: For all sets A and B we have $A \subset A \cup B$ and $B \subset A \cup B$.

Exercise
Prove the transitive property of subsets. For all sets A, B, and C, if $A \subset B$ and $B \subset C$, then $A \subset C$.

2 Set Identities

Theorem 2.1. *Let all sets be referred to be subset of the universal set U. Then, the following hold:*

- *Commutativity: For all sets A and B, we have $A \cup B = B \cup A$ and $A \cap B = B \cap A$.*
- *Associativity: For all sets A, B, and C, we have $(A \cup B) \cup C = A \cup (B \cup C)$ and $(A \cap B) \cap C = A \cap (B \cap C)$.*
- *Distribitivity: For all sets A, B, and C, we have $A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$ and $A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$.*
- *Identity: For all sets A, we have $A \cup \emptyset = A$ and $A \cap U = A$.*

- *Complementarity: For every set A, we have $A \cup A^C = U$ and $A \cap A^C = \emptyset$.*
- *Double complementarity: For every set A, $(A^C)^C = A$.*
- *Idempotency: For every set A, $A \cup A = A$ and $A \cap A = A$.*
- *Universal bound: For every set A, $A \cup U = U$ and $A \cap \emptyset = \emptyset$.*
- *De Morgan's: For all sets A and B: $(A \cup B)^C = A^C \cap B^C$ and $(A \cap B)^C = A^C \cup B^C$.*
- *Absorption: For all sets A and B: $A \cup (A \cap B) = A$ and $A \cap (A \cup B) = A$.*
- *Complements of U and $\emptyset$: $U^C = \emptyset$ and $\emptyset^C = U$*
- *Set diference: $A \setminus B = A \cap B^C$*

I'm not going to provide all the proofs for these here because many of them are obvious. Let's talk aobut De Morgan's though. We want to prove for all sets A and B that $(A \cup B)^C = A^C \cap B^C$. We can get intuition by using a Venn Diagram.

Exercise
Prove that for all sets A and B that $(A \cup B)^C = A^C \cap B^C$

3 In-class exercises

Exercise
Prove: For all sets A, B, and C if $A \subset B$ and $B \subset C^C$, then $A \cap B = \emptyset$.

Exercise
For every integer $n \geq 0$, if a set X ahs n elements then $\mathcal{P}(X)$ has 2^n elements.

Exercise
If a set X has infinitely many elements, such as $\mathbb{N}$ or $\mathbb{R}$, then what can we say about the size of $\mathcal{P}(X)$? Is $|X| = |\mathcal{P}(X)|$? Or $|X| \leq |\mathcal{P}(X)|$? Or $|X| < |\mathcal{P}(X)|$? Which seems better? How do we prove it?

4 The Halting Problem

Let's think about a barber named Joe. Joe is a cool guy. It's a really small town so Joe knows everyone. He shaves all the men, and only the men, who do not shave themselves. But... does Joe shave himself?

This is a weird paradox related to Russell's paradox from yesterday. The answer is both yes and no which makes for a weird issue in logic and math. If the barber shaves himself, then he is a member of the class of men who shave themselves. But no member of that class is shaved by the barber, so he can't shave himself. But... if he does not shave himself, then he belongs to the class of men who do not shave themself. But, we know every many who doesn't shave themself is shaved by the barber, so he does shave himself! Oh no! Paradox all over!

I bring this up again because even though it was resolved by a careful definition in set theory, it led to lots of interesting mathematically ideas and questions. Kurt Gödel got interested in these foundational issues in math, and he showed in 1931, that you could not prove, in a mathematically rigorous way, that math was free of contradictions! Oh no! So we can never prove that our math works. It's a scary time. It's led to a lot of interesting realizations and thoughts.

One question that is related and pertinent to us is, "Can you write a computer program that will look at another program taking some specific input data and tell you if that program using that input data will halt in a finite number of sets?" Disconcertingly, the answer is no and is a very famous result that is the foundation of many ideas in computability theory. This idea will come up again and again in your studies as a computer science student. Let's try to prove this theorem. It's a mind bender, so let's go through it together.

Theorem 4.1. *There is no computer algorithm that will accept any algorithm X and a data set D as input and then will output "halts" or "loops forever" to indicate whether or not X terminates in a finite number of steps when X is run with data set D.*

Proof. We will proceed by proof by contradiction. Let's assume there is an algorithm, `CheckHalt`, such that if algorithm X adn a data set D are input then `CheckHalt(X, D)` prints "halts" if X terminates in a finite number of sets when run with data set D and "loops forever" if X does not terminate in a finite number of steps when run with data set D.

The actual program X, the sequence of characters we write down to define X, can be thought of as a form of data itself. So, we can run `CheckHalt(X, X)`. Let's define a new algorithm `Test` as follows: `Test(X)` loops forever if `CheckHalt(X, X)` prints "halts" and stops if `CheckHalt(X, X)` prints "loops forever."

Now, let's be especially pesky and run `Test` with `Test` as input! If `Test(Test)` terminates after a finite number of steps, then the value of `CheckHalt(Test, Test)` is "halts" so `Test(Test)` actually loops forever, a contradiction of it terminating after a finite number of steps.

But, if `Test(Test)` does not terminate after a finite number of steps, then `CheckHalt(Test, Test)` prints "loops forever" so `Test(Test)` actually terminates!

This means in any scenario `Test(Test)` both halts and loops forever, a horrendous contradiction. But `Test` must exist if `CheckHalt exists`. Therefore, supposition is false and `CheckHalt` cannot exist. □

Unfortunately, the axioms in set theory that we use to avoid Russell's paradox don't deal adequately with all the recursive possibilities of computer algorithms. We try to resolve these in research, one such avenue is called hypersets.

5 Boolean algebra

Definition: *Boolean algebra*
A Boolean algebra is a set B together with two operations, usually denoted + and $\times$, such that for all a and b both $a + b$ and $a \times b$ are in B and the following axioms are assumed to hold:

- Commutative laws: $\forall a, b \in B\, a + b = b + a$ and $a \times b = b \times a$
- Associative law: $\forall a, b, c \in B\, (a + b) + c = a + (b + c)$ and $(a \times b) \times c = a \times (b \times c)$
- Distributive laws: $\forall a, b, c \in B\, a + (b \times c) = (a + b) \times (a + c)$ and $a \times (b + c) = (a \times b) + (a \times c)$. item Identity laws: There exist distinct elements 0 and 1 in B such that for each a in B we have $a + 0 = a$ and $a \times 1 = a$
- Complement laws: For each a in B, there exists an element in B denoted $\overline{a}$ called the negation or complement of a such that $a + \overline{a} = 1$ and $a \times \overline{a} = 0$.