CPSC-354 Report

Zack Klopukh
Chapman University

September 9, 2024

Abstract

(Delete and Replace:) You can safely delete and replace the explanations in this file as they will remain available on the course website. For example, you should replace this abstract with your own. The abstract should be a short summary of the report. It should be written in a way that makes it possible to understand the purpose of the report without reading it.

Contents

1 Introduction

(Delete and Replace): This report will document your learning throughout the course. It will be a collection of your notes, homework solutions, and critical reflections on the content of the course. Something in between a semester-long take home exam and your own lecture notes.[1]

The full report is due at the end of the finals week. It will be graded according to the following guidelines.

Grading guidelines (see also below):

- Is typesetting and layout professional?
- Is the technical content, in particular the homework, correct?
- Did the student find interesting references [BLA] and cites them throughout the report?

[1] One purpose of giving the report the form of lecture notes is that self-explanation is a technique proven to help with learning, see Chapter 6 of Craig Barton, How I Wish I'd Taught Maths, and references therein. In fact, the report can lead you from self-explanation (which is what you do for the weekly deadline) to explaining to others (which is what you do for the final submission). Another purpose is to help those of you who want to go on to graduate school to develop some basic writing skills. A report that you could proudly add to your application to graduate school (or a job application in industry) would give you full points.

- Do the notes reflect understanding and critical thinking?
- Does the report contain material related to but going beyond what we do in class?
- Are the questions interesting?

Do not change the template (fontsize, width of margin, spacing of lines, etc) without asking your first.

2 Week by Week

2.1 Week 1

Notes

Using proofs is important for reasoning on how computers can operate.

Homework

5) Goal: $a + (b + 0) + (c + 0) = a + b + c$

`rw [add_zero b, add_zero c]`
$a + b + c = a + b + c$
rfl

```
 Here we use  add_zero to rewrite and remove the extra zeros so that both sides are equal by reflexivit
```

6) Goal: $a + (b + 0) + (c + 0) = a + b + c$

`rw [add_zero b, add_zero c]`
$a + b + c = a + b + c$
rfl A nearly identical solution as number 5.

7) Goal: $succn = succ(n + 0)$ `rw [add_zero]`
rfl

```
 Here we used add_zero to rewrite n+0 to n which was all it took to show equality through reflexivity.
```

8) Goal: $2 + 2 = 4$
 `rw [four_eq_succ_three]`
2 + 2 = succ 3
 `rw [three_eq_succ_two]`
2 + 2 = succ (succ 2)
 `rw [two_eq_succ_one]`
succ 1 + succ 1 = succ (succ (succ 1))
 `rw [one_eq_succ_zero]`
succ (succ 0) + succ (succ 0) = succ (succ (succ (succ 0)))
 `rw [add_succ]`
succ (succ (succ 0) + succ 0) = succ (succ (succ (succ 0)))
 `rw [add_succ]`
succ (succ (succ (succ 0) + 0)) = succ (succ (succ (succ 0)))
 `rw [add_zero]`
succ (succ (succ (succ 0))) = succ (succ (succ (succ 0)))
 `rfl`
This is the longest proof in the homework and involves multiple rewrittings to show. The process I took was turning both 2's and the 4 into their forms as successors of 0. From that point I added successors to the left side and by doing so I rewrite the two separate 2's into one large successor. Finally by using add

zero, I removed one of the two left terms completely making it a single term on the left and right, both representative of 4.

Comments and Questions

This introduction world taught me how precise proofs need to be. I am beginning to understand how important it is to a machine that proofs are completely concrete, to the point where 2+2=4 becomes a tricky problem. As I go on to take these rewrites for granted, I will with the understanding that under the hood the computer does not go off intuition, but precise and exact steps.

Question of the week: How can systems of algorithmic reasoning as shown in the tutorial world, based in discrete mathematics, be encoded in modern programming languages?

2.2 Week 2

Homework

Question 1: Prove the theorem 'zero_add': For all natural numbers n, we have $0 + n = n$.

Proof:

```
theorem zero_add :  n : , 0 + n = n :=

  intro n,
  induction n with d hd,
  rw add_zero
  rw add_succ
  rw hd
  rfl
```

`This uses induction to prove the zero_add property. It is a simple way to introduce inductive reasoning`

Question 2: Prove the theorem: For all natural numbers a, b, we have $\text{succ}(a) + b = \text{succ}(a + b)$.

Proof:

```
induction b with a b
rw [add_zero]
rw [add_zero]
rfl
rw [add_succ]
rw [b]
rw [add_succ]
rfl
```

This proof uses induction on b to show that adding $\text{succ}(a)$ to b results in $\text{succ}(a + b)$. Inductive reasoning is key to establishing this property.

Question 3: Prove the theorem: On the set of natural numbers, addition is commutative. In other words, if a and b are arbitrary natural numbers, then $a + b = b + a$.

Proof:

```
induction a with b hd
rw [add_zero]
rw [zero_add]
rfl
```

```
rw [add_succ]
rw [succ_add]
rw [succ_eq_add_one]
rw [succ_eq_add_one]
rw [hd]
rfl
```

This proof establishes the commutativity of addition on natural numbers by using induction on a. Each step involves rewrites that simplify the equation until both sides match.

Question 4: Prove the theorem: On the set of natural numbers, addition is associative. In other words, if a, b, and c are arbitrary natural numbers, we have $(a + b) + c = a + (b + c)$.

Proof:

```
induction a with b hd
rw [zero_add]
rw [add_comm]
rw [zero_add]
rfl
rw [succ_add]
rw [succ_add]
rw [succ_add]
rw [succ_eq_add_one]
rw [succ_eq_add_one]
rw [hd]
rfl
```

This proof uses induction on a to establish the associativity of addition on the natural numbers. Each step simplifies the expression to demonstrate that the addition operation is associative.

Question 5: Prove the theorem: If a, b, and c are arbitrary natural numbers, we have $(a+b)+c = (a+c)+b$.

Proof:

```
induction a with b hd
rw [zero_add]
rw [zero_add]
rw [add_comm]
rfl
rw [add_assoc]
nth_rewrite 1 [add_assoc]
nth_rewrite 2 [add_comm]
rfl
```

This proof uses induction on a to show that the expression $(a + b) + c$ can be rearranged to $(a + c) + b$. Various rewrite steps and the commutativity and associativity of addition are used to achieve the result.

Comments and Questions

In this homework, I used lean to complete many proofs I've done in Discrete Mathematics including commutativity and associativity. These are proofs that I did previously use induction to solve, but representing that through lean to a computer makes the challenge extra formal, as there can be no jumps in logic. It's another week where the homework makes me thoughtful on how computers function at a lower level.

What challenges or additional considerations might arise when using a more complicated set like the set of all real numbers compared to the set of natural numbers?

2.3 . . .

. . .

3 Lessons from the Assignments

A proposition is a data type that is built of logical systems that can be either true or false.

(Delete and Replace): Write three pages about your individual contributions to the project.

On 3 pages you describe lessons you learned from the project. Be as technical and detailed as possible. Particularly valuable are *interesting* examples where you connect concrete technical details with *interesting* general observations or where the theory discussed in the lectures helped with the design or implementation of the project.

Write this section during the semester. This is approximately a quarter of apage per week and the material should come from the work you do anyway. Just keep your eyes open for interesting lessons.

Make sure that you use LaTeX to structure your writing (eg by using subsections).

4 Conclusion

(Delete and Replace): (approx 400 words) A critical reflection on the content of the course. Step back from the technical details. How does the course fit into the wider world of software engineering? What did you find most interesting or useful? What improvements would you suggest?

4.1 Week 2

Notes

Homework

Comments and Questions

References

[BLA] Author, Title, Publisher, Year.