

Constructive Logic (15-317), Spring 2023

Assignment 2: Proofs as Programs (60 points)

Instructor: Frank Pfenning

Due: February 9, 2023, 11:59 pm

This assignment will have a written portion and a coding portion. You will submit both portions through Gradescope.

We recommend that you typeset your written solutions. Most students use \LaTeX , but other software is acceptable. If you choose not to typeset your solutions, be aware that your handwriting must be **legible**.

For the coding portion you will use Dcheck. You can find documentation on Dcheck at cs.cmu.edu/~crary/dcheck/dcheck.pdf and a sample file at cs.cmu.edu/~crary/dcheck/example.deriv.

1 Proof Terms (20 points)

Using Dcheck, give derivations with proof terms of each of the following judgements.

Task 1. (5 points)

$$((A \supset B) \wedge (B \supset C)) \supset (A \supset C) \text{ true}$$

Task 2. (5 points)

$$(\neg A \wedge \neg B) \supset \neg(A \vee B) \text{ true}$$

Task 3. (5 points)

$$A \supset \neg\neg A \text{ true}$$

Task 4. (5 points)

$$((A \supset C) \wedge (B \supset C)) \supset ((A \vee B) \supset C) \text{ true}$$

2 Proofs as Programs (20 points)

In this program you will look at the proof-as-programs paradigm not through the lens of theoretical proof terms, but as actual Standard ML programs. We are interested in the propositions:

- a. $(A \wedge B \supset C) \supset (A \supset B \supset C)$
- b. $((A \supset B) \supset B) \supset A$
- c. $(A \supset B) \supset (\neg B \supset \neg A)$
- d. $((A \vee B) \wedge \neg A) \supset B$

Some of these propositions are true, others are not.

Task 5. (20 points) In your SML solution in `hw2.sml`, fill in the definition of the values shown below.

```
val curry : (('a * 'b -> 'c) -> 'a -> 'b -> 'c) option
val abba : ((( 'a -> 'b) -> 'b) -> 'a) option
val contrapositive : (('a -> 'b) -> (('b -> void) -> ('a -> void))) option
val exclusion : (('a, 'b) sum * ('a -> void) -> 'b) option
```

Each field should be `SOME` if the corresponding proposition is true, and `NONE` if it is not. **Do not use exceptions, recursion, or any other “cheat.”**

Your code should type-check in the environment we live-coded, which is included in the solution template `hw2.sml`. Your code will not be autograded, so to facilitate grading please cut-and-paste output in a comment at the end of your file.

3 Verifications & Uses (20 points)

Consider the \odot connective:¹

$$\begin{array}{c}
 \frac{A \text{ true} \quad B \text{ true}}{\odot(A, B, C) \text{ true}} \odot I_1 \quad \frac{A \text{ true} \quad C \text{ true}}{\odot(A, B, C) \text{ true}} \odot I_2 \quad \frac{B \text{ true} \quad C \text{ true}}{\odot(A, B, C) \text{ true}} \odot I_3 \\
 \frac{\overline{A \text{ true}}^x \quad \overline{B \text{ true}}^y \quad \overline{A \text{ true}}^x \quad \overline{C \text{ true}}^z \quad \overline{B \text{ true}}^y \quad \overline{C \text{ true}}^z}{\odot(A, B, C) \text{ true} \quad D \text{ true} \quad D \text{ true} \quad D \text{ true}} \odot E^{(x,y,z)} \\
 D \text{ true}
 \end{array}$$

Task 6. (10 points) Give appropriate rules for \odot in verifications & uses.

Using Dcheck, give a derivation of the following judgement.

Task 7. (10 points)

$$(\neg P \wedge Q) \supset ((P \supset Q) \supset (\neg P \supset \neg Q)) \supset \perp \uparrow$$

Name your derivation task7. Dcheck takes the propositions P and Q to be propositional variables. We restrict the $\downarrow \uparrow$ rule (called UV by Dcheck) to propositional variables, which are only P and Q here.

¹in Latex: `\smiley` using the `wasysym` package