Logic, First Course, Winter 2020. Week 8, Practice Problems. Back to course website

Week 8, Practice Problems

The practice problems fall into four groups:

Before beginning it is probably best to have the handout with all of the rules for Week 8 beside you. (It is also in Lecture 1 from this week.)

Commutativity and associativity

The first five problems could be solved using introduction and elimination rules for conjunction. But rather solve them first by using associativity and commutativity of conjunction. The proofs are often quicker, and steps like these are often appealed to in the context of more difficult proofs.

Example 1.

This first one is just practice applying commutativity of conjunction to a part of the given formula. There is a proof of this one using commutativity which is a total of two lines.

exercise

```
(a ∧ (b ∧ c)) ⊢ (a ∧ (c ∧ b))

1. a∧(b∧c) :assumption
```

Example 2.

This next one extends the previous one by adding on an application of associativity at the end.

exercise

```
(a ∧ (b ∧ c)) ⊢ ((a ∧ c) ∧ b)

1. a∧(b∧c) :assumption
```

Example 3.

This next one extends the previous one by adding on an application of commutativity at the end.

exercise

```
(a ∧ (b ∧ c)) ⊢ ((c ∧ a) ∧ b)

1. a∧(b∧c) :assumption
```

Example 4.

This one is a slightly different combination of commutativity and associativity than the previous one.

exercise

```
(a ∧ ((b ∧ c) ∧ d)) ⊢ ((b ∧ a) ∧ (d ∧ c))

1. a∧((b∧c)∧d) :assumption
```

Example 5.

This next one can be proven using only associativity. Try doing this one on a sheet of paper before putting it into the proof-checker. If the first step isn't clear to you, on your sheet of paper replace $c \wedge d$ by x (or some other new letter) and then apply associativity. If the second step isn't clear to you, on your sheet of paper replace $a \wedge b$ by y (or some new letter) and then apply associativity.

exercise

```
(a ∧ (b ∧ (c ∧ d))) ⊢ (((a ∧ b) ∧ c) ∧ d)

1. a∧(b∧(c∧d)) :assumption
```

Distribution

Example 6.

This one is just an application of distribution followed by two applications of commutativity. Don't let the presence of negations throw you-- just view the ~a, ~b, ~c as propositional letters in and of themselves.

exercise

```
(¬a ∧ (¬b ∨ ¬c)) ⊢ ((¬b ∧ ¬a) ∨ (¬c ∧ ¬a))

1. ~a∧(~b∨~c) :assumption
```

Example 7.

This one is just like the previous one. Don't let the presence of the arrow distract you-- just start modifying the antecedent just like you did in the previous problem.

exercise

```
((¬a ∧ (¬b ∨ ¬c)) → d) ⊢ (((¬b ∧ ¬a) ∨ (¬c ∧
¬a)) → d)
```

```
1. (~a∧(~b∨~c))→d :assumption
```

Example 8.

For this one, try distribution and then positive disjunctive syllogism. Make sure you are applying the right form of distribution, and don't forget to do some conjunction elimination along the way.

exercise

```
(a ∨ (b ∧ c)), (¬c ∧ ¬d) ⊢ a
```

```
1. a∨(b∧c) :assumption
2. ~c∧~d :assumption
3. ~c∧~d :assumption
```

Example 9.

For this one try a distribution followed by a couple of commutativities. In the first step, just on a sheet of paper replace $a \wedge b$ by x (or some other letter) and apply distribution. Don't forget that when applying commutativity, you can only do one application per line.

exercise

```
((a ∧ b) ∨ (c ∧ d)) ⊢ ((c ∨ (a ∧ b)) ∧ (d ∨
(a ∧ b)))
```

```
1. (a∧b)∨(c∧d) :assumption
```

Example 10.

For this one, just continue your previous proof by applying distribution twice inside the two conjuncts of the conclusion to Example 9, and then finish with a bunch of commutativity so as to "alphabetize" the various conjuncts.

exercise

```
((a ∧ b) ∨ (c ∧ d)) ⊢ (((a ∨ c) ∧ (b ∨ c)) ∧
((a ∨ d) ∧ (b ∨ d)))
```

```
1. (a∧b)∨(c∧d) :assumption
```

DeMorgan and Disjunctive Syllogism and Arrow

Example 11.

This one is just a couple of applications of Demorgan.

exercise

```
¬(a ∧ (b ∧ c)) ⊢ (¬a ∨ (¬b ∨ ¬c))
```

```
1. ~(a∧(b∧c)) :assumption
```

Example 12.

For this one, start with your proof for Example 11, and then apply disjunctive syllogism a few times followed by modus ponens.

exercise

```
¬(a ∧ (b ∧ c)), ¬¬a, ¬¬b, (¬c → ¬d) ⊢ ¬d
```

```
1. ~(a∧(b∧c)) :assumption
2. ~~a :assumption
3. ~~b :assumption
4. ~c→~d :assumption
```

Example 13.

A proof of this one can be obtained by taking a proof of the previous one and just turning the *a* into ¬¬*a*, and similarly for *b*. Keep in mind that in taking your proof from Example 12, you'll have to change the lines numbers of the justifications in moving them into this proof.

exercise

```
¬(a ∧ (b ∧ c)), a, b, (¬c → ¬d) ⊢ ¬d
```

```
1. ~(a∧(b∧c)) :assumption
2. a :assumption
3. b :assumption
4. ~c→~d :assumption
5. ~~a :DN2
6. ~~b :DN3
7. ~c→~d :assumption
```

Example 14.

For this one, try to apply combinations of disjunctive syllogism and DeMorgan.

exercise

```
¬(p ∧ q), ¬¬p, (¬q → ¬(r ∧ t)) ⊢ (¬r ∨ ¬t)
```

```
1. ~(p∧q) :assumption
2. ~~p :assumption
3. ~q→~(r∧t) :assumption
```

Example 15.

For this one, try to do DeMorgan followed by reasoning by cases (aka disjunction elimination). Further hint: before applying disjunctive syllogism, try flipping *d* ∨ *e* to *e* ∨ *d*.

exercise

```
¬(a ∧ b), (¬a → c), (¬b → c), (¬¬c → (d ∨
e)), ¬e ⊢ d
```

```
1. ~(a∧b) :assumption
2. ~a→c :assumption
3. ~b→c :assumption
4. ~~c→(d∨e) :assumption
5. ~e :assumption
```

Putting it all together

We can put together all the things we have learned in this course, and the last problems in this practice problem set are just an example of this. Consider:

> If Anthony attends or Briana attends then Cynthia attends. But Cynthia does not attend. Therefore, Anthony does not attend and Brianna does not attend.

Suppose that we were not at the meeting, but we knew the first premise due to our long experience with these characters and we knew the second premise because we got a message from Cynthia. Do we know the conclusion? If the argument was valid, we would know the conclusion. And to see whether it is valid, we would first translate into propositional logic, using the key:

> *a* = Anthony attends
>
> *b* = Briana attends
>
> *c* = Cynthia attends

Then we could translate as follows:

First premise:

```
If Anthony attends or Briana attends then Cyn
```

If Anthony attends or Briana attends then Cynthia attends.

Second premise:

```
Cynthia does not attend.
```

Cynthia does not attend.

Conclusion:

```
Anthony does not attend and Brianna does not a
```

Anthony does not attend and Brianna does not attend.

We probably do not need it at this point, but if we wanted we could parse these sentences too (again, ignore the "submit your solution" bit since we are just practicing):

((a ∨ b) → c)

¬c

(¬a ∧ ¬b)

Then to assess for validity, we could patiently fill out the entire truth-table and see that it is a validity:

((a ∨ b) → c), ¬c ⊢ (¬a ∧ ¬b)

abc	((a	∨	b)	→	c),	¬	c	⊢	(¬	a	∧	¬	b)
TTT	T		T		T		T			T			T
TTF	T		T		F		F			T			T
TFT	T		F		T		T			T			F
TFF	T		F		F		F			T			F
FTT	F		T		T		T			F			T
FTF	F		T		F		F			F			T
FFT	F		F		T		T			F			F
FFF	F		F		F		F			F			F

Counterexample Check

Alternatively, given that we can also assess validity via proofs, we could also just do a derivation:

exercise

((a ∨ b) → c), ¬c ⊢ (¬a ∧ ¬b)

```
1. (a∨b)→c :assumption
2. ~c :assumption
```

In the final exam, you will be asked to do a problem like this. But in the exam, we will ask you to translate the problem then assess for validity, without us having displayed the translation. That is, we will just give you the translation exercise and then ask you to assess the argument for validity using either truth-tables or a proof (and the problem will specify which is being asked for).

This is a practice problem set for this course. It is run on the Carnap software, which is an: