4CCS1ELA – Elementary Logic with Applications

Small Group Tutorial 4 (week 9) – Solutions

1. Consider the following first-order sentence

$$\forall x \forall y (N(x) \wedge N(y) \rightarrow \exists z ((Q(z) \wedge ((x < z \wedge z < y) \vee (y < z \wedge z < x))))$$

and an interpretation over the set of real numbers such that

$N(x)$ means 'x is a natural number',
$Q(x)$ means 'x is a rational number',
$x < y$ means 'x is less than y'.

(i) Give an English translation of this sentence.

(ii) Determine if this sentence is true or false under the given interpretation.

Solution

(i) For every pair of natural numbers x and y there exists a rational number z such that x is less that z and z is less than y or y is less that z and z is less than x.

(ii) This sentence is false. To be true an additional premise $x \neq y$ should be added.

2. Assume that $\exists x \exists y P(x, y)$ is true. Which of the following formulae must also to be true? If the formula is true, explain why. Otherwise, give a counterexample.

(i) $\forall x \forall y P(x, y)$.

(ii) $\forall x \exists y P(x, y)$.

(iii) $\exists x \forall y P(x, y)$.

SOLUTION

(i) $\forall x \forall y P(x,y)$. This formula might be false, i.e., $\exists x \exists y P(x,y) \not\models \forall x \forall y P(x,y)$.

Let $D = \{a,b\}$ and $P(a,a) = 1$ but $P(a,b) = P(b,a) = P(b,b) = 0$. Then $\exists x \exists y P(x,y)$ is true under the interpretation $\mathcal{I}$ (witness $x = a$, $y = a$), but $\forall x \forall y P(x,y)$ is false under $\mathcal{I}$ (witness $x = a, y = b$).

Note. For simplicity, instead of writing $\mathcal{I}(P(a,b)) = 1$ etc, we will often write simply $P(a,b) = 1$ if it is clear what interpretation is applied (as it is done in this case).

Another counterexample. Let the domain be the set of natural numbers. Let $P(x,y)$ mean $x < y$. Then $\exists x \exists y P(x,y)$ is true (witness $x = 1, y = 2$), but $\forall x \forall y P(x,y)$ is false (witness $x = 2, y = 1$).

(ii) $\forall x \exists y P(x,y)$. This formula can be false, i.e., $\exists x \exists y P(x,y) \not\models \forall x \exists y P(x,y)$. The first interpretation given in the answer (i) is a counterexample for this case too (witness $x = b$).

Another counterexample. Let the domain be the set of natural numbers. Let $P(x,y)$ mean $x > y$. Then $\exists x \exists y P(x,y)$ is true (witness $x = 2, y = 1$), but $\forall x \exists y P(x,y)$ is false (witness $x = 0$).

(iii) $\exists x \forall y P(x,y)$. This formula can be false, i.e., $\exists x \exists y P(x,y) \not\models \exists x \forall y P(x,y)$. All three interpretations given in the answers (i) and (ii) are counterexamples for this case.

3. Find a counterexample to show that the following argument is not valid:

$$\exists x P(x),\ \exists x (P(x) \to Q(x)) \ \models \exists x Q(x).$$

SOLUTION

We have to find an interpretation $\mathcal{I}$ such that both premises are true under $\mathcal{I}$ but the conclusion is false. A possible counterexample is $D = \{a,b\}$ and $P(a) = 1$ but $P(b) = Q(b) = Q(a) = 0$.

Another counterexample. Let $D = \mathbf{N}$, then set of natural numbers, let P(x) = 'x = x+x', and $Q(x) =$ '$x+1 = x$. Then $P(0) = 1, P(1) = Q(1) = 0$. Therefore, $\exists x Q(x) = 1$ and $\exists x (P(x) \to Q(x)) = 1$, i.e. both premises are true. However $Q(x)$ is false for all x in $\mathbf{N}$. Therefore, this argument is invalid.

4. Find an error in the following attempt to give a formal proof for the (invalid) argument in the previous exercise.

$$\exists x P(x),\ \exists x(P(x) \to Q(x)) \ \models\ \exists x Q(x).$$

1. $\exists x P(x)$		P	
2. $\exists x(P(x) \to Q(x))$		P	
3. $P(a)$	1	EI	Existential Instantiation
4. $P(a) \to Q(a)$	2	EI	Existential Instantiation
5. $Q(a)$	$3, 4$	MP	Modus ponens
6. $\exists x Q(x)$	10	EG	Existential Generalisation

Here, a is a new constant.

SOLUTION

In step 4, the EI rule must use a new constant different of a (because after appearing in step 3 the constant a is not 'unknown' anymore).

5. (a) Express the following specification in the language of predicate logic, using the dictionary below:

> *If every component works properly and all interfaces are functioning then every test-run will terminate.*
>
> Dictionary :
>
> $C(x)$: x is a component
> $W(x)$: x works properly
> $I(x)$: x is an interface
> $F(x)$: x is functioning
> $R(x)$: x is a test-run
> $T(x)$: x will terminate

SOLUTION

$$\forall x(C(x) \to W(x)) \land \forall x(I(x) \to F(x)) \to \forall x(R(x) \to T(x)).$$

Remarks: Of course, this may be expressed by choosing different variables for each quantifier, e.g

$$\forall x(C(x) \to W(x)) \land \forall y(I(y) \to F(y)) \to \forall z(R(z) \to T(z)).$$

By the law of distribution of the universal quantifier over conjunction, this is equivalent to the following, which is therefore also a correct answer :

$$\forall x((C(x) \to W(x)) \land (I(x) \to F(x))) \to \forall x(R(x) \to T(x)).$$

Remarks :
Note however that we cannot merge the last quantifier into the first one. In other words, the following answer is **incorrect**:

$$\forall x((C(x) \to W(x)) \land (I(x) \to F(x)) \to (R(x) \to T(x))).$$

(b) Negate the expression that you constructed, and then use the principles of quantifier interchange, together with equivalences of propositional logic, to move negation signs inwards as far as you can.

SOLUTION

$$\begin{array}{rl}
\neg(\forall x(C(x) \to W(x)) \land \forall x(I(x) \to F(x)) \to \forall x(R(x) \to T(x))) & \equiv \\
\forall x((C(x) \to W(x)) \land \forall x(I(x) \to F(x)) \land \neg\forall x(R(x) \to T(x))) & \equiv \\
\forall x((C(x) \to W(x)) \land \forall x(I(x) \to F(x)) \land \exists x\neg(R(x) \to T(x))) & \equiv \\
\forall x((C(x) \to W(x)) \land \forall x(I(x) \to F(x)) \land \exists x(R(x) \land \neg T(x))) & \equiv
\end{array}$$

(c) Translate the result of your work in (b) back into English.

SOLUTION

Every component works properly and all interfaces are functioning, but some test-run will not terminate.

Stylistic variations, equally correct:
Although every component works properly and all interfaces are functioning, some test-run will not terminate,

or:
Some test-runs won't terminate, even though every component works properly and all interfaces are functioning.