CS 536 — Solution for Homework 1: Logic Review; Expressions, States

Logic Review:

1. (Full parenthesization)
   a. \( p \land \neg r \land s \rightarrow \neg q \lor r \rightarrow \neg p \leftrightarrow \neg s \rightarrow t \equiv (((p \land (\neg r)) \land s) \rightarrow ((\neg q) \lor r) \rightarrow (\neg p)) \leftrightarrow ((\neg s) \rightarrow t) \)
   b. \((p \rightarrow ((q \land r) \rightarrow ((p \rightarrow (\neg q)) \rightarrow r)) \land (\neg (q \land r) \rightarrow (\neg p)))\)
   c. \( \exists m. 0 \leq m < n \land \forall j. 0 \leq j < m \rightarrow b[0] \leq b[j] \land b[j] \leq b[m] \)
      \(\equiv (\exists m. ((0 \leq m < n) \land (\forall j. ((0 \leq j < m) \rightarrow (b[0] \leq b[j])) \land (b[j] \leq b[m])))\)
   d. \((\exists x. (\forall y. f(x) > g(y))) \rightarrow (\forall y. (\exists x. (f(x) \geq g(y))))\)

2. (Minimal parenthesization) All that was required was the final proposition or predicate, but I'm going to show the reasoning in detail, hoping it might help some people. Notation: "operator1 over operator2" means operator 1 has higher precedence (= is stronger than) operator 2. E.g., (\(\neg\) over \(\land\)).
   a. Subscribing the parentheses of \(((\neg(p \lor q) \land r) \rightarrow (((\neg q) \lor r) \rightarrow ((p \lor (\neg r)) \lor q) \land s))\) gives us
      \(1 \rightarrow (\neg(3 \lor q)) \land r) \rightarrow ((a(5 \lor q) \land r) \lor s) \rightarrow (7(8 \lor r) \land q) \lor s \lor (10(9 \land r) \land s) \lor 1\)
      I'll work from the inside out, in phases. Of the most deeply embedded parentheses, pairs 6 (\(\neg \) over \(\land\)), 9 (\(\neg \) over \(\lor\)), and 10 (\(\land \) over \(\lor\)) are redundant but pair 3 is necessary (\(\neg \) over \(\lor\)). Removing pairs 6, 9, and 10 gives us
      \(1 \rightarrow (\neg(3 \lor q)) \land r) \rightarrow ((a(5 \lor q) \land r) \lor s) \rightarrow (7(8 \lor r) \land q) \lor s \lor (10(9 \land r) \land s) \lor 1\)
      Now, we can remove the redundant pairs 2 (\(\lor \) over \(\rightarrow\)), 5 (\(\lor \) over \(\rightarrow\)), and 8 (\(\lor \) is associative):
      \(1 \rightarrow (\neg(3 \lor q)) \land r) \rightarrow ((a(5 \lor q) \land r) \lor s) \rightarrow (7(8 \lor r) \land q) \lor s \lor (10(9 \land r) \land s) \lor 1\)
      The remaining pairs are redundant: 7 (\(\lor \) over \(\rightarrow\)), 4 (\(\rightarrow \) is right-associative), and 1 (the outermost pair).
      We can drop the subscript 3 and get the final answer,
      \(\neg(p \lor q) \land r) \rightarrow \neg q \lor r \rightarrow p \lor r \lor q \land s\).
   b. \(((p \lor (q \lor r)) \lor (q \lor s)) \lor (q \lor r) \lor (p \lor q)) \equiv \)
      \(1 \rightarrow (\neg(3 \lor q) \land r) \rightarrow ((a(5 \lor q) \land r) \lor s) \rightarrow (7(8 \lor r) \land q) \lor s \lor (10(9 \land r) \land s) \lor 1\)
      Pairs 4 and 9 redundant because \(\lor\) and \(\land\) are associative; pair 11 is also redundant (\(\neg \) over \(\land\))
      \(1 \rightarrow (\neg(3 \lor q) \land r) \rightarrow ((a(5 \lor q) \land r) \lor s) \rightarrow (7(8 \lor r) \land q) \lor s \lor (10(9 \land r) \land s) \lor 1\)
      Pairs 3 (\(\lor \) over \(\rightarrow\)), 8 (\(\lor \) over \(\rightarrow\)), and 10 (\(\lor \) over \(\lor\)) are redundant:
      \(1 \rightarrow (\neg(3 \lor q) \land r) \rightarrow ((a(5 \lor q) \land r) \lor s) \rightarrow (7(8 \lor r) \land q) \lor s \lor (10(9 \land r) \land s) \lor 1\)
      Pairs 6 and 7 are redundant because \(\rightarrow\) is right-associative; the outermost pair 1 is also redundant
      \(2 \lor q \lor r) \rightarrow ((5 \lor q) \rightarrow (s \lor p \land r) \lor s) \rightarrow (7(8 \lor r) \land q) \lor s \lor (10(9 \land r) \land s) \lor 1\)
      Pairs 2 and 5 are required because \(\rightarrow\) is right-associative. Erasing subscripts gives us
      \(p \lor q \lor r) \rightarrow ((q \lor r) \rightarrow (s \lor p \land r)) \rightarrow (10 \lor p \land q)\)
      2c. \((\exists i. (((0 \leq i) \land (i < m)) \land (\forall j. (((m \leq j) \land (j < n)) \rightarrow (b[i] = b[j]))))))\). Let me try not subscripting this time. The tests \(\leq\), \(<\), and \(=\) are stronger than \(\land\), \(\lor\), and \(\rightarrow\) (also \(\neg\) and \(\leftrightarrow\) for that matter), so we can delete the parentheses around them and get
      \((\exists i. (((0 \leq i) \land (i < m)) \lor (\forall j. (((m \leq j) \land (j < n)) \rightarrow (b[i] = b[j]))))))\)
The body of \((\forall j \text{ ends at its matching } )\), so we can drop the parentheses around the body: \(\forall j,(\ldots)\). In that body, \(\land\) is stronger than \(\to\), so altogether we can simplify \((\forall j\ldots)\):

\[
(\exists i.(0 \leq i \land i < m) \land (\forall j.m \leq j \land j < n \to b[i] = b[j]))
\]

Similarly, the body of \((\exists i \text{ ends at the matching } )\).

\[
(\exists i.(0 \leq i \land i < m) \land (\forall j.m \leq j \land j < n \to b[i] = b[j]))
\]

Finally, since \(\land\) is associative, we can remove the parentheses in \((\ldots \land \ldots (\ldots))\) and also delete the outermost parentheses to get the final answer

\[
\exists i.0 \leq i \land i < m \land \forall j.m \leq j \land j < n \to b[i] = b[j]
\]

(Note: In real life, we'd abbreviate \(0 \leq i \land i < m\) to \(0 < i < m\) and similarly for \(j\) to get

\[
\exists i.0 < i < m \land \forall j.m < j < n \to b[i] = b[j]
\]

We could even use bounded quantifiers and get \(\exists 0 \leq i < m.\forall m \leq j < n.b[i] = b[j]\). We'd have to rely on the context to know that the quantifiers are over \(i\) and \(j\), not \(m\) or \(n\).

2d. \((\forall x.((\exists y.(p \to q)) \to (\forall z.(q \lor (r \land s))))))\) minimizes to \((\forall x.((\exists y.p \to q) \to (\forall z.q \lor r \land s)))\).

Here's a brief explanation of the highlights: The parentheses of \((\exists y \ldots)\) are necessary to keep the body \(p \to q\) from becoming \(p \to q \to (\forall z\ldots)\). The parentheses of \((\forall z \ldots)\) are redundant because the right parentheses is at the end of the predicate.

In real life, you can certainly argue that for symmetry, we might want to write the non-minimal \((\forall x.(\exists y.p \to q) \to (\forall z.q \lor r \land s)))\).

3. (Syntactic equality)

a. They are \(\neq\): We have \(p \land q \lor \neg r \to \neg p \to q \equiv ((p \land q) \lor \neg r) \to (\neg p \to q)\)

   but \(((p \land q) \lor (\neg r \to ((\neg p) \to q))) \equiv p \land q \lor (\neg r \to (\neg p \to q))\).

b. They are \(\equiv\). Both \((p \to (\neg q \land s) \to (r \land (s \to p)))\) and \(p \to (\neg q \land s \land r \land (s \to p))\) minimize to

   \(p \to \neg q \land s \to r \land (s \to p)\).

c. They are \(\neq\). \(\forall x.\exists y.p \to \exists y.q \to r \equiv \forall x.((p \to \exists y.(q \to r)), but

   (\forall x.\exists y.q \to r \equiv ((\forall x.p) \to (\exists y.q) \to r))\)

d. They are \(\neq\) because of the swapped order of quantifiers and conjuncts. (They are logically equivalent, however: \(\forall x.\forall y.p \land q \iff \forall y.\forall x.q \land p\).)

4. (Tautology, contradiction, or contingency)

a. \(s \land t \lor u \to s \lor u\) is a tautology

b. \((p \to (q \to r)) \equiv ((p \to q) \to r)\) is a contingency. When \(p\) is \(T\) and \(q\) and \(r\) are \(F\), this becomes \(T \leftrightarrow T\)

   which is \(T\), but when \(q\) is \(T\) and \(p\) and \(r\) are \(F\), this becomes \(T \leftrightarrow F\), which is \(F\).

c. \((\forall x \in \mathbb{Z}.\forall y \in \mathbb{Z}.f(x, y) > 0) \to (\exists x \in \mathbb{Z}.\exists y \in \mathbb{Z}.f(x, y) > 0)\) is a tautology. If \(f(x, y) > 0\) for all integers \(x\) and \(y\), then any two integers prove the existentials: Since (for example) \(f(0, 0) > 0\), there exist \(x\) and \(y\) with \(f(x, y) > 0\).
d. \( \neg (\exists x \in \mathbb{Z} . \neg \forall y \in \mathbb{Z} . f(x, y) > 0) \rightarrow (\exists x \in \mathbb{Z} . \exists y \in \mathbb{Z} . f(x, y) \leq 0) \) is a contingency. For convenience, let \( p \equiv (\exists x . \neg \forall y . f(x, y) > 0) \) and \( q \equiv (\exists x . \exists y . f(x, y) \leq 0) \) and let the quantifiers all range over \( \mathbb{Z} \). (So we're being asked about \( \neg p \rightarrow q \).)

Using DeMorgan's law, we know \( p \equiv (\exists x . \neg \forall y . f(x, y) > 0) \equiv (\exists x . \exists y . \neg f(x, y) > 0) \), which \( \equiv (\exists x . \exists y . f(x, y) \leq 0) \) by negation of \( > \). So \( p \equiv q \). Going back to \( \neg p \rightarrow q \), in any given state \( \sigma \), if \( \sigma \models p \), then \( \sigma \models q \), so \( \sigma \models \neg p \rightarrow q \) (since \( F \rightarrow T \)). If \( \sigma \models \neg p \), then \( \sigma \models \neg q \), so \( \sigma \models \neg p \rightarrow q \) (since \( T \rightarrow F \) doesn't hold).

Without more information about function \( f \), we have to assume that \( f(x, y) \) might be \( > 0 \) or \( \leq 0 \). So \( \neg p \rightarrow q \) might or might not be satisfied, which means that altogether, it's a contingency.

5. \( \leftrightarrow \) "\( p \) is sufficient for \( q \)" \( \iff p \rightarrow q \). "\( p \) only if \( q \", \( p \leftrightarrow q ' \), and "\( p \) is necessary for \( q \)" all \( \iff q \rightarrow p \)

6. \( \equiv \) vs \( = \)
   a. \( e_1 \equiv e_2 \) implies \( e_1 = e_2 \): If \( e_1 \equiv e_2 \) then they differ syntactically only in redundant parentheses, so they must have equal values.
   b. \( e_1 = e_2 \) does not imply \( e_1 \equiv e_2 \). Easy examples: (1) \( 2 + 2 = 4 \) but \( 2 + 2 \neq 4 \); (2) \( p \land q = \) (actually, \( \equiv \)) \( q \land p \), but we're taking \( p \land q \neq q \land p \).
   c. \( e_1 \neq e_2 \) implies \( e_1 \neq e_2 \) because it's the contrapositive of part a.

7. (Prove a tautology) [If you omitted the "\( \lor \) associative and commutative" step, that's okay.]

\[
\begin{align*}
p \land \neg(q \land r) & \rightarrow q \land r \rightarrow \neg p & \text{Defn} \\
\equiv p \land \neg(q \land r) & \rightarrow (\neg(q \land r) \lor \neg p) & \text{Defn} \\
\equiv \neg(p \land \neg(q \land r)) & \lor (\neg(q \land r) \lor \neg p) & \text{DeMorgan's law (on \( \neg(\ldots \land \ldots) \)) and \( \neg \neg \) } \\
\equiv (q \land r) & \lor \neg(q \land r) \lor \neg p \lor \neg p & \lor \text{ associative and commutative } \\
\equiv T & \lor \neg p \lor \neg p & \text{Excluded middle} \\
\equiv T & \text{Domination}
\end{align*}
\]

8. (Remove \( \neg \))

\[
\begin{align*}
\neg(\forall x . (\exists y . x \leq y) \lor \exists z . x \geq z) & \text{DeMorgan's Law (\( \neg \forall \leftrightarrow \exists \neg \))} \\
\equiv \exists x . (\neg(\exists y . x \leq y) \lor \exists z . x \geq z) & \text{DeMorgan's Law (\( \neg \lor \leftrightarrow \neg \land \neg \))} \\
\equiv \exists x . (\neg(\exists y . x \leq y) \land \neg \exists z . x \geq z) & \text{DeMorgan's Law (\( \neg \exists \equiv \forall \neg \) and \( \neg \) of \( \leq \))} \\
\equiv \exists x . (\forall y . x > y) \land \forall z . x < z & \text{DeMorgan's Law (\( \neg \exists \equiv \forall \neg \) and \( \neg \) of \( \geq \))}
\end{align*}
\]

9. \( \text{GT}(b, x, m, k) \equiv \forall i . m \leq i < m+k \rightarrow x > b[i] \) is one solution. \( \forall j . 0 \leq i < k \rightarrow x > b[m+j] \) is another.
Expressions and States

10. (Legal expressions) I included some comments; you weren’t required to do that.
   a. \((x < y \ ? \ T : 17)\) Illegal: (\(T\) and 17 aren’t of the same type)
   b. \((i = 3 \ ? \ b[1] : b’[2])[k]\) Illegal: (Conditional can’t yield an array)
   c. \(f(b, b’, x)\) Legal: \(f\) takes two arrays and an integer

11. (Legal and proper state; value of an expression relative to a state)
   a. \(\{x = \text{ten}, y = \text{eight plus one}\} \) and \(x+y\): The state is legal (it’s the same as \(\{x = 10, y = 8+1\}\)), and \(x+y\) evaluates to 19.
   b. \(\{b = \gamma, i = 1\} \) where \(\gamma(0) = 2\) and \(\gamma(1) = 5\), and \(b[b[i]-1]\): The state is legal and proper (it maps \(b\) to an array function and \(i\) to an integer), but the evaluating the expression causes a runtime error because we’re trying to access \(b[4]\).
   c. \(\{c = \alpha, d = 2\alpha, e = 3\alpha\} \) and \(d/c+(0*z)\): The state is legal but not proper because it doesn’t have a binding for \(z\). (Even though the value of \(z\) wouldn’t matter.) Note the binding for \(e\) isn’t relevant, since it doesn’t appear in the expression.