The University of Melbourne
School of Computing and Information Systems
COMP30026 Models of Computation

Sample Answers to Tutorial Exercises, Week 4

15. The connective $\Leftrightarrow$ is part of the language that we study, namely the language of propositional logic. So $A \Leftrightarrow B$ is just a propositional formula.

    The symbol $\equiv$ belongs to a *meta-language*. The meta-language is a language which we use when we reason *about* some language. In this case we use $\equiv$ to express whether a certain relation holds between formulas in propositional logic.

    More specifically, $\Phi \equiv \Psi$ means that we have both $\Phi \models \Psi$ and $\Psi \models \Phi$. In other words, Φ and Ψ have the same value for every possible assignment of truth values to their variables. The two formulas are logically equivalent.

    On the other hand $\Phi \Leftrightarrow \Psi$ is just a propositional formula (assuming Φ and Ψ are propositional formulas). For some values of the variables involved, $\Phi \Leftrightarrow \Psi$ may be false, for other values it may be true. By the definition of validity, $\Phi \Leftrightarrow \Psi$ is *valid* iff it is true for *every* assignment of propositional variables in Φ and Ψ.

    We want to show that $\Phi \equiv \Psi$ iff $\Phi \Leftrightarrow \Psi$ is valid.

    (a) Suppose $\Phi \equiv \Psi$. Then Φ and Ψ have the same values for each truth assignment to their variables[1]. But that means that, when we construct the truth table for $\Phi \Leftrightarrow \Psi$, it will have a t in every row, that is, $\Phi \Leftrightarrow \Psi$ is valid.

    (b) Suppose $\Phi \Leftrightarrow \Psi$ is valid. That means we find a t in each row of the truth table for $\Phi \Leftrightarrow \Psi$. But we get a t for $\Phi \Leftrightarrow \Psi$ iff the values for Φ and Ψ agree, that is, either both are f, or both are t. In other words, Φ and Ψ agree for every truth assignment. Hence $\Phi \equiv \Psi$.

    You may think that this relation between validity and biimplication is obvious and should always be expected, and indeed we will see that it carries over to first-order predicate logic. But there are (still useful) logics in which it does not hold.

16. Let us draw the truth tables.

    (a)

P	Q	P	$\Leftrightarrow$	$((P$	$\Rightarrow$	$Q)$	$\Rightarrow$	$P)$
t	t	t	t	t	t	t	t	t
t	f	t	t	t	f	f	t	t
f	t	f	t	f	t	t	f	f
f	f	f	t	f	t	f	f	f
			$\Uparrow$					

    Hence satisfiable, and in fact valid (all t).

[1] We should perhaps be more careful here, because Φ and Ψ can be logically equivalent without Φ having the exact same set of variables as Ψ—can you see how? So we should say that we consider both of Φ and Ψ to be functions of the *union* of their variables.

(b)

P	Q	$(P$	$\Rightarrow$	$\neg$	$Q)$	$\wedge$	$((P$	$\vee$	$Q)$	$\Rightarrow$	$P)$
t	t	t	f	f	t	f	t	t	t	t	t
t	f	t	t	t	f	t	t	t	f	t	t
f	t	f	t	f	t	f	f	t	t	f	f
f	f	f	t	t	f	t	f	f	f	t	f
						$\Uparrow$					

Hence satisfiable (at least one t), but not valid (not all t). The truth table shows the formula is equivalent to $\neg Q$.

(c)

P	Q	$((P$	$\Rightarrow$	$Q)$	$\Rightarrow$	$Q)$	$\wedge$	$(Q$	$\oplus$	$(P$	$\Rightarrow$	$Q))$
t	t	t	t	t	t	t	f	t	f	t	t	t
t	f	t	f	f	t	t	f	f	f	t	f	f
f	t	f	t	t	t	t	f	t	f	f	t	t
f	f	f	t	f	f	f	f	f	t	f	t	f
							$\Uparrow$					

Hence not satisfiable (and so certainly not valid).

17. If you negate a satisfiable proposition, you can never get a tautology, since at least one truth table row will yield false.

    You will get another satisfiable proposition iff the original proposition is not valid. For example, P is satisfiable (but not valid), and indeed $\neg P$ is satisfiable.

    Finally, if we have a satisfiable formula which is also valid, its negation will be a contradiction. Example: $P \vee \neg P$.

18. (a)

$$\begin{array}{ll}
\neg(A \wedge \neg(B \wedge C)) & \\
\neg A \vee (B \wedge C) & \text{(push negation in)} \\
(\neg A \vee B) \wedge (\neg A \vee C) & \text{(distribute } \vee \text{ over } \wedge\text{)}
\end{array}$$

The result is now in reduced CNF.

(b)

$$\begin{array}{ll}
A \vee (\neg B \wedge (C \vee (\neg D \wedge \neg A))) & \\
(A \vee \neg B) \wedge (A \vee C \vee (\neg D \wedge \neg A)) & \text{(distribute } \vee \text{ over } \wedge\text{)} \\
(A \vee \neg B) \wedge (A \vee C \vee \neg D) \wedge (A \vee C \vee \neg A) & \text{(distribute } \vee \text{ over } \wedge\text{)}
\end{array}$$

The result is in CNF but not RCNF. To get RCNF we need to eliminate the last clause which is a tautology, and we end up with $(A \vee \neg B) \wedge (A \vee C \vee \neg D)$.

(c)

$$\begin{array}{ll}
(A \vee B) \Rightarrow (C \wedge D) & \\
\neg(A \vee B) \vee (C \wedge D) & \text{(unfold } \Rightarrow\text{)} \\
(\neg A \wedge \neg B) \vee (C \wedge D) & \text{(de Morgan)} \\
(\neg A \vee (C \wedge D)) \wedge (\neg B \vee (C \wedge D)) & \text{(distribute } \vee \text{ over } \wedge\text{)} \\
(\neg A \vee C) \wedge (\neg A \vee D) \wedge (\neg B \vee C) \wedge (\neg B \vee D) & \text{(distribute } \wedge \text{ over } \vee\text{)}
\end{array}$$

The result is in RCNF. We could have chosen different orders for the distributions.

(d)

$$\begin{array}{ll} A \land (B \Rightarrow (A \Rightarrow B)) & \\ A \land (\neg B \lor \neg A \lor B) & \text{(unfold both occurrences of } \Rightarrow\text{)} \\ A & \text{(rightmost clause is tautological: remove it)} \end{array}$$

19. Let us follow the method given in a lecture, except we do the double-negation elimination aggressively, as soon as opportunity arises:

$$\begin{array}{ll} \neg((\neg B \Rightarrow \neg A) \Rightarrow ((\neg B \Rightarrow A) \Rightarrow B)) & \\ \neg(\neg(B \lor \neg A) \lor \neg(B \lor A) \lor B) & \text{(unfold } \Rightarrow \text{ and eliminate double negation)} \\ (B \lor \neg A) \land (B \lor A) \land \neg B & \text{(de Morgan for outermost neg; elim double neg)} \end{array}$$

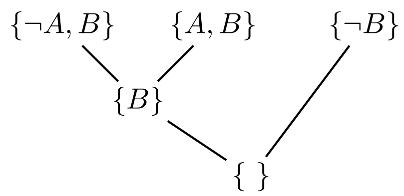
This is RCNF without further reductions.

We could also have used other transformations—sometimes this can shorten the process. For example, we could have rewritten the sub-expression $\neg B \Rightarrow \neg A$ as $A \Rightarrow B$ (the contraposition principle). You may want to check that this does not change the result.

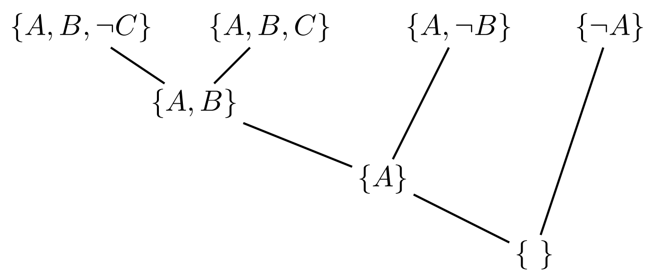
The resulting formula, written as a set of sets of literals:

$$\{\{\neg A, B\}, \{A, B\}, \{\neg B\}\}$$

We can now construct the refutation:



20. Here is a refutation:



From this we conclude that $(A \lor B \lor \neg C) \land \neg A \land (A \lor B \lor C) \land (A \lor \neg B)$ is unsatisfiable.

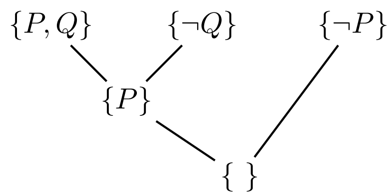
21. (a) $(P \lor Q) \Rightarrow (Q \lor P)$. First negate the formula (why?), to get $\neg((P \lor Q) \Rightarrow (Q \lor P))$. Then we can use the usual techniques to convert the negated proposition to RCNF. Here is a useful shortcut, combining $\Rightarrow$-elimination with one of de Morgan's laws:

$$\neg(A \Rightarrow B) \equiv A \land \neg B.$$

So:

$$\begin{array}{ll}
\neg((P \vee Q) \Rightarrow (Q \vee P)) & \\
(P \vee Q) \wedge \neg(Q \vee P) & \text{(shortcut)} \\
(P \vee Q) \wedge \neg Q \wedge \neg P & \text{(de Morgan)}
\end{array}$$

The result allows for a straight-forward refutation:



(b) $(\neg P \Rightarrow P) \Rightarrow P$. Again, first negate the formula, to get $\neg((\neg P \Rightarrow P) \Rightarrow P)$. Then turn the result into RCNF:

$$\begin{array}{ll}
\neg((\neg P \Rightarrow P) \Rightarrow P) & \\
(\neg P \Rightarrow P) \wedge \neg P & \text{(shortcut from above)} \\
(\neg\neg P \vee P) \wedge \neg P & \text{(unfold } \Rightarrow\text{)} \\
(P \vee P) \wedge \neg P & \text{(eliminate double negation)} \\
P \wedge \neg P & (\vee\text{-absorption})
\end{array}$$

The resolution proof is immediate; we will leave it out.

(c) $((P \Rightarrow Q) \Rightarrow P) \Rightarrow P$. Again, negate the formula, to get $\neg(((P \Rightarrow Q) \Rightarrow P) \Rightarrow P)$. Then turn the result into RCNF:

$$\begin{array}{ll}
\neg(((P \Rightarrow Q) \Rightarrow P) \Rightarrow P) & \\
((P \Rightarrow Q) \Rightarrow P) \wedge \neg P & \text{(shortcut, outermost } \Rightarrow\text{)} \\
((\neg P \vee Q) \Rightarrow P) \wedge \neg P & \text{(unfold } \Rightarrow\text{)} \\
(\neg(\neg P \vee Q) \vee P) \wedge \neg P & \text{(unfold } \Rightarrow\text{)} \\
((\neg\neg P \wedge \neg Q) \vee P) \wedge \neg P & \text{(de Morgan)} \\
((P \wedge \neg Q) \vee P) \wedge \neg P & \text{(double negation)} \\
(P \vee P) \wedge (\neg Q \vee P) \wedge \neg P & \text{(distribution)} \\
P \wedge (\neg Q \vee P) \wedge \neg P & \text{(absorption)}
\end{array}$$

Again this gives an immediate refutation: just resolve $\{P\}$ against $\{\neg P\}$.

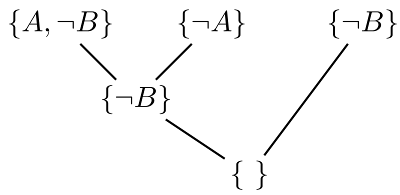
(d) $P \Leftrightarrow ((P \Rightarrow Q) \Rightarrow P)$. Negating the formula, we get $P \oplus ((P \Rightarrow Q) \Rightarrow P)$. Let us turn the resulting formula into RCNF:

$$\begin{array}{ll}
P \oplus ((P \Rightarrow Q) \Rightarrow P) & \\
(P \vee ((P \Rightarrow Q) \Rightarrow P)) \wedge (\neg P \vee \neg((P \Rightarrow Q) \Rightarrow P)) & \text{(eliminate } \oplus\text{)} \\
(P \vee ((P \Rightarrow Q) \Rightarrow P)) \wedge (\neg P \vee ((P \Rightarrow Q) \wedge \neg P)) & \text{(shortcut from above)} \\
(P \vee (\neg(\neg P \vee Q) \vee P)) \wedge (\neg P \vee ((\neg P \vee Q) \wedge \neg P)) & (\Rightarrow\text{-elimination}) \\
(P \vee (\neg\neg P \wedge \neg Q) \vee P) \wedge (\neg P \vee ((\neg P \vee Q) \wedge \neg P)) & \text{(de Morgan)} \\
(P \vee (P \wedge \neg Q) \vee P) \wedge (\neg P \vee ((\neg P \vee Q) \wedge \neg P)) & \text{(double negation)} \\
P \wedge (P \vee \neg Q) \wedge (\neg P \vee ((\neg P \vee Q) \wedge \neg P)) & (\vee\text{-absorption, distribution}) \\
P \wedge (P \vee \neg Q) \wedge (\neg P \vee Q) \wedge \neg P & (\vee\text{-absorption, distribution})
\end{array}$$

Once again, now just resolve $\{P\}$ against $\{\neg P\}$.

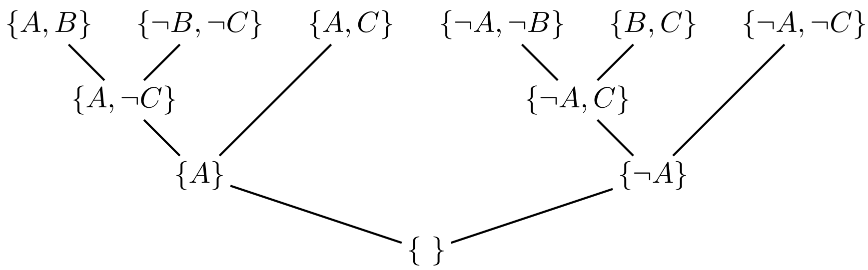
22. (a) $\{\{A, B\}, \{\neg A, \neg B\}, \{\neg A, B\}\}$ stands for the formula $(A \vee B) \wedge (\neg A \vee \neg B) \wedge (\neg A \vee B)$. This is satisfiable by $\{A \mapsto \mathbf{f}, B \mapsto \mathbf{t}\}$.

(b) $\{\{A, \neg B\}, \{\neg A\}, \{B\}\}$ stands for $(A \vee \neg B) \wedge \neg A \wedge B$. A refutation is easy:



(c) $\{\{A\}, \emptyset\}$ stands for $A \wedge \mathbf{f}$, which is clearly not satisfiable.

(d) We have $\{\{A, B\}, \{\neg A, \neg B\}, \{B, C\}, \{\neg B, \neg C\}, \{A, C\}, \{\neg A, \neg C\}\}$. This set is not satisfiable, as a proof by resolution shows:



27. These are the clauses generated:

(a) For each node i generate the clause $B_i \vee G_i \vee R_i$. That comes to $n + 1$ clauses of size 3 each.

(b) For each node i generate three clauses: $(\neg B_i \vee \neg G_i) \wedge (\neg B_i \vee \neg R_i) \wedge (\neg G_i \vee \neg R_i)$. That comes to $3n + 3$ clauses of size 2 each.

(c) For each pair (i, j) of nodes with $i < j$ we want to express $E_{ij} \Rightarrow (\neg(B_i \wedge B_j) \wedge \neg(G_i \wedge G_j) \wedge \neg(R_i \wedge R_j)$. This means for each pair (i, j) we generate three clauses: $(\neg E_{ij} \vee \neg B_i \vee \neg B_j) \wedge (\neg E_{ij} \vee \neg G_i \vee \neg G_j) \wedge (\neg E_{ij} \vee \neg R_i \vee \neg R_j)$. There are $n(n+1)/2$ pairs, so we generate $3n(n+1)/2$ clauses, each of size 3.

Altogether we generate $3n + 3 + 6n + 6 + 9n(n+1)/2$ literals, that is, $9(n+1)(n/2+1)$.