[Tutorial 1 (Propositional Logic) [With solutions]]
This tutorial consists of an assortment of exercises from the slides (Session 1) and an activity of solving a murder mystery puzzle. [Needless to say, you should attempt to do all the exercises from the slides.]

From the lecture slides

[40–45 mins]
First make sure that you are familar with the truth tables for all the connectives (check online if you haven't!). Start by working through the following easy exercises. All the proofs can be done by simply writing down the truth tables.

1. Prove that $A \leftrightarrow B \equiv (A \to B) \wedge (B \to A)$
2. Prove De Morgan's Laws
3. Prove distributivity of $\wedge$
4. Express the all-true formula $\top$ in terms of $\neg, \vee$.
5. Express the all-false formula $\perp$ in terms of $\neg, \wedge$.
6. Prove that $\wedge$ is associative and commutative.
7. Prove Contraposition: $A \to B \equiv \neg B \to \neg A$
8. Prove Modus Ponens: $(A \to B) \wedge A \equiv A \wedge B$

Answer:
2. Prove De Morgan's Laws
2.1 $\neg(A \wedge B) \equiv \neg A \vee \neg B$

A	B	$\neg$	$(A$	$\wedge$	$B)$	$\neg A$	$\vee$	$\neg B$
0	0	**1**		0		1	**1**	1
0	1	**1**		0		1	**1**	0
1	0	**1**		0		0	**1**	1
1	1	**0**		1		0	**0**	0

2.2 $\neg(A \vee B) \equiv \neg A \wedge \neg B$

A	B	$\neg$	$(A$	$\vee$	$B)$	$\neg A$	$\wedge$	$\neg B$
0	0	**1**		0		1	**1**	1
0	1	**0**		1		1	**0**	0
1	0	**0**		1		0	**0**	1
1	1	**0**		1		0	**0**	0

5. Express the all-false formula $\perp$ in terms of $\neg, \wedge$.
We prove that $\neg A \wedge A$ is always false.

A	$\neg$	A	$\wedge$	A
0	1		**0**	
0	1		**0**	
1	0		**0**	
1	0		**0**	

8. Prove Modus Ponens: $(A \to B) \land A \equiv A \land B$

A	B	$(A \to B)$	$\land$	A	A	$\land$	B
0	0	1	**0**			**0**	
0	1	1	**0**			**0**	
1	0	0	**0**			**0**	
1	1	1	**1**			**1**	

Minimum set of logical connectives

The task now is to show that each of the following sets of connectives is sufficient for expressing all propositional formulas:

– $\neg, \land$
– $\neg, \lor$

Explicitly state when you use the substitution principle.
Answer:
1. $\neg, \land$

We prove this by showing that other logic connectives $\lor, \to, \leftrightarrow, \oplus$ can be substituted with only $\neg$ and $\land$.

($\lor$) $A \lor B \equiv \neg(\neg A \land \neg B)$
We start from the equivalence: $C \equiv \neg\neg C$

$$C \equiv \neg\neg C \qquad \text{let } \sigma : C \mapsto A \lor B, \sigma' : C \mapsto A \lor B$$
$$\sigma(C) \equiv \sigma'(C) \text{ trivially holds}$$
$$\Rightarrow \quad A \lor B \equiv \neg\neg(A \lor B) \qquad \cdots (1)$$

$$\neg D = \neg D \qquad \text{let } \sigma : D \mapsto \neg(A \lor B), \sigma' : D \mapsto \neg A \land \neg B$$
$$\sigma(D) \equiv \sigma'(D) \text{ by De Morgan's Laws}$$
$$\Rightarrow \quad \neg\neg(A \lor B) \equiv \neg(\neg A \land \neg B) \qquad \cdots (2)$$

By (1), (2)
$$\Rightarrow \quad A \lor B \equiv \neg\neg(A \lor B) \equiv \neg(\neg A \land \neg B)$$

($\to$) $A \to B \equiv \neg(A \land \neg B)$
We start from the equivalence just proved: $C \lor B \equiv \neg(\neg C \land \neg B)$

$$C \lor B \equiv \neg(\neg C \land \neg B) \qquad \text{let } \sigma : C \mapsto \neg A, \sigma' : C \mapsto \neg A$$
$$\sigma(C) \equiv \sigma'(C) \text{ trivially holds}$$
$$\Rightarrow \quad \neg A \lor B \equiv \neg(\neg\neg A \land \neg B) \qquad \cdots (3)$$

$$\neg(E \land \neg B) \equiv \neg(E \land \neg B) \qquad \text{let } \sigma : E \mapsto \neg\neg A, \sigma' : E \mapsto A$$
$$\sigma(E) \equiv \sigma'(E) \because \neg\neg A \equiv A$$
$$\Rightarrow \quad \neg(\neg\neg A \land \neg B) \equiv \neg(A \land \neg B) \cdots (4)$$

By (3), (4), and $A \to B \equiv \neg A \lor B$
$$\Rightarrow \quad A \to B \equiv \neg A \lor B \equiv \neg(\neg\neg A \land \neg B) \equiv \neg(A \land \neg B)$$

$(\leftrightarrow)$ $A \leftrightarrow B \equiv \neg(A \wedge \neg B) \wedge \neg(B \wedge \neg A)$

We start from trivial equivalence: $C \wedge D \equiv C \wedge D$

$$
\begin{array}{lll}
 & C \wedge D \equiv C \wedge D & \text{let } \sigma : C \mapsto (A \to B), \sigma' : C \mapsto \neg(A \wedge \neg B) \\
 & & \sigma(C) \equiv \sigma'(C) \text{ holds by } (\to) \\
\Rightarrow & (A \to B) \wedge D \equiv \neg(A \wedge \neg B) \wedge D & \text{let } D \mapsto (B \to A), \sigma' : D \mapsto \neg(B \wedge \neg A) \\
 & & \sigma(D) \equiv \sigma'(D) \text{ holds by } (\to) \\
 & & \\
\Rightarrow & (A \to B) \wedge (B \to A) \equiv \neg(A \wedge \neg B) \wedge \neg(B \wedge \neg A) \cdots (5) & \\
 & & \\
 & \text{By (5) and } A \leftrightarrow B \equiv (A \to B) \wedge (B \to A) & \\
\Rightarrow & A \leftrightarrow B \equiv \neg(A \wedge \neg B) \wedge \neg(B \wedge \neg A) &
\end{array}
$$

$(\oplus)$ $A \oplus B \equiv \neg(\neg(A \wedge \neg B) \wedge \neg(B \wedge \neg A))$

We start from trivial equivalence: $\neg C \equiv \neg C$

$$
\begin{array}{lll}
 & \neg C \equiv \neg C & \text{let } \sigma : C \mapsto (A \leftrightarrow B), \sigma' : C \mapsto \neg(A \wedge \neg B) \wedge \neg(B \wedge \neg A) \\
 & & \sigma(C) \equiv \sigma'(C) \text{ holds by } (\leftrightarrow) \\
\Rightarrow & \neg(A \leftrightarrow B) \equiv \neg(\neg(A \wedge \neg B) \wedge \neg(B \wedge \neg A)) \cdots (6) & \\
 & & \\
 & \text{By (6) and } A \oplus B \quad \equiv \neg(A \leftrightarrow B) & \\
\Rightarrow & A \oplus B \equiv \neg(\neg(A \wedge \neg B) \wedge \neg(B \wedge \neg A)) &
\end{array}
$$

2. $\neg, \vee$ From previous proof, we already knew that we can construct all logical connectives with only $\neg$ and $\wedge$. Hence, we only have to prove that $\wedge$ can be substituted by $\neg$ and $\vee$.

$(\wedge)$ $A \wedge B \equiv \neg(\neg A \vee \neg B)$

We start from the equivalence: $C \equiv \neg\neg C$

$$
\begin{array}{llll}
 & C \equiv \neg\neg C & & \text{let } \sigma : C \mapsto A \wedge B, \sigma' : C \mapsto A \wedge B \\
 & & & \sigma(C) \equiv \sigma'(C) \text{ trivially holds} \\
\Rightarrow & A \wedge B \equiv \neg\neg(A \wedge B) & \cdots (1) & \\
 & & & \\
 & \neg D = \neg D & & \text{let } \sigma : D \mapsto \neg(A \wedge B), \sigma' : D \mapsto \neg A \vee \neg B \\
 & & & \sigma(D) \equiv \sigma'(D) \text{ by De Morgan's Laws} \\
\Rightarrow & \neg\neg(A \wedge B) \equiv \neg(\neg A \vee \neg B) & \cdots (2) & \\
 & & & \\
 & \text{By (1), (2)} & & \\
\Rightarrow & A \wedge B \equiv \neg\neg(A \wedge B) \equiv \neg(\neg A \vee \neg B) & &
\end{array}
$$

Validity/satisfiability

1. Argue that the following formula is valid.

$$((\text{Eat} \to \neg\text{Starve}) \wedge \text{Eat}) \to \neg\text{Starve}$$

2. Prove satisfiability, and disprove validity of:

$$((\text{Eat} \to \neg\text{Starve}) \wedge \neg\text{Starve}) \to \text{Eat}$$

[After you've completed the murder mystery riddle (next activity), at home you may try to formalise and prove validity of: "If Eric studies, he does not fail exams. If Eric does not play too often, he studies. Eric fails exams. Thus, Eric plays too often."]
Answer.

1. $(((\text{Eat} \to \neg\text{Starve}) \wedge \text{Eat}) \to \neg\text{Starve})$ is valid

$$\begin{array}{lll} & ((\text{Eat} \to \neg\text{Starve}) \wedge \text{Eat}) \to \neg\text{Starve} & \\ \equiv & (\text{Eat} \wedge \neg\text{Starve}) \to \neg\text{Starve} & \text{by Modus Ponens} \\ \equiv & \neg(\text{Eat} \wedge \neg\text{Starve}) \vee \neg\text{Starve} & \text{by } A \to B \equiv \neg A \vee B \\ \equiv & (\neg\text{Eat} \vee \neg\neg\text{Starve}) \vee \neg\text{Starve} & \text{by De Morgan's Laws} \\ \equiv & \neg\text{Eat} \vee (\text{Starve} \vee \neg\text{Starve}) & \text{by Associativity of } \vee \text{ and } A \equiv \neg\neg A \\ \equiv & \neg\text{Eat} \vee \top & \text{by } A \vee \neg A \equiv \top \\ \equiv & \top & \text{by } A \vee \top \equiv \top \end{array}$$

2. $((\text{Eat} \to \neg\text{Starve}) \wedge \neg\text{Starve}) \to \text{Eat}$ is satisfiable but not valid.
The formula is evaluated to 1 when Starve $= 1$ and Eat $= 0$; hence, it's satisfiable.
The formula is evaluated to 0 when Starve $= 0$ and Eat $= 0$; hence, it's not valid.

3. Eric plays too often.
We construct following formulae to formalise the premises

$$\begin{array}{rl} Premise := & Study \to \neg Fail \\ \wedge & \neg PlayOften \to Study \\ \wedge & Fail \end{array}$$

The goal is to show $(Premise \to PlayOften)$ is a valid conjecture,
and $(Premise \to PlayOften)$ is valid iff $\neg(Premise \to PlayOften)$ is unsatisfiable.

$$\begin{array}{ll} & \neg(Premise \to PlayOften) \\ \equiv & \neg(\neg Premise \vee PlayOften) \\ \equiv & Premise \wedge \neg PlayOften \end{array}$$

We therefore check if $Premise \wedge \neg PlayOften$ is unsatisfiable.
The solver returns UNSAT; hence the conjecture is valid.
Eric indeed plays too often.

Solving a murder mystery riddle with logic

[40–45 mins]
In this activity, we are going to apply *formal* logic to solve a murder mystery riddle.

Learning goals:
- Recognizing logic signals in natural language.
- Applying *formal* logic to systematically solve the puzzle.

A warm up exercise

[10–15 mins]
Let's first solve something boring, but will prepare you for the murder mystery riddle:

Let's play a game. Flip a coin. Heads, I win. Tails, you lose. (*)

Your task is to argue that I always win.

Step 1. Carefully formulate the stated facts in propositional logic. Firstly, you must carefully pick your propositions. Here are the possible propositions:

– H — head
– T — tail
– I — I win
– Y — you win

Task: Write a propositional formula F for (*)

Answer:

$$H \longrightarrow I \wedge T \longrightarrow \neg Y.$$

Step 2. Clearly state some reasonable assumptions that arise from ambiguity of natural language. Here they are:

(A1) $H \oplus T$ (or equivalently, $H \rightarrow \neg T \wedge \neg H \rightarrow T$)
(A2) $I \oplus Y$ (or equivalently, $I \rightarrow \neg Y \wedge \neg I \rightarrow Y$)

Task 3. Formally argue that I holds, assuming F is true. [Hint: $H \vee \neg H$ is valid and analyse the two cases separately.]

To argue that I is true, we consider the two cases H and $\neg H$ separately.

Case 1: $H = 1$. In this case, the first conjunct of F implies that I.

Case 2: $\neg H = 1$ (i.e. $H = 0$). Then, **(A1)** implies that $T = 1$. Therefore, the second conjunct of F implies that $\neg Y$. By **(A2)**, it follows that I.

Conclusion: all possible cases lead to I. Therefore, I is true.

Task 4. Construct a truth table to confirm this. You can automate this using the propositional formulas solver:

`http://logictools.org/index.html`

Simply enter your formula and inspect that the formula is only true for the rows that $I = 1$.

Alternatively, you can show this without using truth table. Simply prove that $F \wedge \neg I$ is unsatisfiable (i.e. all rows in the truth table are 0). This can be done by entering this formula in the formula box of the solver and make sure that you get "Clause set is false for all possible assignments to variables." in the Result box. [Note: Set "using" to `dpll:old` (since `dpll:better` seems to have a bug).]

Answer: Simply put the following line into the formula box:

```
(H -> I) & (T -> -Y) & (H xor T) & (I xor Y) & -I
```

The murder mystery riddle

[30–35 mins]

There are three suspects for a murder: Adam, Brown, and Clark. It has been established beyond any reasonable doubt that exactly one of them is the killer. [So, two are innocents, one is guilty.] Your task is to figure out which one. You questioned Adam, Brown, and Clark one-by-one. Adam says "I didn't do it. The victim was old acquaintance of Brown's. But Clark hated him." Brown states "I didn't do it. I didn't know the guy. Besides I was out of town all week." Finally, Clark says "I didn't do it. I saw both Adam and Brown downtown with the victim that day; one of them must have done it." Assuming that the innocent men are telling the truth, but that the guilty man might not be, discover the killer.

Task 1. Carefully formulate the stated facts in propositional logic. Firstly, you must carefully pick your propositions. To get you started, here are some possible propositions:

- $I(A)$ — Adam is innocent.
- $H(C, V)$ — Clark hates the victim
- $T(B)$ — Brown was in town on the day of the murder
- $W(B, V)$ — Brown was with the victim on the day of the murder

Solution: Here are all the propositions:

- $I(A)$, $I(B)$, $I(C)$ to denote whether Adam, Brown, and Clark (respectively) is innocent.
- $F(B, V)$ — B is a friend (acquaintance) of V.
- $H(C, V)$ — Clark hates the victim
- $T(B)$ — Brown was in town on the day of the murder
- $W(A, V)$ — Adam was with the victim on the day of the murder
- $W(B, V)$ — Brown was with the victim on the day of the murder

– $K(B, V)$ — Brown knows the victim

Here are the formalised statements from the puzzle:

1.1 $I(A) \to F(B, V)$
1.2 $I(A) \to H(C, V)$
2.1 $I(B) \to \neg T(B)$
2.2 $I(B) \to \neg K(B, V)$
3.1 $I(C) \to W(A, V)$
3.2 $I(C) \to W(B, V)$
4 $(I(A) \wedge I(B) \wedge \neg I(C)) \vee (I(A) \wedge \neg I(B) \wedge I(C)) \vee (\neg I(A) \wedge I(B) \wedge I(C))$

Note that (4) simply state that two men are innocent and one is guilty.

Task 2. Clearly state some reasonable assumptions that arise from the semantics of English language. Here is an example:

(X) $W(B, V) \longrightarrow T(B)$, i.e., Brown being with the victim on the day of murder means that Brown was in town on the day of the murder.

Answer: the other assumption that we need to make is:

$$F(B, V) \to K(B, V) \tag{Y}$$

This makes sense since if Brown is a friend of the victim, then he must know the victim.

Task 3. Make a conjecture who is the killer and formally argue that this is the case.

Solution: The killer is Brown. Here is a proof:

5 $I(A) \to K(B, V)$ by (1.1) and (Y)
6 $I(C) \to T(B)$ by (3.2) and (X)
7 $K(B, V) \to \neg I(B)$ — contrapositive of (2.2)
8 $I(A) \to \neg I(B)$ by (5) and (7)
9 $T(B) \to \neg I(B)$ — contrapositive of (2.1)
10 $I(C) \to \neg I(B)$ by (6) and (9)

Consider the three cases (i.e. disjuncts) of (4). For the first and the third disjunct (with Adam guilty and Clark guilty, respectively), item (8) and (9) will, respectively, yield $I(B) \wedge \neg I(B)$, an impossibility. So, the only possibility is the case

$$I(A) \wedge \neg I(B) \wedge I(C)$$

proving that Brown is guilty.

Task 4. Confirm this with the propositional formulas solver:

`http://logictools.org/index.html`

Set "using" to `dpll:old` (since `dpll:better` seems to have a bug).
Solution: copy and paste the following long formula:

```
(IA -> FBV) & (IA -> HCV) & (IB -> -TB) & (IB -> -KBV) & (IC -> WAV) &
(IC -> WBV) & ((IA & IB & -IC) or (IA & -IB & IC) or (-IA & IB & IC)) &
(WBV -> TB) & (FBV -> KBV) & IB
```

Of course, we put `IB` for the same reason as the warm-up exercise.