COMP30026 Models of Computation

Logic Concepts

Harald Søndergaard

Lecture Week 2 Part 2 (Zoom)

Semester 2, 2020

This Lecture is Being Recorded

From the Break: Knights and Knaves Puzzle

On the island of Knights and Knaves, everyone is a knight or knave. Knights always tell the truth. Knaves always lie.

Today there is a census on the island!

You are a census taker, going from house to house. Fill in what you know about each of these three houses.

- **In house 1:** Husband: We are both knaves.

From the Break: Knights and Knaves Puzzle

On the island of Knights and Knaves, everyone is a knight or knave. Knights always tell the truth. Knaves always lie.

Today there is a census on the island!

You are a census taker, going from house to house. Fill in what you know about each of these three houses.

- **In house 1:** Husband: We are both knaves.
- **In house 2:** Wife: At least one of us is a knave.

From the Break: Knights and Knaves Puzzle

On the island of Knights and Knaves, everyone is a knight or knave. Knights always tell the truth. Knaves always lie.

Today there is a census on the island!

You are a census taker, going from house to house. Fill in what you know about each of these three houses.

- **In house 1:** Husband: We are both knaves.
- **In house 2:** Wife: At least one of us is a knave.
- **In house 3:** Husband: If I am a knight then so is my wife.

From the Break: Knights and Knaves Puzzle

On the island of Knights and Knaves, everyone is a knight or knave. Knights always tell the truth. Knaves always lie.

Today there is a census on the island!

You are a census taker, going from house to house. Fill in what you know about each of these three houses.

- **In house 1:** Husband: We are both knaves.
- **In house 2:** Wife: At least one of us is a knave.
- **In house 3:** Husband: If I am a knight then so is my wife.

If you like these puzzles, Raymond Smullyan has written several books that you will like.

Logic and Computer Science

Before we get to more sophisticated logic and its applications, let us establish a vocabulary and some important concepts.

> *"The relationship between computation and mathematical logic will be as fruitful in the next century as that between analysis and physics in the last."*
>
> *John McCarthy*

Validity and Satisfiability

A propositional formula is valid if no truth assignment makes it false. Otherwise it is non-valid.

It is unsatisfiable if no truth assignment makes it true. Otherwise it is satisfiable.

A valid propositional formula is a tautology.

An unsatisfiable propositional formula is a contradiction.

Tautology Example

$(\neg P \land Q) \Rightarrow (P \Rightarrow R)$ is valid:

P	Q	R	$\neg P \land Q$	$P \Rightarrow R$	$(\neg P \land Q) \Rightarrow (P \Rightarrow R)$
f	**f**	**f**	**f**	**t**	**t**
f	**f**	**t**	**f**	**t**	**t**
f	**t**	**f**	**t**	**t**	**t**
f	**t**	**t**	**t**	**t**	**t**
t	**f**	**f**	**f**	**f**	**t**
t	**f**	**t**	**f**	**t**	**t**
t	**t**	**f**	**f**	**f**	**t**
t	**t**	**t**	**f**	**t**	**t**

Logically, "Valid" Means Vacuous

In ordinary discourse, we may praise somebody by saying "the point you make is valid."

But in formal logic, a valid statement is not that laudable; in a sense it is void of information. For example, $A \Rightarrow A$ is a vacuous statement, and valid.

"If Trump is sane then Trump is sane" tells us nothing about whether Trump is sane—the statement is true whatever the case may be.

You don't even have to know who Trump is, or what it means to be sane, in order to agree: The statement is inherently true.

Validity Checking in Haskell

Given a truth table for a proposition P, it is easy to check if P is valid: The conjunction of all the entries in P's column must be true.

To check the validity of 3-place Haskell predicate:

```
valid3 :: (Bool -> Bool -> Bool -> Bool) -> Bool
valid3 f
  = and [f p q r | p <- bs, q <- bs, r <- bs]
    where
      bs = [False, True]
```

Validity Checking in Haskell

Given a truth table for a proposition P, it is easy to check if P is valid: The conjunction of all the entries in P's column must be true.

To check the validity of 3-place Haskell predicate:

```
valid3 :: (Bool -> Bool -> Bool -> Bool) -> Bool
valid3 f
  = and [f p q r | p <- bs, q <- bs, r <- bs]
    where
      bs = [False, True]
```

Poll 2: What does a function to check satisfiability look like?

Contradiction Example

$P \land Q \land (\neg Q \Leftrightarrow (\neg P \lor Q))$ is unsatisfiable.

Again, we can just complete the truth table. However, it is often possible to apply faster reasoning.

In this case, P and Q are conjuncts of the formula, so if a truth assignment maps either to **f**, the formula evaluates to **f**.

And if P and Q are both mapped to **t**, the third conjunct becomes $(\neg \mathbf{t} \Leftrightarrow (\neg \mathbf{t} \lor \mathbf{t}))$, which again evaluates to **f**.

Substitution Preserves Validity + Unsatisfiability

Validity is preserved by substitution of propositional letters by formulas.

We saw that $(\neg P \land Q) \Rightarrow (P \Rightarrow R)$ is valid, and hence

$$(\neg(A \lor B) \land (B \Leftrightarrow C)) \Rightarrow ((A \lor B) \Rightarrow (B \Leftrightarrow C))$$

is valid.

(Different letters can be replaced by the same formula, but of course, all occurrences of a letter must be replaced by the same formula.)

A formula is unsatisfiable iff its negation is valid.

It follows that substitution also preserves unsatisfiability.

Poll 3

Does substitution preserve satisfiability?

Models, Logical Consequence, and Equivalence

Let θ be a truth assignment and Φ be a propositional formula. If θ makes Φ true then θ is a model of Φ.

Ψ is a logical consequence of Φ iff every model of Φ is a model of Ψ as well.

In that case we write $\boxed{\Phi \models \Psi}$

If $\Phi \models \Psi$ and $\Psi \models \Phi$ both hold, that is, Φ and Ψ have exactly the same models, then Φ and Ψ are (logically) equivalent.

In that case we write $\boxed{\Phi \equiv \Psi}$

Poll 4

Of the following statements, which allow us to conclude $P \Rightarrow Q$?

- P
- $\neg P$
- Q
- $P \Rightarrow (Q \wedge R)$
- $(P \vee R) \Rightarrow Q$
- $\neg P \vee Q$
- $\neg Q \Rightarrow \neg P$
- $P \Rightarrow (Q \vee R)$
- $(P \Rightarrow Q) \vee R$

Substitution Preserves Logical Equivalence

If $\Phi \equiv \Psi$ and Φ' and Ψ' are the results of replacing each occurrence of letter P (in both) with formula Υ, then $\Phi' \equiv \Psi'$.

Interchange of Equivalents

Replacing equals by equals yields equals. If

1. Φ is a sub-formula of Υ,
2. $\Phi \equiv \Psi$, and
3. Υ' is the result of replacing Φ in Υ by Ψ,

then $\Upsilon \equiv \Upsilon'$.

Interchange of equivalents preserves not only logical equivalence.

It also preserves logical consequence, validity, and unsatisfiability.

Unlike substitution, it even preserves satisfiability.

Some Equivalences

Absorption:	$P \land P \equiv P$
	$P \lor P \equiv P$
Commutativity:	$P \land Q \equiv Q \land P$
	$P \lor Q \equiv Q \lor P$
Associativity:	$P \land (Q \land R) \equiv (P \land Q) \land R$
	$P \lor (Q \lor R) \equiv (P \lor Q) \lor R$
Distributivity:	$P \land (Q \lor R) \equiv (P \land Q) \lor (P \land R)$
	$P \lor (Q \land R) \equiv (P \lor Q) \land (P \lor R)$

More Equivalences

$\Leftrightarrow$ and $\oplus$ are also commutative and associative.

Double negation:	$P \equiv \neg\neg P$
De Morgan:	$\neg(P \land Q) \equiv \neg P \lor \neg Q$
	$\neg(P \lor Q) \equiv \neg P \land \neg Q$
Implication:	$P \Rightarrow Q \equiv \neg P \lor Q$
Contraposition:	$\neg P \Rightarrow \neg Q \equiv Q \Rightarrow P$
	$P \Rightarrow \neg Q \equiv Q \Rightarrow \neg P$
	$\neg P \Rightarrow Q \equiv \neg Q \Rightarrow P$
Biimplication:	$P \Leftrightarrow Q \equiv (P \land Q) \lor (\neg P \land \neg Q)$

Quiz

Find an alternative way of writing biimplication—as a conjunction.

Last Equivalences

Let $\bot$ be any unsatisfiable formula and let $\top$ be any valid formula.

Duality:	$\neg\top \equiv \bot$ $\neg\bot \equiv \top$
Negation from absurdity:	$P \Rightarrow \bot \equiv \neg P$
Identity:	$P \lor \bot \equiv P$ $P \land \top \equiv P$
Dominance:	$P \land \bot \equiv \bot$ $P \lor \top \equiv \top$
Contradiction:	$P \land \neg P \equiv \bot$
Excluded middle:	$P \lor \neg P \equiv \top$

Poll 5

Which of these claims hold?

1. $P \Rightarrow Q \equiv (Q \Leftrightarrow (P \vee Q))$
2. $(P \Rightarrow Q) \wedge (P \Rightarrow R) \equiv P \Rightarrow (Q \wedge R)$
3. $(P \Rightarrow R) \wedge (Q \Rightarrow R) \models (P \wedge Q) \Rightarrow R$

Next Week Same Time Tune In

Learn how symbolic manipulation beats truth tables.

We shall be . . .

$\boxed{\text{Mechanising deduction based on propositional logic}}$