CMPU-145: Foundations of Computer Science Spring, 2019

Chapter 8: Propositional Logic

CMPU 145 – Foundations of Computer Science
Peter Lemieszewski

Propositional Logic

- Some charts from last week…
  - Plus a few exemplar boolean expressions.

MoreTautological implication (from Makinson)

Table 8.5 Some important tautological implications

Name	LHS	RHS
Simplification, $\wedge^-$	$\alpha\wedge\beta$	α
	$\alpha\wedge\beta$	β
Conjunction, $\wedge^+$	α, β	$\alpha\wedge\beta$
Disjunction, $\vee^+$	α	$\alpha\vee\beta$
	β	$\alpha\vee\beta$
Modus ponens, MP, $\rightarrow^-$	$\alpha, \alpha\rightarrow\beta$	β
Modus tollens, MT	$\neg\beta, \alpha\rightarrow\beta$	$\neg\alpha$
Disjunctive syllogism, DS	$\alpha\vee\beta, \neg\alpha$	β
Transitivity	$\alpha\rightarrow\beta, \beta\rightarrow\gamma$	$\alpha\rightarrow\gamma$
Material implication	β	$\alpha\rightarrow\beta$
	$\neg\alpha$	$\alpha\rightarrow\beta$
Limiting cases	γ	$\beta\vee\neg\beta$
	$\alpha\wedge\neg\alpha$	γ

Tautological Equivalence

- When each of two formulae tautologically implies the other, we have a tautological equivalence.

One might see such substitutions when a compiler optimizes some code

Table 8.7 Some important tautological equivalences for $\neg, \wedge, \vee$

Name	LHS	RHS
Double negation	α	$\neg\neg\alpha$
Commutation for $\wedge$	$\alpha\wedge\beta$	$\beta\wedge\alpha$
Association for $\wedge$	$\alpha\wedge(\beta\wedge\gamma)$	$(\alpha\wedge\beta)\wedge\gamma$
Commutation for $\vee$	$\alpha\vee\beta$	$\beta\vee\alpha$
Association for $\vee$	$\alpha\vee(\beta\vee\gamma)$	$(\alpha\vee\beta)\vee\gamma$
Distribution of $\wedge$ over $\vee$	$\alpha\wedge(\beta\vee\gamma)$	$(\alpha\wedge\beta)\vee(\alpha\wedge\gamma)$
Distribution of $\vee$ over $\wedge$	$\alpha\vee(\beta\wedge\gamma)$	$(\alpha\vee\beta)\wedge(\alpha\vee\gamma)$
Absorption	α	$\alpha\wedge(\alpha\vee\beta)$
	α	$\alpha\vee(\alpha\wedge\beta)$
Expansion	α	$(\alpha\wedge\beta)\vee(\alpha\wedge\neg\beta)$
	α	$(\alpha\vee\beta)\wedge(\alpha\vee\neg\beta)$
de Morgan	$\neg(\alpha\wedge\beta)$	$\neg\alpha\vee\neg\beta$
	$\neg(\alpha\vee\beta)$	$\neg\alpha\wedge\neg\beta$
	$\alpha\wedge\beta$	$\neg(\neg\alpha\vee\neg\beta)$
	$\alpha\vee\beta$	$\neg(\neg\alpha\wedge\neg\beta)$
Limiting cases	$\alpha\wedge\neg\alpha$	$\beta\wedge\neg\beta$
	$\alpha\vee\neg\alpha$	$\beta\vee\neg\beta$

A closer examination of de Morgan's Equivalences

Makinson refers to systematic correspondence (i.e. they are analogous) between
tuatological equivalence and
boolean identities between sets

Arguably, the most important are de Morgan's equivalences. From set theory (p20)

- $-\cup_{i\in I} \{B_i\} = \cap_{i\in I} \{-B_i\}$ or $-(A \cup B) = -A \cap -B$
- $-\cap_{i\in I} \{B_i\} = \cup_{i\in I} \{-B_i\}$ or $-(A \cap B) = -A \cup -B$

And, restated, using English alphabet.

- ¬(p v q) = ¬p ^ ¬q
- ¬(p ^ q) = ¬p v ¬q

- We can use these equivalences to simplify boolean expressions ourselves.

de Morgan	$\neg(\alpha\wedge\beta)$	$\neg\alpha\vee\neg\beta$
	$\neg(\alpha\vee\beta)$	$\neg\alpha\wedge\neg\beta$
	$\alpha\wedge\beta$	$\neg(\neg\alpha\vee\neg\beta)$
	$\alpha\vee\beta$	$\neg(\neg\alpha\wedge\neg\beta)$

Simplifying Boolean Expressions: Cheat Sheet

Or

```
p v p  == p
p v ¬p == 1
p v 1  == 1
p v 0  == p
p v q  == q v p
(pvq)v r == p v (q v r)
```

And

```
q ^ q  == q
q ^ ¬q == 0
q ^ 1  == q
q ^ 0  == 0
p ^ q  == q ^ p
p^(q^r) == (p^q)^r
```

With de Morgan (again)

```
¬(p v q)     == ¬p ^ ¬q
¬(p ^ q)     == ¬p v ¬q
```

Double negation

```
¬ ¬ p    == p      (double negation)
```

Absorption (*3)

```
P^(p v q) == p
p v (p^q) == p
p v (¬p^q)== p v q
```


Simplifying Boolean Expressions: 2 Examples

1. **p^q^r v ¬p v p^¬q^r == ?**
2. **p^q v p^(q v r) v q^(q v r) == ?**

- Imagine if we were assigned the task of optimizing expressions like this as part of a compiler.
- We can apply Table 8.7 (or our cheat sheet!) to simplify these expressions.

Simplifying Boolean Expressions: Example 1

```
1. p^q^r v ¬p v p^¬q^r      == (associativity)
2. p^q^r v p^¬q^r v ¬p      == (distrib. of ^ over v. . .)
3. p^r^(q v ¬q) v ¬p        == (. . . from right to left)
4. p^r^(1) v ¬p             == (p v ¬p == 1)
5. p^r v ¬p                 == (p^1 == p)
6. ¬p v p^r                 == (associativity)
7. ¬¬p v ¬p^r               == (replace p with (¬p) to see…)
8. ¬p v r (answer)          == (another form of absorption rule)
```

Simplifying Boolean Expressions: Example 2

```
1. p^q v p^(q v r) v q^(q v r) == (distribution of ^ over v)
2. p^q v p^q v p^r v q^q v q^r == (left to right this time!)
3. p^q v p^q v p^r v q^q v q^r == (p ^ p == p)
4. p^q v p^q v p^r v q v q^r   == (p v p == p)
5. p^q v p^r v q v q^r         == (de Morgan)
6. p^q v p^r v q               == (associativity)
7. q v p^q v p^r               == (de Morgan)
8. q v p^r                     == (answer)
```


Definitions: Also From Last Week

Additional Let R be any formula. Then:

- R is a tautology iff `v(R) = 1` for every valuation `v`, i.e. iff every row of R's truth table is 1 (always TRUE).
- R is a contradiction iff `v(R) = 0` for every valuation `v`, i.e. iff Every row of of R's truth table is 0 (always FALSE).
- R is contingent iff it is neither a tautology nor a contradiction. , i.e. iff Some rows of R's truth table is 1 and some rows of R's truth table is 0.
  - (i.e. Sometimes TRUE, sometimes FALSE, depending on input).

Tautology or Contradiction or Contingent?

```
a. p v ¬p
b. ¬(p v ¬p)
c. p v ¬q
d. ¬(p v ¬q)
e. (p^(¬p v q)) -> q
f. ¬(p v q) <-> (¬p ^ ¬q)
g. p ^ ¬p
h. p -> ¬p
i. p <-> ¬p
j. (r^s) v ¬(r^s)
k. (r->s) <-> ¬(r->s)
```


Tautology or Contradiction or Contingent?

```
a. p v ¬p
b. ¬(p v ¬p)
c. p v ¬q
d. ¬(p v ¬q)
e. (p^(¬p v q)) -> q
f. ¬(p v q) <-> (¬p ^ ¬q)
g. p ^ ¬p
h. p -> ¬p
i. p <-> ¬p
j. (r^s) v ¬(r^s)
k. (r->s) <-> ¬(r->s)
```

Tautology or Contradiction or Contingent?

a. p v ¬p
b. ¬(p v ¬p)
c. p v ¬q
d. ¬(p v ¬q)
e. (p^(¬p v q)) -> q
f. ¬(p v q) <-> (¬p ^ ¬q)
g. p ^ ¬p
h. p -> ¬p
i. p <-> ¬p
j. (r^s) v ¬(r^s)
k. (r->s) <-> ¬(r->s)

Tautology or Contradiction or Contingent?

a. p v ¬p
b. ¬(p v ¬p)
c. p v ¬q
d. ¬(p v ¬q)
e. (p^(¬p v q)) -> q
f. ¬(p v q) <-> (¬p ^ ¬q)
g. p ^ ¬p
h. p -> ¬p
i. p <-> ¬p
j. (r^s) v ¬(r^s)
k. (r->s) <-> ¬(r->s)

Why Repeat This Stuff?

- On the final exam,
  - There is one question on simplifying boolean expressions
  - There is another question on identifying boolean expressions as a Tautology or a Contradiction or a Contingent.