Informatics 1

Computation and Logic

Boolean Algebra CNF DNF

Michael Fourman

Basic Boolean operations

Boole (1815 – 1864)

$\mathbf{1}, \top$	true, top
$\vee$	disjunction, or
$\wedge$	conjunction, and
$\neg$	negation, not
$\mathbf{0}, \bot$	false, bottom

$$\mathbb{Z}_2 = \{0, 1\}$$

+	0	1
0	0	1
1	1	0

×	0	1
0	0	0
1	0	1

–	
0	0
1	1

$$x \wedge y \equiv xy$$
$$x \vee y \equiv x + y - xy$$
$$\neg x \equiv 1 - x$$

Here, we use arithmetic mod 2
The same equations work if we use ordinary arithmetic!

∨	0	1
0	0	1
1	1	1

∧	0	1
0	0	0
1	0	1

¬	
0	1
1	0

3

If we use one bit (binary unit, 0 or 1) to code each truth value, with
0 ~ ⊥ and 1 ~ ⊤
then we can express the logical operations algebraically.
It doesn't matter whether we interpret these expressions in Z or in Z_2

The algebra of sets

$$\mathcal{P}(S) = \{X \mid X \subseteq S\}$$

$X \vee Y = X \cup Y$	union
$X \wedge Y = X \cap Y$	intersection
$\neg X = S \setminus Y$	complement
$0 = \emptyset$	empty set
$1 = S$	entire set

4

Derived Operations

Definitions:

$$x \to y \equiv \neg x \vee y \qquad \text{implication}$$
$$x \leftarrow y \equiv x \vee \neg y$$
$$x \leftrightarrow y \equiv (\neg x \wedge \neg y) \vee (x \wedge y) \qquad \text{equality (iff)}$$
$$x \oplus y \equiv (\neg x \wedge y) \vee (x \wedge \neg y) \qquad \text{inequality (xor)}$$

Some equations:

$$x \leftrightarrow y = (x \to y) \wedge (x \leftarrow y)$$
$$x \oplus y = \neg(x \leftrightarrow y)$$
$$x \oplus y = \neg x \oplus \neg y$$
$$x \leftrightarrow y = \neg(x \oplus y)$$
$$x \leftrightarrow y = \neg x \leftrightarrow \neg y$$

an algebraic proof

$$\begin{aligned} (x \leftrightarrow y) \leftrightarrow z &= \neg(x \leftrightarrow y) \leftrightarrow \neg z && (a \leftrightarrow b = \neg a \leftrightarrow \neg b) \\ &= (x \oplus y) \leftrightarrow \neg z && (\neg(a \leftrightarrow b) = a \oplus b) \\ &= (x \oplus y) \oplus z && (a \leftrightarrow \neg b = a \oplus b) \end{aligned}$$

Once we know the rules for iff and xor shown on the right, we can give an algebraic proof that the xor combination of three variables is the same as their iff combination

Boolean connectives

Some equalities:

$$x \vee y = \neg(\neg x \wedge \neg y) \qquad x \wedge y = \neg(\neg x \vee \neg y)$$
$$\neg x = x \to 0 \qquad x \vee y = \neg x \to y$$

We will see that $\wedge$, $\vee$, $\neg$ and $\perp$ are sufficient to define any boolean function. These equations show that $\{\wedge, \neg, \perp\}$, $\{\vee, \neg, \perp\}$, and $\{\to, \perp\}$ are all sufficient sets.

Boolean Algebra

$x \vee (y \vee z) = (x \vee y) \vee z$	$x \wedge (y \wedge z) = (x \wedge y) \wedge z$	associative
$x \vee (y \wedge z) = (x \vee y) \wedge (x \vee z)$	$x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z)$	distributive
$x \vee y = y \vee x$	$x \wedge y = y \wedge x$	commutative
$x \vee 0 = x$	$x \wedge 1 = x$	identity
$x \vee 1 = 1$	$x \wedge 0 = 0$	annihilation
$x \vee x = x$	$x \wedge x = x$	idempotent
$x \vee \neg x = 1$	$\neg x \wedge x = 0$	complements
$x \vee (x \wedge y) = x$	$x \wedge (x \vee y) = x$	absorbtion
$\neg(x \vee y) = \neg x \wedge \neg y$	$\neg(x \wedge y) = \neg x \vee \neg y$	de Morgan
$\neg\neg x = x$	$x \to y = \neg x \leftarrow \neg y$	

The equations above the gap define a Boolean algebra. Those below the line follow from these.

Exercise 2.1

Which of the following rules are *not* valid for arithmetic?
Which of the rules are *not* valid for arithmetic in Z_2?

$x + (y + z) = (x + y) + z$	$x \times (y \times z) = (x \times y) \times z$	associative
$x + (y \times z) = (x + y) \times (x + z)$	$x \times (y + z) = (x \times y) + (x \times z)$	distributive
$x + y = y + x$	$x \times y = y \times x$	commutative
$x + 0 = x$	$x \times 1 = x$	identity
$x + 1 = 1$	$x \times 0 = x$	annihilation
$x + x = x$	$x \times x = x$	idempotent
$x + (x \times y) = x$	$x + (x \times y) = x$	absorbtion
$x + -x = 1$	$x \times -x = 0$	complements

Exercise 2.4 (for mathematicians)

In any Boolean algebra, define,

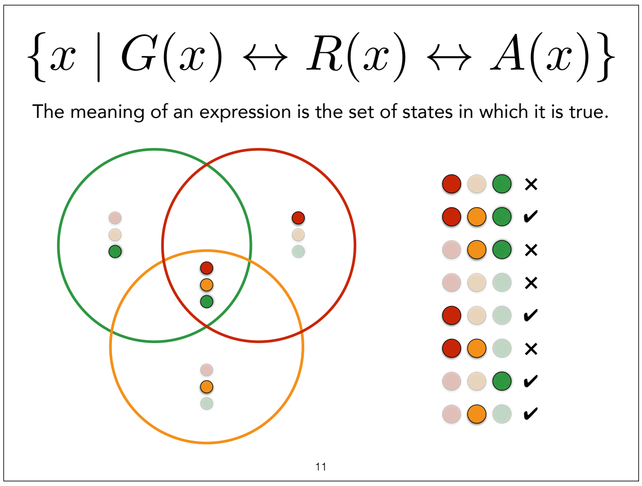
$$x \leq y \equiv x \wedge y = x$$

1. Show that, for any x, y, and z,

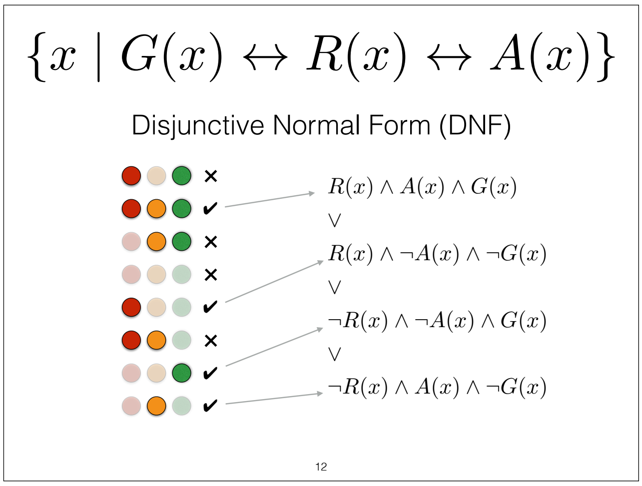
$$\begin{gathered} 0 \leq x \text{ and } x \leq x \text{ and } x \leq 1 \\ x \to y = \top \text{ iff } x \leq y \\ \text{if } x \leq y \text{ and } y \leq z \text{ then } x \leq z \\ \text{if } x \leq y \text{ and } y \leq x \text{ then } x = y \\ \text{if } x \leq y \text{ then } \neg y \leq \neg x \end{gathered}$$

2. Show that, in any Boolean Algebra,

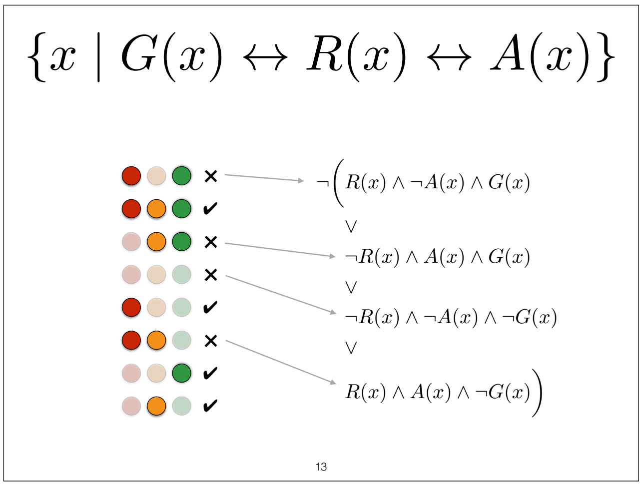
$$x \wedge y = x \text{ iff } x \vee y = y$$



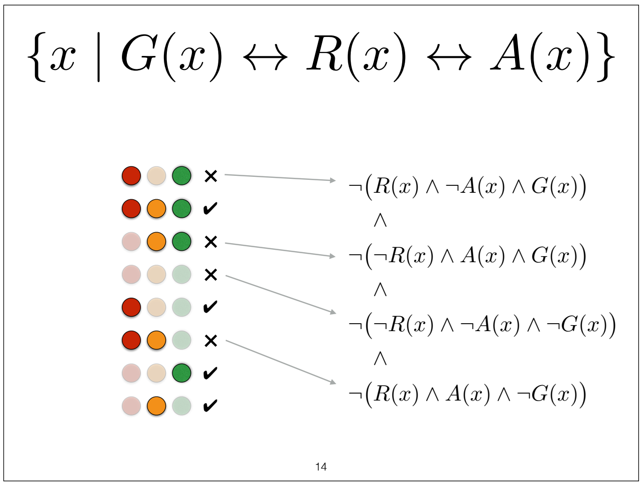
To determine whether to expressions are equivalent, we can check whether they give the same values for all 2^n states of the system.
The meaning of an expression is the set of states in which it is true.



A boolean function of three variables is given by a truth table with eight entries
We can easily write down a disjunction of terms, each one of which corresponds to a single state in which the function is true.



Here we say that we are not in a state where the function is false



Using de Morgan, this becomes a conjunction of negated conjunctions.

$$\{x \mid G(x) \leftrightarrow R(x) \leftrightarrow A(x)\}$$

Conjunctive Normal Form (CNF)

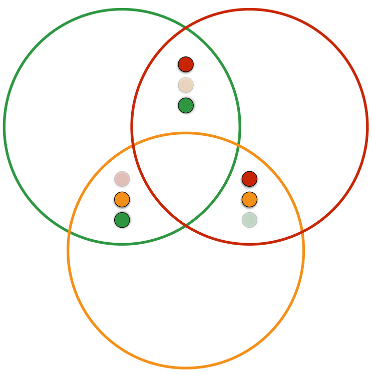
×	$(\neg R(x) \lor A(x) \lor \neg G(x))$
✔	$\land$
×	$(R(x) \lor \neg A(x) \lor \neg G(x))$
×	$\land$
✔	$(R(x) \lor A(x) \lor G(x))$
×	$\land$
✔	$(\neg R(x) \lor \neg A(x) \lor G(x))$
✔	

Using de Morgan again, this becomes a conjunction of disjunctions.

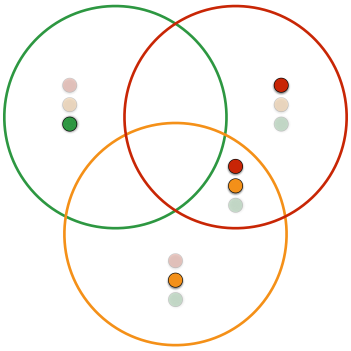
We will return to CNF later.

CNF

Exercise 2.2 Generate CNF for this subset



Exercise 2.3 Generate CNF for this subset



To produce conjunctive normal form (CNF)

eliminate $\longrightarrow$ $\leftrightarrow$

push negations in

push $\vee$ **inside** $\wedge$

$$\neg(a \to b) = a \wedge \neg b \qquad a \to b = \neg a \vee b$$

$$\neg(a \vee b) = \neg a \wedge \neg b \qquad \neg(a \vee b) = \neg a \wedge \neg b$$

$$\neg 0 = 1 \qquad \neg\neg a = a \qquad \neg 1 = 0$$

$$a \vee 1 = 1 \qquad a \wedge 0 = 0$$

$$a \vee 0 = a \qquad a \wedge 1 = a$$

$$a \vee (b \wedge c) = (a \vee b) \wedge (a \vee c) \qquad (a \wedge b) \vee c = (a \vee c) \wedge (b \vee c)$$

(A?B:C)

if A then B else C