COMP2111 Week 3
Term 1, 2019
Propositional Logic II

Summary of topics

- Well-formed formulas
- Boolean Algebras
- Valuations
- CNF/DNF
- Proof
- Natural deduction

Definition: Boolean Algebra

A *Boolean algebra* is a structure $(T, \vee, \wedge, ', 0, 1)$ where

- $0, 1 \in T$
- $\vee : T \times T \to T$ (called **join**)
- $\wedge : T \times T \to T$ (called **meet**)
- $' : T \to T$ (called **complementation**)

and the following laws hold for all $x, y, z \in T$:

commutative:
- $x \vee y = y \vee x$
- $x \wedge y = y \wedge x$

associative:
- $(x \vee y) \vee z = x \vee (y \vee z)$
- $(x \wedge y) \wedge z = x \wedge (y \wedge z)$

distributive:
- $x \vee (y \wedge z) = (x \vee y) \wedge (x \vee z)$
- $x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z)$

identity: $x \vee 0 = x, \quad x \wedge 1 = x$

complementation: $x \vee x' = 1, \quad x \wedge x' = 0$

Examples of Boolean Algebras

The set of subsets of a set X:

- T : $\mathrm{Pow}(X)$
- $\wedge$: $\cap$
- $\vee$: $\cup$
- $'$: ^{c}
- 0 : $\emptyset$
- 1 : X

Laws of Boolean algebra follow from Laws of Set Operations.

Examples of Boolean Algebras

The two element Boolean Algebra :

$$\mathbb{B} = (\{\texttt{true}, \texttt{false}\}, \&\&, \|, !, \texttt{false}, \texttt{true})$$

where $!, \&\&, \|$ are defined as:

- $!\texttt{true} = \texttt{false}$; $!\texttt{false} = \texttt{true}$,
- $\texttt{true} \;\&\&\; \texttt{true} = \texttt{true}$; ...
- $\texttt{true} \;\|\; \texttt{true} = \texttt{true}$; ...

NB

We will often use $\mathbb{B}$ for the two element set $\{true, false\}$. For simplicity this may also be abbreviated as $\{T, F\}$ or $\{1, 0\}$.

Examples of Boolean Algebras

- Cartesian products of $\mathbb{B}$
- Functions from a set S to $\mathbb{B}$
- Examples in tutorial (sets of natural numbers)

Derived laws

The following are all derivable from the Boolean Algebra laws.

Idempotence	$x \wedge x = x$
	$x \vee x = x$
Double complementation	$(x')' = x$
Annihilation	$x \wedge 0 = 0$
	$x \vee 1 = 1$
de Morgan's Laws	$(x \wedge y)' = x' \vee y'$
	$(x \vee y)' = x' \wedge y'$

Duality

If E is an expression made up with $\wedge, \vee, ', 0, 1$ and variables; then $\text{dual}(E)$ is the expression obtained by replacing $\wedge$ with $\vee$ and vice-versa; and 0 with 1 and vice-versa.

Theorem (Principle of Duality)

If you can show $E_1 = E_2$ holds in all Boolean Algebras[a], then $dual(E_1) = dual(E_2)$ holds in all Boolean Algebras.

[a] by using the Boolean Algebra Laws

Duality formally

A Boolean Algebra **expression** is defined as follows:

- 0, 1 are expressions
- A variable, x, y, $\ldots$, is an expression.
- If E is an expression then E' is an expression.
- If E_1 and E_2 are expressions, then $(E_1 \wedge E_2)$ and $(E_1 \vee E_2)$ are expressions.

Duality formally

If $\textsc{Exp}$ is the set of expressions, we define $\mathsf{dual} : \textsc{Exp} \rightarrow \textsc{Exp}$ as follows:

- $\mathsf{dual}(0) = 1$, $\mathsf{dual}(1) = 0$
- $\mathsf{dual}(x) = x$ for all variables x
- $\mathsf{dual}(E') = \mathsf{dual}(E)'$ for all expressions E
- $\mathsf{dual}((E_1 \wedge E_2)) = (\mathsf{dual}(E_1) \vee \mathsf{dual}(E_2))$ for all expressions E_1 and E_2
- $\mathsf{dual}((E_1 \vee E_2)) = (\mathsf{dual}(E_1) \wedge \mathsf{dual}(E_2))$ for all expressions E_1 and E_2

Duality example

$$
\begin{aligned}
\text{dual}((x \vee (x \wedge y))) &= (\text{dual}(x) \wedge \text{dual}((x \wedge y))) \\
&= (x \wedge \text{dual}((x \wedge y))) \\
&= (x \wedge (\text{dual}(x) \vee \text{dual}(y))) \\
&= (x \wedge (x \vee y))
\end{aligned}
$$

Duality example

$$
\begin{aligned}
\mathsf{dual}((x \vee (x \wedge y))) &= (\mathsf{dual}(x) \wedge \mathsf{dual}((x \wedge y))) \\
&= (x \wedge \mathsf{dual}((x \wedge y))) \\
&= (x \wedge (\mathsf{dual}(x) \vee \mathsf{dual}(y))) \\
&= (x \wedge (x \vee y))
\end{aligned}
$$

Duality example

$$
\begin{aligned}
\text{dual}((x \vee (x \wedge y))) &= (\text{dual}(x) \wedge \text{dual}((x \wedge y))) \\
&= (x \wedge \text{dual}((x \wedge y))) \\
&= (x \wedge (\text{dual}(x) \vee \text{dual}(y))) \\
&= (x \wedge (x \vee y))
\end{aligned}
$$

Duality example

$$
\begin{aligned}
\mathrm{dual}((x \vee (x \wedge y))) &= (\mathrm{dual}(x) \wedge \mathrm{dual}((x \wedge y))) \\
&= (x \wedge \mathrm{dual}((x \wedge y))) \\
&= (x \wedge (\mathrm{dual}(x) \vee \mathrm{dual}(y))) \\
&= (x \wedge (x \vee y))
\end{aligned}
$$

Summary of topics

- Well-formed formulas
- Boolean Algebras
- Valuations
- CNF/DNF
- Proof
- Natural deduction

Valuations

A **truth assignment** (or **model**) is a function $v : \textsc{Prop} \to \mathbb{B}$

We can extend v to a function $[\![\cdot]\!]_v : \textsc{WFFs} \to \mathbb{B}$ recursively:

- $[\![\top]\!]_v = \texttt{true}$, $[\![\bot]\!]_v = \texttt{false}$
- $[\![p]\!]_v = v(p)$
- $[\![\neg\varphi]\!]_v = ![\![\varphi]\!]_v$
- $[\![(\varphi \wedge \psi)]\!]_v = [\![\varphi]\!]_v \;\&\&\; [\![\psi]\!]_v$
- $[\![(\varphi \vee \psi)]\!]_v = [\![\varphi]\!]_v \;||\; [\![\psi]\!]_v$
- $[\![(\varphi \to \psi)]\!]_v = ![\![\varphi]\!]_v \;||\; [\![\psi]\!]_v$
- $[\![(\varphi \leftrightarrow \psi)]\!]_v = (![\![\varphi]\!]_v \;||\; [\![\psi]\!]_v) \;\&\&\; (![\![\psi]\!]_v \;||\; [\![\varphi]\!]_v)$

Valuations

A **truth assignment** (or **model**) is a function $v : \textsc{Prop} \to \mathbb{B}$

We can extend v to a function $[\![\cdot]\!]_v : \textsc{WFFs} \to \mathbb{B}$ recursively:

- $[\![\top]\!]_v = \texttt{true}$, $[\![\bot]\!]_v = \texttt{false}$
- $[\![p]\!]_v = v(p)$
- $[\![\neg\varphi]\!]_v = ![\![\varphi]\!]_v$
- $[\![(\varphi \wedge \psi)]\!]_v = [\![\varphi]\!]_v \;\&\&\; [\![\psi]\!]_v$
- $[\![(\varphi \vee \psi)]\!]_v = [\![\varphi]\!]_v \;||\; [\![\psi]\!]_v$
- $[\![(\varphi \to \psi)]\!]_v = ![\![\varphi]\!]_v \;||\; [\![\psi]\!]_v$
- $[\![(\varphi \leftrightarrow \psi)]\!]_v = (![\![\varphi]\!]_v \;||\; [\![\psi]\!]_v) \;\&\&\; (![\![\psi]\!]_v \;||\; [\![\varphi]\!]_v)$

Valuations

A **truth assignment** (or **model**) is a function $v : \textsc{Prop} \to \mathbb{B}$

We can extend v to a function $[\![\cdot]\!]_v : \mathrm{WFF}\textsc{s} \to \mathbb{B}$ recursively:

- $[\![\top]\!]_v = \texttt{true}$, $[\![\bot]\!]_v = \texttt{false}$
- $[\![p]\!]_v = v(p)$
- $[\![\neg\varphi]\!]_v = \,![\![\varphi]\!]_v$
- $[\![(\varphi \wedge \psi)]\!]_v = [\![\varphi]\!]_v \;\&\&\; [\![\psi]\!]_v$
- $[\![(\varphi \vee \psi)]\!]_v = [\![\varphi]\!]_v \;||\; [\![\psi]\!]_v$
- $[\![(\varphi \to \psi)]\!]_v = \,![\![\varphi]\!]_v \;||\; [\![\psi]\!]_v$
- $[\![(\varphi \leftrightarrow \psi)]\!]_v = (![\![\varphi]\!]_v \;||\; [\![\psi]\!]_v) \;\&\&\; (![\![\psi]\!]_v \;||\; [\![\varphi]\!]_v)$

Valuations

A **truth assignment** (or **model**) is a function $v : \textsc{Prop} \to \mathbb{B}$

We can extend v to a function $[\![\cdot]\!]_v : \textsc{WFFs} \to \mathbb{B}$ recursively:

- $[\![\top]\!]_v = \texttt{true}$, $[\![\bot]\!]_v = \texttt{false}$
- $[\![p]\!]_v = v(p)$
- $[\![\neg\varphi]\!]_v = ![\![\varphi]\!]_v$
- $[\![(\varphi \wedge \psi)]\!]_v = [\![\varphi]\!]_v \;\&\&\; [\![\psi]\!]_v$
- $[\![(\varphi \vee \psi)]\!]_v = [\![\varphi]\!]_v \;||\; [\![\psi]\!]_v$
- $[\![(\varphi \to \psi)]\!]_v = ![\![\varphi]\!]_v \;||\; [\![\psi]\!]_v$
- $[\![(\varphi \leftrightarrow \psi)]\!]_v = (![\![\varphi]\!]_v \;||\; [\![\psi]\!]_v) \;\&\&\; (![\![\psi]\!]_v \;||\; [\![\varphi]\!]_v)$

Valuations

A **truth assignment** (or **model**) is a function $v : \textsc{Prop} \to \mathbb{B}$

We can extend v to a function $[\![\cdot]\!]_v : \textsc{WFFs} \to \mathbb{B}$ recursively:

- $[\![\top]\!]_v = \texttt{true}$, $[\![\bot]\!]_v = \texttt{false}$
- $[\![p]\!]_v = v(p)$
- $[\![\neg\varphi]\!]_v = ![\![\varphi]\!]_v$
- $[\![(\varphi \wedge \psi)]\!]_v = [\![\varphi]\!]_v \;\&\&\; [\![\psi]\!]_v$
- $[\![(\varphi \vee \psi)]\!]_v = [\![\varphi]\!]_v \;||\; [\![\psi]\!]_v$
- $[\![(\varphi \to \psi)]\!]_v = ![\![\varphi]\!]_v \;||\; [\![\psi]\!]_v$
- $[\![(\varphi \leftrightarrow \psi)]\!]_v = (![\![\varphi]\!]_v \;||\; [\![\psi]\!]_v) \;\&\&\; (![\![\psi]\!]_v \;||\; [\![\varphi]\!]_v)$

Valuations

A **truth assignment** (or **model**) is a function $v : \textsc{Prop} \to \mathbb{B}$

We can extend v to a function $[\![\cdot]\!]_v : \textsc{WFFs} \to \mathbb{B}$ recursively:

- $[\![\top]\!]_v = \texttt{true}$, $[\![\bot]\!]_v = \texttt{false}$
- $[\![p]\!]_v = v(p)$
- $[\![\neg\varphi]\!]_v = ![\![\varphi]\!]_v$
- $[\![(\varphi \wedge \psi)]\!]_v = [\![\varphi]\!]_v \ \&\& \ [\![\psi]\!]_v$
- $[\![(\varphi \vee \psi)]\!]_v = [\![\varphi]\!]_v \ || \ [\![\psi]\!]_v$
- $[\![(\varphi \to \psi)]\!]_v = ![\![\varphi]\!]_v \ || \ [\![\psi]\!]_v$
- $[\![(\varphi \leftrightarrow \psi)]\!]_v = (![\![\varphi]\!]_v \ || \ [\![\psi]\!]_v) \ \&\& \ (![\![\psi]\!]_v \ || \ [\![\varphi]\!]_v)$

Valuations

A **truth assignment** (or **model**) is a function $v : \textsc{Prop} \to \mathbb{B}$

We can extend v to a function $[\![\cdot]\!]_v : \text{WFFs} \to \mathbb{B}$ recursively:

- $[\![\top]\!]_v = \texttt{true}$, $[\![\bot]\!]_v = \texttt{false}$
- $[\![p]\!]_v = v(p)$
- $[\![\neg\varphi]\!]_v = ![\![\varphi]\!]_v$
- $[\![(\varphi \wedge \psi)]\!]_v = [\![\varphi]\!]_v \;\&\&\; [\![\psi]\!]_v$
- $[\![(\varphi \vee \psi)]\!]_v = [\![\varphi]\!]_v \;||\; [\![\psi]\!]_v$
- $[\![(\varphi \to \psi)]\!]_v = ![\![\varphi]\!]_v \;||\; [\![\psi]\!]_v$
- $[\![(\varphi \leftrightarrow \psi)]\!]_v = (![\![\varphi]\!]_v \;||\; [\![\psi]\!]_v) \;\&\&\; (![\![\psi]\!]_v \;||\; [\![\varphi]\!]_v)$

Valuations

A **truth assignment** (or **model**) is a function $v : \textsc{Prop} \to \mathbb{B}$

We can extend v to a function $[\![\cdot]\!]_v : \mathrm{WFF}\textsc{s} \to \mathbb{B}$ recursively:

- $[\![\top]\!]_v = \texttt{true}$, $[\![\bot]\!]_v = \texttt{false}$
- $[\![p]\!]_v = v(p)$
- $[\![\neg\varphi]\!]_v = \,![\![\varphi]\!]_v$
- $[\![(\varphi \wedge \psi)]\!]_v = [\![\varphi]\!]_v \,\&\&\, [\![\psi]\!]_v$
- $[\![(\varphi \vee \psi)]\!]_v = [\![\varphi]\!]_v \,\|\, [\![\psi]\!]_v$
- $[\![(\varphi \to \psi)]\!]_v = \,![\![\varphi]\!]_v \,\|\, [\![\psi]\!]_v$
- $[\![(\varphi \leftrightarrow \psi)]\!]_v = (![\![\varphi]\!]_v \,\|\, [\![\psi]\!]_v) \,\&\&\, (![\![\psi]\!]_v \,\|\, [\![\varphi]\!]_v)$

Satisfiability, Validity and Equivalence

A formula φ is

- **satisfiable** if $[\![\varphi]\!]_v = \texttt{true}$ for some model v (v **satisfies** φ)
- **valid** or a **tautology** if $[\![\varphi]\!]_v = \texttt{true}$ for all models v
- **unsatisfiable** or a **contradiction** if $[\![\varphi]\!]_v = \texttt{false}$ for all models v

Logical equivalence

Two formulas, φ and ψ, are **logically equivalent**, $\varphi \equiv \psi$, if $[\![\varphi]\!]_v = [\![\psi]\!]_v$ for all models v.

Theorem

$\equiv$ *is an equivalence relation.*

Example

- Commutativity: $(p \vee q) \equiv (q \vee p)$
- Double negation: $\neg\neg p \equiv p$
- Contrapositive: $(p \to q) \equiv (\neg q \to \neg p)$
- De Morgan's: $(p \vee q)' \equiv p' \wedge q'$

Theorem

$\varphi \equiv \psi$ *if, and only if,* $(\varphi \leftrightarrow \psi)$ *is a tautology.*

Logical equivalence

Two formulas, φ and ψ, are **logically equivalent**, $\varphi \equiv \psi$, if $[\![\varphi]\!]_v = [\![\psi]\!]_v$ for all models v.

Theorem

$\equiv$ *is an equivalence relation.*

Example

- Commutativity: $(p \vee q) \equiv (q \vee p)$
- Double negation: $\neg\neg p \equiv p$
- Contrapositive: $(p \rightarrow q) \equiv (\neg q \rightarrow \neg p)$
- De Morgan's: $(p \vee q)' \equiv p' \wedge q'$

Theorem

$\varphi \equiv \psi$ *if, and only if,* $(\varphi \leftrightarrow \psi)$ *is a tautology.*

Logical equivalence

Two formulas, φ and ψ, are **logically equivalent**, $\varphi \equiv \psi$, if $[\![\varphi]\!]_v = [\![\psi]\!]_v$ for all models v.

Theorem

$\equiv$ *is an equivalence relation.*

Example

- Commutativity: $(p \vee q) \equiv (q \vee p)$
- Double negation: $\neg\neg p \equiv p$
- Contrapositive: $(p \to q) \equiv (\neg q \to \neg p)$
- De Morgan's: $(p \vee q)' \equiv p' \wedge q'$

Theorem

$\varphi \equiv \psi$ *if, and only if,* $(\varphi \leftrightarrow \psi)$ *is a tautology.*

Theories and entailment

A set of formulas is a **theory**

A model v *satisfies* a theory T if $[\![\varphi]\!]_v = \texttt{true}$ for all $\varphi \in T$

A theory T **entails** a formula φ, $T \models \varphi$, if $[\![\varphi]\!]_v = \texttt{true}$ for all models v which satisfy T

Example

- $T_1 = \{p\}$, $T_2 = \emptyset$, $T_3 = \{\bot\}$
- $v : p \longrightarrow \texttt{true}$ satisfies T_1 and T_2 but not T_3
- $T_1 \models (p \vee p)$ and $T_3 \models (p \vee p)$ but T_2 does not model $(p \vee p)$

Theories and entailment

A set of formulas is a **theory**

A model v *satisfies* a theory T if $[\![\varphi]\!]_v = \mathtt{true}$ for all $\varphi \in T$

A theory T **entails** a formula φ, $T \models \varphi$, if $[\![\varphi]\!]_v = \mathtt{true}$ for all models v which satisfy T

Theorem

The following are equivalent:

- $\varphi_1, \varphi_2, \ldots, \varphi_n \models \psi$
- $\emptyset \models ((\varphi_1 \wedge \varphi_2) \wedge \ldots \varphi_n) \to \psi$
- $((\varphi_1 \wedge \varphi_2) \wedge \ldots \varphi_n) \to \psi$ *is a tautology*
- $\emptyset \models \varphi_1 \to (\varphi_2 \to (\ldots \to \varphi_n) \to \psi)) \ldots)$

Summary of topics

- Well-formed formulas
- Boolean Algebras
- Valuations
- CNF/DNF
- Proof
- Natural deduction

Terminology and Rules

- For readability we assume associativity of $\wedge$ and $\vee$, and write $\overline{\varphi}$ for $\neg\varphi$.
- A **literal** is an expression p or $\overline{p}$, where p is a propositional atom.
- A propositional formula is in CNF (conjunctive normal form) if it has the form

$$\bigwedge_i C_i$$

  where each **clause** C_i is a disjunction of literals e.g. $p \vee q \vee \overline{r}$.
- A propositional formula is in DNF (disjunctive normal form) if it has the form

$$\bigvee_i C_i$$

  where each clause C_i is a conjunction of literals e.g. $p \wedge q \wedge \overline{r}$.

Motivation

- Finding satisfying assignments of formulas in DNF is straightforward
- Disproving validity of formulas in CNF is straightforward
- Karnaugh maps can be used to simplify formulas

- CNF and DNF are named after their top level operators; no deeper nesting of $\wedge$ or $\vee$ is permitted.
- We can assume in every clause (disjunct for the CNF, conjunct for the DNF) any given variable (literal) appears only once; preferably, no literal and its negation together.
  - $x \vee x = x, \quad x \wedge x = x$
  - $x \wedge \overline{x} = 0, \quad x \vee \overline{x} = 1$
  - $x \wedge 0 = 0$, $x \wedge 1 = x$, $x \vee 0 = x$, $x \vee 1 = 1$
- A preferred form for an expression is DNF, with as few terms as possible. In deriving such minimal simplifications the two basic rules are **absorption** and **combining the opposites**.

Fact

1. *Absorption:* $x \vee (x \wedge y) \equiv x$
2. *Combining the opposites:* $(x \wedge y) \vee (x \wedge \overline{y}) \equiv x$

Theorem

For every Boolean expression ϕ, there exists an equivalent expression in conjunctive normal form and an equivalent expression in disjunctive normal form.

Proof.

We show how to apply the equivalences already introduced to convert any given formula to an equivalent one in CNF, DNF is similar. □

Step 1: Push Negations Down

Using **De Morgan's** laws and the **double negation** rule

$$\overline{x \vee y} \equiv \overline{x} \wedge \overline{y}$$
$$\overline{x \wedge y} \equiv \overline{x} \vee \overline{y}$$
$$\overline{\overline{x}} \equiv x$$

we push negations down towards the atoms until we obtain a formula that is formed from literals using only $\wedge$ and $\vee$.

Step 2: Use Distribution to Convert to CNF

Using the distribution rules

$$x \vee (y_1 \wedge \ldots \wedge y_n) = (x \vee y_1) \wedge \ldots \wedge (x \vee y_n)$$
$$(y_1 \wedge \ldots \wedge y_n) \vee x = (y_1 \vee x) \wedge \ldots \wedge (y_n \vee x)$$

we obtain a CNF formula.

CNF/DNF in Propositional Logic

Using the equivalence

$$A \rightarrow B \quad \equiv \quad \neg A \vee B$$

we first eliminate all occurrences of $\rightarrow$

Example

$$\neg(\neg p \wedge ((r \wedge s) \rightarrow q)) \equiv \neg(\neg p \wedge (\neg(r \wedge s) \vee q))$$

Step 1:

Example

$$\begin{aligned}\overline{p(\overline{rs} \vee q)} &= \overline{\overline{p}} \vee \overline{\overline{rs} \vee q} \\ &= p \vee \overline{\overline{rs}} \wedge \overline{q} \\ &= p \vee rs\overline{q}\end{aligned}$$

Step 2:

Example

$$\begin{aligned}p \vee rs\overline{q} &= (p \vee r)(p \vee s\overline{q}) \\ &= (p \vee r)(p \vee s)(p \vee \overline{q}) \qquad \text{CNF}\end{aligned}$$

Canonical Form DNF

Given a Boolean expression E, we can construct an equivalent DNF E^{dnf} from the lines of the truth table where E is true: Given an assignment v from $\{x_1 \ldots x_i\}$ to $\mathbb{B}$, define the literal

$$\ell_i = \begin{cases} x_i & \text{if } v(x_i) = \texttt{true} \\ \overline{x_i} & \text{if } v(x_i) = \texttt{false} \end{cases}$$

and a product $t_v = \ell_1 \wedge \ell_2 \wedge \ldots \wedge \ell_n$.

Example

If $v(x_1) = \texttt{true}$ and $v(x_2) = \texttt{false}$ then $t_v = x_1 \wedge \overline{x_2}$

The **canonical DNF** of E is

$$E^{dnf} = \bigvee_{[\![E]\!]_v = \texttt{true}} t_v$$

Example

If E is defined by

x	y	E
F	F	T
F	T	F
T	F	T
T	T	T

then $E^{dnf} = (\overline{x} \wedge \overline{y}) \vee (x \wedge \overline{y}) \vee (x \wedge y)$
Note that this can be simplified to either

$$\overline{y} \vee (x \wedge y)$$

or

$$(\overline{x} \wedge \overline{y}) \vee x$$

Canonical CNF

After pushing negations down, the negation of a DNF is a CNF (and vice versa).

$\Rightarrow$ Given an expression E, we can obtain an equivalent CNF by finding a DNF for $\neg E$ and then applying De Morgan's laws.

$\Leftrightarrow$ Look at rows in the truth table of E that contain `false` and *negate* the literals.

Example

If E is defined by

x	y	E
F	F	F
F	T	F
T	F	T
T	T	F

then $E^{cnf} = (x \vee y) \wedge (x \vee \overline{y}) \wedge (\overline{x} + \overline{y})$.

Karnaugh Maps

For up to four variables (propositional symbols) a diagrammatic method of simplification called **Karnaugh maps** works quite well. For every propositional function of $k = 2, 3, 4$ variables we construct a rectangular array of 2^k cells. We mark the squares corresponding to the value `true` with eg "$+$" and try to cover these squares with as few rectangles with sides 1 or 2 or 4 as possible.

Example

	yz	$y\bar{z}$	$\bar{y}\bar{z}$	$\bar{y}z$
x	$+$	$+$		$+$
$\bar{x}$	$+$		$+$	$+$

For optimisation, the idea is to cover the $+$ squares with the minimum number of rectangles. One *cannot* cover any empty cells.

- The rectangles can go 'around the corner'/the actual map should be seen as a *torus*.
- Rectangles must have sides of 1, 2 or 4 squares (three adjacent cells are useless).

Example

	yz	$y\bar{z}$	$\bar{y}\bar{z}$	$\bar{y}z$
x	$+$	$+$		$+$
$\bar{x}$	$+$		$+$	$+$

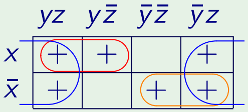
$$E = (x \wedge y) \vee (\bar{x} \wedge \bar{y}) \vee z$$

Canonical form would consist of writing all cells separately (6

For optimisation, the idea is to cover the $+$ squares with the minimum number of rectangles. One *cannot* cover any empty cells.

- The rectangles can go 'around the corner'/the actual map should be seen as a *torus*.
- Rectangles must have sides of 1, 2 or 4 squares (three adjacent cells are useless).

Example



$$E = (x \wedge y) \vee (\bar{x} \wedge \bar{y}) \vee z$$

Canonical form would consist of writing all cells separately (6

Summary of topics

- Well-formed formulas
- Boolean Algebras
- Valuations
- CNF/DNF
- Proof
- Natural deduction

Motivation

Given a theory T and a formula φ, how do we show $T \models \varphi$?

- Consider all valuations v (SEMANTIC approach)
- Use a sequence of equivalences and *deductive rules* to show that φ is a logical consequence of T (SYTACTIC approach)

Formal proofs

A formal way to show that a formula logically follows from a theory.

- Highly disciplined way of reasoning (good for computers)
- A sequence of formulas where each step is a deduction based on earlier steps
- Based entirely on rewriting formulas – no semantic interpretations needed