Summary of topics

- Well-formed formulas (SYNTAX)
- Boolean Algebras
- Valuations (SEMANTICS)
- CNF/DNF
- Proof
- Natural deduction
Summary of topics

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

Let $PROP = \{p, q, r, \ldots\}$ be a set of propositional letters. Consider the alphabet

$$\Sigma = PROP \cup \{\top, \bot, \neg, \land, \lor, \rightarrow, \leftrightarrow, (, )\}.$$ 

The well-formed formulas (wffs) over $PROP$ is the smallest set of words over $\Sigma$ such that:

- $\top, \bot$ and all elements of $PROP$ are wffs
- If $\varphi$ is a wff then $\neg \varphi$ is a wff
- If $\varphi$ and $\psi$ are wffs then $(\varphi \land \psi), (\varphi \lor \psi), (\varphi \rightarrow \psi)$, and $($\varphi \leftrightarrow \psi)$ are wffs.
Examples

The following are well-formed formulas:

- $(p \land \neg \top)$
- $\neg (p \land \neg \top)$
- $\neg \neg (p \land \neg \top)$

The following are **not** well-formed formulas:

- $p \land \land$
- $p \land \neg \top$
- $(p \land q \land r)$
- $\neg (\neg p)$
To aid readability some conventions and binding rules can and will be used.

- Parentheses omitted if there is no ambiguity (e.g. $p \land q$)
- $\neg$ binds more tightly than $\land$ and $\lor$, which bind more tightly than $\rightarrow$ and $\leftrightarrow$ (e.g. $p \land q \rightarrow r$ instead of $((p \land q) \rightarrow r)$
Conventions

To aid readability some conventions and binding rules can and will be used.

- Parentheses omitted if there is no ambiguity (e.g. \( p \land q \))
- \( \neg \) binds more tightly than \( \land \) and \( \lor \), which bind more tightly than \( \rightarrow \) and \( \leftrightarrow \) (e.g. \( p \land q \rightarrow r \) instead of \( ((p \land q) \rightarrow r) \))

Other conventions (rarely used/assumed in this course):

- \( ' \) or \( \bar{\ } \) for \( \neg \)
- \( + \) for \( \lor \)
- \( \cdot \) or juxtaposition for \( \land \)
- \( \land \) binds more tightly than \( \lor \)
- \( \land \) and \( \lor \) associate to the left: \( p \lor q \lor r \) instead of \( (((p \lor q) \lor r) \)
- \( \rightarrow \) and \( \leftrightarrow \) associate to the right: \( p \rightarrow q \rightarrow r \) instead of \( (p \rightarrow (q \rightarrow r)) \)
Parse trees

The structure of well-formed formulas (and other grammar-defined syntaxes) can be shown with a parse tree.

Example

\[((P \land \neg Q) \lor \neg (Q \rightarrow P))\]
Parse trees

The structure of well-formed formulas (and other grammar-defined syntaxes) can be shown with a parse tree.

Example

\[((P \land \neg Q) \lor \neg(Q \rightarrow P))\]
Parse trees

The structure of well-formed formulas (and other grammar-defined syntaxes) can be shown with a parse tree.

Example

\[((P \land \neg Q) \lor \neg (Q \rightarrow P))\]
The structure of well-formed formulas (and other grammar-defined syntaxes) can be shown with a **parse tree**.

**Example**

\[ ((P \land \neg Q) \lor \neg(Q \rightarrow P)) \]
Parse trees

The structure of well-formed formulas (and other grammar-defined syntaxes) can be shown with a parse tree.

Example

\[((P \land \neg Q) \lor \neg (Q \rightarrow P))\]
Formally, we can define a parse tree as follows:
A parse tree is either:

- (B) A node containing $\top$;
- (B) A node containing $\bot$;
- (B) A node containing a propositional variable;
- (R) A node containing $\neg$ with a single parse tree child;
- (R) A node containing $\land$ with two parse tree children;
- (R) A node containing $\lor$ with two parse tree children;
- (R) A node containing $\rightarrow$ with two parse tree children; or
- (R) A node containing $\leftrightarrow$ with two parse tree children.
Summary of topics

- Well-formed formulas
- Boolean Algebras
- Valuations
- CNF/DNF
- Proof
- Natural deduction
Laws (Week 2 flashback)

Look again at the laws of set operations and boolean equivalences:

\[
\begin{align*}
x \cup y &= y \cup x \\
x \cap y &= y \cap x \\
(x \cup y) \cup z &= x \cup (y \cup z) \\
(x \cap y) \cap z &= x \cap (y \cap z) \\
x \cup (y \cap z) &= (x \cup y) \cap (x \cup z) \\
x \cap (y \cup z) &= (x \cap y) \cup (x \cap z) \\
x \cup \emptyset &= x, \quad x \cap \mathcal{U} = x \\
x \cup x^c &= \mathcal{U}, \quad x \cap x^c = \emptyset \\
x \lor \bot &\equiv x, \quad x \land \top \equiv x \\
x \lor \lnot x &\equiv \top, \quad x \land \lnot x \equiv \bot
\end{align*}
\]

These are the same laws written with different symbols!
Laws (Week 2 flashback)

Look again at the laws of set operations and boolean equivalences:

- \( x \cup y = y \cup x \)
- \( x \cap y = y \cap x \)
- \( (x \cup y) \cup z = x \cup (y \cup z) \)
- \( (x \cap y) \cap z = x \cap (y \cap z) \)
- \( x \cup (y \cap z) = (x \cup y) \cap (x \lor z) \)
- \( x \cap (y \lor z) = (x \cap y) \cup (x \land z) \)
- \( x \cup \emptyset = x, \quad x \cap \mathbb{U} = x \)
- \( x \cup x^c = \mathbb{U}, \quad x \cap x^c = \emptyset \)

- \( x \lor y \equiv y \lor x \)
- \( x \land y \equiv y \land x \)
- \( (x \lor y) \lor z \equiv x \lor (y \lor z) \)
- \( (x \land y) \land z \equiv x \land (y \land z) \)
- \( x \lor (y \land z) \equiv (x \lor y) \land (x \lor z) \)
- \( x \land (y \lor z) \equiv (x \land y) \lor (x \land z) \)
- \( x \lor \bot \equiv x, \quad x \land \top \equiv x \)
- \( x \lor \neg x \equiv \top, \quad x \land \neg x \equiv \bot \)

These are the same laws written with different symbols!
Definition: Boolean Algebra

A Boolean algebra is a structure \((T, \lor, \land, \prime, 0, 1)\) where

- \(0, 1 \in T\)
- \(\lor : T \times T \to T\) (called join)
- \(\land : T \times T \to T\) (called meet)
- \(\prime : T \to T\) (called complementation)

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

**commutativity:** \(x \lor y = y \lor x\)
\(x \land y = y \land x\)

**associativity:** \((x \lor y) \lor z = x \lor (y \lor z)\)
\((x \land y) \land z = x \land (y \land z)\)

**distributivity:** \(x \lor (y \land z) = (x \lor y) \land (x \lor z)\)
\(x \land (y \lor z) = (x \land y) \lor (x \land z)\)

**identity:** \(x \lor 0 = x, \quad x \land 1 = x\)

**complementation:** \(x \lor x' = 1, \quad x \land x' = 0\)
Examples of Boolean Algebras

The set of subsets (power set) of a set $\mathcal{U}$: 

- $\mathcal{T} : \text{Pow}(\mathcal{U})$
- $\land : \cap$
- $\lor : \cup$
- $'$ : $c$
- $0 : \emptyset$
- $1 : \mathcal{U}$
Examples of Boolean Algebras

The two element Boolean Algebra:

\[ \mathbb{B} = (\{\text{true, false}\}, \&\&, ||, !, \text{false, true}) \]

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

- \(!\text{true} = \text{false}; \!\text{false} = \text{true},\)
- \(\text{true} \&\& \text{true} = \text{true}; \ldots\)
- \(\text{true} || \text{true} = \text{true}; \ldots\)

NB

We will often use \(\mathbb{B}\) for the two element set \{\text{true, false}\}. For simplicity this may also be abbreviated as \{T, F\} or \{1, 0\}. 
Examples of Boolean Algebras

Cartesian products of \( \mathbb{B} \), that is \( n \)-tuples of 0’s and 1’s with Boolean operations, e.g. \( \mathbb{B}^4 \):

\[
\begin{align*}
\text{join:} & \quad (1, 0, 0, 1) \lor (1, 1, 0, 0) = (1, 1, 0, 1) \\
\text{meet:} & \quad (1, 0, 0, 1) \land (1, 1, 0, 0) = (1, 0, 0, 0) \\
\text{complement:} & \quad (1, 0, 0, 1)' = (0, 1, 1, 0) \\
0: & \quad (0, 0, 0, 0) \\
1: & \quad (1, 1, 1, 1).
\end{align*}
\]

NB

These are the bitwise operations on 4-bit machine words.
Examples of Boolean Algebras

Functions from any set $S$ to $\mathbb{B}$; their set is denoted $\text{Map}(S, \mathbb{B})$

If $f, g : S \rightarrow \mathbb{B}$ then

- $(f \lor g) : S \rightarrow \mathbb{B}$ is defined by $s \mapsto f(s) \lor g(s)$
- $(f \land g) : S \rightarrow \mathbb{B}$ is defined by $s \mapsto f(s) \land g(s)$
- $f' : S \rightarrow \mathbb{B}$ is defined by $s \mapsto \neg f(s)$
- $0 : S \rightarrow \mathbb{B}$ is the function $f(s) = \text{false}$
- $1 : S \rightarrow \mathbb{B}$ is the function $f(s) = \text{true}$
All those examples were just the power set example in disguise.

- The two-element algebra with \{true, false\} is just the power set algebra with \( U = \{\emptyset\} \), but with different notation.
- A function \( f : S \to \mathbb{B} \) coincides with the membership relation \( (\in) \) of the set \( \{ x : x \in S \text{ and } f(x) = \text{true} \} \)

...
All those examples were just the power set example in disguise.

- The two-element algebra with \{true, false\} is just the power set algebra with \(U = \{\emptyset\}\), but with different notation.
- A function \(f : S \rightarrow \mathbb{B}\) coincides with the membership relation \(\in\) of the set \(\{x : x \in S \text{ and } f(x) = \text{true}\}\)
- ...  

In fact, every (finite) boolean algebra is the power set example in disguise!
Every finite Boolean algebra satisfies: $|T| = 2^k$ for some $k$.
All algebras with the same number of elements are isomorphic, i.e. “structurally similar”, written $\simeq$. Therefore, studying one such algebra describes properties of all.
The algebras mentioned above are all of this form

- $n$-tuples $\simeq \mathbb{B}^n$
- $\text{Pow}(S) \simeq \mathbb{B}^{|S|}$
- $\text{Map}(S, \mathbb{B}) \simeq \mathbb{B}^{|S|}$
Duality revisited

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

**Theorem (Principle of Duality)**

If $E_1 = E_2$ holds in all Boolean Algebras\(^a\), then $\text{dual}(E_1) = \text{dual}(E_2)$ holds in all Boolean Algebras.

\(^a\)i.e. is provable using the Boolean Algebra Laws

In Week 2, we convinced ourselves of this with some handwaving.
Duality revisited

If \((T, \lor, \land, ', 0, 1)\) is a Boolean algebra, then the dual algebra \((T, \land, \lor, ', 1, 0)\) is also a Boolean Algebra. For example:

- \(T : \text{Pow}(X)\)
- \(\land : \cup\)
- \(\lor : \cap\)
- \(' : c\)
- \(0 : X\)
- \(1 : \emptyset\)

The principle of duality follows immediately from this observation.
A Boolean Algebra expression is defined inductively 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 \((E_1 \land E_2)\) and \((E_1 \lor E_2)\) are expressions.

Use \( \text{Exp} \) for the set of such expressions.
Dualising, formally

We define dual : \( \text{Exp} \to \text{Exp} \) recursively as follows:

- \( \text{dual}(0) = 1 \), \( \text{dual}(1) = 0 \)
- \( \text{dual}(x) = x \) for all variables \( x \)
- \( \text{dual}(E') = \text{dual}(E)' \) for all expressions \( E \)
- \( \text{dual}((E_1 \land E_2)) = (\text{dual}(E_1) \lor \text{dual}(E_2)) \) for all expressions \( E_1 \) and \( E_2 \)
- \( \text{dual}((E_1 \lor E_2)) = (\text{dual}(E_1) \land \text{dual}(E_2)) \) for all expressions \( E_1 \) and \( E_2 \)
Example dual

\[\text{dual}((x \lor (x \land y))) = (\text{dual}(x) \land \text{dual}((x \land y))))\]
Example dual

\[
dual((x \lor (x \land y))) = (\dual(x) \land \dual((x \land y))) = (x \land \dual((x \land y)))
\]
Example dual

dual((x ∨ (x ∧ y)))  = (dual(x) ∧ dual((x ∧ y)))
= (x ∧ dual((x ∧ y)))
= (x ∧ (dual(x) ∨ dual(y))))
Example dual

dual((x ∨ (x ∧ y)))) = (dual(x) ∧ dual((x ∧ y))))
= (x ∧ dual((x ∧ y))))
= (x ∧ (dual(x) ∨ dual(y))))
= (x ∧ (x ∨ y))
Boolean algebras: punchline

- Boolean algebras allow us to talk about set operations, propositions and bitwise operations on machine words at the same time.
- Properties proved in boolean algebra hold for all these instances: no more double labour.
- Calculation with set operations and calculation with propositional logic is, ultimately, the same thing. Therefore, you can freely change between the two perspectives in specifications and proofs.
Summary of topics

- Well-formed formulas
- Boolean Algebras
- Valuations
- CNF/DNF
- Proof
- Natural deduction
A **truth assignment** (or **valuation**, or **model**) is a function 

\[ \nu : \text{Prop} \to \mathbb{B} \]

In week 2, we said:

*Two formulas \( \phi, \psi \) are **logically equivalent**, denoted \( \phi \equiv \psi \) if they have the same truth value for all truth valuations.*

But we never really defined valuation of formulas—instead, we handwaved it by saying “draw a truth table”. 
Valuations

Let \( \nu : \text{PROP} \rightarrow \mathbb{B} \) be a valuation.

We can extend \( \nu \) to a function \( [\cdot]_\nu : \text{WFFs} \rightarrow \mathbb{B} \) recursively:
Valuations

Let $v : \text{PROP} \to \mathbb{B}$ be a valuation.

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

- $[\top]_v = \text{true}$, $[\bot]_v = \text{false}$
Valuations

Let \( \nu : \text{PROP} \rightarrow \mathbb{B} \) be a valuation.

We can extend \( \nu \) to a function \( \cdot \) : \text{WFFs} \rightarrow \mathbb{B} \) recursively:

- \([\top]_\nu = \text{true}\)
- \([\bot]_\nu = \text{false}\)
- \([p]_\nu = \nu(p)\)
Valuations

Let \( \nu : \text{PROP} \rightarrow \mathbb{B} \) be a valuation.

We can extend \( \nu \) to a function \([\cdot]_\nu : \text{WFFs} \rightarrow \mathbb{B}\) recursively:

- \([\top]_\nu = \text{true}\), \([\bot]_\nu = \text{false}\)
- \([p]_\nu = \nu(p)\)
- \([-\varphi]_\nu = !([\varphi]_\nu)\)
Valuations

Let \( \nu : \text{PROP} \rightarrow \mathbb{B} \) be a valuation.

We can extend \( \nu \) to a function \( [\cdot]_\nu : \text{WFFs} \rightarrow \mathbb{B} \) recursively:

- \( [\top]_\nu = \text{true} \), \( [\bot]_\nu = \text{false} \)
- \( [p]_\nu = \nu(p) \)
- \( [\neg \varphi]_\nu = ![\varphi]_\nu \)
- \( [(\varphi \land \psi)]_\nu = [\varphi]_\nu \& [\psi]_\nu \)
- \( [(\varphi \lor \psi)]_\nu = [\varphi]_\nu \lor [\psi]_\nu \)
- \( [(\varphi \rightarrow \psi)]_\nu = ![\varphi]_\nu \lor [\psi]_\nu \)
- \( [(\varphi \leftrightarrow \psi)]_\nu = (![\varphi]_\nu \lor [\psi]_\nu) \& (![\psi]_\nu \lor [\varphi]_\nu) \)
Valuations

Let $\nu : \text{PROP} \rightarrow \mathbb{B}$ be a valuation.

We can extend $\nu$ to a function $[\cdot]_\nu : \text{WFFs} \rightarrow \mathbb{B}$ recursively:

- $[\top]_\nu = \text{true}$, $[\bot]_\nu = \text{false}$
- $[p]_\nu = \nu(p)$
- $[\neg \phi]_\nu = ![[\phi]_\nu$
- $[[\phi \land \psi]]_\nu = [\phi]_\nu \land [\psi]_\nu$
- $[[\phi \lor \psi]]_\nu = [\phi]_\nu \lor [\psi]_\nu$
- $[[\phi \rightarrow \psi]]_\nu = ![[\phi]_\nu \lor [\psi]_\nu$
- $[[\phi \leftrightarrow \psi]]_\nu = (![\phi]_\nu \lor [\psi]_\nu) \land ([\phi]_\nu \lor ![[\psi]_\nu \lor [\phi]_\nu)$
Valuations

Let \( \nu : \text{PROP} \rightarrow \mathbb{B} \) be a valuation.

We can extend \( \nu \) to a function \([\cdot]_\nu : \text{WFFs} \rightarrow \mathbb{B}\) recursively:

- \([\top]_\nu = \text{true}, \quad [\bot]_\nu = \text{false}\)
- \([p]_\nu = \nu(p)\)
- \([\neg \varphi]_\nu = ![[\varphi]]_\nu\)
- \([((\varphi \land \psi))]_\nu = [[\varphi]]_\nu \land [[\psi]]_\nu\)
- \([((\varphi \lor \psi))]_\nu = [[\varphi]]_\nu \lor [[\psi]]_\nu\)
- \([((\varphi \rightarrow \psi))]_\nu = ![[\varphi]]_\nu \lor [[\psi]]_\nu\)
Valuations

Let \( \nu : \text{PROP} \rightarrow \mathbb{B} \) be a valuation.

We can extend \( \nu \) to a function \( [\cdot]_\nu : \text{WFFs} \rightarrow \mathbb{B} \) recursively:

- \( [\top]_\nu = \text{true} \), \( [\bot]_\nu = \text{false} \)
- \( [p]_\nu = \nu(p) \)
- \( [\neg \phi]_\nu = ![[\phi]_\nu] \)
- \( [(\phi \land \psi)]_\nu = [\phi]_\nu \land [\psi]_\nu \)
- \( [(\phi \lor \psi)]_\nu = [\phi]_\nu \lor [\psi]_\nu \)
- \( [(\phi \rightarrow \psi)]_\nu = ![[\phi]_\nu \lor [\psi]_\nu] \)
- \( [(\phi \leftrightarrow \psi)]_\nu = (![[\phi]_\nu \lor [\psi]_\nu]) \land (![[\psi]_\nu \lor [\phi]_\nu]) \)
A formula $\varphi$ is

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

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

**Theorem**

$\equiv$ is an equivalence relation.
Logical equivalence

Two formulas, \( \varphi \) and \( \psi \), are **logically equivalent**, \( \varphi \equiv \psi \), if \([\varphi]_v = [\psi]_v\) for all models \( v \).

**Theorem**

\( \equiv \) is an equivalence relation.

**Example**

- Commutativity: \((p \lor q) \equiv (q \lor p)\)
- Double negation: \(\neg\neg p \equiv p\)
- Contrapositive: \((p \rightarrow q) \equiv (\neg q \rightarrow \neg p)\)
- De Morgan’s: \((p \lor q)' \equiv p' \land q'\)
Logical equivalence

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

**Theorem**

$\equiv$ is an equivalence relation.

**Example**

- Commutativity: $(p \lor q) \equiv (q \lor p)$
- Double negation: $\neg\neg p \equiv p$
- Contrapositive: $(p \rightarrow q) \equiv (\neg q \rightarrow \neg p)$
- De Morgan’s: $(p \lor q)' \equiv p' \land q'$

**Theorem**

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

A set of (well-formed) formulas is a **theory**

A model \( v \) **satisfies** a theory \( T \) if \( [\varphi]_v = \text{true} \) for all \( \varphi \in T \)

A theory \( T \) **entails** a formula \( \varphi \), \( T \models \varphi \), if \( [\varphi]_v = \text{true} \) for all models \( v \) which satisfy \( T \)

**Example**

- \( T_1 = \{ p \} \), \( T_2 = \emptyset \), \( T_3 = \{ \bot \} \)
- \( v : p \mapsto \text{true} \) satisfies \( T_1 \) and \( T_2 \) but not \( T_3 \)
- \( T_1 \models (p \lor p) \) and \( T_3 \models (p \lor p) \) but \( T_2 \) does not entail \( (p \lor p) \)
Theories and entailment

A set of (well-formed) formulas is a **theory**

A model \( \nu \) *satisfies* a theory \( T \) if \([\varphi]_\nu = \text{true}\) for all \( \varphi \in T \)

A theory \( T \) **entails** a formula \( \varphi \), \( T \models \varphi \), if \([\varphi]_\nu = \text{true}\) for all models \( \nu \) which satisfy \( T \)

**Theorem**

*The following are equivalent:*

1. \( \varphi_1, \varphi_2, \ldots, \varphi_n \models \psi \)
2. \( \emptyset \models ((\varphi_1 \land \varphi_2) \land \ldots \varphi_n) \rightarrow \psi \)
3. \( ((\varphi_1 \land \varphi_2) \land \ldots \varphi_n) \rightarrow \psi \) is a tautology
4. \( \emptyset \models \varphi_1 \rightarrow (\varphi_2 \rightarrow (\ldots \rightarrow \varphi_n \rightarrow \psi)) \ldots \)
Summary of topics

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