Propositional Logic

Definition: A *propositional logic* is a triple $L = (P, C, A)$ in which:

- P is a set of *proposition names*.
- $C = \{\neg, \rightarrow, \bot\}$ is the set of *logical connectives.*
- $A = \{ (,) \}$ is the set of auxiliary symbols.
- $\text{Elements}(P) \cap (A \cup C) = \emptyset$.

Here Elements(P) denotes the set of all characters used in elements of P. For example, if the elements of P are strings (as in the blocks-world example), then Elements(P) is the set of all characters making up those strings.

L is said to be *finite* precisely when P is a finite set.

Notational convention: Uppercase letters, usually from the beginning of the alphabet, as well as such letters subscripted, are typically used to denote proposition names. Examples: A, B, C, A_1, C_{15}.

The example of the previous section presents a slight problem relative to this definition, since the proposition names contain parentheses, which are elements of A. This is easily remedied by using square brackets instead. For example, On_table(x,y) becomes On_table[x,y].

Definition: The class of *(strict) well-formed formulas* over $\mathcal{L}$ (denoted SWF($\mathcal{L}$)), is the smallest set of strings over the alphabet $\mathcal{P} \cup \mathcal{A} \cup \mathcal{C}$ which is closed under the following rules:

(a) $\perp \in$ SWF($\mathcal{L}$).

(b) $A \in \mathcal{P}$ implies that $A \in$ SWF($\mathcal{L}$).

(c) $\varphi \in$ SWF($\mathcal{L}$) implies that $(\neg\varphi) \in$ SWF($\mathcal{L}$).

(d) $\varphi_1, \varphi_2 \in$ SWF($\mathcal{L}$) implies that

$$(\varphi_1 \rightarrow \varphi_2) \in \text{SWF}(\mathcal{L}).$$

Examples: Let $\mathcal{P} = \{A, B, C\}$. The following are strict well-formed formulas:

$\perp$	$(\neg A)$	$(A \rightarrow B)$
A	$((\neg A) \rightarrow B)$	$(\perp \rightarrow B)$
$(\neg\perp)$	$((A \rightarrow B) \rightarrow B)$	$(A \rightarrow (B \rightarrow B))$

Examples: The following are not strict well-formed formulas:

$\neg A$	$A \rightarrow B$	$((A \rightarrow B))$
$(A \vee B)$	$(A \wedge B)$	$(A \equiv B)$

Notes:

- The connectives $\vee$, $\wedge$, and $\equiv$ are not included explicitly.
- Parentheses must be used in very specific ways.

We will relax these restrictions shortly.

Truth and interpretations:

The *truth values* are the elements of {0,1}. 0 is interpreted as false, and 1 as true. Numbers are used, rather than the more conventional F and T, because this use facilitates other definitions.

An interpretation defines a possible world for the logic. More formally, an *interpretation* (or *valuation*) for L is a function

$$v : \mathsf{P} \to \{0,1\}.$$

For $A \in \mathsf{P}$,

- v(A) = 1 if A is true in the interpretation.
- v(A) = 0 if A is false in the interpretation.

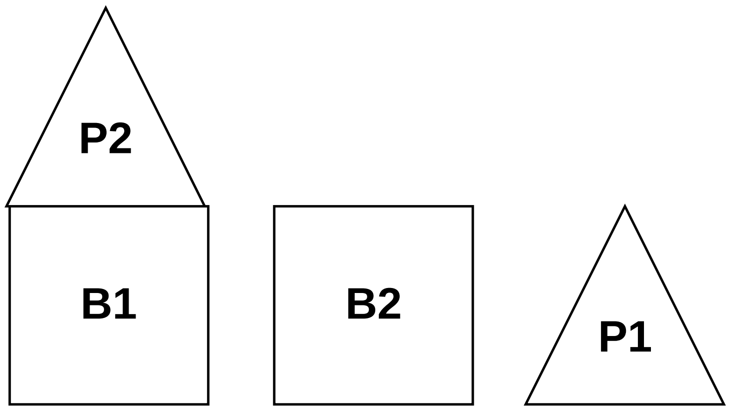
Interp(L) denotes the set of all interpretations for L.

Example: Let P = {A, B, C}. Then there are eight distinct interpretations, identified in the table below.

v(A)	v(B)	v(C)
0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1
1	1	0
1	1	1

In general, if there are n proposition letters, then there are 2^n distinct interpretations.

Example:
Using the propositional logic of the previous section, provide the interpretation describing the world above.



Predicate	Interp.
Is_cube[B1]	1
Is_cube[B2]	1
Is_cube[P1]	0
Is_cube[P2]	0
Is_pyramid[B1]	0
Is_pyramid[B2]	0
Is_pyramid[P1]	1
Is_pyramid[P2]	1
On_table[B1]	1
On_table[B2]	1
On_table[P1]	1
On_table[P2]	0

Predicate	Interp.
On[B1,B1]	0
On[B1,B2]	0
On[B1,P1]	0
On[B1,P2]	0
On[B2,B1]	0
On[B2,B2]	0
On[B2,P1]	0
On[B2,P2]	0
On[P1,B1]	0
On[P1,B2]	0
On[P1,P1]	0
On[P1,P2]	0
On[P2,B1]	1
On[P2,B2]	0
On[P2,P1]	0
On[P2,P2]	0

Note that there are 2^{28} possible interpretations, of which only 13 represent reality. Clearly, the constraints rule out most of the possibilities!

Suppose that we eliminate those which are either true or else false in all interpretations. We then get the following list.

Predicate	Interp.
On_table[B1]	1
On_table[B2]	1
On_table[P1]	1
On_table[P2]	0

Predicate	Interp.
On[B1,B2]	0
On[B2,B1]	0
On[P1,B1]	0
On[P1,B2]	0
On[P2,B1]	1
On[P2,B2]	0

Here there are only 2^{10} or about 1024 interpretations, many more than 13, but far fewer than 2^{28}, which is about 256 million.

Moral (to be repeated many times): It pays to choose your modelling tools carefully.

Extending an interpretation to formulas:

The semantics of $\neg$ is quite obvious. It may be represented on a basic level by using the following truth table, with φ taken to be any strict wff whatever.

φ	$(\neg\varphi)$
0	1
1	0

The semantics of $\rightarrow$ is represented in the table below.

φ_1	φ_2	$(\varphi_1 \rightarrow \varphi_2)$
0	0	1
0	1	1
1	0	0
1	1	1

This may seem a bit strange, since if φ_1 is false, the formula is always true. Call φ_1 the *antecedent* and φ_2 the *consequent*. If the antecedent is false, it does not mean that the consequent is true. Rather, it means that the statement is true. Consider

"If we are in Umeå, then it must be cold."

Another way to say this is

"Either we are not in Umeå, or else it is cold."

With the latter interpretation, things make more sense.

Here is a more formal definition of the semantics.

Definition: Let $v : \mathcal{P} \rightarrow \{0,1\}$ be an interpretation of $\mathcal{L}$. Define $\bar{v}$: SWF(L) $\rightarrow \{0,1\}$ as follows.

(a) $\bar{v}(\perp) = 0$.

(b) $\bar{v}(A) = v(A)$, for $A \in \mathcal{P}$.

(c) $\bar{v}((\neg\varphi)) = 1 - \bar{v}(\varphi)$, for $\varphi \in$ SWF(L).

(d) $\bar{v}((\varphi_1 \rightarrow \varphi_2)) = \max(1 - \bar{v}(\varphi_1), \bar{v}(\varphi_2))$,
for $\varphi_1, \varphi_2 \in$ SWF(L).

Note that this is an inductive definition.

Example: Suppose that $\mathcal{P} = \{A, B, C\}$, and that $v : A \mapsto 1, B \mapsto 0, C \mapsto 1$. Then

$$
\begin{aligned}
&\bar{v}((\neg(A \rightarrow (B \rightarrow (\neg C)))))\\
&\quad = 1 - \bar{v}((A \rightarrow (B \rightarrow (\neg C))))\\
&\quad = 1 - \max(1 - \bar{v}(A), \bar{v}((B \rightarrow (\neg C))))\\
&\quad = 1 - \max(0, \max(1 - \bar{v}(B), \bar{v}((\neg C))))\\
&\quad = 1 - \max(0, \max(1, 1 - \bar{v}(C)))\\
&\quad = 1 - \max(0, \max(1, 0))\\
&\quad = 1 - 1\\
&\quad = 0.
\end{aligned}
$$

Other logical connectives:

The other logical connectives ($\vee$, $\wedge$, $\equiv$) are defined as abbreviations of formulas using the connectives $\neg$ and $\rightarrow$. Formally, we have the following.

Definition:
$(\varphi_1 \vee \varphi_2)$ is an abbreviation for $((\neg\varphi_1) \rightarrow \varphi_2)$.
$(\varphi_1 \wedge \varphi_2)$ is an abbreviation for $(\neg(\varphi_1 \rightarrow (\neg\varphi_2)))$.
$(\varphi_1 \equiv \varphi_2)$ is an abbreviation for
$$((\varphi_1 \rightarrow \varphi_2) \wedge (\varphi_2 \rightarrow \varphi_1)).$$
$\top$ is an abbreviation for $(\neg\bot)$.

Do these abbreviations make sense? Yes.
The key thing to "believe" is that $(\varphi_1 \rightarrow \varphi_2)$ and $((\neg\varphi_1) \vee \varphi_2)$ have the same meaning.

Let WF(L) denote the class of all well-formed formulas over L. This class includes the elements of SWF(L), as well as formulas involving the abbreviations identified above. From now on, we will use wff or wf as an abbreviation for well-formed formula.

Facts: Let $v : \mathsf{P} \rightarrow \{0,1\}$ be an interpretation of L.
Extend $\bar{v}$: WF(L) $\rightarrow \{0,1\}$ as follows.
(a) $\bar{v}((\varphi_1 \vee \varphi_2)) = \max(\bar{v}(\varphi_1), \bar{v}(\varphi_2))$.
(b) $\bar{v}((\varphi_1 \wedge \varphi_2)) = \min(\bar{v}(\varphi_1), \bar{v}(\varphi_2))$.
(c) $\bar{v}((\varphi_1 \equiv \varphi_2)) = 1 - \text{abs}(\bar{v}(\varphi_1) - \bar{v}(\varphi_2))$.
(d) $\bar{v}(\top) = 1$. $\square$

Models and Semantic Entailment:

Models:

- Given a wff φ,

  $\mathrm{Mod}(\varphi) = \{v \in \mathrm{Interp}(L) \mid \bar{v}(\varphi) = 1\}$.
- The notation $v \models \varphi$ is sometimes used to denote that $v \in \mathrm{Mod}(\varphi)$.
- Given a set Φ of wff's,

  $\mathrm{Mod}(\Phi) =$

  $\{v \in \mathrm{Interp}(L) \mid \bar{v}(\varphi) = 1 \text{ for each } \varphi \in \Phi\}$.

Semantic entailment:

- Write $\Phi \models \varphi$ to denote that $\mathrm{Mod}(\Phi) \subseteq \mathrm{Mod}(\varphi)$.

- Write $\models \varphi$ to denote that all interpretations belong to $\mathrm{Mod}(\varphi)$. (In terms of the definition below, φ is a tautology.)

Definitions:

- If $\mathrm{Mod}(\varphi) \neq \emptyset$, φ is *satisfiable.*

- If $\mathrm{Mod}(\varphi) = \emptyset$, φ is *unsatisfiable.*

- If $\mathrm{Mod}(\varphi) = \mathrm{Interp}(L)$, φ is called a *tautology.*

- φ_1 and φ_2 are *logically equivalent* if $\mathrm{Mod}(\varphi_1) = \mathrm{Mod}(\varphi_2)$.

Some logical equivalences:

The following equivalences are all a consequence of the fact that the truth system of propositional logic forms a Boolean algebra.

Zero of $\bot$:

$(\varphi \wedge \bot) \equiv \bot$

Identity of $\bot$:

$(\varphi \vee \bot) \equiv \varphi$

Identity of complements:

$(\varphi \wedge (\neg\varphi)) \equiv \bot$

Idempotence:

$(\varphi \wedge \varphi) \equiv \varphi$

Associative identity:

$((\varphi_1 \wedge \varphi_2) \wedge \varphi_3) \equiv ((\varphi_1 \wedge (\varphi_2 \wedge \varphi_3))$

Commutative identity:

$(\varphi_1 \wedge \varphi_2) \equiv (\varphi_2 \wedge \varphi_1)$

Distributive identity:

$((\varphi_1 \wedge (\varphi_2 \vee \varphi_3)) \equiv ((\varphi_1 \wedge \varphi_2) \vee (\varphi_1 \wedge \varphi_3))$

Absorbtion identity:

$((\varphi_1 \wedge (\varphi_1 \vee \varphi_2)) \equiv \varphi_1$

de Morgan’s identity:

$(\neg (\varphi_1 \wedge \varphi_2)) \equiv ((\neg\varphi_1) \vee (\neg\varphi_2))$

Duals of logical equivalences:

Meta-rule: Every one of the logical equivalences on the previous slide has an equally valid dual, obtained by replacing each connective by its dual, as summarized in the table below.

Connective	Dual
$\wedge$	$\vee$
$\vee$	$\wedge$
$\bot$	$\top$
$\top$	$\bot$
$\equiv$	$\equiv$
$\neg$	$\neg$

Examples:

The dual of de Morgan's identity:

$(\neg\ (\varphi_1 \vee \varphi_2)) \equiv ((\neg\varphi_1) \wedge (\neg\varphi_2))$

The dual of the identity of complements:

$(\varphi \vee (\neg\varphi)) \equiv \top$.

The dual of the zero of $\bot$:

$(\varphi \vee \top) \equiv \top$

This should perhaps be called the "unit of $\top$" identity.

Exercise: Identify the duals of the other identities, and understand their meanings.

Further notational conveniences:

- Because $\wedge$ and $\vee$ are associative, it is convenient to drop parentheses surrounding cascades of these operations.

  Example: $((\varphi_1 \wedge \varphi_2) \wedge \varphi_3)$ and $((\varphi_1 \wedge (\varphi_2 \wedge \varphi_3))$ may each be written as just $(\varphi_1 \wedge \varphi_2 \wedge \varphi_3)$.

- Because $\wedge$ and $\vee$ are commutative, it is convenient to consider pairs differing only in the order of the elements to be the same.

  Example: $(\varphi_1 \wedge \varphi_2)$ and $(\varphi_2 \wedge \varphi_1)$ are the same.

- Parentheses surrounding negated atoms may be omitted.

  Example: $((\neg\varphi_1) \wedge \varphi_2)$ may be written $(\neg\varphi_1 \wedge \varphi_2)$.

Precedence:

The most common precedence schedule is as follows:

Symbol	Arity	Precedence	Meaning
$\wedge$	2	2	Logical conjunction (and)
$\vee$	2	3	Logical disjunction (or)
$\neg$	1	1	Logical negation (not)
$\rightarrow$	2	4	Logical implication (implies)
$\leftrightarrow$ $\equiv$	2	4	Logical equivalence (iff)

However it is **not** a good idea to omit parentheses in formulas involving distinct binary operations. These precedences are not well known, and it is very easy to make a mistake.

Example: $(\varphi_1 \wedge \varphi_2 \vee \varphi_3)$ has the "official" meaning $((\varphi_1 \wedge \varphi_2) \vee \varphi_3)$, but these precedences are not well known, and it is very easy to be misunderstood. Do not use them in this course.

Rather, assume that the following simpler precedence rules hold, and use parentheses to resolve all ambiguities.

Symbol	Arity	Precedence	Meaning
$\wedge$	2	2	Logical conjunction (and)
$\vee$	2	2	Logical disjunction (or)
$\neg$	1	1	Logical negation (not)
$\rightarrow$	2	2	Logical implication (implies)
$\leftrightarrow$ $\equiv$	2	2	Logical equivalence (iff)

Complete sets of connectives:

We have already seen that all logical connectives may be realized with combinations of the elements of $\{\rightarrow, \neg\}$. Call such a set *complete.* It is easy to see that $\{\neg, \vee\}$ and $\{\neg, \wedge\}$ are also complete sets.

Interestingly, there are two single operations which form complete sets by themselves.

NAND (not and) or Sheffer stroke:

φ_1	φ_2	$(\varphi_1 \mid \varphi_2)$
0	0	1
0	1	1
1	0	1
1	1	0

NOR (not or) or Pierce arrow:

φ_1	φ_2	$(\varphi_1 \downarrow \varphi_2)$
0	0	1
0	1	0
1	0	0
1	1	0

These operations have application in combinational circuit design, as we shall see shortly.

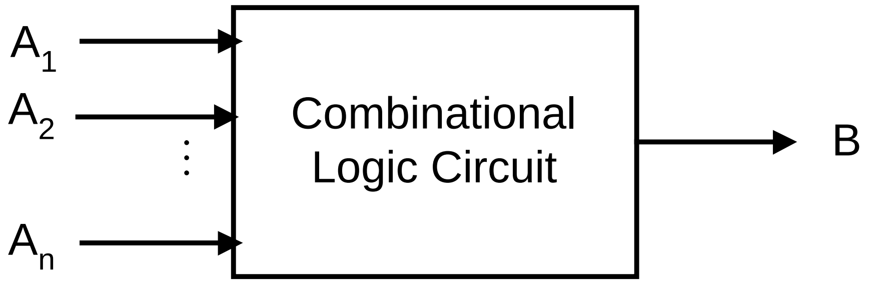
Exercise: Show these to be complete.

Truth functions and combinational logic:

In the design of computer hardware, one is often presented with the problem of designing a circuit which realizes a logical function

$$g: \{0,1\}^n \rightarrow \{0,1\}.$$

This can be envisioned as a problem in which one must build a circuit with n input lines and one output line, as illustrated below. Such devices are called combinational logic circuits.



Every wff which contains at most the propositions $\{A_1, A_2, .., A_n\}$ defines the behavior of such a circuit.

For example, suppose n=3, and that

$$\varphi = (A_1 \wedge (A_3 \rightarrow A_2))$$

The truth table of φ defines the behavior of the circuit.

A_1	A_2	A_3	$(A_1 \wedge (A_3 \rightarrow A_2))$
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

An extremely important question, from an engineering point of view, is that which asks which functions of the form

$$g: \{0,1\}^n \rightarrow \{0,1\}$$

may be realized via a wff φ, as in the preceding example. Amazingly, the answer is that all such functions may be so realized.

To illustrate this, let us reverse engineer the above design. We start with the following truth table.

A_1	A_2	A_3	B
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

For each row of the table which generates a 1 result, write down the conjunction ("and") of propositions and their negations which produces that result.   The fifth row of the table yields:

$$(A_1 \wedge \neg A_2 \wedge \neg A_3)$$

The seventh and eight yield:

$$(A_1 \wedge A_2 \wedge \neg A_3)$$

$$(A_1 \wedge A_2 \wedge A_3)$$

To complete the solution, we just form the disjunction ("or") of these results.

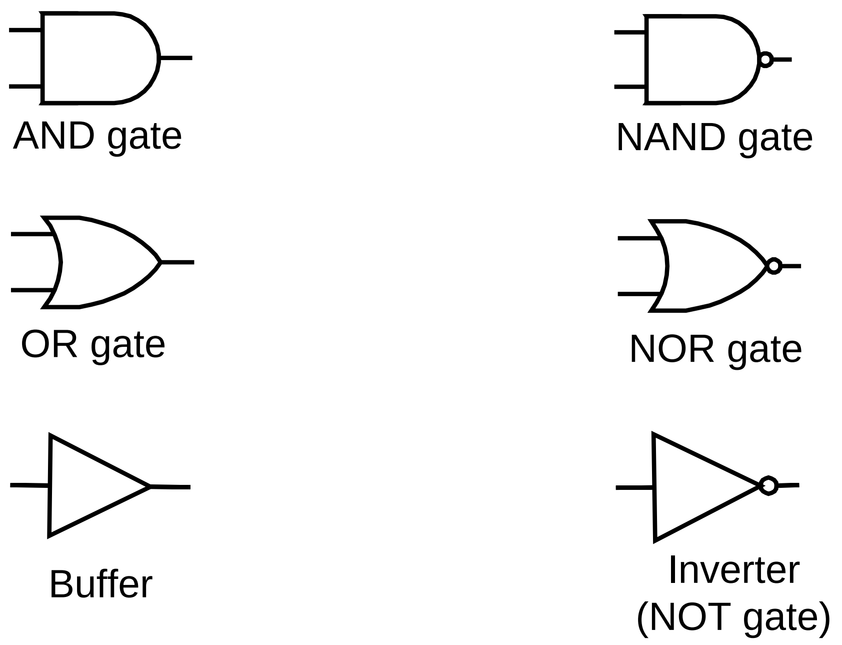
$$(A_1 \wedge \neg A_2 \wedge \neg A_3) \vee (A_1 \wedge A_2 \wedge \neg A_3) \vee (A_1 \wedge A_2 \wedge A_3)$$

It is easy to see that this formula is equivalent to

$$(A_1 \wedge (A_3 \rightarrow A_2)).$$

Combinational logic circuit elements:

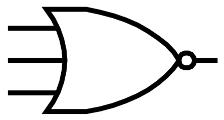
The following types of logic gates are commonly used in the design of combinational logic circuits for computers.



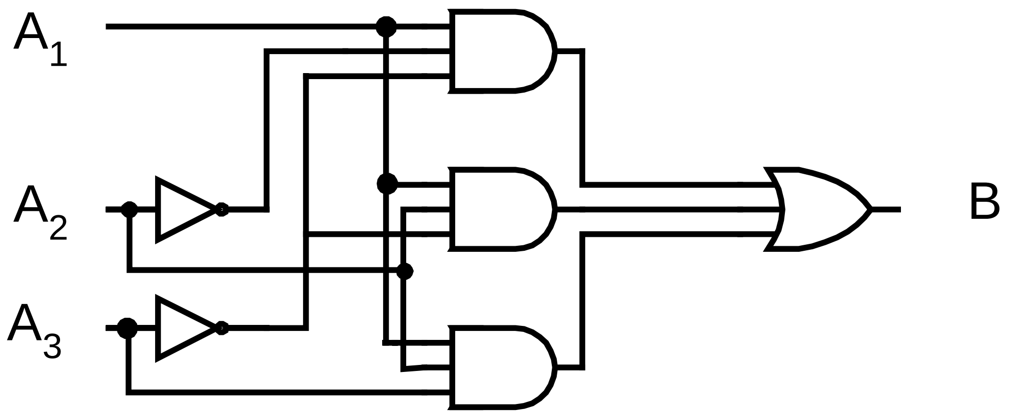
- NAND and NOR gates are the most commonly used, because of their completeness, and because such “inverting” circuits are the easiest to realize with a minimum number of transistors.

- A buffer is used just to amplify a weak signal. It is shown here as the complement of an inverter.

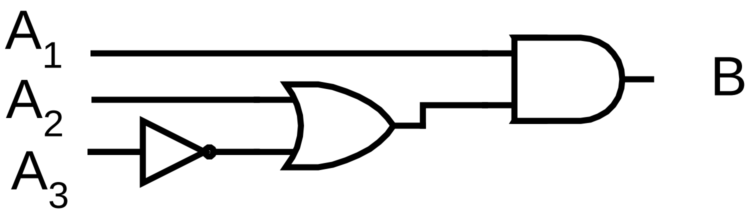
- Often, these gates are available with more than two inputs. Here is an example of a three-input NOR gate:



Here is a realization of
$(A_1 \wedge \neg A_2 \wedge \neg A_3) \vee (A_1 \wedge A_2 \wedge \neg A_3) \vee (A_1 \wedge A_2 \wedge A_3)$
using AND and OR gates (with three inputs), together with inverters.



Here is a simpler realization of the same function, but based upon $(A_1 \wedge (A_3 \rightarrow A_2))$



- Formulating a realization of such a circuit using NAND and/or NOR gates, and in particular with a minimum number of such gates, is a special science, which is studied in courses for computer hardware design (datorteknik).