A Minimal System of Propositional Logic

The Short Form

Language

Our "official" language of propositional logic, $\mathcal{L}_P$, is defined as follows:

- $\mathcal{L}_P$ has the following symbols:
  - *i.* the *atomic formulas* A_0, A_1, A_2, $\dots$;
  - *ii.* the *connectives* $\neg$ ["not"] and $\rightarrow$ ["if , then "]; and
  - *iii.* the *grouping symbols* (and).
- The *formulas* of $\mathcal{L}_P$ are then defined as follows:
  - *a.* Every atomic formula is a formula.
  - *b.* If α is a formula, then $(\neg\alpha)$ is a formula.
  - *c.* If α and β are formulas, then $(\alpha \rightarrow \beta)$ is a formula.
  - *d.* A string of symbols of $\mathcal{L}_P$ is a formula only if it is obtainedcan be built from the symbols of $\mathcal{L}_P$ by finitely many applications of rules a, b, and c.

Informally, we often use the connectives $\vee$ ["or"], $\wedge$ ["and"], and $\leftrightarrow$ ["if and only if"]. We can think of them as abbreviations:

$(\alpha \vee \beta)$ is an abbreviation for $((\neg\alpha) \rightarrow \beta)$.
$(\alpha \wedge \beta)$ is an abbreviation for $(\neg\,(\alpha \rightarrow (\neg\beta)))$.
$(\alpha \leftrightarrow \beta)$ is an abbreviation for $(\neg\,((\alpha \rightarrow \beta) \rightarrow (\neg(\beta \rightarrow \alpha))))$, *i.e.* for $(\alpha \rightarrow \beta) \wedge (\beta \rightarrow \alpha))$.

Semantics

The intended interpretation of the formulas in this system works as follows. We have two *truth values*, T ["true"] and F ["false"], which interact with the connectives according to the following *truth tables*:

α	$(\neg\alpha)$
T	F
F	T

and

α	β	$(\alpha \rightarrow \beta)$
T	T	T
T	F	F
F	T	T
F	F	T

If you know the truth values of all of the atomic formulas in a formula, you can apply these rules to work out the truth value of the formula.

One could, in principle, do proofs in this way. Suppose we are given a set, possibly empty, of formulas Σ, the *premisses* or *hypotheses*, and a formula α, the desired *conclusion*. Then "Σ *entails* α", often written as $\Sigma \vDash \alpha$, if every possible assignment of truth values to atomic formulas that makes every formula in Σ true, also makes α true. Since every additional atomic formula involved doubles the size of the truth table that must be checked, this is usually not a very efficient way to do proofs, though it is handy in small cases.

Proofs

Proofs in our system of propositional logic work as follows:

- The *logical axiom schema* of the system are:
  - *A*1. $(\alpha \rightarrow (\beta \rightarrow \alpha))$
  - *A*2. $((\alpha \rightarrow (\beta \rightarrow \gamma)) \rightarrow ((\alpha \rightarrow \beta) \rightarrow (\alpha \rightarrow \gamma)))$
  - *A*3. $(((\neg\beta) \rightarrow (\neg\alpha)) \rightarrow (((\neg\beta) \rightarrow \alpha) \rightarrow \beta))$

  Every instance of one these schema, where we plug some particular formulas of $\mathcal{L}_P$ in for α, β, and/or γ, is a *logical axiom*.

- The system has a single rule of procedure, *Modus Ponens*, usually abbreviated as MP when referred to in a deduction:

  MP. Given the formulas α and $(\alpha \to \beta)$, we may infer the formula β.

- *Deductions* or *formal proofs* are then defined as follows:

  Given a set, possibly empty, of formulas Σ, the *premisses* or *hypotheses*, a *deduction* of a formula α from Σ, is a sequence of formulas $\varphi_1, \varphi_2, \ldots, \varphi_n$ such that each formula φ_k in the sequence

  1. is a logical axiom,
  2. is in Σ (*i.e.* it is a premiss or hypothesis), or
  3. follows from some formulas φ_i and φ_j earlier in the sequence (so i, $j < k$),

  and φ_n, the last formula in the sequence, is α. "Σ *proves* α", often written as $\Sigma \vdash \alpha$, means that there is a deduction of α using Σ as the set of premisses.

Deductions are usually a more efficient way to proceed once the number of formulas involved is more than three or four. For many small cases, however, deductions could be rather more work. For example, suppose we wish to prove that for any formula φ, $(\varphi \to \varphi)$ is true. (Note that the set of hypotheses here is empty.) A deduction proving this that is as short as possible is:

1. $(\varphi \to ((\varphi \to \varphi) \to \varphi)) \to ((\varphi \to (\varphi \to \varphi)) \to (\varphi \to \varphi))$	A2
2. $\varphi \to ((\varphi \to \varphi) \to \varphi)$	A1
3. $(\varphi \to (\varphi \to \varphi)) \to (\varphi \to \varphi)$	1,2 MP
4. $\varphi \to (\varphi \to \varphi)$	A1
5. $\varphi \to \varphi$	3,4 MP

By comparison, doing this with a truth table is ridiculously short:

φ	$(\varphi \to \varphi)$
T	T
F	T

Thus, whether a truth assignment makes φ true or false, it must make $(\varphi \to \varphi)$ true.

There are some partly informal shortcuts available when using deductions in practice. First, if you prove something generic like $(\varphi \to \varphi)$, you can just cite the fact in later deductions without doing it from scratch. Second, one can often make use of the Deduction Theorem, which states that $\Sigma \vdash (\alpha \to \beta)$ if and only if $\Sigma \cup \{\alpha\} \vdash \beta$.

The Deduction Theorem would reduce the effort needed in the above example quite a bit. Instead of proving $\vdash (\varphi \to \varphi)$ directly, we could use the Deduction Theorem to prove $\{\varphi\} \vdash$ instead. This is much easier to do:

1. φ	Premiss

Yes, you can have one-line deductions! It does need the conclusion to be a premiss or a logical axiom.

In practice, we can freely use both deductions and truth table-based arguments, since they turn out to be equivalent. The Soundness and Completeness Theorems for propositional logic guarantee that $\Sigma \vdash \alpha$ if and only if $\Sigma \vDash \alpha$.