MAU22C00 Lecture 5

John Stalker

Trinity College Dublin

The grammar of zeroeth order logic, again

```
%start statement
%%
statement  : expression ;
expression : variable
           | ( expression binop expression )
           | [ expression binop expression ]
           | { expression binop expression }
           | ( ¬ expression )
           | [ ¬ expression ]
           | { ¬ expression } ;
variable   : letter | variable ! ;
letter     : p | q | r | s | u ;
binop      : ∧ | ∨ | ⊃ | ⊼ | ⊻ | ≡ | ≢ | ⊂ ;
```

Ambiguity?

This language is unambiguous. Why?

Any expression longer than a single variable is bracketed by parentheses of some type.

The rules which expand to such expressions all have a single Boolean operator joining one or two expressions.

There may be more operators in those expressions, but they are contained within their own set of parentheses.

We can identify which rule (alternate) was used by looking at the type of parentheses and the one operator which is not further parenthesised.

For example $[(\neg p) \wedge (q \vee r)]$ must have been generated by

```
[ expression ∧ expression ]
```

This allows top down parsing: start with the whole statement, figure out which rule generated it, figure out which rule generated the expression(s) within it, repeat until we reach the level of variables.

Extralinguistic conventions

When talking *about* the language I'll use the upper case letters P, Q, R, etc. to stand for expressions. This is useful when talking about the structure of expressions, e.g. we could have a rule of inference "in any statement any expression of the form $(P \wedge Q)$ can be replaced by $(Q \wedge P)$."

P, Q, and R could be replaced by variables like p, q, and r, but they could also be replaced by longer expressions like $(q \vee r)$, $(\neg q)$, and $\{[p \wedge (\neg q)] \vee [(\neg p) \wedge q]\}$.

Note that P, Q, etc. don't exist as tokens *within* the language.

Whenever I use this notation the different types of parentheses are considered interchangeable, so "in any statement any expression of the form $[P \wedge Q]$ can be replaced by $[Q \wedge P]$" is also implied, along with "in any statement any expression of the form $\{P \wedge Q\}$ can be replaced by $\{Q \wedge P\}$", etc.

Semantics

An interpretation of the language is an assignment of the values “true” and “false” to the variables, and to larger expressions as described by the truth tables:

P	Q	$(P \wedge Q)$
F	F	F
F	T	F
T	F	F
T	T	T

P	Q	$(P \vee Q)$
F	F	F
F	T	T
T	F	T
T	T	T

Semantics, continued

P	$(\neg P)$
F	T
T	F

P	Q	$(P \supset Q)$
F	F	T
F	T	T
T	F	F
T	T	T

Since every expression is built up using these four operations, *in an unambiguous way*, once we've assigned values to the variables this determines the value of any expression.

Example: $\{[(p \supset q) \wedge (q \supset r)] \supset (p \supset r)\}$

p	q	r	(p ⊃ q)	(q ⊃ r)	[(p ⊃ q) ∧ (q ⊃ r)]	(p ⊃ r)	{[(p ⊃ q) ∧ (q ⊃ r)] ⊃ (p
F	F	F	T	T	T	T	T
F	F	T	T	T	T	T	T
F	T	F	T	F	F	T	T
F	T	T	T	T	T	T	T
T	F	F	F	T	F	F	T
T	F	T	F	T	F	T	T
T	T	F	T	F	F	F	T
T	T	T	T	T	T	T	T

Tautologies and satisfiability

A statement which is true in every interpretation, i.e. no matter what truth values are assigned to its variables, is called a *tautology*.

The truth table on the previous slide shows that $\{[(p \supset q) \wedge (q \supset r)] \supset (p \supset r)\}$ is a tautology.

This is an informal proof. At this point no formal proof is possible, since we haven't introduced any axioms or rules of inference.

A statement which is true in some interpretation, i.e. for some assignment of truth values to its variables, is called *satisfiable*.

Every tautology is satisfiable but not every satisfiable statement is a tautology.

For example p is satisfiable but not a tautology.

Types of proofs in zeroeth order logic

- Informal
  - appeal to intuition, e.g. intuitively we know that any statement of the form $[P \vee (\neg P)]$ must be true.
  - truth tables, as above
  - other case by case analysis, as below
  - tableaux, to be introduced below
- Formal
  - traditional axiomatic formal systems, to be introduced later
  - natural deductive systems, also to be introduced later

Types of proofs in zeroeth order logic, continued

- Semiformal
  - a derivation from the axioms of a formal system using the given rules of inference plus derived rules of inference.
  - an informal argument plus an algorithm for converting that type of informal arguement into a formal proof in one of the formal systems, e.g. a tableau plus and an algorithm for generating natural deductive proofs from tableaux.

What is a derived rule of inference? Something which derives from a set of statements another statement, which can be shown (informally) to be derivable from them using the axioms and rules of inference of the formal system.

For example, if we have a rule of inference allowing us to replace every occurence of a single variable with an arbitrary expression then a derived rule might allow us to replace multiple variables at once. It's not a rule of inference but it's clear that anything you can derive with it you could also derive using the original rule repeatedly.

Case by case analysis without truth tables

When you want to prove something is true it's helpful to ask "how could this be false?".

For example, how could $\{[(\neg p) \wedge (\neg q)] \supset [\neg(p \vee q)]\}$ fail to be true?

It's a statement of the form $(P \supset Q)$, where P is $[(\neg p) \wedge (\neg q)]$ and Q is $[\neg(p \vee q)]$.

Remember our convention that different types of parentheses are considered interchangeable?

We can tell that it's a $\supset$ type expression because $\supset$ is the only operator inside only one set of parentheses.

For $(P \supset Q)$ to be false P would need to be true and Q would need to be false.

In our case, $[(\neg p) \wedge (\neg q)]$ needs to be true and $[\neg(p \vee q)]$ needs to be false.

Case by case analysis, continued

We “want” $[(\neg p) \wedge (\neg q)]$ to be true and $[\neg(p \vee q)]$ to be false.

If $[(\neg p) \wedge (\neg q)]$ is true then so are $(\neg p)$ and $(\neg q)$.

So both p and q are false.

If $[\neg(p \vee q)]$ is false then $(p \vee q)$ is true.

Then p is true or q is true.

But we already know both are false.

So $\{[(\neg p) \wedge (\neg q)] \supset [\neg(p \vee q)]\}$ can't fail to be true. In other words, it's a tautology.

Is this any faster than writing down a truth table? Probably not, but it generalises better.

Another example

How about $\{[p \supset (q \supset r)] \supset [(p \supset q) \supset (p \supset r)]\}$?

Again we have something of the form $(P \supset Q)$, so for this to be false P must be true and Q must be false, i.e. $[p \supset (q \supset r)]$ is true and $[(p \supset q) \supset (p \supset r)]$ is false.

$[(p \supset q) \supset (p \supset r)]$ is also of the form $(P \supset Q)$. For it to be false we need $(p \supset q)$ to be true and $(p \supset r)$ to be false.

$(p \supset r)$ is also of the form form $(P \supset Q)$. For it to be false we need p to be true and r to be false.

So $[p \supset (q \supset r)]$, $(p \supset q)$ and p are all true and r is false.

How can $[p \supset (q \supset r)]$ be true? It's of the form $(P \supset Q)$ and so can be true if P is false or Q is true.

In this case that means p is false or $(q \supset r)$ is true.

But p is true so $(q \supset r)$ must be true.

Another example, continued

Where are we? We wanted to show $\{[p \supset (q \supset r)] \supset [(p \supset q) \supset (p \supset r)]\}$ is true, so we assumed it was false and have found that $(q \supset r)$, $(p \supset q)$ and p must be true and r must be false.

If $(p \supset q)$ is true then p is false or q is true, but p is true so q is true.

If $(q \supset r)$ is true then q is false or r is true, but q is true and r is false, so we have a contradiction.

Therefore $\{[p \supset (q \supset r)] \supset [(p \supset q) \supset (p \supset r)]\}$ must be true.

The main problem with such arguments is keeping track of what we know and, if we need to split things into cases, which case or subcase we're in.

The tableau method was created as a bookkeeping device to keep track of this.

A tableau

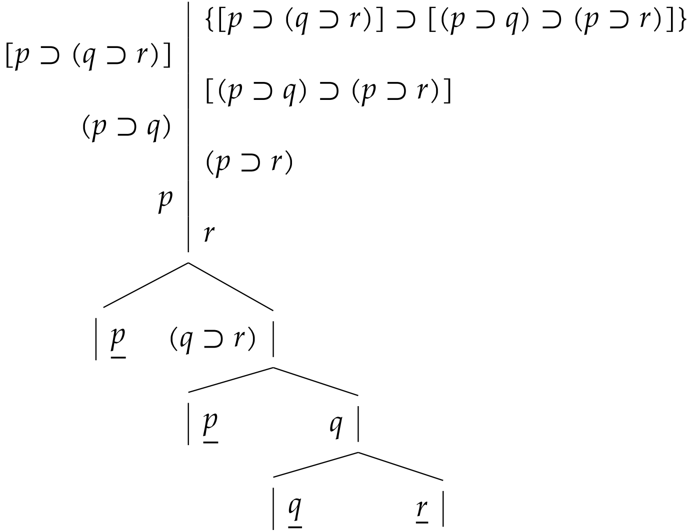

Figure 1: An analytic tableau