

## Assignment 1

### Natural Deduction and Structural Induction Due: 11am on Monday 29th August 2011

---

The submission of your assignment must be done via the assignment boxes in the student foyer. Failure to submit by the due date will result in late penalties of ten percent per weekday. Other arrangements that are required because of truly exceptional circumstances need to be negotiated before your deadline.

Your submission must be well-presented on clean A4 paper with a fully completed standard cover page, including your *tutor's name* and your *tutorial group*. Failure to follow this instruction will result in a penalty. The COMP2600 Assignments page has a link to an appropriate cover page.

## 1 Truth Tables in Propositional Logic

### Exercise 1.1

Construct truth-tables showing whether the following propositions are valid theorems (tautologies) or not; if not use the truth tables to find truth values for the variables which provide a counterexample.

$$(1) \neg(p \wedge q) \leftrightarrow (\neg p \vee \neg q)$$

$$(2) (\neg(p \rightarrow q) \vee \neg(q \rightarrow r)) \leftrightarrow (r \rightarrow p)$$

## 2 Natural Deduction in Propositional Logic

### Exercise 2.1

Construct natural deduction proofs for the following derived rules of propositional logic. You may only use the eight introduction and elimination rules discussed in lectures. Do not use truth tables or algebraic laws, or any of the “derived rules” obtained in lectures.

$$(1) \frac{\neg(p \vee q)}{\neg p \wedge \neg q}$$

$$(2) \frac{p \rightarrow (q \vee r) \quad q \rightarrow r}{p \rightarrow r}$$

### 3 Natural Deduction in Predicate Logic

#### Exercise 3.1

Construct natural deduction proof for the following derived rule of propositional logic. You may only use the introduction and elimination rules discussed in lectures. Do not use algebraic laws, or any of the “derived rules” obtained in lectures.

$$\frac{\exists x. P(x) \rightarrow Q}{(\forall x. P(x)) \rightarrow Q}$$

Note the implied bracketing of the top line; it is  $\exists x. (P(x) \rightarrow Q)$

### 4 Structural Induction

#### Exercise 4.1

Here is the usual Haskell definition of a binary tree:

```
data Tree a = Nul | Node a (Tree a) (Tree a)
```

We can count the values stored at the nodes by the Haskell function `count`. We can also count the “leaves”, the empty trees at the bottom, using the Haskell function `countNul`.

```
count Nul = 0 -- (C1)
```

```
count (Node x t1 t2) = 1 + count t1 + count t2 -- (C2)
```

```
countNul Nul = 1 -- (CN1)
```

```
countNul (Node x t1 t2) = countNul t1 + countNul t2 -- (CN2)
```

Prove, using structural induction on trees, that

```
countNul t = 1 + count t
```

#### Exercise 4.2

A tail-recursive function to sum the members of a list is given by `sumTR`. In fact `sumTR xs acc` adds all the members of the list `xs` to the number `acc`

```
sumTR [] acc = acc -- (STR1)
```

```
sumTR (x:xs) acc = sumTR xs (x + acc) -- (STR2)
```

Prove by induction the following property of `sumTR`

```
sumTR xs (sumTR ys acc) = sumTR (ys ++ xs) acc
```

Hint: you will need to think about whether to prove it by induction on `xs` or `ys`

State clearly what property  $P$  is being proved by induction, including any quantifiers needed in the statement of  $P$  and in the inductive hypothesis.