CS571

- Notes 11
- A sample axiomatic proof

Multiplication by repeated addition

```plaintext
p := 0;
n := x;
while n greater 0 do
  p := p plus y;
n := n minus 1
end
```

- x and y are placeholders for constants
- Pre-condition: x ≥ 0
- Post-condition: p = x*y
- Specification: [x ≥ 0, p = x*y]

Proof strategy

- Find a suitable invariant, I, for the loop
- Prove (I \land b) \subseteq \mathcal{C}(I) for the loop body
- Assert {I} while … {I \land \neg b}
- Work backwards and/or forwards from the loop to the beginning and/or end
- Prove: pre-condition implied by initial assertion, and post-condition implies the final assertion

Finding the invariant

- No known mechanical method
- Should include all variables that are assigned in the loop body
- Should include something like the loop exit test
- Should be as simple, but as strong as possible
- Any invariant will not yield a proof
- Can induce it from a trace of the loop:

<table>
<thead>
<tr>
<th>p</th>
<th>n</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>0</td>
<td>3</td>
</tr>
</tbody>
</table>

- So invariant is
- Add n ≥ 0 to give
- I is (p = (x-n)*y \land n ≥ 0)
Proving the loop

- Work backwards through the loop body:
  \{I[n-1\n]\}n := n minus1{I}, and
  \{I[p+y\p][n-1\n]\}p := p plus y{I[n-1\n]}
- i.e. \{p+y=(x-(n-1))^\*y \land (n-1)=0\} assigns{I}
- Pre-condition is \{p=(x-n)^\*y \land n>0\}
- This is exactly \{l \land b\}, so the loop axiom can be used to give:
  \{l\}while...{I \land ~b\}, or
- Post-condition is \{p=(x-n)^\*y \land n \geq 0 \land ~n>0\)
  which is p=x\*y, since n must be zero

Completing the proof

- \{I[x\n]\}n := x{I}
- \{I[x\n][0\p]\}p := 0{I[x\n]}
- Pre-condition is \{0=(x-x)^\*y \land x \geq 0\}
- This implied by x \geq 0, so the proof is complete.
- No strengthening or weakening was required,
  just th axioms, and basic manipulation of expressions to show equivalence.