COMP110: Principles of Computing

7: Algorithm Strategies

Recursion and induction

A boolean identity

$$\neg(X_1 \vee X_2 \vee \cdots \vee X_n) = \neg X_1 \wedge \neg X_2 \wedge \cdots \wedge \neg X_n$$

Proving the identity

- We can verify the formula for individual values of n
- (e.g. by drawing a truth table with all 2^n possible values of $X_1, \ldots, X_n$)
- How do we **prove** it for **all** n?
- We can use **proof by induction**

Case $n = 1$

$$\neg(X_1) = \neg X_1$$

Case $n = 2$

$$\neg(X_1 \vee X_2) = \neg X_1 \wedge \neg X_2$$

Exercise Sheet ii, question 3(a)

Case $n = k, k > 2$

- Suppose we have already proved the formula for all $n < k$
- Use this to show that the formula holds for $n = k$

$$\begin{aligned}
\neg(X_1 \vee X_2 \vee \cdots \vee X_k) &= \neg(X_1 \vee (X_2 \vee \cdots \vee X_k)) \\
&= \neg X_1 \wedge \neg(X_2 \vee \cdots \vee X_k) \text{ } (n = 2 \text{ case}) \\
&= \neg X_1 \wedge (\neg X_2 \wedge \cdots \wedge \neg X_k) \text{ } (n = k - 1 \text{ case})
\end{aligned}$$

Completing the proof

- We know:
  - The formula works for $n = 1$ and $n = 2$
  - If the formula works for $n = k - 1$, then it works for $n = k$
- The formula works for $n = 1$ and $n = 2$
- Therefore the formula works for $n = 2 + 1 = 3$
- Therefore the formula works for $n = 3 + 1 = 4$
- ...
- Therefore the formula works for all positive integers n

A formula for summation

$$\sum_{i=1}^{n} i = \frac{1}{2}n(n+1)$$

- $n = 1$: $1 = \frac{1}{2} \times 1 \times 2$
- $n = 2$: $1 + 2 = \frac{1}{2} \times 2 \times 3 = 3$
- $n = 3$: $1 + 2 + 3 = \frac{1}{2} \times 3 \times 4 = 6$
- ...

Proving the formula

- We can verify the formula for individual values of n
- How do we **prove** it for **all** n?
- We can use **proof by induction**

Proving the formula

Base case

- $n = 1$: $1 = \frac{1}{2} \times 1 \times 2$

Inductive assumption

- $\sum_{i=1}^{k-1} i = \frac{1}{2}(k-1)k$

Therefore

- $\sum_{i=1}^{k} i = \left(\sum_{i=1}^{k-1} i\right) + k$
- $= \frac{1}{2}(k-1)k + k$ (by inductive assumption)
- $= \frac{1}{2}k^2 - \frac{1}{2}k + k$
- $= \frac{1}{2}k^2 + \frac{1}{2}k$
- $= \frac{1}{2}k(k-1)$

So **if** the formula works for $n = k - 1$, **then** it works for $n = k$

Completing the proof

- We know:
  - The formula works for $n = 1$
  - If the formula works for $n = k - 1$, then it works for $n = k$
- The formula works for $n = 1$
- Therefore the formula works for $n = 1 + 1 = 2$
- Therefore the formula works for $n = 2 + 1 = 3$
- Therefore the formula works for $n = 3 + 1 = 4$
- ...
- Therefore the formula works for all positive integers n

Exercise

Prove

$$\sum_{i=1}^{n} i^2 = \frac{n(n+1)(2n+1)}{6}$$

Thinking inductively

- I want to prove something for all n
- Given k, if I had already proved $n = k - 1$ then I could prove $n = k$
- I can also prove $n = 1$
- Therefore by induction I can prove the result for all n

Recursion

- A **recursive** function is a function that **calls itself**

```
def factorial(n):
    if n <= 1:
        return 1
    else:
        return n * factorial(n-1)
```

Thinking recursively

- I want to solve a problem
- If I already had a function to solve smaller instances of the problem, I could use it to write my function
- I can solve the smallest possible problem
- Therefore I can write a recursive function

Exercise

- **Write** a pseudocode function to calculate the total size of all files in a directory and its subdirectories
- You may use the following functions in your pseudocode:
  - LISTDIR(directory): return a list of names of all files and folders in the given directory
  - GETSIZE(filename): return the size, in bytes, of the given file
  - ISDIR(name), ISFILE(name): determine whether the given name refers to a file or a directory

procedure CALCDIRSIZE(directory)
    ... ▷ return total size in bytes
end procedure