COMP250 - Lecture 12

Proofs by induction → Proving running time, correctness for recursive algs.

Sel. Sort : pick max → put in last pos.

"Silly" Selection Sort Recursive version

Algorithm SelectionSort (a, n)

Input : Array a of size n

Output : a should be sorted

```
// "Base case"
if n ≤ 1 return;
int indmax ← findMaxIndex(a, n)
swap (a, indmax, n)
selectionSort (a, n-1)
return;
```

"Tail recursion" → compiler turns such things into loops (for)

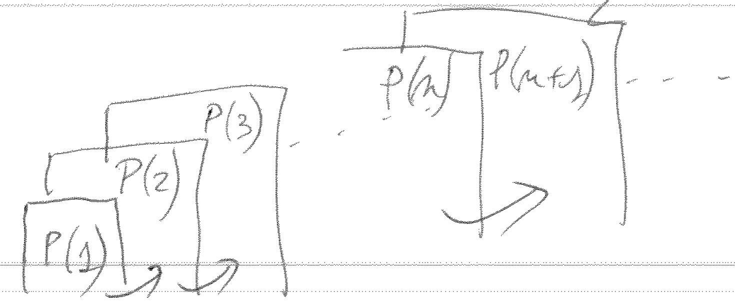
"Base case" for code → "Base case for induction

Recursive call → Induction step

Proof by induction

For all natural numbers n, $P(n)$ is true

$$1+2+\dots+n = \frac{n(n+1)}{2} \quad \forall n \geq 1$$



Show $P(1)$ true (base case)

Show if $P(n)$ true, then $P(n+1)$ must also be true (Induction step)

Later on: do this with more complicated structures

E.g. Show that $\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$

Base case: $n=1$ $\quad 1 = \frac{1 \cdot \cancel{2}}{\cancel{2}}$ Yay!

Induction step: suppose $\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$ "$P(n)$"

to show $\sum_{i=1}^{n+1} i = \frac{(n+1)(n+2)}{2}$ "$P(n+1)$"

$$\sum_{i=1}^{n+1} i = \underbrace{\sum_{i=1}^{n} i}_{} + (n+1) \underset{\text{ind hyp}}{=} \frac{n(n+1)}{2} + (n+1) = \frac{(n+1)(n+2)}{2}$$

L12-3

E.g. Show that $2^n \leq n! \;\; \forall n \geq 4$

$n! = 1 \cdot 2 \cdot 3 \cdots n$

Base case: $n=4$ $\quad 2 \cdot 2 \cdot 2 \cdot 2 \leq 1 \cdot 2 \cdot 3 \cdot 4$

$16 \leq 24$ yay!

Induction step: Suppose $2^n \leq n!$

To show: $2^{n+1} \leq (n+1)!$

$$2^n \cdot 2 \overset{?}{\leq} n! \cdot (n+1)$$

We know from ind hyp $2^n \leq n!$

$$\frac{\begin{array}{r} 2^n \leq n! \\ 2 \leq (n+1) \end{array}}{2^{n+1} \leq (n+1)!}$$

Yay!

E.g. Prove that $1^3+2^3+\dots+n^3=(1+2+\dots+n)^2$

Base case: $n=1$ $\quad 1^3=1^2$ Yay!

Induction step: Suppose $\sum_{i=1}^{n} i^3 = \left(\sum_{i=1}^{n} i\right)^2$

$$\sum_{i=1}^{n+1} i^3 = \underbrace{\sum_{i=1}^{n} i^3}_{\text{ind hyp}} + (n+1)^3 = \left(\sum_{i=1}^{n} i\right)^2 + (n+1)^3$$

$$= \underbrace{\left(\sum_{i=1}^{n} i\right)^2}_{} + (n+1)\underbrace{(n+1)^2}_{} =$$

$$n(n+1)^2 + (n+1)^2$$

Aiming for: $\left(\sum_{i=1}^{n} i + (n+1)\right)^2$

$$= \left(\sum_{i=1}^{n} i\right)^2 + (n+1)^2 + \underbrace{n(n+1)(n+1)}_{2\cdot\underbrace{\frac{n(n+1)}{2}}_{}(n+1)}$$

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

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

$$= \left(\sum_{i=1}^{n} i + (n+1)\right)^2 = \left(\sum_{i=1}^{n+1} i\right)^2 \quad \text{Yay!}$$

Blue print for proving correctness of recursive algs

Prove base call is correct

"Put our faith in the recursion!"

Assume recursive call worked correctly → prove that where we called from also works

Sorting: $a[j] \le a[i] \quad \forall j \le i$ Correctness property

Base case: $n = 1$ "Smallest input structure"
Trivially true

Induction step:

int indmax ← findMaxIndex(a, n)

↳ $a[indmax] \ge a[i] \ \forall i \ne indmax$ by correctness of findMaxIndex

swap(a, n, indmax)

↳ $a[n] \ge a[i] \ \forall i \le n$

selectionSort(a, n-1)

↳ induction hypothesis: rec. call worked so:

$\forall j \le i, \ i \le n-1, \quad a[j] \le a[i]$

$\Rightarrow$

$\Rightarrow \forall j \le i, \ i \le n \quad a[j] \le a[i]$

L12-6

- Comparing this proof technique to loop invariants, this is more "automated" (no need to guess at a loop invariant)
- What is true before/after each piece (fn. calls eg)

  what is true before: precondition

  what is true after: post condition

  At any point, we have a "collection" of statements true so far → deduce new ones

  Programming languages ~~may~~ can produce proofs in some cases (res. at McGill on this)