Lecture 6

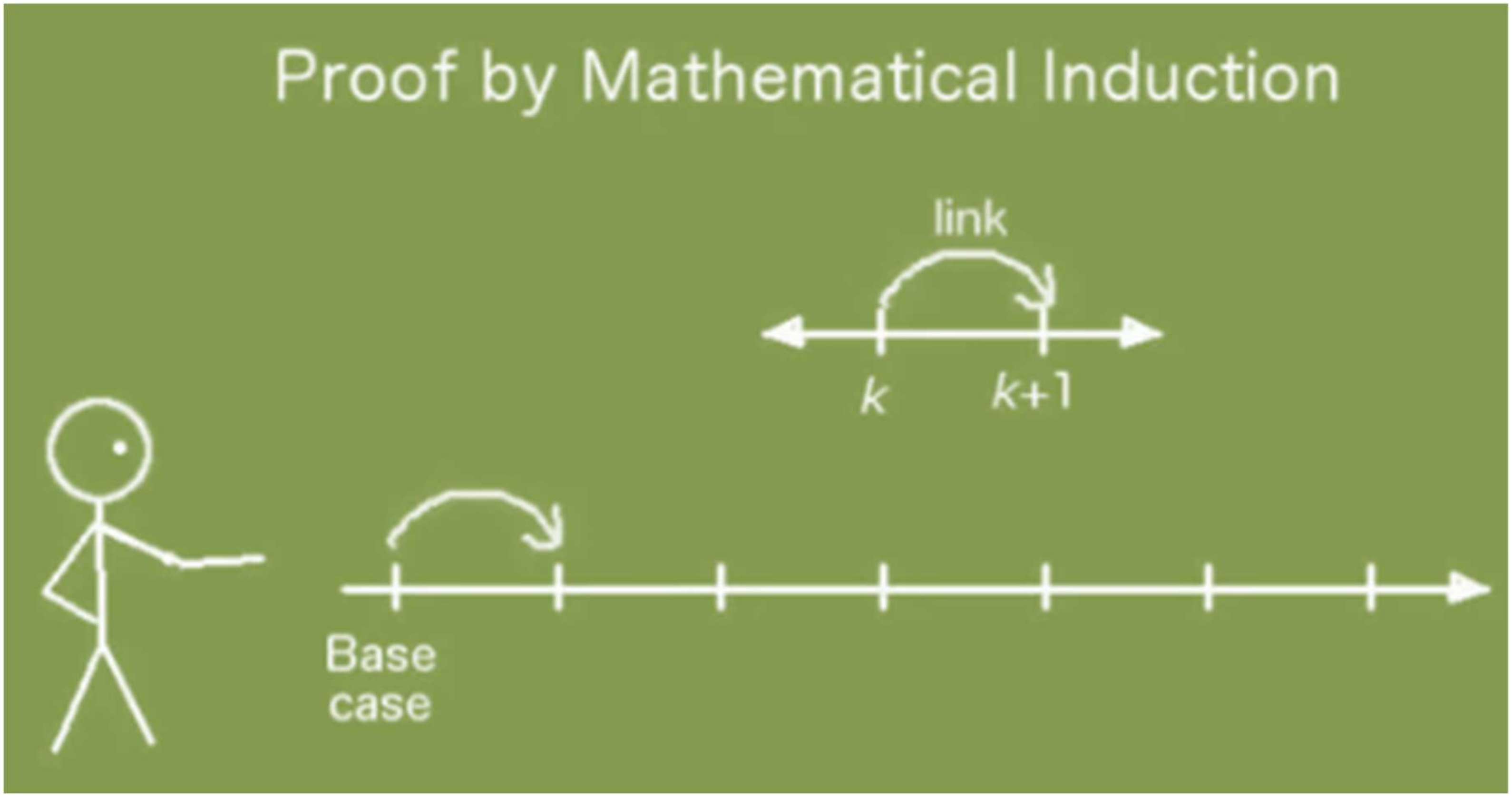
Correctness of Recursive Algorithm: Quick Review of Mathematical Induction, Proving Correctness of Recursive Algorithm using Induction, and Illustrative Examples.

Review of Mathematical Induction

A Quick Review

Not everything that can be counted counts, and not everything that counts can be counted.
—Albert Einstein (1879–1955)

Using induction, we can conclude infinitely many statements are true just by checking two things.



An **infinite sequence** if it is stated for n numbers, by induction method it is to be proved that for any **natural number** it is true. Mathematical Induction two steps. (1) **Base Step**; (2) **Inductive Step.**

Principle of Mathematical Induction

Let $P(n)$ be a predicate defined for integers n.

Suppose the following statements are true:

1. *Basis step:*

   $P(a)$ *is true for some fixed* $a \in \boldsymbol{Z}$.

2. *Inductive step:* *For all integers* $k \geq a$,

   if $P(k)$ *is true then* $P(k+1)$ *is true.*

Then for all integers $n \geq a$, $P(n)$ *is true.*

Example: Sum of Odd Integers

- ***Proposition:*** $1 + 3 + \ldots + (2n-1) = n^2$

  for all integers $n \geq 1$.

- ***Proof (by induction):***

  1) Basis step:

  The statement is true for $n=1$: $1=1^2$.

  2) Inductive step:

  Assume the statement is true for some $k \geq 1$

  (inductive hypothesis),

  show that it is true for $k+1$.

Example: Sum of Odd Integers

- ***Proof (cont.):***

The statement is true for k:

$$1+3+\ldots+(2k-1) = k^2 \qquad \textbf{(1)}$$

We need to show it for k+1:

$$1+3+\ldots+(2(k+1)-1) = (k+1)^2 \qquad \textbf{(2)}$$

Showing (2):

$$1+3+\ldots+(2(k+1)-1) = 1+3+\ldots+(2k+1) = \qquad \text{by (1)}$$

$$1+3+\ldots+(2k-1)+(2k+1) = k^2+(2k+1) = (k+1)^2 .$$

We proved the basis and inductive steps,

so, we conclude that the given statement true. ■

PRINCIPLE OF MATHEMATICAL INDUCTION:

Let P(n) be a propositional function defined for all positive integers n. P(n) is true for every positive integer n if

1.Basis Step:

The proposition P(1) is true.

2.Inductive Step:

If P(k) is true, then P(k + 1) is true for all integers $k \geq 1$.

i.e. $\forall k \quad p(k) \rightarrow P(k+1)$

EXAMPLE:

Use Mathematical Induction to prove that

$$1+2+3+\cdots+n=\frac{n(n+1)}{2} \quad \text{for all integers } n \geq 1$$

SOLUTION:

Let

$$P(n): 1+2+3+\cdots+n=\frac{n(n+1)}{2}$$

1.Basis Step: P(1) is true.

For n = 1, left hand side of P(1) is the sum of all the successive integers starting at 1 and ending at 1, so LHS = 1 and RHS is

$$R.H.S=\frac{1(1+1)}{2}=\frac{2}{2}=1$$

so, the proposition is true for n = 1.

2. Inductive Step: Suppose P(k) is true for, some integers $k \geq 1$.

(1) $1+2+3+\cdots+k=\dfrac{k(k+1)}{2}$

To prove P(k + 1) is true. That is,

(2) $1+2+3+\cdots+(k+1)=\dfrac{(k+1)(k+2)}{2}$

Consider L.H.S. of (2), $1+2+3+\cdots+(k+1) = 1+2+3+\cdots+k+(k+1)$

$$= \frac{k(k+1)}{2}+(k+1) \qquad \text{using (1)}$$

$$= (k+1)\left[\frac{k}{2}+1\right]$$

$$= (k+1)\left[\frac{k+2}{2}\right]$$

$$= \frac{(k+1)(k+2)}{2} = \text{RHS of (2)}$$

Hence by principle of Mathematical Induction the given result true for all integers greater or equal to 1.

Correctness of Recursive Algorithm

- To prove the correctness of a recursive algorithm
  - Mathematical induction is used.
- In a mathematical induction we want to prove a statement P(n) for all natural numbers
  - n (possibly starting at an n_0)
  - Also proving the statement for all n≥1).

Example 1 (Binary search algorithm). *Consider the following recursive implementation of binary search algorithm:*

```
function RecBSearch(x, A, s, f)
    if s == f then
        if x == A[s] then
            return s
        else
            return −1
        end if
    else
        m = (s + f) / 2        ▷ Integer Division
        if x ≤ A[m] then
            return RecBSearch(x, A, s, m)
        else
            return RecBSearch(x, A, m + 1, f)
        end if
    end if
end function
```

Precondition:

1. Elements of A comparable with each other and with x
2. Assume array indices start at 0 and hence $0 \leq s \leq f < length(A)$
3. Array A isorted in nondecreasing order ($A[s] \leq \cdots \leq A[f]$)

Postcondition: RecBSearch(x,A,s,f) terminates and returns index p such that:

1. $s \leq p \leq f$ or $p = -1$
2. If $s < p$, then $A[p-1] < x$
3. If $s \leq p \leq f$, then $x = A[p]$

Correctness of Recursive Algorithm

› To prove correctness of the algorithm

Proof. By induction on size $n = f + 1 - s$, we prove (precondition and execution) implies (termination and postcondition). Inductive structure of proof will follow recursive structure of algorithm.

Base case: $n = 1$, i.e., $s = f$. Then, algorithm terminates (lines 2-7 contain no loop or call), and returns s if $x = A[s]$, -1 if $x \neq A[s]$, which satisfies postcondition.

Induction Step: Let $n > 1$ and suppose postcondition holds after execution for all inputs of size k that satisfy precondition, for $1 \leq k < n$ (IH). Consider call RecBSearch(x,A,s,f) when $f + 1 - s = n \geq 2$. Test on line 2 fails, so $s < f$ (since $s \leq f$ by precondition and $s \neq f$ by negation of test) and algorithm executes line 9. Next, test on line 10 executes.

Case 1 ($x \leq A[m]$): Because $m < f$ then $m + 1 - s < f + 1 - s$ and hence by IH, RecBSearch(x,A,s,m) returns index p such that:

1. $s \leq p \leq m$ or $p = -1$
2. if $s < p$, then $A[p - 1] < x$

Correctness of Recursive Algorithm

3. if $s \leq p \leq m$, then $x = A[p]$

Hence,

1. $s \leq p \leq f$ (since $m < f$) or $p = -1$
2. if $s < p$, then $A[p-1] < x$ since $p \leq m$ and because of $\boxed{\text{IH}}$ Inductive Hypothesis
3. becuase we recursed on the first half then if $s \leq p \leq f$ then $p \leq m$, and by IH for $s \leq p \leq m$ then $x = A[p]$

Therefore, current call satisfies postcondition.

Case 2 ($A[m] < x$): Because $s \leq m$ then $s < m+1$ so $f+1-(m+1) < f+1-s$ and hence by IH, RecBSearch(x,A,$m+1$,f) returns index p such that:

1. $m+1 \leq p \leq f$ or $p = -1$
2. if $m+1 < p$, then $A[p-1] < x$
3. if $m+1 \leq p \leq f$ then $x = A[p]$

Hence,

1. $s \leq p \leq f$ (since $s < m+1$) or $p = -1$
2. if $s < p$ then we know that $m+1 \leq p$ since we recursed on the second half. By IH $A[p-1] < x$ for $m+1 < p$ and by the test of line 10, $A[m] < x$ in this case. Therefore, if $s < p$ then $A[p-1] < x$
3. if $s \leq p \leq f$ then $m+1 \leq p \leq f$ since we recursed on the second half and by IH $x = A[p]$

Therefore, current call satisfies postcondition. In all cases, current call satisfies postcondition. Therefore, by induction, RecBSearch is correct.

Correctness of Recursive Algorithm

Example 2. *In this example we prove the correctness of MergeSort algorithm.*

```
function MergeSort(A,s,f)
    if s == f then
        return
    else
        m = (s + f)/2                ▷ Integer Division
        MergeSort(A,s,m)
        MergeSort(A,m+1,f)
        # merge sorted A[s..m] and A[m + 1..f] back into A[s..f]
        for i = s, ··· , f do
            B[i] = A[i]
        end for
        c = s
        d = m + 1
        for i = s, ··· , f do
            if d > f or (c ≤ m and B[c] < B[d]) then
                A[i] = B[c]
                c = c + 1
            else                     ▷ d ≤ f and (c > m or B[c] ≥ B[d])
                A[i] = B[d]
                d = d + 1
            end if
        end for
    end if
end function
```

Precondition:

1. $s, f \in \mathbb{N}$, $0 \leq s \leq f < length(A)$
2. elements of $A[s..f]$ comparable with each other

Postcondition: $A[s..f]$ contains same elements as before, but sorted in non-decreasing order ($A[s] \leq \cdots \leq A[f]$)

Correctness of Recursive Algorithm

Proof. By induction on size $n = f + 1 - s$, we prove precondition and execution implies termination and postcondition, for all inputs of size n. Once again, the inductive structure of proof will follow recursive structure of algorithm.

Base case: Suppose (A,s,f) is input of size $n = f - s + 1 = 1$ that satisfies precondition. Then, $f = s$ so algorithm terminates and returns A unchanged (on line 2), which satisfies postcondition.

Induction Step: Suppose $n > 1$ and, for $1 \leq k < n$, for all inputs of size k that satisfy precondition, algorithm terminates and postcondition holds after execution (IH). Suppose (A,s,f) is input of size $n = f - s + 1 > 1$ that satisfies precondition, and consider call MergeSort(A,s,f). Test on line 2 fails because $f - s + 1 > 1$ iff $f > s$ and hence the algorithm executes line 5. Since $s \leq \lfloor \frac{s+f}{2} \rfloor < f$, IH implies that MergeSort(A,s,m) terminates and the output $A[s..m]$ contains same elements as input $A[s..m]$ but sorted in non-decreasing order. For the same reason, MergeSort(A,$m+1$,f) terminates and output $A[m+1..f]$ contains same elements as input $A[m+1..f]$ but sorted in non-decreasing order. Lines 9-11 copies $A[s..f]$ into $B[s..f]$ (exercise: prove this). Lines 12-22 merge $B[s..m]$ and $B[m+1..f]$ into $A[s..f]$, which satisfies postcondition.

Thank You!!!

Have a good day