Discrete Structures for Computing

CSCE 222

Sandeep Kumar

Many slides based on [Lee19], [Rog21], [GK22]

Induction and Recursion

Chapter 5

Chapter Summary

- Mathematical Induction
- Strong Induction
- Well-Ordering
- Recursive Definitions
- Structural Induction
- Recursive Algorithms

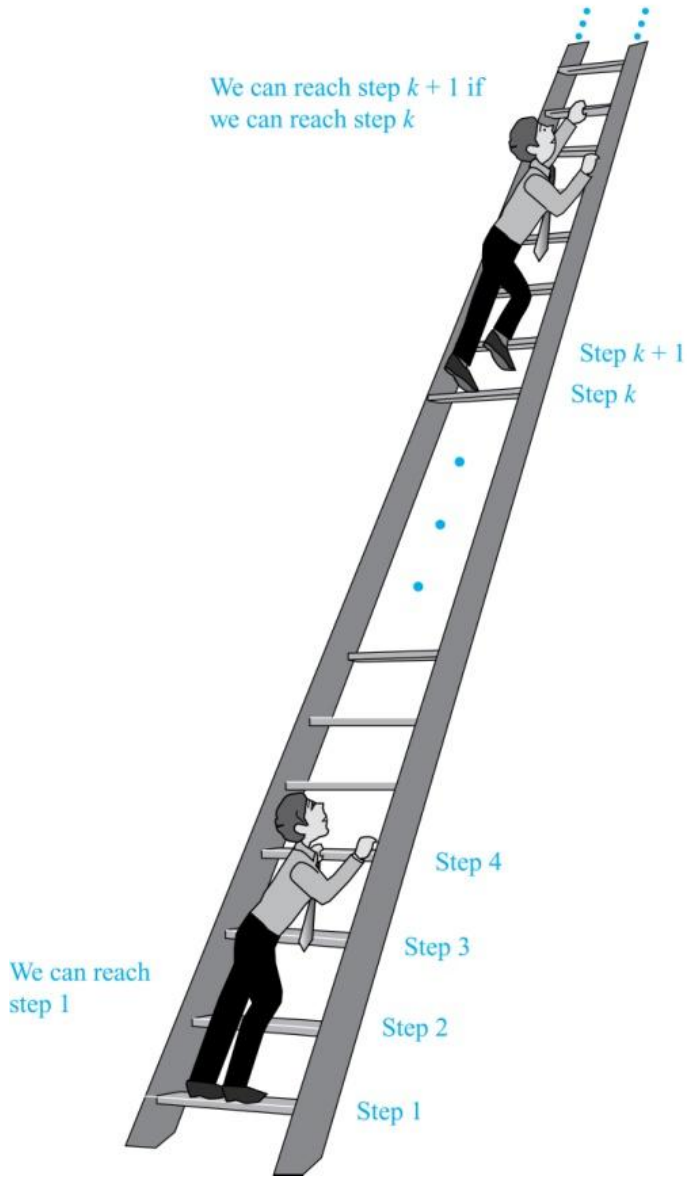
Section Summary

- Mathematical Induction
- Examples of Proof by Mathematical Induction
- Mistaken Proofs by Mathematical Induction
- Guidelines for Proofs by Mathematical Induction

Climbing an Infinite Ladder

Suppose we have an infinite ladder:

1. We can reach the first rung of the ladder.
2. If we can reach a particular rung of the ladder, then we can reach the next rung.
3. From (1), we can reach the first rung.
   - Then by applying (2), we can reach the second rung.
   - Applying (2) again, the third rung. And so on.
   - We can apply (2) any number of times to reach any particular rung, no matter how high up.
4. This example motivates proof by mathematical induction.



Principle of Mathematical Induction

Principle of Mathematical Induction: To prove that $P(n)$ is true for all positive integers n, we complete these steps:

- *Basis Step*: Show that $P(1)$ is true.
- *Inductive Step*: Show that $P(k) \rightarrow P(k+1)$ is true for all positive integers k.

To complete the inductive step, assuming the *inductive hypothesis* that $P(k)$ holds for an arbitrary integer k, show that $P(k+1)$ must be true.

Climbing an Infinite Ladder:

- BASIS STEP: By (1), we can reach rung 1.
- INDUCTIVE STEP: Assume the inductive hypothesis that we can reach rung k. Then by (2), we can reach rung $k+1$.

Hence, $P(k) \rightarrow P(k+1)$ is true for all positive integers k. We can reach every rung on the ladder.

Important Points About Using Mathematical Induction

Mathematical induction can be expressed as the rule of inference

$$P(1) \wedge \forall k(P(k) \rightarrow P(k+1)) \rightarrow \forall n\, P(n)$$

where the domain is the set of positive integers.

In a proof by mathematical induction, we don't assume that $P(k)$ is true for all positive integers! We show that if we assume that $P(k)$ is true, then $P(k+1)$ must also be true.

Proofs by mathematical induction do not always start at the integer 1. In such a case, the basis step begins at a starting point b where b is an integer. We will see examples of this soon.

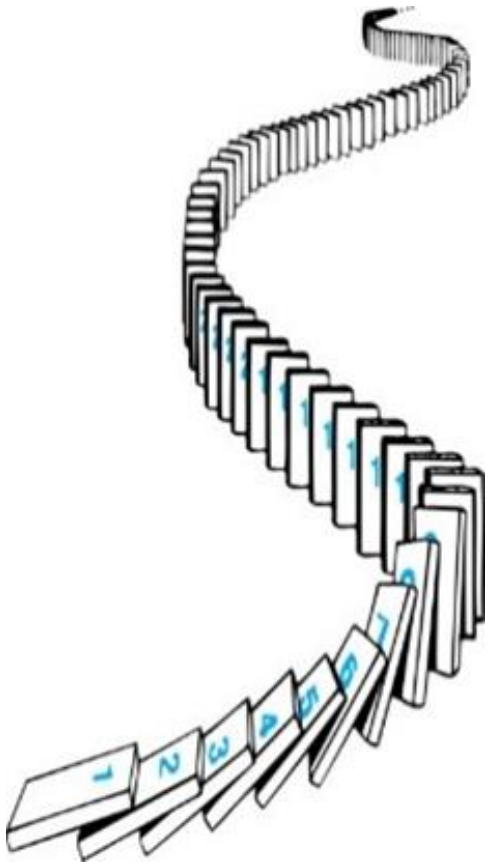
Validity of Mathematical Induction

Mathematical induction is valid because of the *well ordering property*, which states that every nonempty subset of the set of positive integers has a least element (see Section 5.2 and Appendix 1).

- Suppose that $P(1)$ holds and $P(k) \rightarrow P(k+1)$ is true for all positive integers k.
- Assume there is at least one positive integer n for which $P(n)$ is false. Then the set S of positive integers for which $P(n)$ is false is nonempty.
- By the well-ordering property, S has a least element, say m.
- We know that $m \neq 1$ since $P(1)$ holds.
- Since $m > 1$, $(m-1)$ must be a positive integer. Since $(m-1) < m$, it is not in S, so $P(m-1)$ must be true.
- But then, since the conditional $P(k) \rightarrow P(k+1)$ for every positive integer k holds, $P(m)$ must also be true. This contradicts $P(m)$ being false.
- Hence, $P(n)$ must be true for every positive integer n.

Remembering How Mathematical Induction Works

- Consider an infinite sequence of dominoes, labeled $1, 2, 3, \ldots$ where each domino is standing.
- Let $P(n)$ be the proposition that the n^{th} domino is knocked over.
- We know that the first domino is knocked down, i.e., $P(1)$ is true.
- We also know that whenever the k^{th} domino is knocked over, it knocks over the $(k+1)^{st}$ domino, i.e, $P(k) \rightarrow P(k+1)$ is true for all positive integers k.
- Hence, all dominos are knocked over.
- $P(n)$ is true for all positive integers n.



Examples of Mathematical Induction

- Rain on Mars.
  - Today it is raining on Mars.
  - On Mars, it never rains on any day without raining the next day as well.
- Delivering milk.
  - Never leave milk on one day without leaving milk the next day as well.
  - Does this imply that milk will always be delivered?

  What we want is:
  - Never leave milk on one day without leaving milk the next day as well.
  - Leave milk today.

  Let's use recursion:
  - Leave milk today and read this note again tomorrow.

Proving a Summation Formula by Mathematical Induction

Show that[1]

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

- BASIS STEP: $P(1)$ is true since $1(1+1)/2 = 1$.
- INDUCTIVE STEP: Assume true for $P(k)$.

  The inductive hypothesis is $\sum_{i=1}^{k} = \frac{k(k+1)}{2}$. Then,

$$\begin{aligned} 1+2+\ldots+k+(k+1) &= \frac{k(k+1)}{2} + (k+1) \\ &= \frac{k(k+1)+2(k+1)}{2} \\ &= \frac{(k+1)(k+2)}{2} \end{aligned}$$

[1] Once we have this conjecture, mathematical induction can be used to prove it correct.

Conjecturing and Proving Correct a Summation Formula

Conjecture and prove correct a formula for the sum of the first n positive odd integers. We have:

$$1 = 1$$
$$1 + 3 = 4$$
$$1 + 3 + 5 = 9$$
$$1 + 3 + 5 + 7 = 16$$
$$1 + 3 + 5 + 7 + 9 = 25$$

- We can conjecture that the sum of the first n +ve odd integers is n^2,

$$1 + 3 + 5 + \cdots + (2n - 1) = n^2$$

- We prove the conjecture using mathematical induction.

Conjecturing and Proving Correct a Summation Formula

- BASIS STEP: $P(1)$ is true since $1^2 = 1$.
- INDUCTIVE STEP: $P(k) \rightarrow P(k+1)$ for every positive integer k.

  Assume the inductive hypothesis holds and then show that $P(k+1)$ holds has well.
- So, assuming $P(k)$, it follows that:

$$\begin{aligned} 1+3+5+\cdots+(2k-1)+(2k+1) &= \overbrace{[1+3+5+\cdots+(2k-1)]}^{P(k)}+(2k+1) \\ &= k^2+(2k+1) \text{ by the induction hypothesis} \\ &= k^2+2k+1 \\ &= (k+1)^2 \end{aligned}$$

- Hence, we have shown that $P(k+1)$ follows from $P(k)$. Therefore the sum of the first n positive odd integers is n^2.

Proving Inequalities

Use mathematical induction to prove that $n < 2^n$ for all positive integers n.

Let $P(n)$ be the proposition that $n < 2^n$.

- BASIS STEP: $P(1)$ is true since $1 < 2^1 = 2$.
- INDUCTIVE STEP: Assume $P(k)$ holds, i.e., $k < 2^k$, for an arbitrary positive integer k.
- Must show that $P(k+1)$ holds. Since by the inductive hypothesis, $k < 2^k$, it follows that:

$$k + 1 < 2^k + 1 \leq 2^k + 2^k = 2.2^k = 2^{k+1}$$

Therefore $n < 2^n$ holds for all positive integers n.

Proving Inequalities...

Use mathematical induction to prove that $2^n < n!$, for every integer $n \geq 4$.

Let $P(n)$ be the proposition that $2^n < n!$.

- BASIS STEP: $P(4)$ is true since $2^4 = 16, 4! = 24$.
- INDUCTIVE STEP: Assume $P(k)$ holds, i.e., $2^k < k!$ for an arbitrary integer $k \geq 4$. To show that $P(k+1)$ holds:

$$\begin{aligned} 2^{k+1} &= 2 \cdot 2^k \\ &< 2 \cdot k! \\ &< (k+1)k! \\ &< (k+1)! \end{aligned}$$

Therefore, $2^n < n!$ holds, for every integer $n \geq 4$. Note that here the basis step is $P(4)$, since $P(0)$, $P(1)$, $P(2)$, and $P(3)$ are all false.

Proving Divisibility Results

Use mathematical induction to prove that $n^3 - n$ is divisible by 3, for every positive integer n.

Let $P(n)$ be the proposition that $n^3 - n$ is divisible by 3.

- BASIS STEP: $P(1)$ is true since $1^3 - 1 = 0$, which is divisible by 3 .
- INDUCTIVE STEP: Assume $P(k)$ holds,
  - I.e., $k^3 - k$ is divisible by 3, for an arbitrary positive integer k.
  - To show that $P(k+1)$ follows:

$$\begin{aligned}(k+1)^3 - (k+1) &= \left(k^3 + 3k^2 + 3k + 1\right) - (k+1) \\ &= \left(k^3 - k\right) + 3\left(k^2 + k\right)\end{aligned}$$

- By the inductive hypothesis,
  - The first term $(k^3 - k)$ is divisible by 3 and,
  - The second term is divisible by 3 since it is an integer multiplied by 3.
  - So by part (i) of Theorem 1 in Section 4.1, $(k+1)^3 - (k+1)$ is divisible by 3.

Therefore, $n^3 - n$ is divisible by 3, for every integer positive integer n.

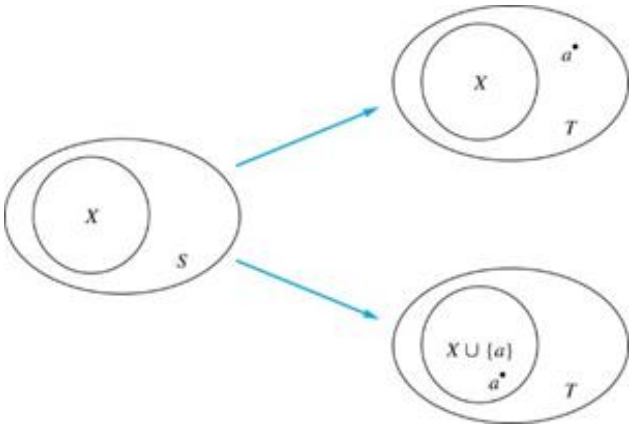
Number of Subsets of a Finite Set

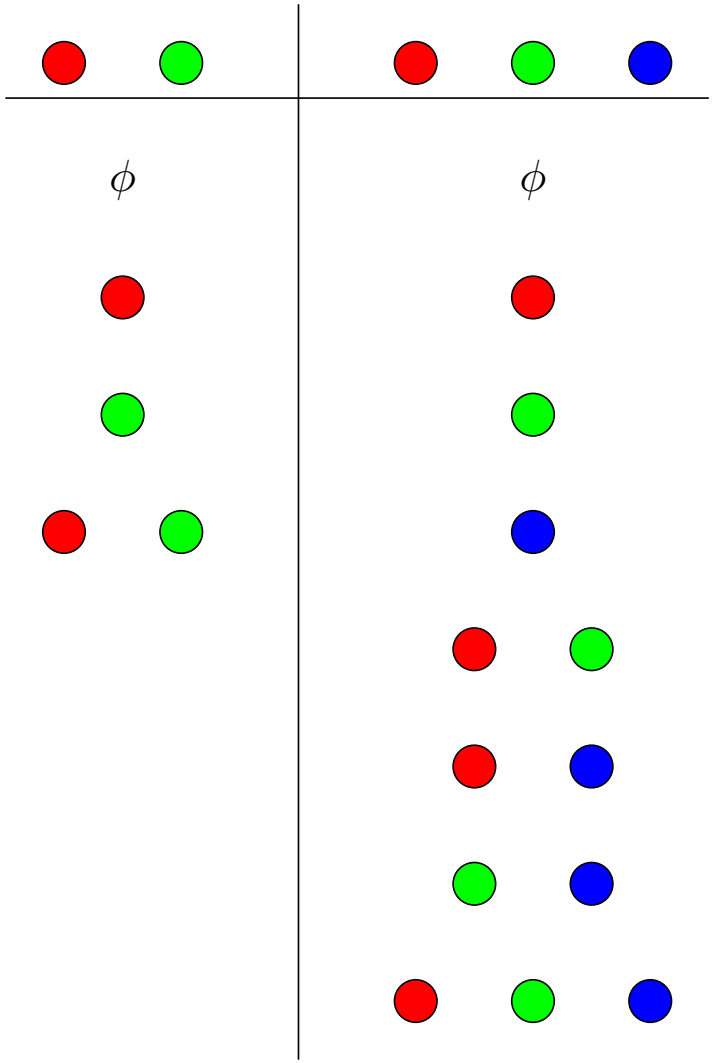
Use mathematical induction to show that if S is a finite set with n elements, where n is a nonnegative integer, then S has 2^n subsets.

Let $P(n)$ be the proposition that a set with n elements has 2^n subsets.

- Basis Step: $P(0)$ is true, because the empty set has only itself as a subset and $2^0 = 1$.
- Inductive Step: Assume $P(k)$ is true for an arbitrary nonnegative integer k. I.e., for an arbitrary nonnegative integer k, every set with k elements has 2^k subsets.
- Let T be a set with $k+1$ elements. Then $T = S \cup \{a\}$, where $a \in T$ and $S = T - \{a\}$. Hence $|S| = k$.
- For each subset X of S, there are exactly two subsets of T, i.e., X and $X \cup \{a\}$.
- By the inductive hypothesis S has 2^k subsets. Since there are two subsets of T for each subset of S, the number of subsets of T is $2 \cdot 2^k = 2^{k+1}$.

Number of Subsets of a Finite Set

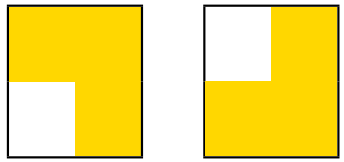




Tiling Checkerboards

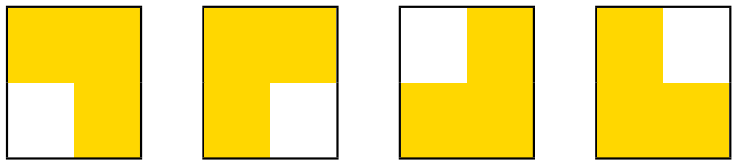
Show that every $2^n \times 2^n$ checkerboard with one square removed can be tiled using right triominoes.

A right triomino is an L-shaped tile which covers three squares at a time.



Let $P(n)$ be the proposition that every $2^n \times 2^n$ checkerboard with one square removed can be tiled using right triominoes. Use mathematical induction to prove that $P(n)$ is true for all positive integers n.

- BASIS STEP: $P(1)$ is true, because each of the four $2^1 \times 2^1$ checkerboards with one square removed can be tiled using one right triomino.

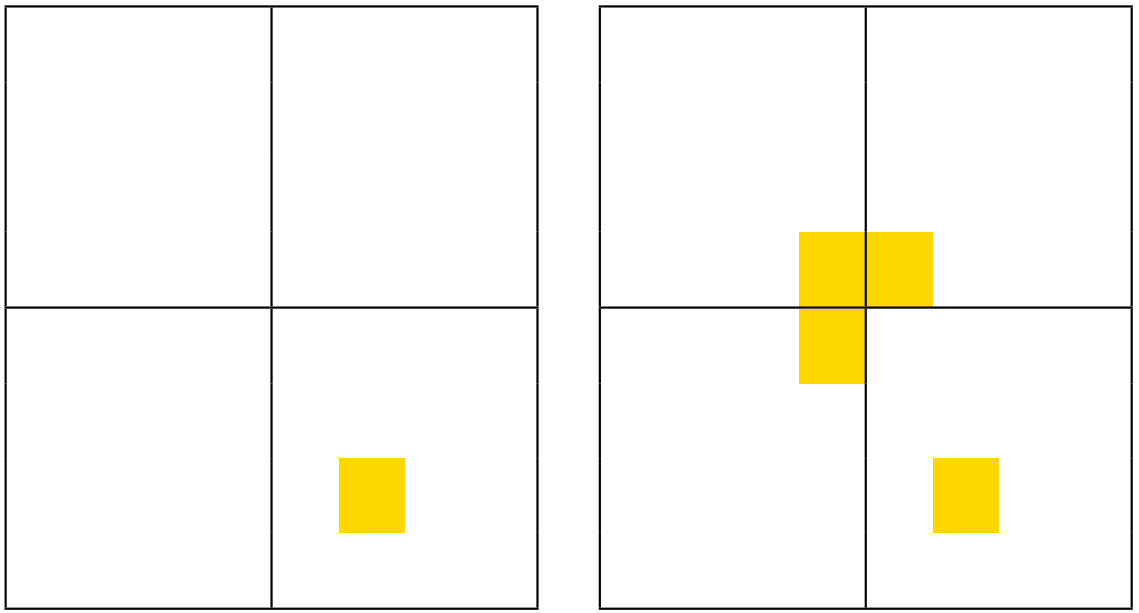


Tiling Checkerboards...

- INDUCTIVE STEP: Assume that $P(k)$ is true for every $2^k \times 2^k$ checkerboard, for some positive integer k.

> Inductive Hypothesis: Every $2^k \times 2^k$ checkerboard, for some positive integer k, with one square removed can be tiled using right triominoes.

Consider a $2^{k+1} \times 2^{k+1}$ checkerboard with one square removed. Split this checkerboard into four checkerboards of size $2^k \times 2^k$, by dividing it in half in both directions.



Tiling Checkerboards...

- Remove a square from one of the four $2^k \times 2^k$ checkerboards.
- By the inductive hypothesis, this board can be tiled.
- Also by the inductive hypothesis, the other three boards can be tiled with the square from the corner of the center of the original board removed.
- We can then cover the three adjacent squares with a triomino.

Hence, the entire $2^{k+1} \times 2^{k+1}$ checkerboard with one square removed can be tiled using right triominoes.

An Incorrect "Proof" by Mathematical Induction

Let $P(n)$ be the statement that every set of n lines in the plane, no two of which are parallel, meet in a common point. Here is a *proof* that $P(n)$ is true for all positive integers $n \geq 2$.

- BASIS STEP: The statement $P(2)$ is true because any two lines in the plane that are not parallel meet in a common point.
- INDUCTIVE STEP: The inductive hypothesis is the statement that
  - $P(k)$ is true for the positive integer $k \geq 2$, i.e., every set of k lines in the plane, no two of which are parallel, meet in a common point.

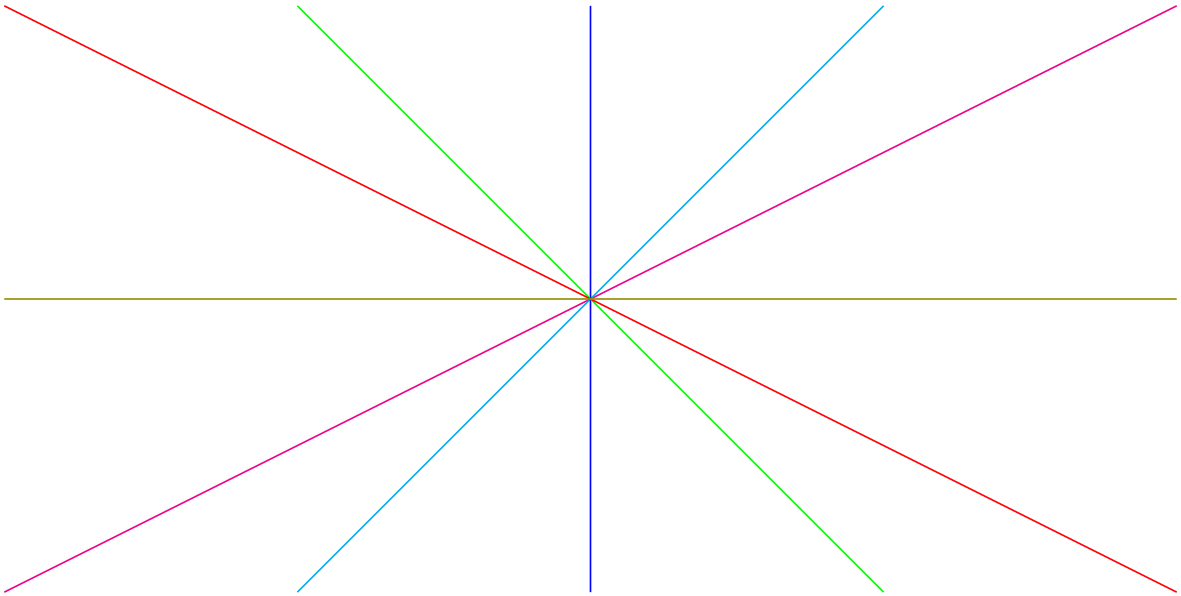
Inductive Hypothesis: Every set of k lines in the plane, where $k \geq 2$, no two of which are parallel, meet in a common point.

An Incorrect "Proof" by Mathematical Induction...

- We must show that if $P(k)$ holds, then $P(k+1)$ holds, i.e.,
  - If every set of $k \geq 2$ lines in the plane, no two of which are parallel, meet in a common point, then
  - Every set of $k+1$ lines in the plane, no two of which are parallel, meet in a common point.
- Consider a set of $k+1$ distinct lines in the plane, no two parallel.
  - By the inductive hypothesis, the first k of these lines must meet in a common point p_1.
  - By the inductive hypothesis, the last k of these lines meet in a common point p_2.
- If p_1 and p_2 are different points,
  - All lines containing both of them must be the same line since two points (p_1 and p_2) determine a line.
  - This contradicts the assumption that the lines are distinct.

An Incorrect "Proof" by Mathematical Induction...

- Hence, $p_1 = p_2$ lies on all $k+1$ distinct lines, and therefore $P(k+1)$ holds.
- Assuming that $k \geq 2$, distinct lines meet in a common point, then every $k+1$ lines meet in a common point.



An Incorrect "Proof" by Mathematical Induction...

- There must be an error in this proof since the conclusion is absurd. But where is the error?
  - $P(k) \rightarrow P(k+1)$ only holds for $k \geq 3$.
  - It is not the case that $P(2)$ implies $P(3)$.
  - The first two lines must meet in a common point p_1 and the second two must meet in a common point p_2.
  - They do not have to be the same point since only the second line is common to both sets of lines.

Guidelines: Mathematical Induction Proofs

Template for Proofs by Mathematical Induction

- Express stmt to be proved as "$\forall n \geq b, P(n)$" for fixed integer b.
- Write the words "Basis Step."
  - Show that $P(b)$ is true.
- Write the words "Inductive Step".
- State, and clearly identify the inductive hypothesis in the form "assume that $P(k)$ is true for an arbitrary fixed integer $k \geq b$."
- State what needs to be proved under the assumption that the inductive hypothesis is true. That is, write out what $P(k+1)$ says.
- Prove the statement $P(k+1)$ making use of the assumption $P(k)$.
  - Be sure that your proof is valid for all integers k with $k \geq b$, taking care that the proof works for small values of k, including $k = b$.
- Clearly identify the conclusion of the inductive step, such as by saying "this completes the inductive step."
- After completing the basis step and the inductive step, state the conclusion, namely, by mathematical induction, $P(n)$ is true for all integers $n \geq b$.

Section Summary

Section 5.2

- Strong Induction
- Example Proofs using Strong Induction
- Well-Ordering Property

Strong Induction

Strong Induction: To prove that $P(n)$ is true for all positive integers n, where $P(n)$ is a propositional function, complete two steps:

- *Basis Step*: Verify that the proposition P(1) is true.
- *Inductive Step*: Show the conditional statement

$$[P(1) \wedge P(2) \wedge \cdots \wedge P(k)] \rightarrow P(k+1)$$

  holds for all positive integers k.

Strong Induction is sometimes called the *second principle of mathematical induction or complete induction*.

Strong Induction and the Infinite Ladder

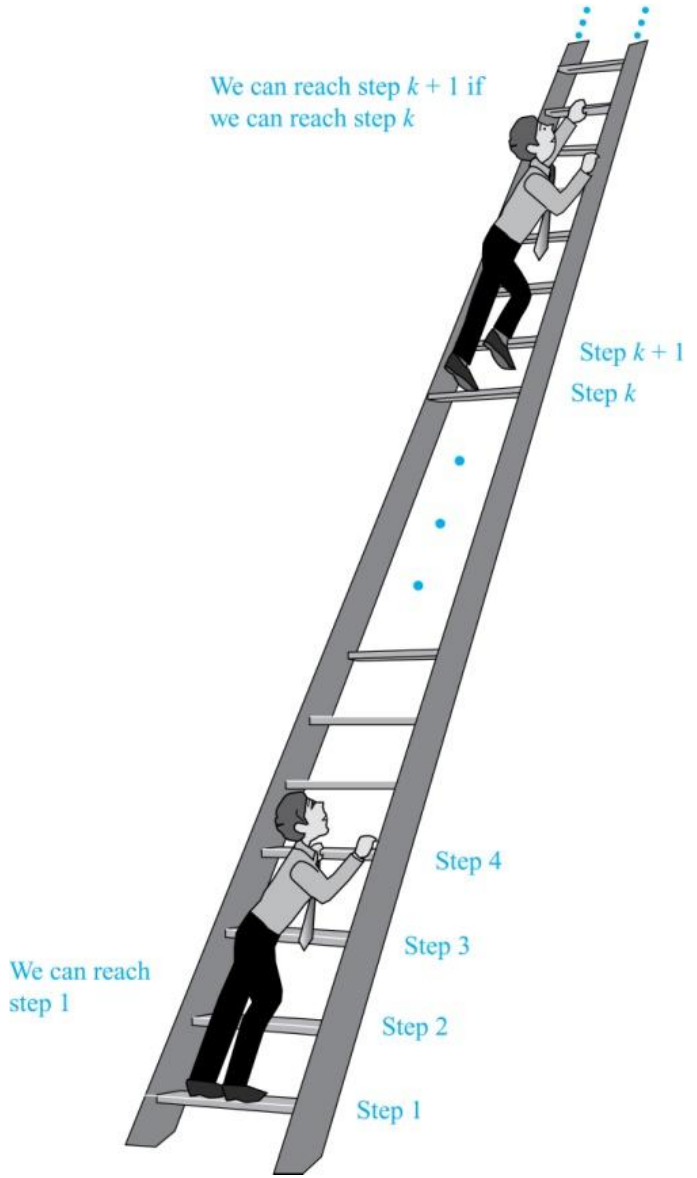
Strong induction tells us that we can reach all rungs if:

- We can reach the first rung of the ladder.
- For every integer k, if we can reach the first k rungs, then we can reach the $(k+1)$st rung.

To conclude that we can reach every rung by strong induction:

- BASIS STEP: $P(1)$ holds.
- INDUCTIVE STEP: Assume $P(1) \wedge P(2) \wedge \cdots \wedge P(k)$ holds for an arbitrary integer k, and show that $P(k+1)$ must also hold.

We will have then shown by strong induction that for every positive integer n, $P(n)$ holds, i.e., we can reach the nth rung of the ladder.



Proof using Strong Induction

Suppose we can reach the first and second rungs of an infinite ladder, and we know that if we can reach a rung, then we can reach two rungs higher. Prove that we can reach every rung. (Try this with mathematical induction.)

Prove the result using strong induction.

- BASIS STEP: We can reach the first two steps.
- INDUCTIVE STEP:
  - The inductive hypothesis is that we can reach the first k rungs, for any $k \geq 2$.
  - We can reach the $(k+1)^{st}$ rung since we can reach the $(k-1)^{st}$ rung by the inductive hypothesis.
- Hence, we can reach all rungs of the ladder.

Which Form of Induction Should Be Used?

- We can always use strong induction instead of mathematical induction.
  - But there is no reason to use it if it is simpler to use mathematical induction. (See page 335 of text.)
- In fact, the principles of mathematical induction, strong induction, and the well-ordering property are all equivalent. (Exercises 41-43, page 365)
- Sometimes it is clear how to proceed using one of the three methods, but not the other two.

Informally—Well Ordering $\rightarrow$ Induction

If the *well-ordering* principle is true, then does it mean that

$$[P(0) \wedge \forall n\, P(n) \rightarrow P(n+1)] \rightarrow \forall n\, P(n)$$

- Assume otherwise. Form the subset $n_1, n_2, \ldots \in N$, where $\neg P$.
- By the well-ordering principle, $\exists$ *smallest* $n \in \{n_1, n_2, \ldots\}$.
- So we have $P(n-1) \wedge \neg P(n)$, which is a contradiction.
- So $\forall n\, P(n)$.

Informally—Weak Induction $\rightarrow$ Strong Induction

If

$$[P(0) \wedge \forall n\, P(n) \rightarrow P(n+1)] \rightarrow \forall n\, P(n)$$

is a valid proof technique, then is

$$[P(0) \wedge (\forall k \leq n\, P(n) \rightarrow P(n+1))] \rightarrow \forall n\, P(n)$$

also a valid proof technique?

- If all we need to show is that $P(k) \rightarrow P(k+1)$, but we are given
- $P(1) \wedge P(2) \wedge \ldots \wedge P(k) \rightarrow P(k+1)$. But
- $P(1) \wedge P(2) \wedge \ldots \wedge P(k) \rightarrow P(k)$, $\therefore$
- $P(k) \rightarrow P(k+1)$.

What about Strong Induction $\rightarrow$ Weak Induction?

Completion of the proof of the Fundamental Theorem of Arithmetic

Show that if n is an integer > 1, then n can be written as the product of primes.

Let $P(n)$ be the proposition that n can be written as a product of primes.

- BASIS STEP: $P(2)$ is true since 2 itself is prime.
- INDUCTIVE STEP: The inductive hypothesis is $P(j)$ is true for all integers $j \mid 2 \leq j \leq k$. To show that $P(k+1)$ must be true under this assumption, two cases need to be considered:
  - If $k+1$ is prime, then $P(k+1)$ is true.
  - Otherwise, $k+1$ is composite and,
    - ★ it can be written as the product of two positive integers a and b with $2 \leq a \leq b < k+1$.
    - ★ By the inductive hypothesis, a and b can be written as the product of primes, and $\therefore$
    - ★ $k+1$ can also be written as the product of those primes.

Hence, every integer > 1 can be written as the product of primes.
Uniqueness proved in Section 4.3.

Proof using Strong Induction

Prove that every amount of postage of 12¢ or more can be formed using just 4¢ and 5¢ stamps.

Let $P(n)$ be the proposition that postage of n¢ can be formed using 4¢ and 5¢ stamps.

- BASIS STEP: $P(12)$, $P(13)$, $P(14)$, and $P(15)$ hold.
  - $P(12) = 3 \times 4$¢ stamps. $P(13) = 2 \times 4$¢ stamps $+ 1 \times 5$¢ stamp.
  - $P(14) = 1 \times 4$¢ stamp $+ 2 \times 5$¢ stamps. $P(15)$ uses 3×5¢ stamps.
- INDUCTIVE STEP:
  - $P(j)$ holds for $12 \leq j \leq k$, where $k \geq 15$.
  - Show that $P(k+1)$ holds.
- Using the inductive hypothesis,
  - $P(k-3)$ holds since $k - 3 \geq 12$.
  - To form postage of $k + 1$¢, add a 4¢ stamp to the postage for $k - 3$¢.

Hence, $P(n)$ holds for all $n \geq 12$.

Proof of Same Example using Mathematical Induction

Prove that every amount of postage of 12¢ or more can be formed using just 4¢ and 5¢ stamps.

Let $P(n)$ be the proposition that postage of n¢ can be formed using 4¢ and 5¢ stamps.

- BASIS STEP: Postage of 12¢ can be formed using three 4¢ stamps.
- INDUCTIVE STEP: $P(k)$ is that postage of k¢ can be formed using 4¢ and 5¢ stamps. To show $P(k+1)$ where $k \geq 12$, we consider two cases:
  - If at least one 4¢ stamp has been used, then a 4¢ stamp can be replaced with a 5¢ stamp to yield a total of $k+1$¢.
  - Otherwise, no 4-cent stamp have been used and at least three 5¢ stamps were used. Three 5¢ stamps can be replaced by four 4¢ stamps to yield a total of $(k+1)$¢.
- Hence, $P(n)$ holds for all $n \geq 12$.

Well-Ordering Property

- *Well-ordering property*: Every nonempty set of nonnegative integers has a least element.
- The well-ordering property is one of the axioms of the positive integers listed in Appendix 1.
- The well-ordering property can be used directly in proofs, as the next example illustrates.
- The well-ordering property can be generalized.
  - Definition: A set is *well ordered if every subset has a least element*.
    - **N** is well ordered under $\leq$.
    - The set of finite strings over an alphabet using lexicographic ordering is well ordered.
  - We will see a generalization of induction to sets other than the integers in the next section.

Well-Ordering Property

Skip for now—searching for a better example

Use the well-ordering property to prove the division algorithm, which states that if a is an integer and d is a positive integer, then there are unique integers q and r with $0 \leq r < d$, such that $a = dq + r$.

Let S be the set of nonnegative integers of the form $a - dq$, where q is an integer. The set is nonempty since $-dq$ can be made as large as needed.

- By the well-ordering property, S has a least element $r = a - dq_0$. The integer r is nonnegative. It also must be the case that $r < d$.
- If it were not, then there would be a smaller nonnegative element in S, namely,

$$a - d(q_0 + 1) = a - dq_0 - d = r - d > 0$$

- Therefore, there are integers q and r with $0 \leq r < d$.

Uniqueness of q and r is Exercise 37.

Check-in

The sum of the interior angles of any triangle is 180°. Now, using this fact and induction, prove that any polygon with $k \geq 3$ vertices has interior angles that sum to $180k - 360$ degrees.

Section Summary

- Recursively Defined Functions
- Recursively Defined Sets and Structures
- Structural Induction
- Generalized Induction

Recursively Defined Functions

Definition: A *recursive* or *inductive definition* of a function consists of two steps.

- BASIS STEP: Specify the value of the function at zero.
- RECURSIVE STEP: Give a rule for finding its value at an integer from its values at smaller integers.

A function $f(n)$ is the same as a sequence $a_0, a_1, \ldots,$ where $f(i) = a_i$. This was done using recurrence relations in Section 2.4.

Recursively Defined Functions

Suppose f is defined by

$$f(0) = 3$$
$$f(n+1) = 2f(n) + 3$$

Find $f(1), f(2), f(3), f(4)$.

$$\begin{aligned} f(1) = 2f(0) + 3 = 2 \cdot 3 + 3 &= 9 \\ f(2) = 2f(1) + 3 = 2 \cdot 9 + 3 &= 21 \\ f(3) = 2f(2) + 3 = 2 \cdot 21 + 3 &= 45 \\ f(4) = 2f(3) + 3 = 2 \cdot 45 + 3 &= 93 \end{aligned}$$

Give a recursive definition of the factorial function $n!$

$$f(0) = 1$$
$$f(n+1) = (n+1) \cdot f(n)$$

Recursively Defined Functions

Give a recursive definition of

$$\sum_{k=0}^{n} a_k$$

- The first part of the definition is $\sum_{k=0}^{0} a_k = a_0$.
- The second part is

$$\sum_{k=0}^{n+1} a_k = a_{n+1} + \left(\sum_{k=0}^{n} a_k \right)$$

Python recursive definition of a series summation

```
def rec(n: int):
    if n == 0:
        return a0

    return an + rec(n-1)
```

Fibonacci Numbers

The Fibonacci numbers are defined as follows:

$$f_0 = 0, f_1 = 1, f_n = f_{n-1} + f_{n-2}$$

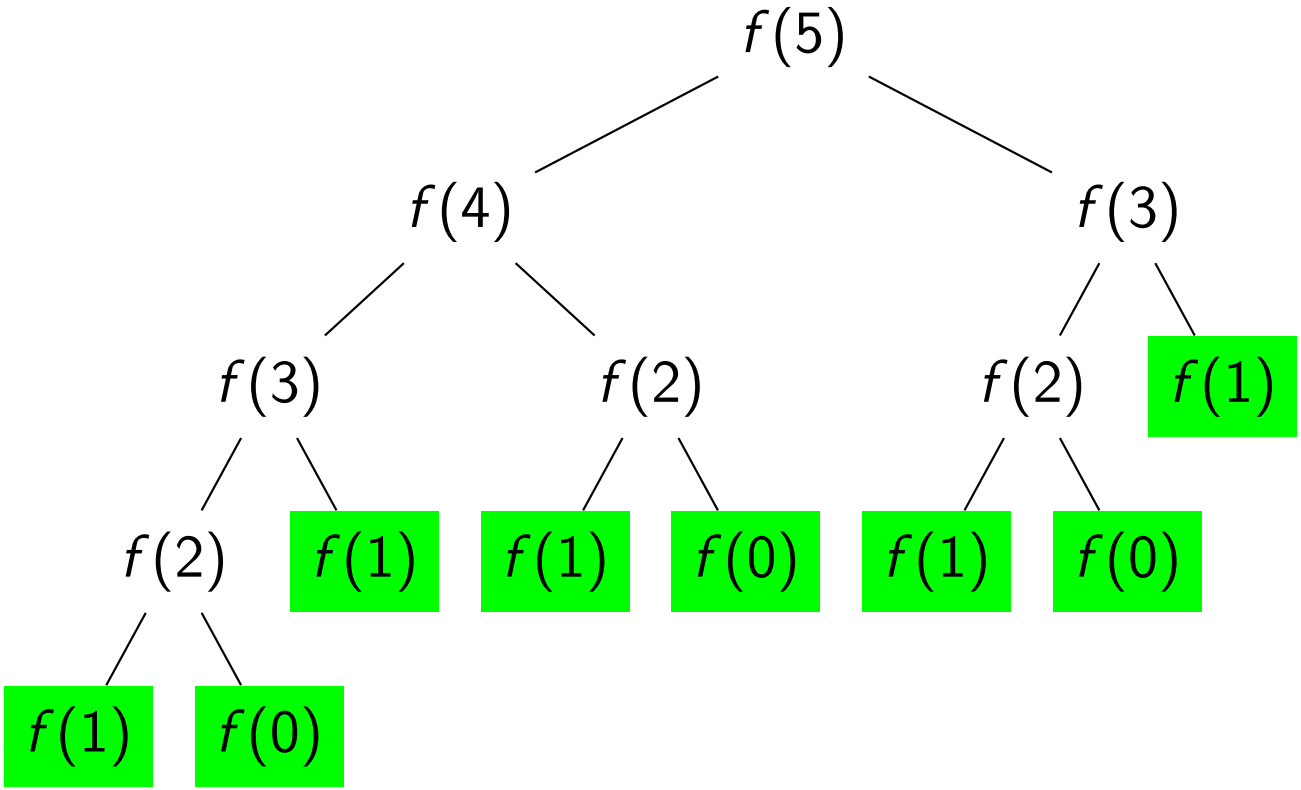
Find f_2, f_3, f_4, f_5.

$$f_2 = f_1 + f_0 = 1 + 0 = 1$$
$$f_3 = f_2 + f_1 = 1 + 1 = 2$$
$$f_4 = f_3 + f_2 = 2 + 1 = 3$$
$$f_5 = f_4 + f_3 = 3 + 2 = 5$$

In Chapter 8, we will use the Fibonacci numbers to model population growth of rabbits. This was an application described by Fibonacci himself.

Next, we use strong induction to prove a result about the Fibonacci numbers.

Recursively Defined Functions. . .Fibonacci call graph



Fibonacci Numbers

FYIO

Show that whenever $n \geq 3$, $f_n > \alpha^{n-2}$, where $\alpha = (1 + \sqrt{5})/2$.

Let $P(n)$ be the statement $f_n > \alpha^{n-2}$. Use strong induction to show that $P(n)$ is true whenever $n \geq 3$.

BASIS STEP:

- $P(3)$ holds since $f_3 = 2 > \alpha^1$ because $\alpha < 2$.
- $P(4)$ holds since $f_4 = 3 > \alpha^2$.
  - $\alpha^2 = (1 + 5 + 2\sqrt{5})/4 = (3 + \sqrt{5})/2 < 3$.
  - $\sqrt{5} < 3$.

Fibonacci Numbers

FYIO

INDUCTIVE STEP: Assume that $P(j)$ holds, i.e., $f_j > \alpha^{j-2}$ for all integers j with $3 \leq j \leq k$, where $k \geq 4$. Show that $P(k+1)$ holds, i.e., $f_{k+1} > \alpha^{k-1}$.

- Since $\alpha^2 = \alpha + 1$ (because α is a solution of $x^2 - x - 1 = 0$),

$$\alpha^{k-1} = \alpha^2 \cdot \alpha^{k-3} = (\alpha+1) \cdot \alpha^{k-3} = \alpha \cdot \alpha^{k-3} + 1 \cdot \alpha^{k-3} = \alpha^{k-2} + \alpha^{k-3}$$

- By the inductive hypothesis, because $k \geq 4$ we have

$$f_{k-1} > \alpha^{k-3}, \quad f_k > \alpha^{k-2}$$

- Therefore, it follows that

$$f_{k+1} = f_k + f_{k-1} > \alpha^{k-2} + \alpha^{k-3} = \alpha^{k-1}$$

- Hence, $P(k+1)$ is true.

Lam?'s Theorem

Let a and b be positive integers with $a \geq b$. Then the number of divisions used by the Euclidian algorithm to find $gcd(a, b)$ is $\leq$ five times the number of decimal digits in b.

Proof: When we use the Euclidian algorithm to find $gcd(a, b)$ with $a \geq b$,

n divisions are used with $a = r_0, b = r_1$.

$$
\begin{aligned}
a = r_0 &= r_1 q_1 + r_2 && 0 \leq r_2 < r_1, \\
b = r_1 &= r_2 q_2 + r_3 && 0 \leq r_3 < r_2, \\
&\vdots \\
r_{n-2} &= r_{n-1} q_{n-1} + r_n && 0 \leq r_n < r_{n-1}, \\
r_{n-1} &= r_n q_n
\end{aligned}
$$

$r_n < r_{n-1}$ and $r_n \mid r_{n-1}$. So $q_n \geq 2$.

Since each quotient $q_1, q_2, \ldots, q_{n-1}$ is at least 1 and $q_n \geq 2$:

$$
\begin{aligned}
r_n &\geq 1 = f_2, \\
r_{n-1} &\geq 2r_n \geq 2f_2 = f_3, \\
r_{n-2} &\geq r_{n-1} + r_n \geq f_3 + f_2 = f_4, \\
&\vdots \\
r_2 &\geq r_3 + r_4 \geq f_{n-1} + f_{n-2} = f_n, \\
b = r_1 &\geq r_2 + r_3 \geq f_n + f_{n-1} = f_{n+1}
\end{aligned}
$$

Lam?'s Theorem

- It follows that if n divisions are used by the Euclidian algorithm to find $gcd(a, b)$ with $a \geq b$, then $b \geq f_{n+1}$.
- By Example 4, $f_{n+1} > \alpha^{n-1}$, for $n > 2$, where $\alpha = (1 + \sqrt{5})/2$.
- Therefore, $b > \alpha^{n-1}$.
- Because $\log_{10} \alpha \sim 0.208 > 1/5$, $\log_{10} b > (n-1) \log_{10} \alpha > (n-1)/5$. Hence,

$$n - 1 < 5 \log_{10} b$$

- Suppose that b has k decimal digits. Then $b < 10^k$ and $\log_{10} b < k$. It follows that $n - 1 < 5k$ and since k is an integer, $n \leq 5k$.
- As a consequence of Lam?'s Theorem, $O(\log b)$ divisions are used by the Euclidian algorithm to find $gcd(a, b)$ whenever $a > b$.
- By Lam?'s Theorem, the number of divisions needed to find $gcd(a, b)$ with $a > b$ is $\leq 5(\log_{10} b + 1)$ since the number of decimal digits in b (which equals $\lfloor log_{10} b \rfloor + 1$) is $\leq \log_{10} b + 1$.

Recursively Defined Sets and Structures

Recursive definitions of sets have two parts:

- The basis step specifies an initial collection of elements.
- The recursive step gives the rules for forming new elements in the set from those already known to be in the set.

Sometimes the recursive definition has an exclusion rule, which specifies that the set contains nothing other than those elements specified in the basis step and generated by applications of the rules in the recursive step.

We will always assume that the exclusion rule holds, even if it is not explicitly mentioned.

We will later develop a form of induction, called structural induction, to prove results about recursively defined sets.

Recursively Defined Sets and Structures

Example: Subset of Integers S:

- Basis Step: $3 \in S$.
- Recursive Step: If $x \in S$ and $y \in S$, then $x + y \in S$.

Initially 3 is in S, then $3 + 3 = 6$, then $3 + 6 = 9$, etc.

Example: The natural numbers N.

- Basis Step: $0 \in N$.
- Recursive Step: If n is in N, then $n + 1$ is in N.

Initially 0 is in S, then $0 + 1 = 1$, then $1 + 1 = 2$, etc.

Strings

Definition: The set Σ^* of strings over the alphabet Σ:

- Basis Step: $\lambda \in \Sigma^*$ (λ is the empty string)
- Recursive Step: If w is in Σ^* and x is in Σ, then $wx \in \Sigma^*$.

Example: If $\Sigma = \{0, 1\}$, the strings in Σ^* are the set of all bit strings, $\lambda, 0, 1, 00, 01, 10, 11$, etc.

Example: If $\Sigma = \{a, b\}$, show that $aab \in \Sigma^*$.

- Since $\lambda \in \Sigma^*$ and $a \in \Sigma$, $a \in \Sigma^*$.
- Since $a \in \Sigma^*$ and $a \in \Sigma$, $aa \in \Sigma^*$.
- Since $aa \in \Sigma^*$ and $b \in \Sigma$, $aab \in \Sigma^*$.

String Concatenation

Two strings can be combined via the operation of *concatenation*. Let Σ be a set of symbols and Σ^* be the set of strings formed from the symbols in Σ. We can define the concatenation of two strings, denoted by $\cdot$, recursively as follows.

- Basis Step: If $w \in \Sigma^*$, then $w \cdot \lambda = w$.
- Recursive Step: If $w_1 \in \Sigma^*$ and $w_2 \in \Sigma^*$ and $x \in \Sigma$, then

$$w_1 \cdot (w_2 x) = (w_1 \cdot w_2)x$$

Often $w_1 \cdot w_2$ is written as $w_1 w_2$. If $w_1 = abra$ and $w_2 = cadabra$, the concatenation $w_1 w_2 = abracadabra$.

Python recursive string concatenation

```
def concat(w1: str, w2: str):
    if w2 == '':
        return w1
    else:
        return concat(w1, w2[:-1]) + w2[-1]
```

Length of a String

Give a recursive definition of $l(w)$, the length of the string w.

The length of a string can be recursively defined by:

$$l(\lambda) = 0;$$
$$l(wx) = l(w) + 1 \text{ if } w \in \Sigma^* \text{ and } x \in \Sigma$$

Python recursive definition of l(x)

```
def l(x: str):
    if x == '':
        return 0
    return l(x[:-1])+1
```

Balanced Parentheses

Give a recursive definition of the set of balanced parentheses P.

- Basis Step: $() \in P$
- Recursive Step: If $w \in P$, then
  1. $()w \in P$,
  2. $(w) \in P$, and
  3. $w() \in P$.
- Show that $((())())$ is in P.
- Why is $))((()$ not in P?

```
def balanced_paren() -> str:
    c = random.randint(1, 4)
    if c == 1:
        return "()"
    elif c == 2:
        return f"(){balanced_paren()}"
    elif c == 3:
        return f"({balanced_paren()})"
    else:
        return f"{balanced_paren()}()"
```

Well-Formed Formulae in Propositional Logic

The set of *well-formed formulae* in propositional logic involving T, F, propositional variables, and operators from the set $\{\neg, \wedge, \vee, \rightarrow, \leftrightarrow\}$.

- Basis Step: T, F, and s, where s is a propositional variable, are well-formed formulae.
- Recursive Step: If E and F are well formed formulae, then $(\neg E)$, $(E \wedge F)$, $(E \vee F)$, $(E \rightarrow F)$, $(E \leftrightarrow F)$, are well-formed formulae.

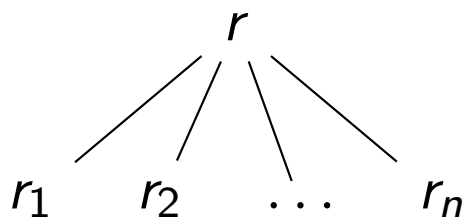
Examples:

- $((p \vee q) \rightarrow (q \wedge F))$ is a well-formed formula.
- $pq\wedge$ is not a well-formed formula.
  - Can also show this by induction. If $pq\wedge$ is well-formed, then it's generated at *some* recursive step.

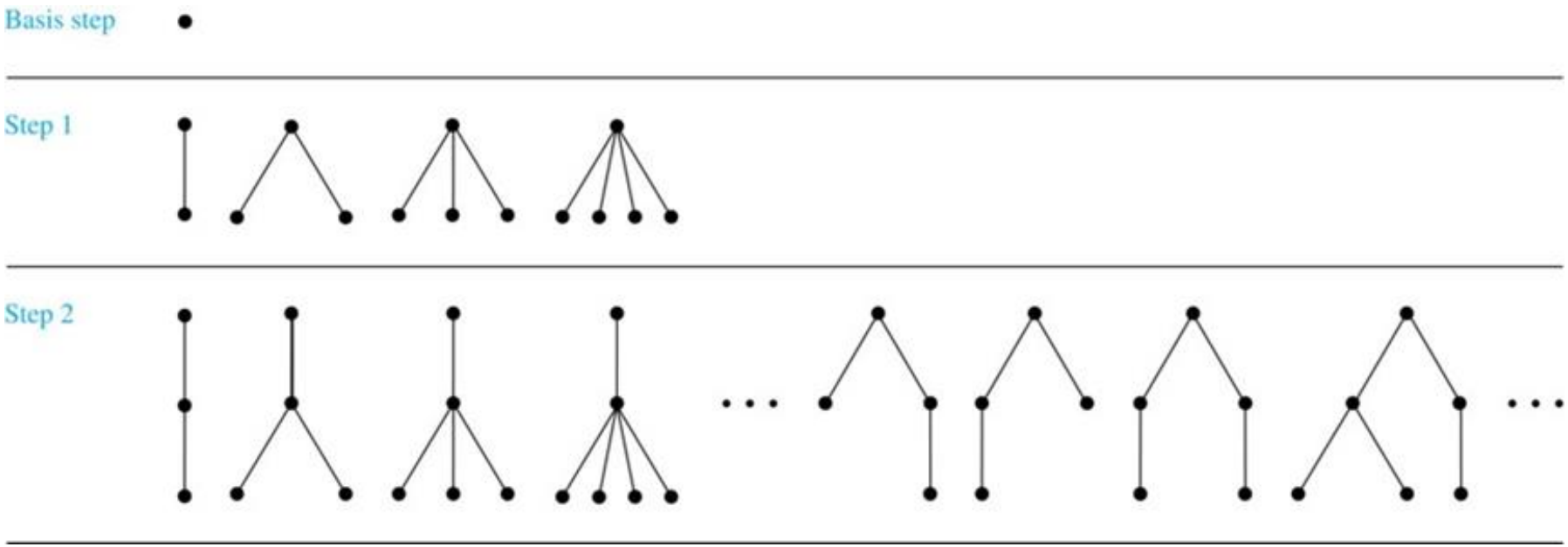
Rooted Trees

The set of *rooted trees*, where a rooted tree consists of a set of vertices containing a distinguished vertex called the *root*, and edges connecting these vertices, can be defined recursively by these steps:

- Basis Step: A single vertex r is a rooted tree.
- Recursive Step: Suppose that $T_1, T_2, \ldots, T_n$ are disjoint rooted trees with roots $r_1, r_2, \ldots, r_n$, respectively. Then the graph formed by
  - Starting with a root r, which is not in any of the rooted trees $T_1, T_2, \ldots, T_n$, and
  - Adding an edge from r to each of the vertices $r_1, r_2, \ldots, r_n$, is also a rooted tree.



Building Up Rooted Trees



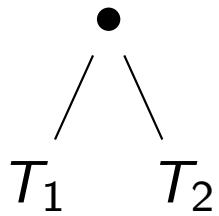
Trees are studied extensively in Chapter 11.

Next we look at a special type of tree, the full binary tree.

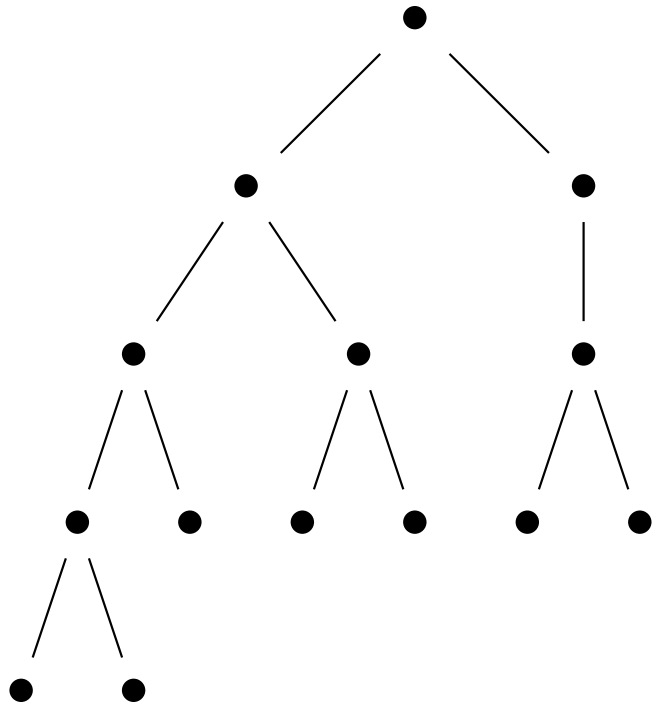
Full Binary Trees

The set of full binary trees can be defined recursively by these steps.

- *Basis Step*: There is a full binary tree consisting of only a single vertex r.
- *Recursive Step*: If T_1 and T_2 are disjoint full binary trees, there is a full binary tree, denoted by $T_1 \cdot T_2$, consisting of
  - A root r,
  - Together with edges connecting the root to each of the roots of the left subtree T_1 and the right subtree T_2.



Is this a full binary tree?



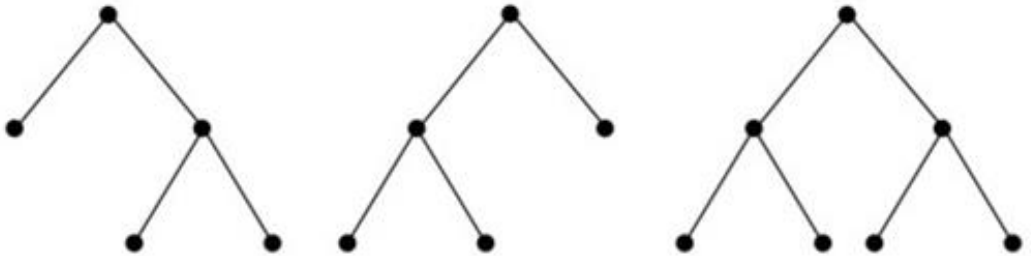
Go to https://schnekli-tamu.uc.r.appspot.com/poll to cast your vote.

Building Up Full Binary Trees

Basis step

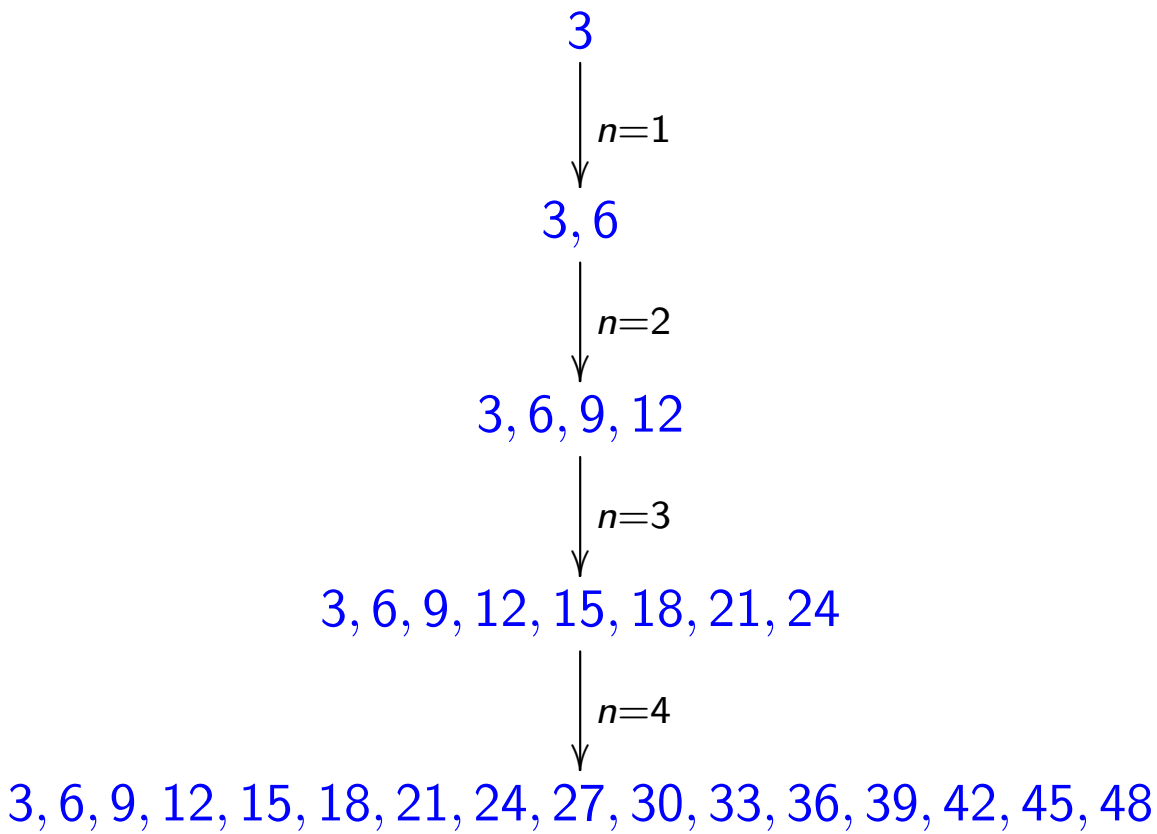
Step 1

Step 2



Induction and Recursively Defined Sets

Show that the set S defined by specifying that $3 \in S$ and that if $x \in S$ and $y \in S$, then $x + y$ is in S, is the set of all positive integers that are multiples of 3.



Induction and Recursively Defined Sets

Let A be the set of all positive integers divisible by 3. To prove that $A = S$, show that A is a subset of S and S is a subset of A.

- $A \subset S$: Let $P(n)$ be the statement that $3n$ belongs to S.
  - Basis Step: $3 \cdot 1 = 3 \in S$, by the first part of recursive definition.
  - Inductive Step: Assume $P(k)$ is true. By the second part of the recursive definition, if $3k \in S$, then
    - Since $3 \in S, 3k + 3 = 3(k + 1) \in S$.
    - Hence, $P(k + 1)$ is true.
- $S \subset A$:
  - Basis Step: $3 \in S$ by the first part of recursive definition, and $3 = 3 \cdot 1$.
  - Inductive Step: The second part of the recursive definition adds $x + y$ to S, if both x and y are in S.
    - If x and y are both in A, then both x and y are divisible by 3.
    - By part (i) of Theorem 1 of Section 4.1, it follows that $x + y$ is divisible by 3.

We used mathematical induction to prove a result about a recursively defined set. Next, we study a more direct form of induction for proving results about recursively defined sets.

Structural Induction

To prove a property of the elements of a recursively defined set, we use structural induction.

- *Basis Step*: Show that the result holds for all elements specified in the basis step of the recursive definition.
- *Recursive Step*: Show that if the statement is true for each of the elements used to construct new elements in the recursive step of the definition, the result holds for these new elements.

The validity of structural induction can be shown to follow from the principle of mathematical induction.

A proof by structural induction is identical in form to a proof by strong induction *on the number of applications of the inductive-case rules used to generate the object*.

Full Binary Trees

The *height* $h(T)$ of a full binary tree T is defined recursively as follows:

- Basis Step: The height of a full binary tree T consisting of only a root r is $h(T) = 0$.
- Recursive Step: If T_1 and T_2 are full binary trees, then the full binary tree $T = T_1 \cdot T_2$ has height

$$h(T) = 1 + \max(h(T_1), h(T_2))$$

The number of vertices $n(T)$ of a full binary tree T satisfies the following recursive formula:

- Basis Step: The number of vertices of a full binary tree T consisting of only a root r is $n(T) = 1$.
- Recursive Step: If T_1 and T_2 are full binary trees, then the full binary tree $T = T_1 \cdot T_2$ has the number of vertices

$$n(T) = 1 + n(T_1) + n(T_2)$$

Structural Induction and Binary Trees

If T is a full binary tree, then $n(T) \leq 2^{h(T)+1} - 1$.

Proof: Use structural induction.

- BASIS STEP: The result holds for a full binary tree consisting only of a root, $n(T) = 1$ and $h(T) = 0$. Hence, $n(T) = 1 \leq 2^{0+1} - 1 = 1$.
- RECURSIVE STEP: Assume $n(T_1) \leq 2^{h(T_1)+1} - 1$ and also

$$n(T_2) \leq 2^{h(T_2)+1} - 1$$

whenever T_1 and T_2 are full binary trees. Then,

$$\begin{aligned}
n(T_2) &\leq 2^{h(T_2)+1} - 1 \text{ whenever } T_1 \text{ and } T_2 \text{ are full binary trees.} \\
n(T) &= 1 + n(T_1) + n(T_2), \text{ by recursive formula of } n(T) \\
&\leq 1 + \left(2^{h(T_1)+1} - 1\right) + \left(2^{h(T_2)+1} - 1\right), \text{ by inductive hypothesis} \\
&\leq 2 \cdot \max\left(2^{h(T_1)+1}, 2^{h(T_2)+1}\right) - 1 \\
&= 2 \cdot 2^{\max(h(T_1), h(T_2)+1)} - 1, \quad \max\left(2^x, 2^y\right) = 2^{\max(x,y)} \\
&= 2 \cdot 2^{h(t)} - 1 \\
&= 2^{h(t)+1} - 1
\end{aligned}$$

Generalized Induction

FYIO

- *Generalized induction* is used to prove results about sets other than the integers that have the well-ordering property (explored in more detail in Chapter 9).
- For example, consider an ordering on $N \times N$ ordered pairs of non-negative integers. Specify that (x_1, y_1) is less than or equal to (x_2, y_2) if either $x_1 < x_2$, or $x_1 = x_2 \wedge y_1 < y_2$. This is called the lexicographic ordering.
- Strings are also commonly ordered by a *lexicographic ordering*.
- The next example uses generalized induction to prove a result about ordered pairs from $N \times N$.

Generalized Induction

FYIO

Suppose that $a_{m,n}$ is defined for $(m, n) \in N \times N$ by $a_{0,0} = 0$ and

$$a_{m,n} = \begin{cases} a_{m-1,n} + 1 & \text{if } n = 0 \text{ and } m > 0 \\ a_{m,n-1} + n & \text{if } n > 0 \end{cases}$$

Show that $a_{m,n} = m + n(n+1)/2$ is defined for all $(m, n) \in N \times N$.

Recursive Algorithms

Section 5.4

- Recursive Algorithms
- Proving Recursive Algorithms Correct
- Recursion and Iteration
- Merge Sort

Recursive Algorithms

Definition: An algorithm is called *recursive* if it solves a problem by reducing it to an instance of the same problem with smaller input.

For the algorithm to terminate, the instance of the problem must eventually be reduced to some initial case for which the solution is known.

Recursive Factorial Algorithm

Give a recursive algorithm for computing $n!$, where n is a nonnegative integer.

```
procedure factorial(n: nonnegative integer)
  if n = 0 then
    return 1
  else
    return n · factorial(n − 1)
{output is n!}
```

Recursive Exponentiation Algorithm

Give a recursive algorithm for computing a^n, where a is a nonzero real number and n is a nonnegative integer.

```
procedure power(a: nonzero real number, n: nonnegative integer)
  if n = 0 then
    return 1
  else
    return a × power(a, n − 1)
{output is a^n}
```

Recursive GCD Algorithm

Give a recursive algorithm for computing the greatest common divisor of two nonnegative integers a and b with $a < b$.

Use the reduction $gcd(a, b) = gcd(b \bmod a, a)$ and the condition $gcd(0, b) = b, b > 0$.

```
procedure gcd(a,b: nonnegative integers with a < b)
  if a = 0 then
    return b
  else
    return gcd(b mod a, a)

{output is gcd(a, b)}
```

Recursive Binary Search Algorithm

Assume we have $a_1, a_2, \ldots, a_n$, an increasing sequence of integers. Initially i is 1 and j is n. We are searching for x.

```
procedure binarysearch(i, j, x: integers, 1 ≤ i ≤ j ≤ n)
  m := ⌊ (i + j)/2 ⌋
  if x = a_m then
    return m
  else if(x < a_m ∧ i < m) then
    return binarysearch(i, m − 1, x)
  else if(x > a_m ∧ j > m) then
    return binarysearch(m + 1, j, x)
  else
    return 0
{output is location of x in a_1,...,a_n if present, otherwise 0}
```

Bibliography I


Ashutosh Gupta and S. Krishna.
Cs 228: Logic for computer science 2022.
https://www.cse.iitb.ac.in/ akg/courses/2022-logic/, January 2022.


Hyunyoung Lee.
Discrete structures for computing.
Class slides for TAMU CSCE 222, 2019.


Phillip Rogaway.
Ecs20 fall 2021 lecture notes, Fall 2021.