CS243: Discrete Structures

Strong Induction and Recursively Defined Structures

İsıl Dillig

Announcements

▶ Homework 4 is due today
▶ Homework 5 is out today
▶ Covers induction (last lecture, this lecture, and next lecture)
▶ Homework 5 due next Thursday Nov. 1
▶ 7 questions, all of them require proofs ⇒ start early!

Review

▶ Induction is used to prove universally quantified properties about natural numbers and other countably infinite sets
▶ Consists of a base case and inductive step
▶ Base case: prove property about the least element(s)
▶ Inductive step: assume \( P(k) \) and prove \( P(k+1) \)
▶ The assumption that \( P(k) \) is true is called inductive hypothesis

Example (review)

▶ Prove the following statement by induction:
\[
\forall n \in \mathbb{Z}^+. \sum_{i=1}^{n} i = \frac{(n)(n+1)}{2}
\]
▶ Base case: \( n = 1 \). In this case, \( \sum_{i=1}^{1} i = 1 \) and \( \frac{(1)(1+1)}{2} = 1 \); thus, the base case holds.
▶ Inductive step: By the inductive hypothesis, we assume \( P(k) \):
\[
\sum_{i=1}^{k} i = \frac{(k)(k+1)}{2}
\]
Now, we want to show \( P(k+1) \):
\[
\sum_{i=1}^{k+1} i = \frac{(k+1)(k+2)}{2}
\]
▶ First, observe:
\[
\sum_{i=1}^{k+1} i = \sum_{i=1}^{k} i + (k+1)
\]
By the inductive hypothesis, \( \sum_{i=1}^{k} i = \frac{(k)(k+1)}{2} \); thus:
\[
\sum_{i=1}^{k+1} i = \frac{(k)(k+1)}{2} + (k+1)
\]
Rewrite left hand side as:
\[
\sum_{i=1}^{k+1} i = \frac{k^2 + 3k + 2}{2} - \frac{(k+1)(k+2)}{2}
\]
Since we proved both base case and inductive step, property holds.

Example (review), cont.

Plan for Today

▶ Strong induction
▶ Recursive definitions
▶ Proving properties of recursively defined functions

Işıl Dillig, CS243: Discrete Structures Strong Induction and Recursively Defined Structures 6/34
Strong Induction

- **Strong induction** is a proof technique that is a slight variation on mathematical (regular) induction.
- Just like regular induction, you need to prove base case and inductive step, but the inductive step is slightly different.
  - **Regular induction**: assume $P(k)$ holds and prove $P(k+1)$.
  - **Strong induction**: assume $P(1), P(2), \ldots, P(k)$; prove $P(k+1)$.
- Regular induction and strong induction are equivalent, but strong induction can sometimes make proofs easier.

Motivation for Strong Induction

- Prove that if $n$ is an integer greater than 1, then it is either a prime or can be written as the product of primes.
- Let's first try to prove the property using regular induction.
  - **Base case**: $n=2$. Since 2 is a prime number, $P(2)$ holds.
  - **Inductive step**: Assume $k$ is either a prime or the product of primes.
- But this doesn’t really help us prove the property about $k+1$.
- **Claim**: Proven much easier using strong induction!

Proof Using Strong Induction

Prove that if $n$ is an integer greater than 1, then it is either a prime or can be written as the product of primes.

- **Base case**: same as before.
- **Inductive step**: Assume each of $2, 3, \ldots, k$ is either prime or product of primes.
- Now, we want to prove the same thing about $k+1$.
- Two cases: $k+1$ is either (i) prime or (ii) composite.
- If it is prime, property holds.

Proof, cont.

- If composite, $k+1$ can be written as $pq$ where $2 \geq p, q \geq k$.
  - Because a composite number has a divisor distinct from 1 and itself.
  - By the IH, $p, q$ are either primes or product of primes.
- Thus, $k+1$ can also be written as product of primes.
- **Observe**: Much easier to prove this property using strong induction!

A Word about Base Cases

- In all examples so far, we had only one base case.
  - i.e., only proved the base case for one integer.
- In some inductive proofs, there may be multiple base cases.
  - i.e., prove base case for the first $m$ numbers.
- Perfectly fine to have inductive proofs with multiple base cases.
- In such proofs, inductive step only needs to consider numbers greater than $m$.

Example

- Prove that every integer $n \geq 12$ can be written as $n = 4a + 5b$ for some non-negative integers $a, b$.
  - Proof by strong induction on $n$ and consider 4 base cases.
  - Base case 1 ($n=12$):
  - Base case 2 ($n=13$):
  - Base case 3 ($n=14$):
  - Base case 4 ($n=15$):
Example, cont.

Prove that every integer \( n \geq 12 \) can be written as \( n = 4a + 5b \) for some non-negative integers \( a, b \).

- **Inductive hypothesis:** Suppose every \( 12 \leq i \leq k \) can be written as \( i = 4a + 5b \).
- **Inductive step:** We want to show \( k + 1 \) can also be written this way for \( k + 1 \geq 16 \).
- **Observe:** \( k + 1 = (k - 3) + 4 \)
- By IH, \( k - 3 = 4a + 5b \) for some \( a, b \) because \( k - 3 \geq 12 \)
- But then, \( k + 1 \) can be written as \( 4(a + 1) + 5b \)
- Would this proof work if we only showed base case for \( n = 12 \)?

Recursive Definitions

- **Definitions of structures that refer to themselves are called recursive definitions**
- Picture below is “defined” recursively because each picture contains a smaller version of itself

Recursive Definitions in Math

- **Recursive definitions come up a lot in discrete math**
- Consider the factorial function: \( n! = 1 \cdot 2 \cdot 3 \ldots \cdot n \)
- This is a direct definition, but easier to define factorial recursively:
  \[
  1! = 1 \\
  n! = (n - 1)! \cdot n
  \]
- **Definition is recursive** because we use ! when defining !

General Structure of Recursive Definitions

- Recursive definitions consist of base case and recursive step
- Base case defines function for least element in the domain
- Recursive step shows how to compute \( f(k + 1) \) assuming \( f(k) \) can be computed
- For factorial, base case is \( 1! = 1 \)
- For factorial, recursive step is \( n! = (n - 1)! \cdot n \)
- Recursive definitions are similar to proofs by induction
- In fact, recursive definitions sometimes called inductive definitions

Recursively Defined Function Example

- Here is another recursive definition:
  \[
  f(0) = 3 \\
  f(n + 1) = 2f(n) + 3 \quad (n \geq 1)
  \]
- What is \( f(1) \)?
- What is \( f(2) \)?
- What is \( f(3) \)?
Recursively Defined Sequences

- Just like functions, sequences can also be defined recursively.
- For example, consider the following sequence:
  
  \[ 1, 3, 9, 27, 81, \ldots \]

- What is a recursive definition of this sequence?

- Base case:
- Recursive step:

Recursive Definition Examples

- Consider \( f(n) = 2n + 1 \) where \( n \) is non-negative integer
- What’s a recursive definition for \( f(n) \)?
- Consider the sequence \( 1, 4, 9, 16, \ldots \)
- What is a recursive definition for this sequence?
- Recursive definition of function defined as \( f(n) = \sum_{i=1}^{n} i \)?

Recursive Definitions of Important Functions

- Some important functions/sequences defined recursively
- Factorial function:
  
  \[
  f(1) = 1 \\
  f(n) = n \cdot f(n - 1) \quad (n \geq 2)
  \]

- Fibonacci numbers: \( 0, 1, 1, 2, 3, 5, 8, 13, 21, \ldots \)
  
  \[
  a_0 = 0 \\
  a_1 = 1 \\
  a_n = a_{n-1} + a_{n-2} \quad (n \geq 2)
  \]
- Just like there can be multiple base cases in inductive proofs, there can be multiple base cases in recursive definitions

Inductive Proofs for Recursively Defined Structures

- Recursive definitions and inductive proofs are very similar
- Therefore, it’s natural to use induction to prove properties about recursively defined functions and sequences
- In these proofs, base case of induction shows property holds for base case of recursive definition
- Similarly, the inductive step shows the property holds for the recursive part of the definition

Example 1

- Consider the function defined recursively as follows:
  
  \[
  f(0) = 1 \\
  f(n) = f(n - 1) + 3
  \]
- Prove that \( f(n) = 3n + 1 \)
- We’ll prove this by regular mathematical induction
- Base case:

Example 1, cont.

- Inductive step: We need to show \( f(k + 1) = 3(k + 1) + 1 \)
  assuming \( f(k) = 3k + 1 \)
- Using the recursive case of definition, \( f(k + 1) = f(k) + 3 \)
- From IH, \( f(k) = 3k + 1 \)
- Thus, \( f(k + 1) = 3k + 1 + 3 = 3(k + 1) + 1 \)
Example 2

- Let \( f_n \) denote the \( n \)’th element of the Fibonacci sequence
- **Proof:** For \( n \geq 3 \), \( f_n > \alpha^{n-2} \) where \( \alpha = \frac{1 + \sqrt{5}}{2} \)
- **Proof is by strong induction** on \( n \) with two base cases
  - **Base case 1** (\( n=3 \)): \( f_3 = 2 \), and \( \alpha < 2 \), thus \( f_3 > \alpha \)
  - **Base case 2** (\( n=4 \)): \( f_4 = 3 \) and \( \alpha^2 = \frac{3 + \sqrt{5}}{2} < 3 \)
- **Inductive step:** We need to show \( f_{k+1} > \alpha^{k-1} \) for \( k + 1 > 4 \)
  - Using IH, we can assume \( f_i > \alpha^{i-2} \) for \( 3 \leq i \leq k \)
  - First, rewrite \( \alpha^{k-1} \) as \( \alpha^2 \alpha^{k-3} \)
  - \( \alpha^2 = \left( \frac{1 + \sqrt{5}}{2} \right)^2 = \frac{\sqrt{5} + 3}{2} = \alpha + 1 \)
  - Thus, \( \alpha^{k-1} = (\alpha + 1)(\alpha^{k-3}) = \alpha^{k-2} \alpha^{k-3} \)

Example, cont.

- \( \alpha^{k-1} = \alpha^{k-2} \alpha^{k-3} \)
- By recursive definition, we know \( f_{k+1} = f_k + f_{k-1} \)
- Furthermore, by inductive hypothesis:
  - \( f_k > \alpha^{k-2} \) and \( f_{k-1} > \alpha^{k-3} \)
- Therefore, \( f_{k+1} > \alpha^{k-2} \alpha^{k-3} = \alpha^{k-1} \)

More Examples

- **Give a recursive definition of the set \( E \) of all even integers:**
  - **Base case:**
  - **Recursive step:**

- **Give a recursive definition of \( \mathbb{N} \), the set of all natural numbers:**
  - **Base case:**
  - **Recursive step:**

Strings and Alphabets

- **Recursive definitions play important role in study of strings**
- **Strings are defined over an alphabet \( \Sigma \)**
  - Example: \( \Sigma_1 = \{ a, b \} \)
  - Example: \( \Sigma_2 = \{ \} \)
- **Examples of strings over \( \Sigma_1 \):** \( a, b, ab, ba, bb, \ldots \)
- **Set of all strings formed from \( \Sigma \) forms language called \( \Sigma^* \)**
  - \( \Sigma^*_2 = \{ e, 0, 0.00, 000.\ldots \} \)
Recursive Definition of Strings

- The language $\Sigma^*$ has natural recursive definition:
  - Base case: $\epsilon \in \Sigma^*$ (empty string)
  - Recursive step: If $w \in \Sigma^*$ and $x \in \Sigma$, then $wx \in \Sigma^*$
- Since $\epsilon$ is the empty string, $\epsilon s = s$
- Consider the alphabet $\Sigma = \{0, 1\}$
- How is the string “1” formed according to this definition?
- How is “10” formed?

Recursive Definitions of String Operations

- Many operations on strings can be defined recursively.
- Consider function $l(w)$ which yields length of string $w$
- Example: Give recursive definition of $l(w)$
  - Base case:
  - Recursive step:

Another Example

- The reverse of a string $s$ is $s$ written backwards.
- Example: Reverse of “abc” is “bca”
- Give a recursive definition of the $\text{reverse}(s)$ operation
  - Base case:
  - Recursive step:

Palindromes

- A palindrome is a string that reads the same forwards and backwards
- Give a recursive definition of the set $P$ of all palindromes over the alphabet $\Sigma = \{a, b\}$
  - Base cases:
  - Recursive step: