Discrete Mathematics: Lecture 15

- Last time:
  - Chap 5.1: Mathematical induction
  - Chap 5.2: Strong induction and well-ordering
- Today:
  - Chap 5.3: Recursive definitions and structural induction
- Assignment 5: Due in two weeks
- Next time:
  - Chap 5.4: Recursive algorithms
  - Chap 5.5 Program correctness

Review of last time

- Simple induction, strong induction, well-ordering property
- Equivalence of the three

The well-ordering property (良序性质)

Every nonempty set of nonnegative integers has a least element.

Example proofs of using the property

- If a is an integer and d is a positive integer, then there are unique integers q and r with $0 \leq r < d$ and $a = dq + r$.
- In a round-robin tournament (循环锦标赛) every player plays every other player exactly once and each match has a winner and a loser. We say that the players $p_1, p_2, \ldots, p_m$ form a cycle if p_1 beats p_2, p_2 beats p_3, ..., and p_m beats p_1. Show that if there is a cycle of length m $(m \geq 3)$, there must be a cycle of 3.

Equivalence to simple induction

- the well-ordering property $\Rightarrow$ the principle of simple induction
  - Let S be the set of $n \in \mathbf{N}$ such that $P(n)$ does not hold
- the principle of simple induction $\Rightarrow$ the well-ordering property
  - Assume to the contrary
  - Let $P(n)$: for all $i \leq n$, $i \notin S$
  - We prove that $P(n)$ holds for all $n \in \mathbf{N}$

Recursively defined functions

- To define a function with the set of nonnegative integers as its domain, we use two steps:
  - Basis step: specify $f(0)$
  - Recursive step: give a rule for defining $f(n)$ from $f(k)$ where $k < n$
- Example: Give a recursive definition of $F(n) = n!$
- Recursively defined functions are well-defined, that is, if f is recursively defined, then for all $n \in \mathbf{N}$, $f(n)$ is uniquely defined.
  - proof by complete induction

Fibonacci numbers

- Definition: $f_0 = 0$, $f_1 = 1$, $f_n = f_{n-1} + f_{n-2}$, where $n \geq 2$
- Show that whenever $n \geq 3$, $f_n > \alpha^{n-2}$, where $\alpha = (1 + \sqrt{5})/2$

LAME's theorem

Let a and b be positive integers with $a \geq b$. Then the number of divisions used by the Euclidean algorithm to find $gcd(a, b)$ is ≤ 5 times the number of decimal digits in b.

Recursively defined sets and structures

To define a set

- Basis step: specify an initial collection of elements
- Recursive step: give rules for forming new elements from existing elements
- Exclusion rule: the set contains nothing other than those specified in the basis step or generated by applications of the recursive step

Exclusion rule is often omitted.

Recursive definition of strings

Recall that a string over an alphabet Σ is a finite sequence of symbols from Σ

The set Σ^* of strings over an alphabet Σ can be defined recursively by

- Basis step: $\lambda \in \Sigma^*$, where λ represents the empty string
- Recursive step: If $w \in \Sigma^*$ and $x \in \Sigma$, then $wx \in \Sigma^*$

Recursively defined functions on recursively defined sets

- Two strings can be combined via the operation of concatenation. We define a function $\cdot : \Sigma^* \times \Sigma^* \to \Sigma^*$. Let $w \in \Sigma^*$
  - Basis step: $w \cdot \lambda = w$
  - Recursive step: If $w' \in \Sigma^*$ and $x \in \Sigma$, then $w \cdot (w'x) = (w \cdot w')x$
- The length of a string can be recursively defined by
  - Basis step: $l(\lambda) = 0$
  - Recursive step: If $w \in \Sigma^*$ and $x \in \Sigma$, then $l(wx) = l(w) + 1$

Well-formed formulae (wff) for compound propositions

- Recursive definition
  - Basis step: T, F, and p are wffs, where p is a propositional variable
  - Recursive step: If E and F are wffs, so are $(\neg E)$, $(E \vee F)$, $(E \wedge F)$, $(E \rightarrow F)$, $(E \leftrightarrow F)$
- Examples: $((p \vee q) \rightarrow (q \wedge F))$
- Counter-examples: $\neg \wedge pq$

Rooted trees

The set of rooted trees can be recursively defined:

- 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 T_1, $T_2, \ldots, T_n$, and adding an edge to each of r_1, $r_2, \ldots, r_n$, is also a rooted tree.

Binary trees

- Extended binary trees
  - Basis step: The empty set is an extended binary tree.
  - Recursive step: If T_1 and T_2 are extended binary trees, there is an extended 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 when these trees are not empty.
- Full binary trees
  - Basis step: A single vertex is a full binary tree.
  - Recursive step: If T_1 and T_2 are 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.

Recursive definitions for full binary trees

- The height $h(T)$ of a full binary tree T
  - Basis step: If T consists of only a root, $h(T) = 0$
  - 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 $h(T) = \max(h(T_1), h(T_2)) + 1$.
- The number of vertices $n(T)$ in a full binary tree T
  - Basis step: If T consists of only a root, $n(T) = 1$
  - Recursive step: If T_1 and T_2 are full binary trees, then the full binary tree $T = T_1 \cdot T_2$ has $n(T) = n(T_1) + n(T_2) + 1$.

Structural induction

To prove results about recursively defined sets

- Basis step: show that the result hold for all initial elements
- Recursive step: show that if the result holds for all elements used to construct new elements, then the result holds for all new elements

simple induction $\Rightarrow$ structural induction

- Let $Q(n)$: The result holds for all elements generated by at most n applications of the recursive step

Example proofs

- Show that every wff for compound propositions has an equal number of left and right parentheses.
- Show that $l(xy) = l(x) + l(y)$, where $x, y \in \Sigma^*$.
- If T is a full binary tree, then $n(T) \leq 2^{h(T)+1} - 1$