Topic 6: Induction & Recursion

Mathematical Induction

- Can be used only to prove results obtained in some other way, *not* a tool for discovering formulae or theorems
- it can be used to prove *formulas, inequalities, divisibility, properties of subsets and their cardinality*.

> **Principle of Mathematical Induction**
> To prove that $P(n)$ is true for all positive integers n, where $P(n)$ is a propositional function, we compelte two steps:
> *Basis Step:* We verify that $P(1)$ is true.
> *Inductive Step:* We show that the conditional statement $P(k) \to P(k+1)$ is true for all positvie integers k.

Strong Induction and Well-Ordering

- Mathematicl Induction $\equiv$ Strong Induction $\equiv$ Well-Ordering

Strong Induction

Basis step: $P(1)$ is true.

Inductive step: $\forall k(P(1), P(2), \ldots, P(k) \to P(k+1)) \Rightarrow \therefore \forall n P(n)$.

Direct Proof: $p \to q$

Proof by contrapositive: $\neg q \to \neg p$

Proof by contradiction: Assume $\neg p$ is true, prove $\neg p$ is false, thus p is true.

Well-Ordering Property: every nonempty set of nonnegative integers has a least element.

Recursive Definitions

- An equation defines a sequence based on a rule that produces the next term as a function of the previous.

Basis Step: Initial value of the function.

Recursive Step: Give a rule for finding its value at an integer from its values at smaller integers.

Linear recurrences

- A relation that each term of the sequence is a linear function of previous terms.

Linear Homogenous Recurrences: $a_n = c_1 a_{n-1} + c_2 a_{n-2} + \cdots + c_k a_{n-k}$. $c_1, \ldots, c_k \in \mathbb{R}$. k is the degree of the relation.

Linear Non-homogenous Recurrences: $a_n = c_1 a_{n-1} + c_2 a_{n-2} + \cdots + c_k a_{n-k} + f(n)$. $c_1, \ldots, c_k \in \mathbb{R}$. k is the degree of the relation.

Arithmetic sequences: the *difference* between consecutive terms is a constant.

Geometric sequences: the *ratio* between consecutive terms is a constant.

Divide and conquer recurrence: divide problem into smaller subproblems and solve them recursively.