Digital Logic Design: a rigorous approach ©

Chapter 2: Induction and Recursion

Guy Even Moti Medina

School of Electrical Engineering Tel-Aviv Univ.

March 16, 2020

Book Homepage:
http://www.eng.tau.ac.il/~guy/Even-Medina

Induction - an example

Definition

$$S_n \triangleq \sum_{i=1}^{n} i \, .$$

Note: $S_0 = 0, S_1 = 1, S_2 = 1 + 2 = 3, \ldots$

Theorem

For every $n \in \mathbb{N}$:

$$S_n = \frac{n \cdot (n+1)}{2} \, . \tag{1}$$

Proof by induction

Abstract formulation: denote by P the set of all natural numbers n that satisfy a property we are interested in. Our goal is to prove that every n satisfies this property, namely, that $P = \mathbb{N}$.

The proof consists of three steps:

1. Induction basis: prove that $0 \in P$.
2. Induction hypothesis: assume that $n \in P$.
3. Induction step: prove that if $n \in P$, then $n + 1 \in P$.

Theorem

Let $P \subseteq \mathbb{N}$. If (i) $0 \in P$ and (ii) $n \in P$ implies that $(n + 1) \in P$, for every $n \in \mathbb{N}$, then $P = \mathbb{N}$.

Complete Induction

Theorem

Let $P \subseteq \mathbb{N}$. *If (i)* $0 \in P$ *and (ii)* $n \in P$ *implies that* $(n+1) \in P$, *for every* $n \in \mathbb{N}$, *then* $P = \mathbb{N}$.

Theorem (Complete Induction)

Let $P \subseteq \mathbb{N}$. *Assume that (i)* $0 \in P$ *and (ii) for every* $n \in \mathbb{N}$, $\{0, \ldots, n\} \subseteq P$ *implies that* $(n+1) \in P$. *Then,* $P = \mathbb{N}$.

Induction - another example

A generalization of De Morgan's law to more than two sets. Here, the statement is about sets, not numbers.

Theorem

Let $n \geq 2$. For every n sets $A_1, \dots, A_n$,

$$U \setminus (A_1 \cup \cdots \cup A_n) = \bar{A}_1 \cap \cdots \cap \bar{A}_n. \tag{2}$$

Recursion

A method to define a function (or other structures) for large arguments from small arguments.
Advantages: simple and suits induction.
A recursive definition of a function $f : \mathbb{N} \to \mathbb{N}$ has two parts:

1. the base cases - for small values of n
2. reduction rules - for large values of n

Recursion: the factorial function

Definition

the factorial function $f : \mathbb{N}^+ \to \mathbb{N}^+$ is defined recursively by:

1. Base case: $f(1) = 1$.
2. Reduction rule: $f(n+1) = f(n) \cdot (n+1)$.

Claim

$f(n) = 1 \cdot 2 \cdot \cdots \cdot n$.

Proof.

By induction on n. $\square$

Notation: denote $f(n)$ by $n!$

Recursion: Fibonacci sequence

Definition

We define the function $g : \mathbb{N} \to \mathbb{N}$ recursively as follows.

(i) Base case: $g(0) = 0$ and $g(1) = 1$.

(ii) Reduction rule: $g(n+2) = g(n+1) + g(n)$.

Following the reduction rule we obtain:

$$
\begin{aligned}
g(2) &= g(1) + g(0) = 1 + 0 = 1. \\
g(3) &= g(2) + g(1) = 1 + 1 = 2. \\
g(4) &= g(3) + g(2) = 2 + 1 = 3. \\
g(5) &= g(4) + g(3) = 3 + 2 = 5.
\end{aligned}
$$

Self-reference does not lead to an infinite loop. Why? Self references are to smaller arguments so the chain of self-references eventually ends with a base case.

Recursion: Fibonacci sequence (cont.)

Recall: $g(0) = 0$, $g(1) = 1$, and $g(n+2) = g(n+1) + g(n)$.
Denote the golden ratio by $\varphi \triangleq \frac{1+\sqrt{5}}{2}$. $\varphi \approx 1.62$ is a solution of $x^2 = x + 1$.

Lemma

$\forall n \in \mathbb{N} \quad g(n) \leq \varphi^{n-1}$

Proof: induction on n.

Recursion: conclusions

- a way to define a function, a structure, or even an algorithm.
- bases cases for $n \leq n_0$
- reduction rules for $n > n_0$
- easy to formulate
- easy to prove properties using induction.

One-to-one and Onto Functions

Definition

Let $f : A \to B$ denote a function from A to B.

1. The function f is one-to-one if $a \neq a'$ implies that $f(a) \neq f(a')$.
2. The function f is onto if, for every $b \in B$, there exists an $a \in A$ such that $f(a) = b$.
3. The function f is a bijection if it is both onto and one-to-one.

Comparing Cardinalities

Lemma

Let A and B denote two finite sets. If there exists a one-to-one function $f : A \to B$, then $|A| \leq |B|$.

Lemma

Let A and B denote two finite sets. If there exists an onto function $f : A \to B$, then $|A| \geq |B|$.

Pigeonhole Principle

- If there exists a one-to-one function $f : A \to B$, then $|A| \leq |B|$.
- The contrapositive statement: if $|A| > |B|$, then every function $f : A \to B$ is not one-to-one.

We are now ready to formalize the Pigeonhole Principle, as follows.

The Pigeonhole Principle

Let $f : A \to \{1, \ldots, n\}$, and $|A| > n$, then f is not one-to-one, i.e., there are $a_1, a_2 \in A$; $a_1 \neq a_2$, such that $f(a_1) = f(a_2)$.