Digital Logic Systems

Recitation 2: Sequences and Series & Directed Graphs

Guy Even   Moti Medina

School of Electrical Engineering Tel-Aviv Univ.

March 12, 2019

Function that maps functions

Consider the following set of functions

$$F \triangleq \left\{ f_n \mid f_n : \{0,1\}^n \to \{0,1\} \text{ where } n \in \{2,..,16\} \text{ ; } f_n(a) = 1 \leftrightarrow a = 1^n \right\}$$

Consider the function π defined as follows:

$$\pi : F \to \{2,..,16\} \text{ ; } \pi(f_n) = n$$

Question

Is π a bijection? Yes:

1. *One-to-one, since for a given n, there exists a unique AND_n function.*
2. *Onto, since for every n, there is a valid AND_n function.*

Sequences

1. A sequence is a function from $\mathbb{N}$ to $\mathbb{R}$.
2. Abuse-of-Notation: We usually denote a sequence by f_n where the n^{th} element of the sequence is also denoted f_n.
3. Three popular families of sequences: Arithmetic $a_n = a_0 + n \cdot d$, Geometric $b_n = b_0 \cdot q^n$, Harmonic $\frac{1}{n}$
4. You can recognize a sequence as a member of a specific family if you can parameterize it accordingly:

Example

- $h_n \triangleq -7 \cdot n$ is arithmetic with $a_0 = 0, d = -7$
- $f_n \triangleq 5 \cdot 10^n$ is geometric with $b_0 = 5, q = 10$

Series

1. Series is a sum over a certain number of sequence's elements
2. Notation: sum over elements 0 through n is denoted S_n.
3. We are interested in finding closed-form expressions for series. A closed mathematical expression doesn't contain $\sum$ or Π operators, and can be evaluated by a small number of well known operations $(+,-,\cdot,/)$.
4. $S_n^{Arithmetic} = \sum_{i=0}^{n} a_i = a_0 \cdot (n+1) + d \cdot \frac{n \cdot (n+1)}{2}$
5. $S_n^{Geometric} = \sum_{i=0}^{n} b_i = b_0 \cdot \frac{q^{n+1}-1}{q-1}$

Finding a Closed Form Expression

Example

Write a closed form expression for $\sum_{i=0}^{3n-1} x_i$. Where the sequence

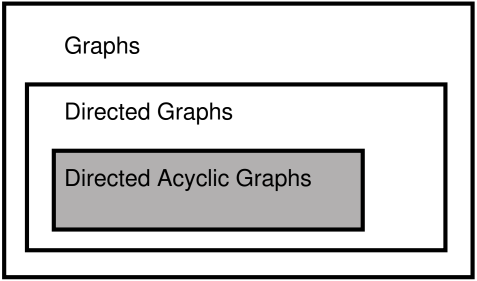
$$x_i = \begin{cases} 0 \text{ if } mod(i,3) = 1 \\ 2^i \text{ otherwise} \end{cases}$$

Solution

$$\begin{aligned}
\sum_{i=0}^{3n-1} x_i &= 2^0 + 0 + 2^2 + 2^3 + 0 + 2^5 + 2^6 + 0 + 2^8 + \ldots + 0 + 2^{3n-1} \\
&= (2^0 + 2^3 + 2^6 + \ldots + 2^{3n-3}) + (2^2 + 2^5 + 2^8 + \ldots + 2^{3n-1}) \\
&= (2^0 + 2^3 + 2^6 + \ldots + 2^{3n-3}) + 4 \cdot (2^0 + 2^3 + 2^6 + \ldots + 2^{3n-3}) \\
&= 5 \cdot \sum_{i=0}^{n-1} 2^{3i} = 5 \cdot \sum_{i=0}^{n-1} 8^i = 5 \cdot \frac{8^n - 1}{8 - 1} = \frac{5}{7} \cdot (8^n - 1)
\end{aligned}$$

Directed Graphs - Definitions

1. A **graph** is defined by a set of vertices V and edges E. $G = (V, E)$.
2. An **edge** $e \in E$ can be directed, in this case $e = (u, v)$ means that the edge is directed from vertex $u \in V$ to $v \in V$.
3. A **vertex** $v \in V$ can be characterized by its $deg_{in}(v)$ which stands for the number of incoming edges, and by $deg_{out}(v)$ which stands for the number of outgoing edges.



DAGs

1. **DAG** stands for Directed Acyclic Graph.
2. Every DAG has at least 1 **source** and at least 1 **sink**.
3. Vertices of any DAG can be sorted in a **Topological Ordering**
4. Topological ordering is described by the **labeling function** π

Definition

A bijection $\pi : V \to \{0, ..., n-1\}$ is a topological ordering if

$$(u, v) \in E \Rightarrow \pi(u) < \pi(v)$$

In other words:

$$\pi(v) < \pi(u) \Rightarrow (u, v) \notin E$$

5. For a given DAG, there can be **multiple** topological orderings.

Algorithm for topological ordering

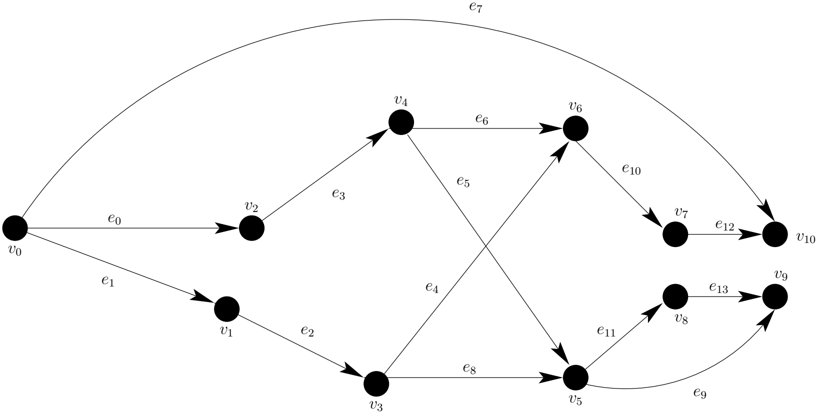
Algorithm $\text{TS}(V, E)$ outputs an ordering: $\pi(u) = 0, \pi(v) = 1, \ldots$
Notation:

$$E_v \triangleq \{e \mid e \text{ enters } v \text{ or emanates from } v\}.$$

Algorithm 1 $\text{TS}(V, E)$ - An algorithm for sorting the vertices of a DAG $G = (V, E)$ in topological ordering.

1. Base Case: If $|V| = 1$, then let $v \in V$ and return $(\pi(v) = 0)$.
2. Reduction Rule:
   1. Let $v \in V$ denote a sink.
   2. return $(\text{TS}(V \setminus \{v\}, E \setminus E_v)$ extended by $(\pi(v) = |V| - 1))$

example of DAG



Algorithm: longest path lengths

Algorithm 2 longest-path-lengths(V, E) - An algorithm for computing the lengths of longest paths in a DAG. Returns a delay function $d(v)$.

1. topological sort: $(v_0, \ldots, v_{n-1}) \leftarrow TS(V, E)$.
2. For $j = 0$ to $(n-1)$ do
    1. If v_j is a source then $d(v_j) \leftarrow 0$.
    2. Else

$$d(v_j) = 1 + \max\left\{d(v_i) \mid i < j \text{ and } (v_i, v_j) \in E\right\}.$$

Algorithm: longest path (not just length)

Question

Given a DAG $G = (V, E)$, design an algorithm that prints the vertices along the longest path.

Hint

You can maintain an auxiliary data structure during the run of the algorithm.

Algorithm: longest path (not just length)

Algorithm 3 longest-path(V, E) - An algorithm for computing the longest path in a DAG. Outputs a delay function $d(v)$ and prints out the sequence of vertices that form the longest path.

1. topological sort: $(v_0, \ldots, v_{n-1}) \leftarrow TS(V, E)$.
2. For $j = 0$ to $(n-1)$ do
   1. If v_j is a source then $d(v_j) \leftarrow 0$.
   2. Else

$$prev(v_j) = \arg\max_{v_i} \left\{ d(v_i) \mid i < j \text{ and } (v_i, v_j) \in E \right\}.$$

$$d(v_j) = 1 + d(prev(v_j))$$

3. $v_j := \arg\max_{v_i} \left\{ d(v_i) \mid v_i \in V \right\}$
4. While (True):
   1. Print v_j
   2. If $d(v_j) = 0$: break;
   3. Else: $v_j \leftarrow prev(v_j)$