EE360C: Algorithms

A Review of Discrete Mathematics

Spring 2019

Department of Electrical and Computer Engineering
University of Texas at Austin

Sets

Set Definitions

- a **set** is a collection of *distinguishable* objects, called **members** or **elements**
    - if x is an element of a set S, we write $x \in S$
    - if x is not an element of set S, we write $x \notin S$
- two sets are equal (i.e., $A = B$) if they contain exactly the same elements
- some special sets:
    - $\emptyset$ is the set with no elements
    - $\mathbb{Z}$ is the set of integer elements
    - $\mathbb{R}$ is the set of real number elements
    - $\mathbb{N}$ is the set of natural number elements

Set Operators

- **subset**: if $x \in A$ implies $x \in B$, then $A \subseteq B$
- **proper subset**: if $A \subseteq B$ and $A \neq B$ then $A \subset B$
- **intersection**: $A \cap B = \{x : x \in A \text{ and } x \in B\}$
- **union**: $A \cup B = \{x : x \in A \text{ or } x \in B\}$
- **difference**: $A - B = \{x : x \in A \text{ and } x \notin B\}$

Relations

Relation Definitions

A **binary relation** R on two sets A and B is a subset of the Cartesian product $A \times B$. If $(a, b) \in R$, we sometimes write $a\,R\,b$.

Consider the relations "$=$", "$<$", and "$\leq$" for each of the following.

- **reflexive**: $R \subseteq A \times A$ is reflexive if $a\,R\,a$ for all $a \in A$
- **symmetric**: R is symmetric if $a\,R\,b$ implies $b\,R\,a$ for all $a, b \in A$
- **transitive**: R is transitive if $a\,R\,b$ and $b\,R\,c$ imply $a\,R\,c$ for all $a, b, c \in A$
- **antisymmetric**: R is antisymmetric if $a\,R\,b$ and $b\,R\,a$ imply $a = b$.

More Relation Definitions

A relation that is reflexive, symmetric, and transitive is an **equivalence relation.** If R is an equivalence relation on set A, then for $a \in A$, the **equivalence class** of a is the set $[a] = \{b \in A : aRb\}$.

Consider $R = \{(a, b) : a, b \in \mathbf{N}$ and $a + b$ is an even number$\}$. Is it reflexive? Is it symmetric? Is it transitive?

A relation that is reflexive, antisymmetric, and transitive is a **partial order**.

A partial order on A is a **total order** if for all $a, b \in A$, $a\ R\ b$ or $b\ R\ a$ hold.

Functions

Function Definitions

Given sets A and B, a **function** f is a binary relation on $A \times B$ s.t. $\forall a \in A$, there exists exactly one $b \in B$ s.t. $(a, b) \in f$.

- A is the **domain** of f (a is an **argument** to the function)
- B is the **co-domain** of f (b is the **value** of the function)

We often write functions as:

- $f : A \to B$
- if $(a, b) \in f$, we write $b = f(a)$

f assigns an element of B to each element of A. No element of A is assigned to two different elements of B, but the same element of B can be assigned to two different elements of A.

More Function Definitions

- A **finite sequence** is a function whose domain is $\{0, 1, \ldots, n-1\}$, often written as $\langle f(0), f(1), \ldots, f(n-1)\rangle$
- An **infinite sequence** is a function whose domain is the set of **N** natural numbers ($\{0, 1, \ldots\}$).
- When the domain of f is a Cartesian product, e.g., $A = A_1 \times A_2 \times \ldots \times A_n$, we write $f(a_1, a_2, \ldots, a_n)$ instead of $f((a_1, a_2, \ldots, a_n))$
- We call each a_i an argument of f even though the argument is really the n-tuple $(a_1, a_2, \ldots, a_n)$

And Still More Function Definitions

If $f : A \to B$ is a function and $b = f(a)$, then we say that b is the **image** of a under f.

- The **range** of f is the image of its domain (i.e., $f(A)$).
- A function is a **surjection** if its range is its codomain. (This is sometimes referred to as mapping A **onto** B.)
  - $f(n) = \lfloor n/2 \rfloor$ is a surjective function from $\mathbb{N}$ to $\mathbb{N}$
  - $f(n) = 2n$ is not a surjective function from $\mathbb{N}$ to $\mathbb{N}$
  - $f(n) = 2n$ is a surjective function from $\mathbb{N}$ to the even numbers

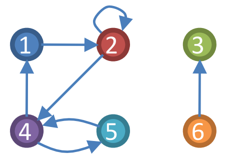
The Last of the Function Definitions

- A function is an **injection** if distinct arguments to f produce distinct values, i.e., $a \neq a'$ implies $f(a) \neq f(a')$. (This is sometimes referred to as a **one-to-one function**.)
  - $f(n) = \lfloor n/2 \rfloor$ is not an injective function from $\mathbb{N}$ to $\mathbb{N}$
  - $f(n) = 2n$ is an injective function from $\mathbb{N}$ to $\mathbb{N}$
- A function is a **bijection** if it is both injective and surjective. (This is sometimes referred to as a **one-to-one correspondence**.)

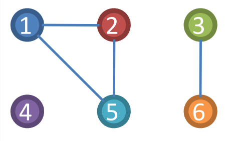
Graphs

Types of Graphs

A **directed graph** (or **digraph**) G is a pair (V, E) where V is a finite set (of "vertices") and E (the "edges") is a subset of $V \times V$.



An **undirected graph** G is a pair (V, E) where V is a finite set (of "vertices") and E (the "edges") is a set of unordered pairs of edges $\{u, v\}$, where $u \neq v$.



Properties of Edges

- If (u, v) is an edge in a *digraph* G, then (u, v) is **incident from** or **leaves** u and is **incident to** or **enters** v.
- If (u, v) is an edge in an undirected graph G, then (u, v) is **incident to** both u and v.
- In both cases, v is **adjacent** to u; in a digraph adjacency is not necessarily symmetric.
- The **degree** of a vertex in an undirected graph is the number of edges incident to it (which is the same as the number of vertices adjacent to it).
- The **out-degree** of a vertex in a digraph is the number of edges leaving it.
- The **in-degree** of a vertex in a digraph is the number of edges entering it.

Paths in Graphs

A **path** from u to v is a sequence of vertices $\langle v_0, v_1, \ldots, v_k \rangle$ s.t. $u = v_0$, $v = v_k$, and $(v_{i-1}, v_i) \in E$ for $i = 1, 2, \ldots, k$.

- The **length** of a path is the number of edges
- The path **contains** the vertices $v_0, v_1, \ldots, v_k$ *and* the edges $(v_0, v_1), (v_1, v_2), \ldots, (v_{k-1}, v_k)$
- v is **reachable** from u if there is a path from u to v
- A path is **simple** if all its vertices are distinct
- A **subpath** of a path p is any $\langle v_i, v_{i+1}, \ldots, v_j \rangle$ where $0 \leq i \leq j \leq k$. (p is a subpath of itself)
- In a digraph, $\langle v_0, v_1, \ldots, v_k \rangle$ is a **cycle** if $v_0 = v_k$ and $k \geq 1$. A cycle is **simple** if all vertices except $v_0 = v_k$ are distinct.
- In an undirected graph, a path $\langle v_0, v_1, \ldots, v_k \rangle$ forms a **cycle** if $v_0 = v_k$, $k \geq 3$ and $v_1, v_2, \ldots, v_k$ are distinct.
- An **acyclic** graph has no cycles.

Connectivity in Graphs

An undirected graph is **connected** if each pair of vertices is connected by a path.

- The **connected components** are the equivalence classes of vertices under the “is reachable from” relation

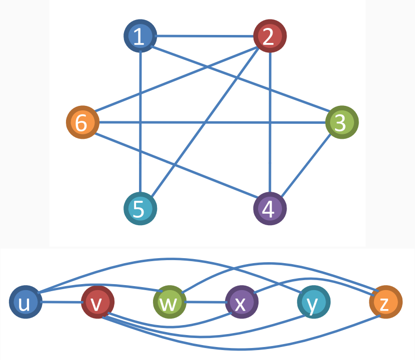
A directed graph is **strongly connected** if every two vertices are reachable from one another

- The **strongly connected components** of a digraph are the equivalence classes of vertices under the “are mutually reachable” relation
- A digraph is strongly connected if it has exactly one strongly connected component

Graph Isomorphism

$G = (V, E)$ is **isomorphic** to $G' = (V', E')$ if there is a 1-to-1 onto function $f : V \to V'$ such that $(u, v) \in E$ if and only if $(f(u), f(v)) \in E'$

- conceptually, we "relabel" G to get G'



Subgraphs and Transformations

The graph $G' = (V', E')$ is a **subgraph** of $G = (V, E)$ if $V' \subseteq V$ and $E' \subseteq E$

- Given $V' \subseteq V$, the **subgraph induced by** V' is $G' = (V', (V' \times V') \cap E)$, or, equivalently, $E' = \{(u, v) \in E : u, v \in V'\}$

Given an undirected graph $G = (V, E)$, the **directed version** of G is the graph $G' = (V, E')$, where $(u, v) \in E'$ if and only if $(u, v) \in E$

- Conceptually, we introduce two edges for each original edge

Given a directed graph $G = (V, E)$, the **undirected version** of G is the graph $G' = (V, E')$ where $(u, v) \in E'$ if $u \neq v$ and $(u, v) \in E$.

- Conceptually, we remove directionality and self-loops

Special Graphs

- **complete graph**: an undirected graph in which every pair of vertices is adjacent
- **bipartite graph**: an undirected graph in which the vertex set can be partitioned into two sets V_1 and V_2 such that every edge in the graph is of the form (x, y) where $x \in V_1$ and $y \in V_2$.
- **forest**: an acyclic undirected graph
- **tree**: a connected, acyclic undirected graph
- **dag**: directed acyclic graph

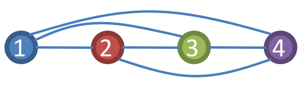


Figure 2: A complete graph

Additional Types of Graphs

- **multigraph**: like an undirected graph but can have multiple edges between vertices and self-loops
- **hypergraph**: like an undirected graph, but each **hyperedge** can connect an arbitrary number of vertices

Trees

Theorem (Properties of Trees)

Let $G = (V, E)$ be an undirected graph. Then the following are equivalent statements:

1. *G is a tree.*
2. *Any two vertices of G are connected by a unique simple path.*
3. *G is connected, but if any edge is removed from E, the resulting graph will not be connected.*
4. *G is connected and $|E| = |V| - 1$*
5. *G is acyclic and $|E| = |V| - 1$*
6. *G is acyclic, but if any edge is added to E, the resulting graph contains a cycle*

Rooted Trees

A **rooted tree** is a tree in which one vertex is distinguished from the others.

- the distinguished vertex is called the **root**
- a vertex in a rooted tree is often called a **node**

Let r be the root of a rooted tree T. For any node x, there is a unique path from r to x.

- any node y on a path from r to x is an **ancestor** of x
- if y is an ancestor of x, then x is a **descendant** of y
- every node is its own ancestor and descendant
- a **proper ancestor (descendant)** is an ancestor (descendant) that is not the node itself
- the **subtree rooted at** x is the tree induced by the descendants of x

More on Rooted Trees

If the last edge of the path from r to x is (y, x), then y is the **parent** of x and x is the **child** of y

- The root is the only node with no parent
- **siblings**: two nodes that share the same parent
- **leaf**: a node with no children (aka an **external node**)
- **internal node**: a non-leaf node

The number of children of a node x in a rooted tree T is called the **degree** of x.

The length of a path from r to x is called the **depth** of x.

- The largest depth of any node in T is the **height** of T

An **ordered tree** is a rooted tree in which the children at each node are ordered.

Binary Trees

Binary trees are defined recursively. A **binary tree** T is a structure defined on a finite set of nodes that either:

1. contains no nodes (we call this **empty** or **null** or NIL)
2. is composed of three disjoint sets of nodes: a **root node**, a **left subtree**, and a **right subtree**

If the left subtree of a binary tree is nonempty, its root is called the **left child**; similar definition of the **right child**.

A **full binary tree** is a binary tree in which each node is either a leaf or has degree 2.

A binary tree is not just an ordered tree in which each node has degree at most two. Left and right children matter.

Questions