COMP 251: Algorithms & Data Structures

Owen Lewis

Winter 2018

Contents

1 Overview of Graph Theory

1.1 Definitions

A graph $G = (V, E)$ is a set V of vertices (a.k.a. nodes) and a set E of edges (denoting vertex pairs). We set $n = |V|$, and $m = |E|$. A graph is said to be *undirected* when for any edge $(u, v) \in E$ there exists an edge $(v, u) \in E$ for some nodes u, and v. A graph is said to be *directed* if it is not undirected. In other words, the edge set of a directed graph consists of ordered pairs where the edge set of an undirected graph consists of unordered pairs.

A *walk* is a set of vertices $\{v_0, v_1, \ldots, v_\ell\}$ such that $(v_i, v_{i+1}) \in E$, $\forall\ 0 \leq i \leq \ell$. A walk where $v_0 = v_\ell$ is said to be a *circuit* or a *closed walk*. A circuit where every edge in the graph is used exactly once is known as an *Eulerian circuit*. A *cycle* is a walk $\{v_0, v_1, \ldots, v_\ell\}$ such that every vertex is distinct except $v_0 = v_\ell$. A cycle where every vertex of the graph is used exactly once is known as a *Hamiltonian cycle*. A walk where every vertex is distinct is said to be a *path*.

A graph is said to be *connected* if for each $u, v \in V$ there exists a walk from u to v. A graph is said to be *disconnected* id it is not connected. Each connected subgraph of a graph is called a *component*. A connected graph therefore has exactly one component.

A connected component with no cycles is called a *tree*. A graph whose components are all trees is said to be a *forrest*. A tree is said to be *spanning* if it contains every vertex in the graph. A vertex in a tree with at most one neighbour is called a *leaf*.

A *matching* is a set of vertex-disjoint edges i.e. each edge is incident to at most one other edge in a matching. A matching is said to be *perfect* if every vertex is incident to exactly one edge in the matching.

A *clique* is a set of pairwise adjacent vertices. In *independent set* (a.k.a. a *stable set*) is set of pairwise non-adjacent vertices.

A *bipartite graph* is a graph such that the vertex set V can be partitioned as $V = X \cup Y$ where each edge has one node in X and the other node in Y. Note that X and Y are necessarily independent sets.

1.2 Some Theorems for Undirected Graphs

Theorem: (Handshaking Lemma) Let $G = (V, E)$ be an undirected graph, let $\Gamma(v) := \{u : (u, v) \in E\}$ be the set of neighbours of a node v, and let the *degree* $\deg(v)$ of a vertex v equal the cardinality of $\Gamma(v)$. Then there are an even number of vertices with odd degree.
Proof: First note that since we're double-counting the number of pairs where

(v, e) is an edge incident to v

$$2 \cdot |E| = \sum_{v \in V} \deg(v)$$

Since the degree of a vertex is either even or odd, we can partition V into a set of odd-degree vertices $\mathcal{O}$, and a set of even-degree vertices $\mathcal{E}$. This gives us

$$\sum_{v \in V} \deg(v) = \sum_{v \in \mathcal{O}} \deg(v) + \sum_{v \in \mathcal{E}} \deg(v)$$

which implies

$$\sum_{v \in \mathcal{O}} \deg(v) = 2 \cdot |E| - \sum_{v \in \mathcal{E}} \deg(v)$$

since both the $2 \cdot |E|$ term is even (obvious) and the $\sum_{v \in \mathcal{E}} \deg(v)$ term is even (sum of even numbers) then the $\sum_{v \in \mathcal{O}} \deg(v)$ term must also be even. □

Theorem: (Euler's Theorem) If G is an undirected graph then G contains an Eulerian circuit if and only if every vertex has even degree.
Proof: Easy proof by induction

Lemma: A tree T with $n \geq 2$ vertices has at least one leaf vertex.
Proof: Trees are connected so there exists no vertices with degree 0 when $n \geq 2$. Suppose each vertex has degree of at least 2. Then consider the longest path $P \subseteq T$, $P = \{v_1, v_2, \ldots, v_{\ell-1}, v_\ell\}$. Since $\deg(v_\ell) \geq 2$, $\exists$ a neighbour (of v_ℓ) $x \in P$ with $x \neq v_{\ell-1}$. If $x = v_{\ell+1}$ then P is not the longest path, a contradiction. Therefore, for P to be the longest path, x must be somewhere else in P, but this creates a cycle, another contradiction. Thus there must exist at least one node v such that $0 < \deg(v) < 2$ – a leaf. □

Theorem: A tree with n vertices has exactly $n - 1$ edges.
Proof: Simple proof by induction.
Base case: A tree with one vertex trivially has 0 edges.
Induction Hypothesis: Assume any tree with $n - 1$ vertices has $n - 2$ edges.
Inductive Step: Take a tree with $n \geq 2$ vertices. By the previous lemma this tree contains a leaf vertex v. This implies that $T \setminus \{v\}$ is a tree with $n - 1$ vertices and by the induction hypothesis $T \setminus \{v\}$ is a tree with $n - 2$ edges, which imples that T is a tree with $n - 1$ edges. □

Theorem: (Hall's Theorem) Let $G = (X \cup Y, E)$ with $|X| = |Y|$ be a bipartite graph. G contains a perfect matching if and only if $\forall\ B \subseteq X$, $|\Gamma(B)| \geq |B|$ (Hall's condition).
Proof: Firstly, the ($\Rightarrow$) direction is fairly obvious. If $B \subseteq X$ with $\Gamma(B) < |B|$ then the graph can't have a perfect matching. The ($\Leftarrow$) direction is a bit trickier. Suppose Hall's condition is satisfied. Then, take the maximum cardinality

matching M is the graph. If M is perfect then we are done. Otherwise there must exist an unmached vertex b_0.

- Since Hall's condition holds, we have $|\Gamma(\{b_0\})| \geq |\{b_0\}| = 1$ so b_0 must have at least one neighbour s_0.
- Suppose s_0 is matched in M to b_1.
- Since Hall's condition holds, we have $|\Gamma(\{b_0, b_1\})| \geq |\{b_0, b_1\}| = 2$ so $\{b_0, b_1\}$ must have at least one neighbour $s_1 \neq s_0$.
- Suppose s_1 is matched in M to b_2.
- Since Hall's condition holds, we have $|\Gamma(\{b_0, b_1, b_2\})| \geq |\{b_0, b_1, b_2\}| = 3$ so $\{b_0, b_1, b_2\}$ must have at least one neighbour $s_2 \notin \{s_0, s_1\}$.
- ...

we repeat this argument as long as we can. Since the graph contains a finite number of vertices this process must terminate, but it can only terminate when we reach an unmatched node s_k. Using the edges we've formed in M we can create a path P from b_0 to s_k that alternates between using non-matching edges and using matching edges. Swapping the matching edges with the non-matching edges gives us one more matching edge (as we have an odd number of edges.) This is still a valid matching as the internal nodes of P are still incident to exactly one matching edge. Also, the end nodes, b_0 and s_k were previously unmatched but are now incident to exactly one edge in the new matching. Thus M isn't the maximum capacity matching – a contradiction. □

1.3 """Data Structures""" for Representing Graphs

1.3.1 Adjacency Matrices

For a graph, an *adjacency matrix* M is a matrix suxh that

1. There is a row for each vertex
2. There is a column for each vertex
3. The $ij - th$ entry is defined as $M_{ij} = \begin{cases} 1, (i,j) \in E \\ 0, (i,j) \notin E \end{cases}$

Note that in an undirected graph the matric is symmetric around the diagonal because $(i, j) \sim (j, i)$. Of course this is not necessarily true of directed graphs.

1.3.2 Adjacency Lists

An *adjacency list* of an undirected graph is such that for each vertex v of V we store a list of its neighbours. For a directed graph we have two lists: one in which we store the in-neighbours of v and one in which we store the out-neighbours of v.

1.3.3 Adjacency Matrices vs. Adjacency Lists

The main difference between the two is the amount of storage required to implement them.

- An adjacency matrix requires we store $\Theta(n^2)$ numbers
- An adjacency list requires we store $\Theta(m)$ numbers

In any graph $m = O(n^2)$. This means that for a sparse graph adjacency lists are highly favourable in terms of space complexity.

Verifying whether an edge exists, however, is much faster in an adjacency matrix – when using the array representation of a matrix it takes $O(1)$ time, where verifying the existance of an edge takes $O(\log n)$ time for an ordered adjacency list (using binary search), and $O(n)$ time if the adjacency list is not ordered (using sequential search).

2 Divide & Conquor

3 Greedy Algorithms

4 Dynamic Programming

5 Network Flows

6 Data Structures