

AlgoLib

The AlgoLib Team

September 21, 2026

Contents

1	Graph Theory	2
1.1	Basic	2
1.2	Structures	2
1.2.1	Trees	2
1.3	Connectivity	2
1.3.1	Reachability and distance	2
1.4	Coloring	3
1.5	Matching	3
1.6	Minors	3
1.7	Spectral	3
1.8	Embeddings	3
2	Graph Algorithms	4
2.1	Traversal	4
2.1.1	Breadth-first search	4
2.2	Search	5
2.3	Shortest Path	5
2.4	MST	5
2.5	Flow	5
2.6	SCC	5
3	Data Structures	6
3.1	Binary Tree	6
3.2	Splay Tree	6
3.3	Union Find	6
3.3.1	Inverse Ackermann	6

Chapter 1

Graph Theory

1.1 Basic

Definition 1.1.1. An (*undirected*) *graph* is a pair $G = (V, E)$ consisting of a set V and a multi-set E of unordered pairs of elements in V . Elements of V are called *vertices* and elements of E are called *edges*. For two vertices $u, v \in V$, we say u and v are *adjacent* if $\{u, v\} \in E$. We call an undirected graph G *simple* if all edges are distinct (i.e., E is a set), and each edge is a pair of distinct vertices.

Definition 1.1.2. A *directed graph* is a pair $G = (V, E)$ consisting of a set V and a multi-set E of ordered pairs of elements in V . Elements of V are called *vertices* and elements of E are called *edges*. We call a directed graph G *simple* if all edges are distinct and each edge is a pair of distinct vertices.

1.2 Structures

1.2.1 Trees

1.3 Connectivity

1.3.1 Reachability and distance

Definition 1.3.1 (Reachability). Let G be a simple graph or a simple directed graph. A vertex v is *reachable* from u if some walk realized in G runs from u to v . In the undirected case reachability is an equivalence relation on $V(G)$; in the directed case it is only a preorder, since walks cannot be reversed.

Definition 1.3.2 (Symmetric orientation). The *symmetric orientation* of a simple graph G is the simple directed graph on $V(G)$ whose arcs are the ordered pairs (u, v) with $\{u, v\} \in E(G)$. A walk is realized in G exactly when it is realized in the symmetric orientation, so reachability and distances agree.

Definition 1.3.3 (Distance). The *distance* $\text{dist}(u, v) \in \mathbb{N} \cup \{\infty\}$ from u to v in a simple directed graph is the least length of a walk from u to v , and ∞ if v is not reachable from u . For a simple graph it is the distance in the symmetric orientation, and is then symmetric in u and v .

1.4 Coloring

1.5 Matching

1.6 Minors

1.7 Spectral

1.8 Embeddings

Chapter 2

Graph Algorithms

2.1 Traversal

2.1.1 Breadth-first search

Throughout, G is a simple directed graph whose vertex set $V(G)$ is finite and given as data, and whose arc membership is decidable. Undirected graphs are searched through their symmetric orientation (Definition 1.3.2), so one search serves both kinds of graph.

Definition 2.1.1 (BFS layers). Let $s \in V(G)$. The *layers* of breadth-first search from s are the sets

$$L_0 = \{s\}, \quad L_{k+1} = N^+(L_k) \setminus (L_0 \cup \dots \cup L_k),$$

where $N^+(L)$ denotes the set of out-neighbours of the vertices of L . The set of vertices *visited* within k rounds is $V_k = L_0 \cup \dots \cup L_k$. When $s \notin V(G)$ every layer is empty. In Lean the search is a state (V_k, L_k) iterated k times from $(\{s\}, \{s\})$, so that the definition reduces and **decide** can evaluate it.

Lemma 2.1.2 (Soundness). *Every vertex of L_k is the endpoint of a walk of length k from s .*

Proof. By induction on k : a vertex of L_{k+1} is an out-neighbour of a vertex of L_k , and appending that arc to the walk given by the induction hypothesis yields a walk of length $k+1$. $\square$

Lemma 2.1.3 (Completeness). *The endpoint of a walk of length ℓ from s lies in V_ℓ .*

Proof. By induction on the walk. The key step is that the out-neighbours of a vertex of V_k lie in V_{k+1} : such a vertex lies in some L_j with $j \leq k$, and its out-neighbours are either already in V_j or in L_{j+1} . $\square$

Lemma 2.1.4 (Termination). *If $L_k \neq \emptyset$ then $k < |V(G)|$. Hence the search from s has exhausted every layer after $|V(G)|$ rounds.*

Proof. An empty layer is followed by an empty layer, so $L_0, \dots, L_k$ are all non-empty. They are pairwise disjoint subsets of $V(G)$, so $k+1 \leq |V_k| \leq |V(G)|$. $\square$

Theorem 2.1.5 (The layers are the spheres). *For every vertex v and every $k \in \mathbb{N}$,*

$$v \in L_k \iff \text{dist}(s, v) = k.$$

Proof. Let $v \in L_k$. Soundness gives $\text{dist}(s, v) \leq k$. For the reverse inequality there is nothing to show when $k = 0$; when $k = j + 1$, a walk from s to v of length at most j would place v in V_j by completeness, whereas L_{j+1} avoids V_j by construction, so $\text{dist}(s, v) > j$, i.e. $\text{dist}(s, v) \geq k$. Conversely, if $\text{dist}(s, v) = k$ then a shortest walk places $v \in V_k$, so $v \in L_j$ for some $j \leq k$, and the first part gives $j = k$. $\square$

Theorem 2.1.6 (BFS computes reachability and distance). *The set $V_{|V(G)|}$ is exactly the set of vertices reachable from s , and the least $k < |V(G)|$ with $v \in L_k$ (or ∞ if there is none) is exactly $\text{dist}(s, v)$.*

Proof. Soundness gives one inclusion and one inequality; completeness together with the termination bound gives the other. $\square$

2.2 Search

2.3 Shortest Path

2.4 MST

2.5 Flow

2.6 SCC

Chapter 3

Data Structures

3.1 Binary Tree

3.2 Splay Tree

3.3 Union Find

3.3.1 Inverse Ackermann

Bibliography