8 GRAPHS

8.1 INTRODUCTION

The structure of mathematics is based on relations between the elements of sets. The sets contain given elements. The relations are also sets and contain tuples which are formed from the elements of sets according to given rules of operation. In this manner, relationships between selected elements of sets are described symbolically.

The relations between elements may be visualized diagrammatically. In the diagram, the elements are represented as vertices, whereas the relationships are represented as edges. Simple relations can thus be represented visually in a plane. A relation for which such a visualization exists is called a graph. The diagram of vertices and edges is often also called a graph.

Since a graph is a visualizable relation, the algebra of relations forms the basis of graph theory. The algebra of relations for finite sets may be transformed into a boolean vector and matrix algebra. Basic definitions and rules of the algebra of relations for finite graphs are treated in Section 8.2.

Various applications require graphs with specific properties. Simple graphs, directed graphs, bipartite graphs, multigraphs and hypergraphs are distinguished with respect to these properties. The nomenclature for graphs varies considerably in the literature. For instance, simple and directed graphs are often also called ordinary graphs and digraphs, respectively. The different classes of graphs, their properties and their relationships are treated in Section 8.3.

Graphs may be classified with respect to their structural properties. For this purpose, the graph is considered as a domain consisting of vertices and edges. An edge sequence in the graph is a chain of connected edges which form either a path or a cycle. The study of paths and cycles leads to a definition of different forms of connectedness of graphs. The removal of some of the vertices and edges of a graph (a cut in the graph) leads to subgraphs; their connectedness is an essential structural property of the graph. The fundamentals of the structural analysis and further classification of simple and directed graphs are treated in Section 8.4.

The determination of paths in networks with specific properties is a basic problem of graph theory. A network is represented as a weighted graph in which the edges are weighted according to the properties under consideration. The various path problems are unified by abstraction. This leads to a path algebra for weighted graphs. The fundamentals of the path algebra and the algebraic methods of solution are treated in Section 8.5.

The determination of flows in networks is a problem in graph theory related to the theory of optimization. As in the case of path problems, the networks for flow problems are represented by weighted graphs. The flows in the network must satisfy the law of conservation of mass and may be bounded by given capacities. To determine optimal flows, the principles of optimization are applied to graph theory. The fundamentals of flows in networks are treated in Section 8.6.

Graph theory has a large spectrum of applications. In computer science, for example, graph theory is applied in the theory of automata, in the theory of networks and in connection with formal languages and data structures. In engineering, it is applied to object-oriented modelling in the study of communications, transport and supply systems and of planning, decision and production processes. Some applications are shown as examples in connection with the theoretical foundations. The theoretical foundations treated here, as well as their applications, are restricted to finite graphs.

8.4 STRUCTURE OF GRAPHS

8.4.1 INTRODUCTION

Structure : The structure of a graph is uniquely determined by the relations of the domain. For example, the structure of a directed graph (V ; R) is determined by the edge relation R. In analogy with vector algebra, topology and group theory, the question arises whether a graph may be decomposed into subgraphs which have simple structural characteristics and yield insight into the essential structural properties of the graph. Paths and cycles are examples of such subgraphs.

The foundations for the structural analysis of graphs are first treated for directed graphs with directed edges and then transferred to simple graphs with undirected edges. Multigraphs and hypergraphs may be transformed into directed and simple graphs, so that the foundations of structural analysis treated here also apply to these graphs.

Paths and cycles : Graphs consist of vertices and edges. An edge sequence in a graph is a chain of connected edges. An open edge sequence with different start and end vertices is a path. A closed edge sequence with identical start and end vertices is a cycle. Paths and cycles can be simple or elementary. Graphs without cycles are called acyclic graphs. Graphs which consist entirely of cycles are called cyclic graphs. Paths and cycles in directed and simple graphs are treated in Sections 8.4.2 and 8.4.5.

Connectedness : A graph is said to be connected if there is an edge sequence between any two vertices. Different forms of connectedness are defined for directed graphs, in particular strong and weak connectedness. Every graph which is not strongly or weakly connected has a unique decomposition into strongly or weakly connected components. The connectedness of directed and simple graphs is treated in Sections 8.4.3 and 8.4.6.

Cuts : The effects of removing edges and vertices on the connectedness of a graph are studied. Edges are cut, or vertices are excised together with the incident edges, and the connectedness of the remaining graph is studied. These considerations lead to a classification of edges and vertices and to the definition of multiple vertex-disjoint connectedness and multiple edge-disjoint connectedness of graphs. Cuts in directed and simple graphs are treated in Sections 8.4.4 and 8.4.7.

8.4.2 PATHS AND CYCLES IN DIRECTED GRAPHS

Introduction : A directed graph $G = (V ; R)$ is a structured set. It consists of the vertex set V and a binary vertex relation R which corresponds to a set of directed edges. The vertex set V is equipped with structure by the vertex relation R. The structural properties of a directed graph are entirely determined by the properties of the relation R.

Various concepts are introduced in order to study the structural properties of directed graphs and to cast them in a mathematical form. The definition of paths and cycles in a directed graph forms the basis of the structural analysis. The existence of paths and cycles between two vertices leads to the formation of the transitive closure R^+ of the relation R. The properties of the transitive closure allow a classification into acyclic, anticyclic and cyclic graphs. The essential concepts and fundamentals for the structural analysis of directed graphs are treated in the following.

Predecessor and successor : A vertex x is called a predecessor of a vertex y if there is an edge from x to y in the graph, so that the ordered vertex pair (x,y) is contained in the relation R. If x is a predecessor of y, then y is called a successor of x.

$$\begin{array}{lcll} x \text{ predecessor of } y & \Leftrightarrow & (x, y) \in R & \Leftrightarrow \\ y \text{ successor of } x & \Leftrightarrow & (y, x) \in R^T & \end{array}$$

A vertex x in a vertex set V may be regarded as a unary point relation in V. In the following, this unary point relation is also designated by x. The predecessorship and the successorship of vertices $x,y \in V$ are formulated as an inclusion using such unary relations :

$$\begin{array}{lcll} x \text{ predecessor of } y & \Leftrightarrow & x\, y^T \sqsubseteq R & \Leftrightarrow \\ y \text{ successor of } x & \Leftrightarrow & y\, x^T \sqsubseteq R^T & \end{array}$$

The set of all predecessors of a vertex $x \in V$ is designated by $t_P(x)$, the set of all successors of x by $t_S(x)$. The sets $t_P(x)$ and $t_S(x)$ are unary relations in V and are determined as follows using the edge relation R :

$$\begin{array}{lll} \text{predecessors of } x : & & t_P(x) = R\, x \\ \text{successors of } x : & & t_S(x) = R^T x \end{array}$$

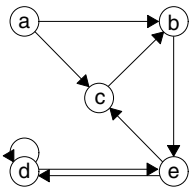
Indegree and outdegree : The number of predecessors of a vertex x is called the indegree of x and is designated by $g_P(x)$. The indegree $g_P(x)$ corresponds to the number of elements in the set $t_P(x)$, and hence to the number of directed edges which end at the vertex x. The number of successors of a vertex x is called the outdegree of x and is designated by $g_S(x)$. The outdegree $g_S(x)$ corresponds to the number of elements in the set $t_S(x)$, and hence to the number of directed edges which emanate from the vertex x.

indegree	$g_P(x) = \lvert t_P(x) \rvert = \lvert R\,x \rvert$
outdegree	$g_S(x) = \lvert t_S(x) \rvert = \lvert R^T x \rvert$

The sum of the indegrees of all vertices $x \in V$ is equal to the number of directed edges of the directed graph, and hence coincides with the number of elements of the relation R. The same is true for the outdegrees.

sum	$\sum_{x \in V} g_P(x) = \sum_{x \in V} g_S(x) = \lvert R \rvert$

Example 1 : Predecessors, successors and degrees in directed graphs



	a	b	c	d	e
a	0	1	1	0	0
b	0	0	0	0	1
c	0	1	0	0	0
d	0	0	0	1	1
e	0	0	1	1	0

R

a	0
b	0
c	0
d	0
e	1

e

a	0
b	1
c	0
d	1
e	0

$\mathbf{t}_P(e)$

a	0
b	0
c	1
d	1
e	0

$\mathbf{t}_S(e)$

x	a	b	c	d	e	
$g_P(x)$	0	2	2	2	2	$\sum g_P(x) = 8$
$g_S(x)$	2	1	1	2	2	$\sum g_S(x) = 8$

The directed graph shown above consists of 5 vertices and 8 directed edges. The relation is specified by a boolean matrix **R**. The unary point relation for the vertex e is shown as a boolean unit vector **e**. The product **R e** yields the boolean vector $\mathbf{t}_P(e)$ for the set of predecessors of e. It is identical with the column of **R** which is associated with the vertex e. The product $\mathbf{R}^T\mathbf{e}$ yields the boolean vector $\mathbf{t}_S(e)$ for the set of successors of e. It is identical with the row of **R** which is associated with the vertex e. The indegrees and the outdegrees of all vertices are compiled. The sum of the indegrees and the sum of the outdegrees are both equal to the number of edges.

Edge sequence : A chain of edges is called an edge sequence if the end vertex of each edge except for the last edge is the start vertex of the following edge.

edge sequence $< (x_0, x_1),\ (x_1, x_2)\ ,...,\ (x_{n-1}, x_n) >$

condition $\bigwedge_{j=1}^{n} ((x_{j-1}, x_j) \in R)$

The start vertex x_0 of the first edge and the end vertex x_n of the last edge are called the start vertex and the end vertex of the edge sequence, respectively. The vertices x_1 to x_{n-1} are called intermediate vertices of the edge sequence. The number n of edges is called the length of the edge sequence. An edge may occur more than once in an edge sequence.

Ancestors and descendants : A vertex x is called an n-th ancestor of a vertex y if there is an edge sequence of length n from x to y in the graph. If x is an n-th ancestor of y, then y is called an n-th descendant of x. A 1-st ancestor or 1-st descendant of x is a predecessor or successor of x, respectively. The n-th ancestors and descendants of x are determined recursively from the relationships for predecessors and successors according to the following rule :

n-th ancestors of x :

$$t_P^{(k)}(x) = R\ t_P^{(k-1)}(x) \quad \text{for} \quad k = 1,...,n \quad \text{with} \quad t_P^{(0)}(x) = x$$

$$t_P^{(n)}(x) = R^n\ x \quad \text{for} \quad n > 0$$

n-th descendants of x :

$$t_S^{(k)}(x) = R^T t_S^{(k-1)}(x) \quad \text{for} \quad k = 1,...,n \quad \text{with} \quad t_S^{(0)}(x) = x$$

$$t_S^{(n)}(x) = (R^n)^T\ x \quad \text{for} \quad n > 0$$

The set of all ancestors of a vertex x is designated by $t_P^+(x)$; it is determined as the union of the sets of n-th ancestors of x. The set $t_S^+(x)$ of all descendants of x is determined analogously. The transitive closure R^+ of a relation R with stability index s, defined in Section 8.2.6, may be used to determine these sets :

ancestors of x :

$$t_P^+(x) = t_P^{(1)}(x) \sqcup ... \sqcup t_P^{(s)}(x) = Rx \sqcup ... \sqcup R^s x = R^+ x$$

descendants of x :

$$t_S^+(x) = t_S^{(1)}(x) \sqcup ... \sqcup t_S^{(s)}(x) = R^T x \sqcup ... \sqcup R^{sT} x = R^{+T} x$$

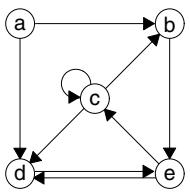
Path : A path from a start vertex x via intermediate vertices to an end vertex y is an edge sequence. In a directed graph, a path may be uniquely represented as a vertex sequence < x,...,y >. A path < x > with the same start and end vertex x contains no edges and is called an empty path. The length of an empty path is 0. There is an empty path for every vertex of a directed graph. The existence of non-empty paths in a directed graph is established as follows :

$$\begin{aligned} &\text{there is a path of length n from x to y} && \Leftrightarrow && xy^T \sqsubseteq R^n \\ &\text{there is a non-empty path from x to y} && \Leftrightarrow && xy^T \sqsubseteq R^+ \end{aligned}$$

Cycle : A non-empty path whose start vertex and end vertex coincide is called a cycle. A loop at a vertex is a cycle of length 1. A cycle which contains no loops is called a proper cycle. If there is a non-empty path from x to y and a non-empty path from y to x, then the concatenation of the two paths yields a cycle through x and y. The existence of cycles in a directed graph is established as follows :

$$\begin{aligned} &\text{there is a cycle of length } n > 0 \text{ through x} && \Leftrightarrow && xx^T \sqsubseteq R^n \\ &\text{there is a cycle through x} && \Leftrightarrow && xx^T \sqsubseteq R^+ \\ &\text{there is a cycle through x and y} && \Leftrightarrow && xy^T \sqsubseteq R^+ \sqcap R^{+T} \end{aligned}$$

Example 2 : Ancestors and descendants, paths and cycles in directed graphs



	a	b	c	d	e
a	0	1	0	1	0
b	0	0	0	0	1
c	0	1	1	1	0
d	0	0	0	0	1
e	0	0	1	1	0

R

	a	b	c	d	e
a	0	0	0	0	1
b	0	0	1	1	0
c	0	1	1	1	1
d	0	0	1	1	0
e	0	1	1	1	1

$\mathbf{R}^2$

	a	b	c	d	e
a	0	1	1	1	1
b	0	1	1	1	1
c	0	1	1	1	1
d	0	1	1	1	1
e	0	1	1	1	1

$\mathbf{R}^+$

The relation R, the product R^2 and the transitive closure R^+ for the graph under consideration are shown as boolean matrices. The second ancestors of a vertex are read off from the column of the matrix $\mathbf{R}^2$ associated with that vertex. The second ancestors of vertex e are a, c, e. The second descendants of a vertex are read off from the row of the matrix $\mathbf{R}^2$ associated with that vertex. The second descendants of vertex b are c, d. In the same way, the ancestors and descendants are read off from the matrix $\mathbf{R}^+$. Vertex a does not have any ancestors, but it has the descendants b, c, d, e.

The existence of paths of length 2 may be read off directly from the elements of the matrix $\mathbf{R}^2$. There are paths of length 2 from vertex a to e, namely < a, b, e > and < a, d, e >. There is no path of length 2 from vertex d to e. There is a path of length 2 from vertex d to vertex d, namely the cycle < d, e, d >. There is a path of length 2 from vertex c to vertex c, namely the improper cycle < c, c, c >. The existence of

non-empty paths may also be read off from the elements of the matrix $\mathbf{R}^+$. There is no path from vertex e to vertex a. There is a non-empty path from vertex e to d, namely $< e, d >$, $< e, c, d >$, $< e, c, b, e, d >$,... . There are non-empty paths from vertex e to e, namely the cycles $< e, d, e >$, $< e, c, c, d, e >$, $< e, c, b, e >$,... .

Acyclic graph : A directed graph $G = (V ; R)$ is said to be acyclic if it does not contain any cycles. The transitive closure R^+ of an acyclic graph is asymmetric. If there is a non-empty path from x to y, then there is no non-empty path from y to x, since otherwise the concatenation of the two paths would yield a cycle.

$$\text{acyclic graph} \quad :\Leftrightarrow \quad R^+ \sqcap R^{+T} = 0$$

Anticyclic graph : A directed graph $G = (V ; R)$ is said to be anticyclic if it does not contain any proper cycles. In contrast to acyclic graphs, an anticyclic graph may contain loops at the vertices. The transitive closure R^+ of an anticyclic graph is antisymmetric.

$$\text{anticyclic graph} \quad :\Leftrightarrow \quad R^+ \sqcap R^{+T} \sqsubseteq I$$

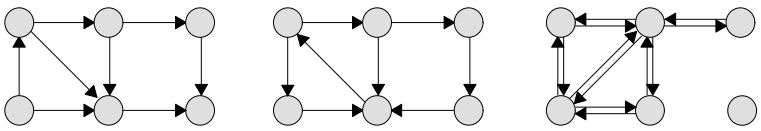
Cyclic graph : A directed graph $G = (V ; R)$ is said to be cyclic if every non-empty path in G belongs to a cycle. The transitive closure R^+ of a cyclic graph is symmetric. If there is a non-empty path from x to y, then there is also a non-empty path from y to x, so that the concatenation of the two paths yields a cycle.

$$\text{cyclic graph} \quad :\Leftrightarrow \quad R^+ = R^{+T}$$

Properties : The following relationships hold between the properties of a relation R and of its transitive closure R^+. If the transitive closure R^+ is asymmetric or antisymmetric, then the relation R is asymmetric or antisymmetric, respectively. If the relation R is symmetric, then the transitive closure R^+ is symmetric. These relationships lead to the following implications :

acyclic graph	$\Rightarrow$	asymmetric graph
anticyclic graph	$\Rightarrow$	antisymmetric graph
cyclic graph	$\Leftarrow$	symmetric graph

Example 3 : Properties of graphs



asymmetric acyclic | asymmetric cyclic | symmetric cyclic

These examples show that while every acyclic graph is asymmetric, not every asymmetric graph is acyclic. They also show that while every symmetric graph is cyclic, not every cyclic graph is symmetric.

Simple path : A non-empty path is said to be simple if it does not contain any edge more than once. The vertices and the edges of a simple path form a subgraph of the directed graph. If the start vertex and end vertex of a simple path are different, the following relationships hold between the indegrees and the outdegrees of the vertices of the corresponding subgraph :

subgraph for a simple path $< x,...,z,...,y >$ with $x \neq y$	
start vertex	$g_S(x) = g_P(x) + 1$
intermediate vertex	$g_S(z) = g_P(z)$
end vertex	$g_S(y) = g_P(y) - 1$

Simple cycle : A simple path whose start vertex and end vertex coincide is called a simple cycle. In the subgraph for a simple cycle, the indegree and the outdegree of each vertex are equal.

subgraph for a simple cycle with vertex z	
vertex	$g_S(z) = g_P(z)$

Eulerian paths and cycles : A simple path with different start and end vertices is called an Eulerian path if it contains all edges of the directed graph. A simple cycle is called an Eulerian cycle if it contains all edges of the directed graph.

Elementary path : A non-empty path is said to be elementary if it does not contain any vertex more than once. The vertices and the edges of an elementary path form a subgraph. If the start vertex and end vertex of an elementary path are different, then the vertices of the corresponding subgraph have the following indegrees and outdegrees :

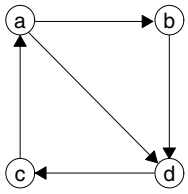
subgraph for an elementary path $< x,...,z,...,y >$ with $x \neq y$		
start vertex	$g_S(x) = 1$	$g_P(x) = 0$
intermediate vertex	$g_S(z) = 1$	$g_P(z) = 1$
end vertex	$g_S(y) = 0$	$g_P(y) = 1$

Elementary cycle : An elementary path whose start vertex and end vertex coincide is called an elementary cycle. In the subgraph of an elementary cycle, the indegree and the outdegree of every vertex are equal to 1. Note that the identical start and end vertex is counted once, not twice.

subgraph for an elementary cycle with vertex z	
vertex	$g_S(z) = g_P(z) = 1$

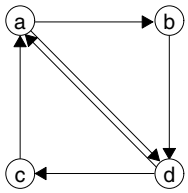
Hamiltonian paths and cycles : An elementary path with different start and end vertices is called a Hamiltonian path if it contains all vertices of the directed graph. An elementary cycle is called a Hamiltonian cycle if it contains all vertices of the directed graph.

Example 4 : Eulerian and Hamiltonian paths and cycles



x	a	b	c	d
$g_S(x)$	2	1	1	1
$g_P(x)$	1	1	1	2

Eulerian paths < a, d, c, a, b, d >, < a, b, d, c, a, d >
Hamiltonian paths < a, b, d, c >, < b, d, c, a >



x	a	b	c	d
$g_S(x)$	2	1	1	2
$g_P(x)$	2	1	1	2

Eulerian cycles < a, d, c, a, b, d, a >, < d, c, a, d, a, b, d >
Hamiltonian cycles < a, b, d, c, a >, < b, d, c, a, b >

8.4.3 CONNECTEDNESS OF DIRECTED GRAPHS

Introduction : In a directed graph $G = (V ; R)$, a vertex may or may not be reachable from another vertex along the directed edges. The concept of reachability forms the basis for a definition of the connectedness of vertices. Different kinds of connectedness may be defined, such as strong and weak connectedness. Directed graphs which are not strongly or weakly connected may be decomposed uniquely into strongly or weakly connected subgraphs. These subgraphs are called strongly or weakly connected components, respectively. Connectedness and decompositions of directed graphs are treated in the following.

Reachability : In a directed graph $G = (V ; R)$, a vertex $y \in V$ is said to be reachable from a vertex $x \in V$ if there is an empty or non-empty path from x to y. Vertex y is reachable from vertex x if and only if the product $x\,y^T$ of the associated point relations x and y is contained in the reflexive transitive closure R^*.

$$\text{y is reachable from x} \quad :\Leftrightarrow \quad x\,y^T \sqsubseteq R^* \qquad\qquad R^* = \mathrm{I} \sqcup R^+$$

Strong connectedness : Two vertices x and y of a directed graph are said to be strongly connected if x is reachable from y and y is reachable from x. A directed graph is said to be strongly connected if all vertices are pairwise strongly connected.

$$\begin{array}{lll} \text{x and y are strongly connected} & :\Leftrightarrow & x\,y^T \sqsubseteq R^* \sqcap R^{*T} \\ \text{the graph is strongly connected} & :\Leftrightarrow & R^* \sqcap R^{*T} = E \;\Leftrightarrow\; R^* = E \end{array}$$

Unilateral connectedness : Two vertices x and y of a directed graph are said to be unilaterally connected if x is reachable from y or y is reachable from x. A directed graph is said to be unilaterally connected if all vertices are pairwise unilaterally connected.

$$\begin{array}{lll} \text{x and y are unilaterally connected} & :\Leftrightarrow & x\,y^T \sqsubseteq R^* \sqcup R^{*T} \\ \text{the graph is unilaterally connected} & :\Leftrightarrow & R^* \sqcup R^{*T} = E \end{array}$$

Weak connectedness : Two vertices x and y of a directed graph $(V ; R)$ are said to be weakly connected if they are strongly connected in the symmetric graph $G = (V ; R \sqcup R^T)$. A directed graph is said to be weakly connected if all vertices are pairwise weakly connected. Since the transitive closure of a symmetric relation is symmetric, this definition may be expressed as follows :

$$\begin{array}{lll} \text{x and y are weakly connected} & :\Leftrightarrow & x\,y^T \sqsubseteq (R \sqcup R^T)^* \\ \text{the graph is weakly connected} & :\Leftrightarrow & (R \sqcup R^T)^* = E \end{array}$$

Connectedness relations : The relation R of a directed graph $G = (V ; R)$ generally contains strong, unilateral and weak connections. A relation which contains only connections of the same type is called a connectedness relation. The connectedness relations for a directed graph G are derived from the relation R and its reflexive transitive closure R^* :

strong connectedness relation	:	$S = R^* \sqcap R^{*T}$
unilateral connectedness relation	:	$P = R^* \sqcup R^{*T}$
weak connectedness relation	:	$C = (R \sqcup R^T)^*$

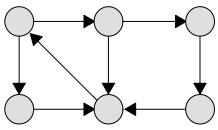
A strongly connected vertex pair is also unilaterally connected ; a unilaterally connected vertex pair is also weakly connected. Hence a strongly connected graph is also unilaterally connected, and a unilaterally connected graph is also weakly connected. For a symmetric graph, the three different kinds of connectedness coincide.

inclusion	:	$R^* \sqcap R^{*T}$	$\sqsubseteq$	$R^* \sqcup R^{*T}$	$\sqsubseteq$	$(R \sqcup R^T)^*$
connectedness	:	strong	$\Rightarrow$	unilateral	$\Rightarrow$	weak

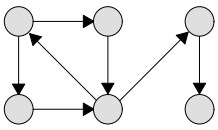
Two different vertices which are strongly connected lie on a cycle. A strongly connected graph is therefore cyclic. The converse is not true in the general case.

strongly connected graph $\Rightarrow$ cyclic graph

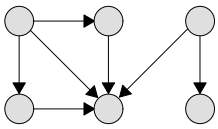
Example 1 : Connectedness of graphs



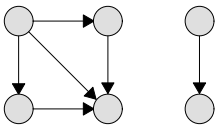
strongly connected



unilaterally connected



weakly connected



not connected

Properties of the connectedness relations : The strong connectedness relation S is reflexive, symmetric and transitive. Reflexivity and symmetry follow directly from the definition. Transitivity follows from the following consideration. If (x, y) and (y, z) are strongly connected vertex pairs, then z is reachable from x via y and x is reachable from z via y. Hence (x, z) is also a strongly connected vertex pair.

The unilateral connectedness relation P is reflexive and symmetric, but generally not transitive. This follows from the following consideration. If (x, y) and (y, z) are unilaterally connected vertex pairs, then it is possible that x is only reachable from y and z is only reachable from y. In this case, neither is x reachable from z, nor is z reachable from x. Thus (x, z) is not a unilaterally connected vertex pair.

The weak connectedness relation C is by definition the strong connectedness relation of an associated symmetric graph. This is reflexive, symmetric and transitive.

A reflexive, symmetric and transitive relation is an equivalence relation. Hence the strong and weak connectedness relations are equivalence relations. The unilateral connectedness relation is generally not an equivalence relation.

Decomposition into connected components : Let $G = (V; R)$ be a directed graph. Its strong connectedness relation $S = R^* \sqcap R^{*T}$ and its weak connectedness relation $C = (R \sqcup R^T)^*$ are equivalence relations. Let Z stand for either of these equivalence relations. The graph $(V; R)$ is connected if the equivalence relation Z is the all relation E. If the graph $(V; R)$ is disconnected, then it may be uniquely decomposed into connected subgraphs. The subgraphs are called the connected components of the graph. The decomposition is carried out in the following steps, independent of the kind of connectedness being considered :

(1) Connected class : The vertex set V of the graph is partitioned into connected classes using the relation Z. A connected class [x] with the vertex x as a representative contains all vertices of V which are connected with x. The class [x] is a unary relation and is determined as follows :

$$[x] = Zx$$

(2) Mapping : The set K of all connected classes is the quotient set V/Z. Each vertex $x \in V$ is mapped to exactly one connected class, yielding a canonical mapping Φ :

$$\Phi : V \to K \qquad \text{with} \qquad K = V/Z$$

(3) Reduced graph : The mapping Φ from the vertex set V of the directed graph $G = (V; R)$ to the set K of connected classes induces the reduced graph $G_K = (K; R_K)$.

$$G_K = (K; R_K) \qquad \text{with} \qquad R_K = \Phi^T R \Phi$$

(4) Connected component : A connected subgraph $G_k := (V_k, R_k)$ of a directed graph $G = (V; R)$. The vertex set V_k contains all vertices of a connected class k of the graph $(V; R)$. The edge set $R_k = R \sqcap (V_k \times V_k)$ contains the edges from R whose vertices belong to V_k. The union of all connected components G_k is generally a partial graph of G, since the union of all vertex sets V_k is the vertex set V and the union of all edge sets R_k is only a subset of the edge set R.

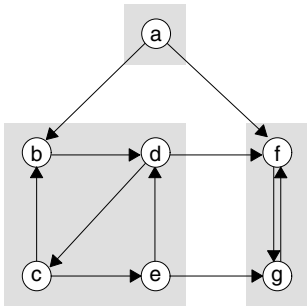
$$\bigsqcup_{k \in K} G_k \sqsubseteq G$$

Decomposition into strongly connected components : The vertex set V of a directed graph $G = (V; R)$ may be decomposed into strongly connected classes using its strong connectedness relation $Z = S = R^* \sqcap R^{*T}$. Two different classes cannot be strongly connected in the reduced graph $G_K = (K; R_K)$, since strongly connected vertices belong to the same class. Each connected component $G_k = (V_k; R_k)$ has a symmetric transitive closure R_k^+ and is therefore a cyclic graph. The reduced graph $G_K = (K; R_K)$ has an antisymmetric transitive closure R_K^+ and is therefore an anticyclic graph.

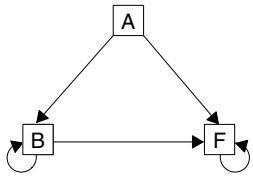
Decomposition into weakly connected components : The vertex set V of a directed graph $G = (V; R)$ may be decomposed into weakly connected classes using its weak connectedness relation $Z = C = (R \sqcup R^T)^*$. Two different classes cannot be weakly connected in the reduced graph $G_K = (K; R_K)$, since weakly connected vertices belong to the same class and the two vertices of an edge are at least weakly connected. Hence every directed graph is the union of its weakly connected components.

$$G = \bigsqcup_{k \in K} G_k$$

Example 2 : Decomposition into strongly connected components



directed graph



reduced graph

relation **R**

	a	b	c	d	e	f	g
a	0	1	0	0	0	1	0
b	0	0	0	1	0	0	0
c	0	1	0	0	1	0	0
d	0	0	1	0	0	1	0
e	0	0	0	1	0	0	1
f	0	0	0	0	0	0	1
g	0	0	0	0	0	1	0

closure $\mathbf{R}^*$

	a	b	c	d	e	f	g
a	1	1	1	1	1	1	1
b	0	1	1	1	1	1	1
c	0	1	1	1	1	1	1
d	0	1	1	1	1	1	1
e	0	1	1	1	1	1	1
f	0	0	0	0	0	1	1
g	0	0	0	0	0	1	1

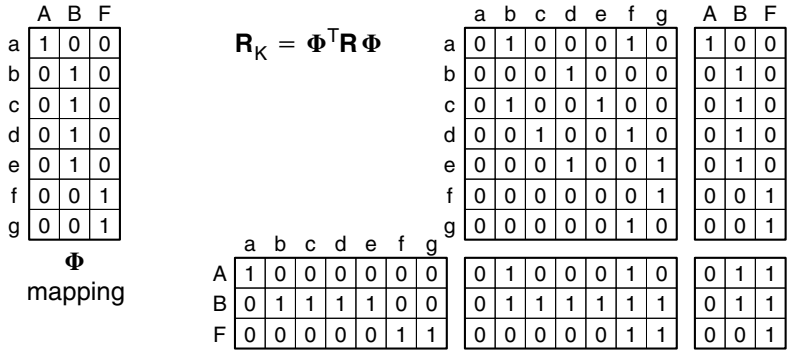
$\mathbf{S} = \mathbf{R}^* \sqcap \mathbf{R}^{*\top}$

	a	b	c	d	e	f	g
a	1	0	0	0	0	0	0
b	0	1	1	1	1	0	0
c	0	1	1	1	1	0	0
d	0	1	1	1	1	0	0
e	0	1	1	1	1	0	0
f	0	0	0	0	0	1	1
g	0	0	0	0	0	1	1

Let the directed graph $G = (V; R)$ shown in the diagram be given. The relation R, the reflexive transitive closure R^* and the strong connectedness relation S are shown as boolean matrices. The graph is not strongly connected, since the reflexive transitive closure R^* is not equal to the all relation E. It is decomposed into its strongly connected components.

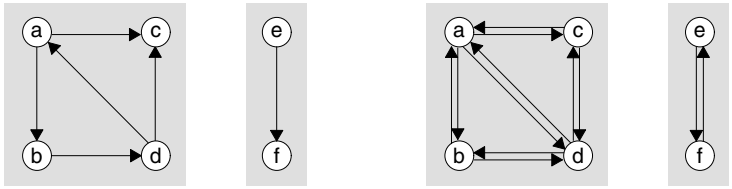
The strongly connected classes [a], [b] and [f] are determined using the connectedness relation S. The class [a] contains the vertex a, the class [b] contains the vertices b, c, d, e, and the class [f] contains the vertices f, g. Each vertex of the graph G is mapped to exactly one strongly connected class. The vertex set is thus partitioned into three strongly connected classes, which are designated by the uppercase letters A, B, F of their representatives a, b, f in order to simplify the diagram.

The boolean matrix for the mapping $\Phi: V \to \{A, B, F\}$ is formed columnwise from the boolean vectors for the unary relations [a], [b], [f]. The edge relation $R_K = \Phi^T R \Phi$ of the reduced graph is calculated as a product of boolean matrices.



The decomposition of the directed graph yields three strongly connected components, which are cyclic graphs. Like the strongly connected classes, they are designated by A, B, F. They form a reduced graph which is anticyclic. In the diagram of the directed graph at the beginning of the example, the strongly connected components are shaded. The reduced graph is shown alongside, with the vertices A, B, F and the directed edges corresponding to the relation R_K calculated above.

Example 3 : Decomposition into weakly connected components



directed graph with weakly connected components

symmetric graph with strongly connected components

The figure shows a directed graph with the vertex set $V = \{a,...,f\}$, as well as the associated symmetric graph. The weakly connected components of the directed graph and the strongly connected components of the symmetric graph are shaded.

The decomposition of a directed graph into its weakly connected classes is reduced to the decomposition of the symmetric graph into its strongly connected classes. The strongly connected components of the symmetric graph are not connected by edges, and hence neither are the weakly connected components of the directed graph. The directed graph is the union of its weakly connected components.

8.4.4 CUTS IN DIRECTED GRAPHS

Introduction : The reachability and connectedness of vertices in a directed graph are treated in the preceding section. In this section, the effects of removing edges or vertices on reachability and connectedness in the remaining graph are studied. For this purpose, the concept of cuts is introduced.

In a directed graph, edges may be cut or vertices may be excised. Cutting an edge means removing the edge from the graph; this leads to a partial graph. Excising a vertex means removing the vertex as well as the incident edges from the graph; this leads to a subgraph.

The vertices and edges of a directed graph are classified according to how their removal affects reachability and connectedness in the graph. This leads to concepts such as vertex cuts and edge cuts or vertex-disjoint and edge-disjoint paths. These are used to define further structural properties of graphs such as edge-disjoint connectedness and vertex-disjoint connectedness. These concepts and the corresponding structural properties of directed graphs are treated in the following.

Basic edge : An edge (x, y) in a directed graph $G = (V ; R)$ is called a basic edge (separating edge) if the vertex y is reachable from the vertex x only via this edge. If the basic edge is removed, then y is no longer reachable from x.

Chord : An edge (x, y) in a directed graph $G = (V ; R)$ is called a chord if the vertex y is also reachable from the vertex x via other edges. The chord (x, y) is the shortest path from x to y.

Basic graph : A partial graph $B = (V ; Q)$ of a directed graph $G = (V ; R)$ is called a basic graph for G if the following conditions are satisfied :

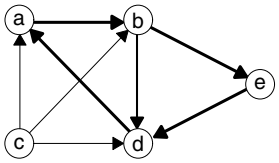
1. If a vertex y is reachable from a vertex x in the directed graph G, then y is also reachable from x in the partial graph B.
2. If an edge from x to y is removed from the partial graph B, then y is no longer reachable from x in the partial graph.

Construction of basic graphs : A basic graph $B = (V ; Q)$ for a directed graph $G = (V ; R)$ is generally not unique. A basic graph contains all basic edges of the graph. It may be iteratively constructed from a directed graph by removing a chord from the current graph in every step. The transitive closure R^+ of the directed graph is identical with the transitive closure Q^+ of a basic graph.

Example 1 : Basic edges and basic graph

The directed graph G = (V ; R) shown below has four basic edges and four chords. The basic edges are highlighted in the diagram. A basic graph B = (V ; Q) of G is shown. It contains the four basic edges and also a chord of the directed graph. The transitive closure R^+ of the directed graph is identical with the transitive closure Q^+ of the basic graph.

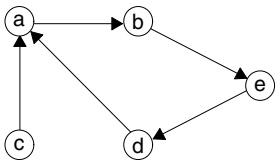
directed graph G = (V ; R)



transitive closure R^+

	a	b	c	d	e
a	1	1	0	1	1
b	1	1	0	1	1
c	1	1	0	1	1
d	1	1	0	1	1
e	1	1	0	1	1

basic graph B = (V ; Q)



transitive closure Q^+

	a	b	c	d	e
a	1	1	0	1	1
b	1	1	0	1	1
c	1	1	0	1	1
d	1	1	0	1	1
e	1	1	0	1	1

Edge cut : Let two different vertices x and y of a directed graph G = (V ; R) be given. An edge set T(x, y) $\sqsubseteq$ R is called an edge cut if removing its edges from the graph G has the effect that y is no longer reachable from x. An edge cut T(x, y) is called a minimal edge cut if no other edge cut S(x, y) contains fewer edges than T(x, y). The number of edges in a minimal edge cut is called the minimal edge cut size and is designated by min t(x, y). If a minimal edge cut contains exactly one edge, then this edge is a basic edge of the directed graph.

Both the set of all edges emanating from the vertex x and the set of all edges ending at the vertex y are edge cuts. Hence the minimal edge cut size is bounded from above by the outdegree of x and the indegree of y.

$$\min t(x, y) \quad \leq \quad \min \{g_S(x),\ g_P(y)\}$$

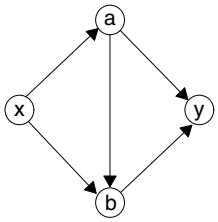
Edge-disjoint paths : Let x and y be two different vertices of a directed graph G = (V ; R). Two simple paths from x to y are said to be edge-disjoint if they have no edge in common. An edge-disjoint path set W(x, y) contains paths from x to y which are pairwise edge-disjoint. It is called a maximal edge-disjoint path set if no other edge-disjoint path set U(x, y) contains more paths than W(x, y). The number of paths in a maximal edge-disjoint path set is called the maximal number of edge-disjoint paths and is designated by max w(x, y).

Every edge emanating from the vertex x and every edge ending at the vertex y can occur in at most one edge-disjoint path from x to y. The maximal number of edge-disjoint paths is therefore bounded from above by the outdegree of x and the indegree of y.

$$\max w(x, y) \leq \min \{g_S(x), g_P(y)\}$$

Example 2 : Construction procedures

The following example illustrates two different procedures for constructing an edge-disjoint path set and a corresponding edge cut. Let the illustrated directed graph with the vertices x and y be given. It possesses a maximal edge-disjoint path set W(x,y) with two paths and three minimal edge cuts T(x,y) with two edges each.



maximal edge-disjoint path set :

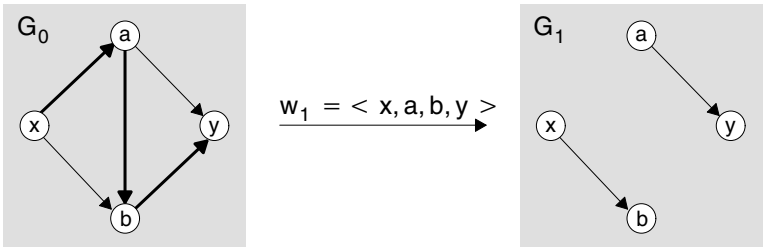
$W(x, y) = \{ < x, a, y >, < x, b, y > \}$

minimal edge cuts :

$T(x, y) : \{(x, a), (x, b)\}, \{(x, a), (b, y)\}, \{(a, y), (b, y)\}$

In the first procedure, edge-disjoint paths of the graph G are constructed in the following steps :

1. Look for a simple path from x to y in the graph $G_0 = G$. Let this path be $w_1 = < x, a, b, y >$. A partial graph G_1 is formed from the graph G_0 by removing all edges of the path w_1 from G_0. This prevents the edges of w_1 from being used again in a later step.

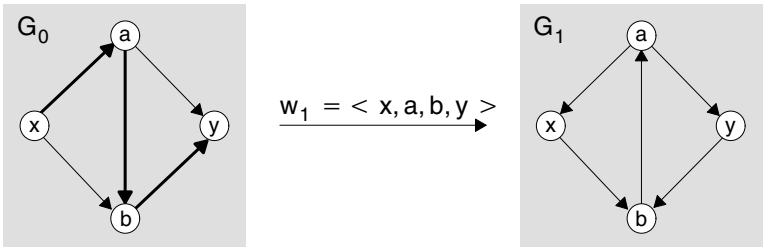


2. Look for a simple path from x to y in the graph G_1. Since there is no such path, the construction procedure terminates.

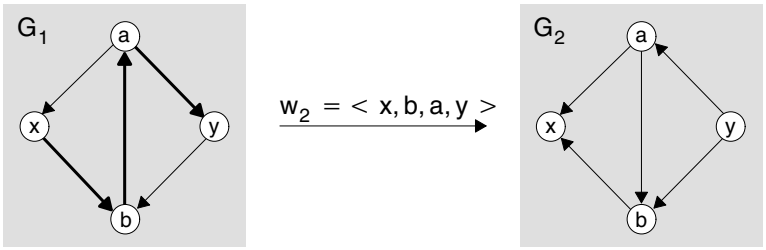
After this construction procedure, the edge-disjoint path set $W(x,y) = \{w_1\}$ consists only of the path $w_1 = < x, a, b, y >$. It is not maximal. A corresponding edge cut $T(x,y) = \{(x,a), (a,b), (b,y)\}$ consists of all edges of the path w_1. It is not minimal and contains a minimal edge cut $\{(x,a), (b,y)\}$ as a subset.

In the second procedure, edge-disjoint paths of the graph G are constructed in the following steps :

1. Look for a simple path from x to y in the graph $G_0 = G$. Let this path be $w_1 = < x, a, b, y >$. A modified graph G_1 is formed from the graph G_0 by reversing all edges of the path w_1 in G_0. This prevents the edges of w_1 from being used again with the same direction in a later step.



2. Look for a simple path from x to y in the graph G_1. Let this path be $w_2 = < x, b, a, y >$. A modified graph G_2 is formed from the graph G_1 by reversing all edges of the path w_2 in G_1. This prevents the edges of w_2 from being used again with the same direction in a later step.



Check whether the path w_2 contains a reversed edge of the path w_1 determined earlier. The path w_2 contains the edge (b,a), which is the reverse of the edge (a,b) in the path w_1. Two shorter paths $\overline{w}_1$ and $\overline{w}_2$ are constructed from the two paths w_1 and w_2. The path $\overline{w}_1 = < x, a, y >$ is the concatenation of the first subpath $< x, a >$ of w_1 and the last subpath $< a, y >$ of w_2. The path $\overline{w}_2 = < x, b, y >$ is the concatenation of the first subpath $<x, b>$ of w_2 and the last subpath $< b, y >$ of w_1. The path $\overline{w}_1$ does not contain the edge (a,b), and the path $\overline{w}_2$ does not contain the reversed edge (b,a). The graph G_2 contains the edge (a,b) in its original direction, so that this edge is available for the construction of further paths. The paths w_1 and w_2 are replaced by $\overline{w}_1$ and $\overline{w}_2$.

3. Look for a simple path from x to y in the graph G_2. Since there is no such path, the construction procedure terminates.

After this construction procedure, the edge-disjoint path set $W(x,y) = \{\overline{w}_1, \overline{w}_2\}$ consists of the paths $\overline{w}_1 = < x, a, y >$ and $\overline{w}_2 = < x, b, y >$. It is maximal. A corresponding edge cut T(x,y) consists of one edge from each edge-disjoint path, for example $T(x,y) = \{(x, a), (b, y)\}$. It is minimal.

Edge-disjoint paths and edge cuts : Let two different vertices x and y of a directed graph $G = (V ; R)$ be given. The maximal number of edge-disjoint paths leading from x to y is equal to the minimal edge cut size for x and y.

$$\max w(x,y) \;=\; \min t(x,y)$$

The proof of this theorem is contained in the following procedure for constructing a maximal edge-disjoint path set and a minimal edge cut.

Construction of an edge-disjoint path set and an edge cut : Let two different vertices x and y in a directed graph $G = (V ; R)$ be given. A set W(x,y) of edge-disjoint paths from x to y is constructed iteratively by modifying the original graph. At the beginning the path set W(x,y) is empty, and the graph G_0 is equal to the original graph G. In each step $k = 1, 2,...$ a path w_k of the path set W(x,y) and a modified graph G_k are determined in the following steps :

1. Look for a simple path w_k from x to y in the graph G_{k-1}. If there is no such path, the path set W(x,y) is complete, and the construction terminates.
2. Form the modified graph G_k by reversing the directions of all edges of the path w_k in the graph G_{k-1}.
3. Check whether the path w_k contains an edge with reversed direction. If the path w_k contains an edge (b,a) with reversed direction, then a path w_i with $i < k$ which was determined earlier contains the edge (a,b) with the original direction. In this case, the paths w_i and w_k are replaced by the two shorter paths $\overline{w}_i$ and $\overline{w}_k$.

   $w_i = <$ x,...,r,a,b,s,...,y> $\qquad$ $w_k = <$ x,...,p,b,a,q,...,y>

   $\overline{w}_i = <$ x,...,r,a,q,...,y> $\qquad$ $\overline{w}_k = <$ x,...,p,b,s,...,y>

   The new path $\overline{w}_i$ does not contain the edge (a,b), and the new path $\overline{w}_k$ does not contain the reversed edge (b,a). Thus the original edge (a,b) is available again for the construction of further paths. Due to the double reversal in steps i and k, this edge is contained in the modified graph G_k. Step 3 is repeated until the path w_k does not contain any edges with reversed direction.

The graph modified by the construction of the edge-disjoint path set W(x,y) is designated by G^I. The directions of all edges of the edge-disjoint paths in the modified graph G^I are reversed with respect to their directions in the original graph G. The modified graph G^I is used as follows to construct an edge cut T(x,y) associated with W(x,y) :

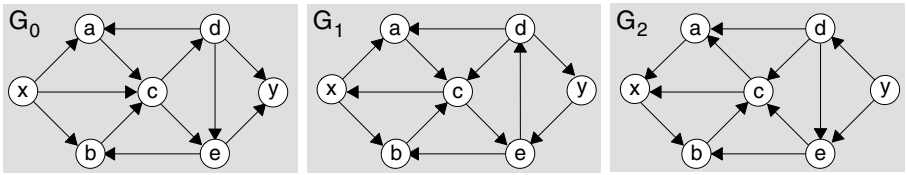
1. In the modified graph G^I, all vertices reachable from the vertex x are determined and collected in the vertex set X. The vertex x belongs to the vertex set X, since it is reachable from itself.
2. All vertices which do not belong to the vertex set X are collected in the vertex set Y. The vertex y belongs to the vertex set Y, since there is no path from x to y in the graph G^I.
3. The edge cut T(x,y) contains all edges (a,b) of the original graph G which lead from a vertex $a \in X$ to a vertex $b \in Y$.

The following relationships hold between the edge-disjoint path set W(x,y) and the corresponding edge cut T(x,y) :

1. The edge cut T(x,y) contains only edges which occur in edge-disjoint paths. If there were an edge (a,b) with $a \in X$ and $b \in Y$ which occured in none of the edge-disjoint paths, then its direction would not be reversed in the modified graph G^I. Hence b would have to be reachable from a in G^I, and would therefore belong to the vertex set X. This contradicts the definition of X.
2. A path $w_k = <x,...,a,b,...,y> \in W(x,y)$ consists of a subpath $w_x = <x,...,a>$ with vertices from X and a subpath $w_y = <b,...,y>$ with vertices from Y. The subpath w_x can only contain vertices from X, since in G^I every vertex of w_x is reachable from a. If a vertex c from w_y would belong to X, b would also have to belong to X, since b is reachable from c in G^I. This contradicts the hypothesis $b \in Y$. Thus from each path $w_k \in W(x,y)$ the edge cut T(x,y) contains exactly one edge (a,b) with $a \in X$ and $b \in Y$.
3. Since the edge cut T(x,y) contains exactly one edge from every path in W(x,y) and all edges in T(x,y) occur in paths from W(x,y), the size t(x,y) of the edge cut T(x,y) is equal to the number w(x,y) of edge-disjoint paths in W(x,y), that is $t(x,y) = w(x,y)$.
4. Each path of an edge-disjoint path set must contain at least one edge of an edge cut. Hence the edge cut size t(x,y) is an upper bound for the maximal number of edge-disjoint paths. Conversely, the number w(x,y) of edge-disjoint paths is a lower bound for the minimal edge cut size. Since $t(x,y) = w(x,y)$, it follows that t(x,y) is minimal and w(x,y) is maximal.

Example 3 : Edge-disjoint paths and edge cuts

original graph G | modified graph | modified graph G^{I}



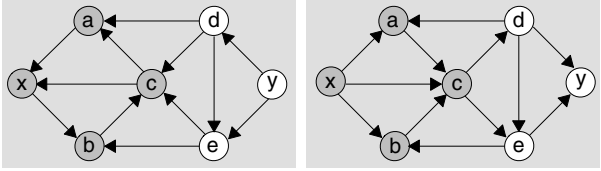
$w_1 = < x, c, d, e, y >$ $w_2 = < x, a, c, e, d, y >$ no further path

$\overline{w}_1 = < x, c, d, y >$ $\overline{w}_2 = < x, a, c, e, y >$ $W(x, y) = \{\overline{w}_1, \overline{w}_2\}$

The set W(x,y) of edge-disjoint paths from x to y in the graph G shown above is constructed iteratively. In the first step, the simple path w_1 is chosen in the original graph $G_0 = G$, and the modified graph G_1 is formed by reversing all edges of w_1 in G_0. In the second step, the simple path w_2 is chosen in the modified graph G_1, and the modified graph G_2 is formed by reversing all edges of w_2 in G_1. Since the edge (d,e) occurs in w_1 and the reversed edge (e,d) occurs in w_2, the paths w_1 and w_2 are replaced by the paths $\overline{w}_1$ and $\overline{w}_2$. In the modified graph G_2, there is no further path from x to y, and hence $W(x,y) = \{\overline{w}_1, \overline{w}_2\}$ is a maximal edge-disjoint path set.

In the modified graph $G^{I} = G_2$, the edges of the paths $\overline{w}_1$ and $\overline{w}_2$ have been reversed. The vertex set X = {x,a,b,c} contains the vertices reachable from x in G^{I}, the vertex set Y = {d,e,y} contains the remaining vertices of G^{I}. The minimal edge cut T(x,y) = {(c,d), (c,e)} associated with the maximal edge-disjoint path set W(x,y) contains the edges of the original graph G which lead from a vertex in X to a vertex in Y. The set T(x,y) contains one edge from each of the paths $\overline{w}_1$ and $\overline{w}_2$.

modified graph G^{I} | original graph G



$X = \{x, a, b, c\}$ $T(x, y) = \{(c, d), (c, e)\}$

$Y = \{d, e, y\}$

Vertex cut : Let two different vertices x and y of a directed graph G = (V ; R) be given. A vertex set $S(x,y) \subseteq V - \{x, y\}$ is called a vertex cut if removing its vertices and the incident edges from the graph G has the effect that y is no longer reachable from x. If there is an edge from x to y, then there is no vertex cut. A vertex cut S(x, y) is called a minimal vertex cut if no other vertex cut Q(x, y) contains fewer vertices

than S(x,y). The number of vertices of a minimal vertex cut is called the minimal vertex cut size and is designated by min s(x,y). If a minimal vertex cut contains exactly one vertex, then this vertex is called a separating vertex of the directed graph.

If there is no edge from x to y, then both the set of all successors of x and the set of all predecessors of y are vertex cuts. The minimal vertex cut size is therefore bounded from above by the outdegree of x and the indegree of y.

$$\min s(x,y) \leq \min \{g_S(x),\ g_P(y)\} \qquad (x,y) \notin R$$

Vertex-disjoint paths : Let two different vertices x and y of a directed graph G = (V ; R) be given. Two elementary paths from x to y are said to be vertex-disjoint if they have no vertex in common other than x and y . A vertex-disjoint path set U(x,y) contains paths from x to y which are pairwise vertex-disjoint. A vertex-disjoint path set U(x,y) is called a maximal vertex-disjoint path set if no other vertex-disjoint path set P(x,y) contains more paths than U(x,y). The number of paths in a maximal vertex-disjoint path set is called the maximal number of vertex-disjoint paths and is designated by max u(x,y).

Every successor of x and every predecessor of y can occur in at most one vertex-disjoint path from x to y. The maximal number of vertex-disjoint paths is therefore bounded from above by the outdegree of x and the indegree of y.

$$\max u(x,y) \leq \min \{g_S(x),\ g_P(y)\}$$

Vertex-disjoint paths and vertex cuts : Let two different vertices x and y of a directed graph G = (V ; R) be given which are not connected by an edge from x to y. The maximal number of vertex-disjoint paths leading from x to y is equal to the minimal vertex cut size for x and y.

$$\max u(x,y) = \min s(x,y)$$

The proof of this theorem is contained in the following procedure for constructing a maximal vertex-disjoint path set and a minimal vertex cut.

Construction of a vertex-disjoint path set and a vertex cut : Let two different vertices x and y in a directed graph G = (V ; R) be given which are not connected by an edge from x to y. The determination of a maximal set U(x,y) of vertex-disjoint paths from x to y and a minimal vertex cut S(x,y) in the graph G is reduced to the determination of a maximal set of edge-disjoint paths and a minimal edge cut in a substitute graph G_E. The substitute graph G_E is constructed from the graph G as follows :

– Every vertex a of G is replaced by two vertices a^I and a^{II} and an edge (a^I, a^{II}).
– Every edge (a,b) of G is replaced by an edge (a^{II}, b^I).

In the substitute graph G_E, a maximal set $W(x^{II},y^{I})$ of edge-disjoint paths from x^{II} to y^{I} and a minimal edge cut $T(x^{II},y^{I})$ are determined. The following relationships hold between the maximal edge-disjoint path set $W(x^{II},y^{I})$ in G_E and the maximal vertex-disjoint path set U(x,y) in G, and between the minimal edge cut $T(x^{II},y^{I})$ in G_E and the minimal vertex cut S(x,y) in G :

(1) There is a one-to-one correspondence of paths $w = < x^{II},...,z^{I}, z^{II},...,y^{I} >$ from x^{II} to y^{I} in G_E with paths $u = < x,...,z,...,y >$ from x to y in G. The vertices z^{I}, z^{II} and the edge (z^{I}, z^{II}) in w correspond to the intermediate vertex z in u. Two paths from x to y in G are vertex-disjoint if and only if the corresponding paths from x^{II} to y^{I} in G_E are edge-disjoint. Thus there is also a one-to-one correspondence between maximal edge-disjoint path sets $W(x^{II},y^{I})$ in G_E and maximal vertex-disjoint path sets U(x,y) in G. Hence the maximal number max $w(x^{II},y^{I})$ of edge-disjoint paths in G_E is equal to the maximal number max u(x,y) of vertex-disjoint paths in G.

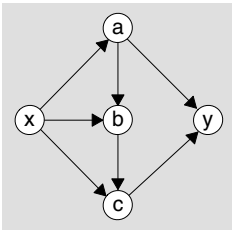
(2) For every maximal edge-disjoint path set $W(x^{II},y^{I})$ in G_E there is a minimal edge cut $T(x^{II},y^{I})$ which contains exactly one edge from every edge-disjoint path. If $T(x^{II},y^{I})$ contains an edge (a^{II},z^{I}) with $z^{I} \neq y^{I}$ or an edge (z^{II},a^{I}) with $z^{II} \neq x^{II}$, then by virtue of the construction of G this edge may be replaced by the edge (z^{I},z^{II}). Then $T(x^{II},y^{I})$ contains only edges of type (z^{I},z^{II}). An edge (z^{I},z^{II}) in G_E corresponds to the vertex z. Thus there is a one-to-one correspondence between minimal edge cuts $T(x^{II},y^{I})$ with edges of type (z^{I},z^{II}) in G_E and minimal vertex cuts S(x,y) with intermediate vertices $z \neq x,y$ in G. Hence the minimal edge cut size min $t(x^{II},y^{I})$ in G_E is equal to the minimal vertex cut size min s(x,y) in G.

(3) Since in the substitute graph G_E the maximal number max $w(x^{II},y^{I})$ of edge-disjoint paths is equal to the minimal edge cut size min $t(x^{II},y^{I})$, it follows by (1) and (2) that in the graph G the maximal number max u(x,y) of vertex-disjoint paths is equal to the minimal vertex cut size min s(x, y).

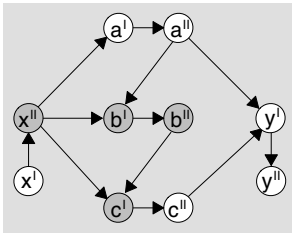
Example 4 : Vertex-disjoint paths and vertex cuts

In the directed graph G shown below, a vertex-disjoint path set and a vertex cut are constructed as follows using the substitute graph G_E and the modified substitute graph G_E^I :

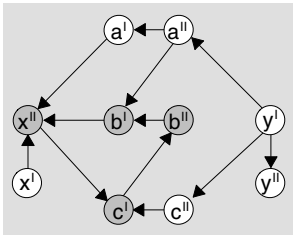
graph G



substitute graph G_E



modified substitute graph G_E^I



$u_1 = < x, a, y >$ $w_1 = < x^{II}, a^I, a^{II}, y^I >$ $X^{II} = \{x^{II}, b^I, b^{II}, c^I\}$

$u_2 = < x, b, c, y >$ $w_2 = < x^{II}, b^I, b^{II}, c^I, c^{II}, y^I >$ $Y^I = \{x^I, a^I, a^{II}, c^{II}, y^I, y^{II}\}$

$U(x, y) = \{u_1, u_2\}$ $W(x^{II}, y^I) = \{w_1, w_2\}$

$S(x, y) = \{a, c\}$ $T(x^{II}, y^I) = \{(x^{II}, a^I), (c^I, c^{II})\} \rightarrow \{(a^I, a^{II}), (c^I, c^{II})\}$

The substitute graph G_E for the graph G is constructed. In the substitute graph G_E, the edge-disjoint paths w_1 and w_2 from x^{II} to y^I are determined. The substitute graph G_E is transformed into the modified substitute graph G_E^I by reversing the direction of every edge in w_1 and w_2. In the modified substitute graph G_E^I, all vertices reachable from x^{II} are collected in the vertex set X^{II}, and all remaining vertices are collected in the vertex set Y^I. The minimal edge cut $T(x^{II}, y^I)$ contains all edges of the substitute graph G_E which lead from a vertex in X^{II} to a vertex in Y^I. The edge (x^{II}, a^I) is replaced by the edge (a^I, a^{II}). The edges (a^I, a^{II}) and (c^I, c^{II}) in the substitute graph G_E correspond to the vertices a and c in the graph G. The edge-disjoint paths in the substitute graph G_E correspond to the vertex-disjoint paths in the graph G.

Multiple edge-disjoint reachability : A vertex y in a directed graph $G = (V ; R)$ is said to be n-fold edge-disjointly reachable from a vertex x if x and y are identical or the maximal number of edge-disjoint paths from x to y is not less than n. The multiple edge-disjoint reachability of vertices forms the basis for the definition of multiple edge-disjoint connectedness.

Multiple edge-disjoint connectedness : Two vertices x and y are said to be n-fold edge-disjointly connected if x is at least n-fold edge-disjointly reachable from y and vice versa. A directed graph is said to be n-fold edge-disjointly connected (n-edge connected) if all vertices are pairwise n-fold edge-disjointly connected. The strong connectedness of a directed graph corresponds to simple (1-fold) edge-disjoint connectedness.

The maximal multiplicity max m of the edge-disjoint connectedness of a directed graph is equal to the minimum of the maximal number of edge-disjoint paths for all vertex pairs (x, y) with $x \neq y$. The upper bound for the maximal number of edge-disjoint paths yields an upper bound for the maximal multiplicity.

$$\max m \;=\; \min_{x \neq y} \{\max w(x,y)\} \;\leq\; \min_{x} \{g_S(x), g_P(x)\}$$

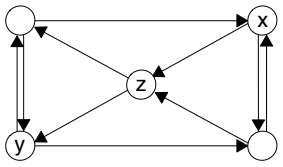
Multiple vertex-disjoint reachability : A vertex y in a directed graph G = (V ; R) is said to be n-fold vertex-disjointly reachable from a vertex x if x and y are identical, if there is an edge from x to y or if the maximal number of vertex-disjoint paths from x to y is not less than n. The multiple vertex-disjoint reachability of a vertex forms the basis for the definition of multiple vertex-disjoint connectedness.

Multiple vertex-disjoint connectedness : Two vertices x and y are said to be n-fold vertex-disjointly connected if x is at least n-fold vertex-disjointly reachable from y and vice versa. A directed graph is said to be n-fold vertex-disjointly connected (n-vertex connected, n-connected) if all vertices are pairwise n-fold vertex-disjointly connected. The strong connectedness of a directed graph corresponds to simple (1-fold) vertex-disjoint connectedness.

The maximal multiplicity max n of the vertex-disjoint connectedness of a directed graph which is not complete is equal to the minimum of the maximal number of vertex-disjoint paths for all vertex pairs (x, y) with $x \neq y$ which are not connected by an edge from x to y.

$$\max n \;=\; \min_{x \neq y} \{\, \max u(x,y) \;\mid\; (x,y) \notin R \,\}$$

Example 5 : Multiple edge- and vertex-disjoint connectedness



directed graph

The directed graph shown above is strongly connected, and hence simply edge-disjointly connected. It is also doubly (2-fold) edge-disjointly connected, since from each vertex each other vertex is reachable via exactly two edge-disjoint paths. The graph has no higher edge-disjoint connectedness, since every vertex has indegree 2 and outdegree 2 and the degree of connectedness is bounded from above by the minimal indegree and outdegree of the vertices.

The directed graph is strongly connected, and therefore simply vertex-disjointly connected. It is not doubly vertex-disjointly connected, since for instance there is only one vertex-disjoint path from the vertex x to the vertex y with the intermediate vertex z as a separating vertex.

8.4.5 PATHS AND CYCLES IN SIMPLE GRAPHS

Introduction : A simple graph $G = (V ; \Gamma)$ consists of a vertex set V and an adjacency relation Γ for the neighborhood of vertices. The adjacency of two vertices is represented by an undirected edge which corresponds to a pair of edges with opposite directions. The graph is free of loops. The adjacency relation Γ is symmetric and antireflexive.

A simple graph is treated as a symmetric and antireflexive special case of a directed graph. The fundamentals of paths and cycles for directed graphs can largely be transferred to simple graphs. The fundamentals of the structural analysis of simple graphs are treated in the following.

Neighbors : Two vertices x and y are called neighbors if there is an undirected edge between x and y in the simple graph, so that the vertex pairs (x, y) and (y, x) are contained in the adjacency relation Γ.

$$\text{x and y are neighbors} \quad \Leftrightarrow \quad (x, y) \in \Gamma \quad \Leftrightarrow \quad (y, x) \in \Gamma$$

If the vertices x and y are represented as unary point relations in V, which are also designated by x and y, then their neighborhood is determined as follows using the algebra of relations :

$$\text{x and y are neighbors} \quad \Leftrightarrow \quad x\, y^{\mathsf{T}} \sqsubseteq \Gamma \quad \Leftrightarrow \quad y\, x^{\mathsf{T}} \sqsubseteq \Gamma$$

The set $t(x)$ of all neighbors of a vertex x is calculated as a unary relation in the vertex set V as follows :

$$\text{neighbors of x} \quad : \quad t(x) = \Gamma x$$

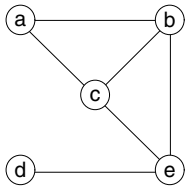
Degree : The number of neighbors of a vertex x is called the degree of the vertex and is designated by g (x). The degree g (x) corresponds to the number of elements in the set t(x) and hence to the number of undirected edges at the vertex x.

$$\text{degree of a vertex} \quad : \quad g(x) = \left| t(x) \right| = \left| \Gamma x \right|$$

If a simple graph contains k undirected edges, then the sum of the degrees of all vertices $x \in V$ is 2k, and hence equal to the number of elements in Γ.

$$\text{sum of degrees} \quad : \quad \sum_{x \in V} g(x) = \sum_{x \in V} \left| \Gamma x \right| = \left| \Gamma \right| = 2k$$

Example 1 : Neighbors and degrees



	a	b	c	d	e
a	0	1	1	0	0
b	1	0	1	0	1
c	1	1	0	0	1
d	0	0	0	0	1
e	0	1	1	1	0

Γ

a	0
b	0
c	0
d	0
e	1

e

a	0
b	1
c	1
d	1
e	0

t(e)

x	a	b	c	d	e
g(x)	2	3	3	1	3

$\sum g(x) = |\Gamma| = 12$

The simple graph shown above consists of 5 vertices and 6 undirected edges. The symmetric adjacency relation is specified by a boolean matrix $\mathbf{\Gamma}$. The unary point relation for the vertex e is shown as a boolean unit vector **e**. The product $\mathbf{\Gamma e}$ yields the boolean vector **t**(e) for all neighbors of e. It coincides with the column of $\mathbf{\Gamma}$ which is associated with the vertex e. The degrees of all vertices are compiled. The sum of the degrees of all vertices is equal to twice the number of undirected edges.

Edge sequence : A chain of edges is called an edge sequence if the end vertex of each edge except for the last edge is the start vertex of the following edge.

edge sequence $< (x_0, x_1), (x_1, x_2) ,..., (x_{n-1}, x_n) >$

condition $\bigwedge_{j=1}^{n} ((x_{j-1}, x_j) \in \Gamma)$

The start vertex x_0 of the first edge and the end vertex x_n of the last edge are called the start vertex and end vertex of the edge sequence, respectively. The vertices x_1 to x_{n-1} are called intermediate vertices of the edge sequence. The number n of edges is called the length of the edge sequence. If there is an edge sequence from x_0 to x_n, then by virtue of the symmetry of simple graphs there is also an edge sequence in the reverse direction from x_n to x_0.

Descendants : A vertex y is called an n-th descendant of a vertex x if there is an edge sequence of length n from x to y in the graph. If y is an n-th descendant of x, then x is also an n-th descendant of y, since for an edge sequence from x to y there is also a reverse edge sequence from y to x. The descendants are calculated as for directed graphs using the n-th power Γ^n and the transitive closure Γ^+ of the adjacency relation Γ.

n-th descendants of x : $t^{(n)}(x) = \Gamma^n x$

all descendants of x : $t^+(x) = \Gamma^+ x$

Path : A path from a start vertex x via intermediate vertices to an end vertex y is an edge sequence. In a simple graph, it can be uniquely represented as a vertex sequence $< x,...,y >$. A path $< x >$ with the same start and end vertex x contains no edges and is called an empty path. The length of an empty path is 0. There is an empty path for every vertex of a simple graph. The existence of non-empty paths in a simple graph is established as follows :

$$\begin{array}{lcl} \text{there is a path of length n from } x \text{ to } y & \Leftrightarrow & xy^{\mathsf{T}} \sqsubseteq \Gamma^{n} \\ \text{there is a non-empty path from } x \text{ to } y & \Leftrightarrow & xy^{\mathsf{T}} \sqsubseteq \Gamma^{+} \end{array}$$

Cycle : A non-empty path whose start and end vertex coincide is called a cycle. Due to the symmetry of the adjacency relation Γ, a simple graph contains a large number of trivial cycles. If a path is first traversed in one direction and then retraced in the other direction, a trivial cycle is obtained. However, trivial cycles are not significant for the structure of simple graphs. Since the conditions for the existence of cycles in directed graphs also hold for trivial cycles when applied to simple graphs, they cannot be used to study simple graphs.

Simple path : A non-empty path is said to be simple if it does not contain any undirected edge more than once. The vertices and the undirected edges of a simple path form a subgraph of the simple graph. If the start vertex and end vertex of a simple path are different, then the degrees of the vertices in the subgraph have the following properties :

subgraph for a simple path $< x,...,z,...,y >$ with $x \neq y$

start vertex	:	$g(x)$ odd
intermediate vertex	:	$g(z)$ even
end vertex	:	$g(y)$ odd

Simple cycle : A simple path whose start vertex and end vertex coincide is called a simple cycle. In the subgraph for a simple cycle, the degree of every vertex is even.

subgraph for a simple cycle with vertex z

vertex	:	$g(z)$ even

Eulerian paths and cycles : A simple path with different start and end vertices is called an Eulerian path if it contains all undirected edges of the simple graph. A simple cycle is called an Eulerian cycle if it contains all undirected edges of the simple graph.

Elementary path : A non-empty path is said to be elementary if it does not contain any vertex more than once. The vertices and the undirected edges of an elementary path form a subgraph of the simple graph. If the start vertex and end vertex of a simple path are different, then the degrees of the vertices in the subgraph are :

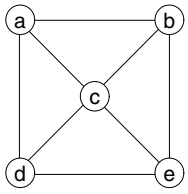
subgraph for a simple path $< x,..., z,...,y >$ with $x \neq y$
start vertex : $g(x) = 1$
intermediate vertex : $g(z) = 2$
end vertex : $g(y) = 1$

Elementary cycle : An elementary path whose start vertex and end vertex coincide is called an elementary cycle. In the subgraph for an elementary cycle, the degree of every vertex is 2. Note that the identical start and end vertex of the cycle is counted once, not twice.

subgraph for an elementary cycle with vertex z
vertex z : $g(z) = 2$

Hamiltonian paths and cycles : An elementary path with different start and end vertices is called a Hamiltonian path if it contains all vertices of the simple graph. An elementary cycle is called a Hamiltonian cycle if it contains all vertices of the simple graph.

Example 2 : Paths and cycles



x	a	b	c	d	e
g(x)	3	3	4	3	3

simple paths	< a, c, e, d, c, b >	< a, b, e, c, b >
elementary paths	< a, b, c >	< b, c, d >
simple cycles	< a, b, c, d, e, c, a >	< e, b, c, d, a, c, e >
elementary cycles	< a, b, c, a >	< a, b, e, c, a >

The simple graph shown above does not contain any Eulerian cycles, since the degrees of vertices a,b,d,e are odd and hence the necessary condition for a simple cycle containing all edges of the graph is not satisfied. However, the graph contains several Hamiltonian cycles. For example, the cycle $< a, b, c, e, d, a >$ is a Hamiltonian cycle.

8.4.6 CONNECTEDNESS OF SIMPLE GRAPHS

Introduction : The connectedness properties of directed graphs may be transferred directly to simple graphs. The symmetry property of simple graphs leads to essential simplifications. The different forms of connectedness of directed graphs coincide for simple graphs and are all referred to as simple connectedness. The fundamentals for the connectedness and the decomposition of simple graphs are treated in the following.

Connectability : In a simple graph $G = (V ; \Gamma)$ two vertices $x, y \in V$ are said to be connectable if there is an empty or non-empty path between x and y. The vertices x and y are connectable if and only if the product $x y^T$ of the associated point relations is contained in the reflexive transitive closure Γ^*.

x and y are connectable $:\Leftrightarrow \quad x\, y^T \sqsubseteq \Gamma^* \qquad \Gamma^* = \mathrm{I} \sqcup \Gamma^+$

Simple connectedness : Two vertices x and y in a simple graph are simply connected if x and y are connectable. A simple graph is simply connected if all vertices in V are pairwise simply connected. A distinction between strong, unilateral and weak connectedness is not possible for simple graphs, since the adjacency relation Γ is symmetric, which implies $\Gamma^* \sqcap \Gamma^{*T} = \Gamma^* \sqcup \Gamma^{*T} = (\Gamma \sqcup \Gamma^T)^* = \Gamma^*$.

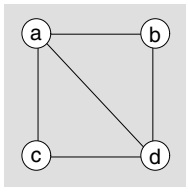
x and y are simply connected $:\Leftrightarrow \quad x y^T \sqsubseteq \Gamma^*$
the graph is simply connected $:\Leftrightarrow \quad \Gamma^* = \mathrm{E}$

Connectedness relation : Like the strong and the weak connectedness relation for directed graphs, the simple connectedness relation $Z = \Gamma^*$ for simple graphs is an equivalence relation. It forms the basis for a decomposition of simple graphs into their simply connected components.

Decomposition into simply connected components : A simple graph $G = (V ; \Gamma)$ may be decomposed into simply connected components using the simple connectedness relation $Z = \Gamma^*$. The vertex set V is mapped to the quotient set $K = V/Z$. The vertex set V_k of a connected component $G_k := (V_k ; \Gamma_k)$ contains all vertices of a connected class $k \in K$. The edge set $\Gamma_k := \Gamma \sqcap (V_k \times V_k)$ contains the edges from Γ whose vertices belong to V_k. There are no edges between the elements of the reduced graph. The simple graph is the union of its simply connected components G_k.

$$G = \bigsqcup_{k \in K} G_k$$

Example : Simple connectedness of a graph



G_1

G_2

	a	b	c	d	e	f
a	0	1	1	1	0	0
b	1	0	0	1	0	0
c	1	0	0	1	0	0
d	1	1	1	0	0	0
e	0	0	0	0	0	1
f	0	0	0	0	1	0

Γ

	a	b	c	d	e	f
a	1	1	1	1	0	0
b	1	1	1	1	0	0
c	1	1	1	1	0	0
d	1	1	1	1	0	0
e	0	0	0	0	1	1
f	0	0	0	0	1	1

closure Γ^*

The reflexive transitive closure of the graph shown above is represented as a boolean matrix Γ^*, from which the simply connected classes may be read off directly. The class [a] corresponds to the column for a in the matrix Γ^*. It contains the vertices a, b, c, d and is the vertex set for the simply connected component G_1. The class [e] corresponds to the column e in the matrix Γ^*. It contains the vertices e, f and is the vertex set for the simply connected component G_2.

8.4.7 CUTS IN SIMPLE GRAPHS

Introduction : The connectability and connectedness of vertices in a simple graph are treated in the preceding section. In this section, the effects of removing edges or vertices on the connectability and connectedness in the remaining graph are studied. For this purpose, the concept of cuts is introduced as in the case of directed graphs.

Edges are classified into bridges and cycle edges according to how their removal affects the connectedness of the graph. If a bridge is cut, the connectedness of the graph is partially lost; if a cycle edge is cut, connectedness is preserved. Acyclic graphs contain only bridges, cyclic graphs contain only cycle edges. Simple cyclic connectedness is defined for simple graphs. Graphs which are not simply cyclically connected may be uniquely decomposed into simply cyclically connected components.

If the connectedness of a graph is partially lost by the excision of a vertex, this vertex is called an articulation vertex. A graph without articulation vertices is elementarily cyclically connected. A graph with articulation vertices may be uniquely decomposed into elementarily cyclically connected blocks.

The definitions of simple and elementary cyclic connectedness allow a deeper analysis of the connectedness properties of simple graphs. They are special cases of multiple edge- and vertex-disjoint connectedness, which is described for directed graphs in Section 8.4.4. The concepts and fundamentals for cyclic connectedness of graphs are treated in the following.

Bridge : An undirected edge between two vertices x and y in a simple graph $G = (V ; \Gamma)$ is called a bridge if x and y are connectable only via this edge. If a bridge from x to y is removed from the simple graph, then x and y are no longer connectable. If a bridge is removed from a simply connected graph, the graph is divided into two simply connected components.

Cycle edge : An undirected edge between two vertices x and y in a simple graph $G = (V ; \Gamma)$ is called a cycle edge if it is contained in a simple cycle. If a cycle edge between x and y is removed from the simple graph, x and y remain connectable. If a cycle edge is removed from a simply connected graph, simple connectedness is preserved.

Decomposition of the adjacency relation : Every undirected edge of a simple graph $G = (V ; \Gamma)$ is either a bridge or a cycle edge. The adjacency relation Γ may therefore be uniquely decomposed into a part Γ_B for the bridges and a part Γ_Z for the cycle edges.

adjacency relation for G	:	$\Gamma = \Gamma_B \sqcup \Gamma_Z \quad \text{with} \quad \Gamma_B \sqcap \Gamma_Z = \emptyset$
adjacency relation for bridges	:	Γ_B
adjacency relation for cycle edges	:	Γ_Z

Simple acyclic and cyclic graphs : A simple acyclic graph contains only bridges and no cycle edges. A simple cyclic graph contains only cycle edges and no bridges.

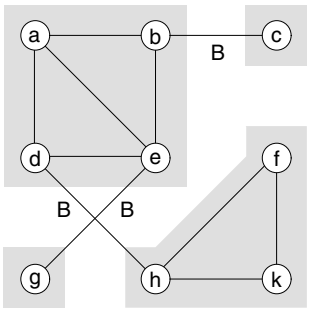
simple acyclic graph	$:\Leftrightarrow$	$\Gamma_Z = \emptyset$
simple cyclic graph	$:\Leftrightarrow$	$\Gamma_B = \emptyset$

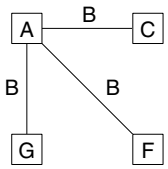
Simple cyclic connectedness : Two vertices x and y are said to be simply cyclically connected if x and y are identical or there is a simple cycle in which they both occur. A simple graph is said to be simply cyclically connected if all vertices are pairwise simply cyclically connected.

x and y are simply cyclically connected	$:\Leftrightarrow$	$x y^T \sqsubseteq \Gamma_Z^*$
the graph is simply cyclically connected	$:\Leftrightarrow$	$\Gamma_Z^* = E$

Decomposition into simply cyclically connected components : The simple cyclic connectedness relation $Z = \Gamma_Z^*$ is an equivalence relation. A simple graph $G = (V ; \Gamma)$ may therefore be uniquely decomposed into simply cyclically connected components. The decomposition is carried out as in the case of directed graphs. Every simply cyclically connected component is a simple cyclic subgraph. The reduced simple graph is a simple acyclic graph if the loops at the vertices are disregarded.

Example 1 : Decomposition into simply cyclically connected components





reduced graph without loops
B bridges

bridge part Γ_B

	a	b	c	d	e	f	g	h	k
a	0	0	0	0	0	0	0	0	0
b	0	0	1	0	0	0	0	0	0
c	0	1	0	0	0	0	0	0	0
d	0	0	0	0	0	0	0	1	0
e	0	0	0	0	0	0	1	0	0
f	0	0	0	0	0	0	0	0	0
g	0	0	0	0	1	0	0	0	0
h	0	0	0	1	0	0	0	0	0
k	0	0	0	0	0	0	0	0	0

cycle part Γ_Z

	a	b	c	d	e	f	g	h	k
a	0	1	0	1	1	0	0	0	0
b	1	0	0	0	1	0	0	0	0
c	0	0	0	0	0	0	0	0	0
d	1	0	0	0	1	0	0	0	0
e	1	1	0	1	0	0	0	0	0
f	0	0	0	0	0	0	0	1	1
g	0	0	0	0	0	0	0	0	0
h	0	0	0	0	0	1	0	0	1
k	0	0	0	0	0	1	0	1	0

closure Γ_Z^*

	a	b	c	d	e	f	g	h	k
a	1	1	0	1	1	0	0	0	0
b	1	1	0	1	1	0	0	0	0
c	0	0	1	0	0	0	0	0	0
d	1	1	0	1	1	0	0	0	0
e	1	1	0	1	1	0	0	0	0
f	0	0	0	0	0	1	0	1	1
g	0	0	0	0	0	0	1	0	0
h	0	0	0	0	0	1	0	1	1
k	0	0	0	0	0	1	0	1	1

The simple graph shown above is simply connected, but not simply cyclically connected. The adjacency relations for the bridge part Γ_B and for the cycle part Γ_Z as well as the reflexive transitive closure Γ_Z^* for the cycle part are shown as boolean matrices. The undirected edges (b,c), (d,h), (e,g) are bridges. All remaining edges are cycle edges. The simply cyclically connected classes can be read off directly from the boolean matrix for the reflexive transitive closure Γ_Z^*. The simple graph possesses the simply cyclically connected classes A, C, G, F with the vertex sets {a,b,d,e}, {c}, {g}, {f,h,k}. These classes form the vertex set of the reduced graph, which is a simple acyclic graph except for the loops. The bridges of the simple graph induce edges between the connected classes of the reduced graph.

Articulation vertex : A vertex a of a simple graph $G = (V ; \Gamma)$ is called an articulation vertex if two different vertices $x \neq a$ and $y \neq a$ are connectable only via a. If the articulation vertex a is removed from the simple graph together with its edges, then x and y are no longer connectable. If an articulation vertex is excised from a simply connected graph, the graph is divided into several simply connected components.

Elementary cyclic connectedness : Two vertices x and y of a simple graph are said to be elementarily cyclically connected if x and y are identical, they are neighbors or there is an elementary cycle in which they both occur. A simple graph is said to be elementarily cyclically connected if all vertices are pairwise elementarily cyclically connected. An elementarily cyclically connected graph does not contain any articulation vertices.

Block : A subgraph of a simple graph $G = (V ; \Gamma)$ is called a block if it is elementarily cyclically connected. A block is said to be proper if it is not contained in another block as a subgraph. If a simple graph contains an elementary cycle, then all vertices and all undirected edges of this cycle belong to a proper block.

Relationships between blocks : A simple graph $G = (V ; \Gamma)$ may possess several proper blocks. Two different proper blocks have the following properties :

(1) Two different proper blocks have either one vertex or no vertices in common.

(2) If two different proper blocks have a vertex in common, this vertex is an articulation vertex of the simple graph.

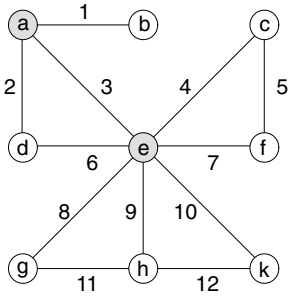
(3) Two different proper blocks are not connected by edges.

Block decomposition : A simple graph $G = (V ; \Gamma)$ may be uniquely decomposed into proper blocks $B_e = (V_e ; \Gamma_e)$. Since two different proper blocks have at most one vertex in common and are not connected by edges, every edge of the simple graph is associated with a unique proper block. The edge sets Γ_e of all blocks are therefore disjoint subsets of the edge set Γ of the simple graph. The vertex sets V_e of the blocks are generally not disjoint subsets of the vertex set V of the simple graph, since articulation vertices are contained in different vertex sets V_e.

$$G = \bigsqcup_e B_e$$

The block structure of a simple graph is represented in a block graph. The vertices of the block graph correspond to the proper blocks. The edges of the block graph indicate that the two proper blocks share a common vertex, which is an articulation vertex of the simple graph. A block graph may also be represented as a hypergraph in which every hyperedge corresponds to an articulation vertex.

Example 2 : Articulation vertices and block decomposition



proper blocks

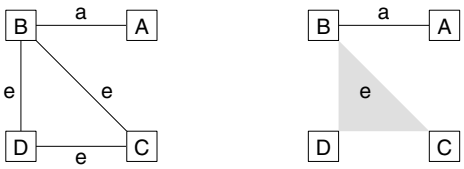
A = ({a,b} ; {1})

B = ({a,d,e} ; {2,3,6})

C = ({c,e,f} ; {4,5,7})

D = ({e,g,h,k} ; {8,9,10,11,12})

In the simple graph shown above, lowercase letters identify vertices and numbers identify edges. The simple graph is simply connected, but not elementarily cyclically connected. For example, the vertices a and k do not lie on an elementary cycle. The graph has the articulation vertices a and e. For example, the vertices b and d are connectable only via a, the vertices c and g only via e. The graph possesses four proper blocks A, B, C and D, which are elementarily cyclically connected. The vertices and edges of the blocks are specified above. Every undirected edge is contained in exactly one block. The corresponding block graph with the blocks as vertices and the articulation vertices as edges is shown as a simple graph and as a hypergraph.



simple graph hypergraph

Multiple edge- and vertex-disjoint connectedness : The fundamentals for multiple edge- and vertex-disjoint connectedness of directed graphs are described in Section 8.4.4. They may be directly transferred to simple graphs, taking into account the symmetry of these graphs. Undirected edges take the place of directed edges, and connectability takes the place of reachability of vertices. The forms of connectedness of simple graphs treated here are special cases of multiple edge- or vertex-disjoint connectedness which are particularly important in applications to practical problems. Simple connectedness corresponds to simple edge-disjoint connectedness and simple vertex-disjoint connectedness. Simple cyclic connectedness corresponds to two-fold edge-disjoint connectedness, and elementary cyclic connectedness corresponds to two-fold vertex-disjoint connectedness.

8.4.8 ACYCLIC GRAPHS

Introduction : The acyclicity of a graph leads to special structural properties of the graph. In studying these properties, a distinction is made between directed acyclic graphs with directed edges and simple acyclic graphs with undirected edges.

Directed acyclic graphs possess an order structure. The vertex set is an ordered set. The directed edges describe the order relation in the vertex set. Due to the order structure, the vertices can be sorted. Edges can be removed from the graph in such a manner that the order structure is preserved. The minimal structure-preserving edge set is unique. The vertex set and the minimal edge set form the basic graph.

Simple acyclic graphs do not have an order structure, since their edges are undirected. They form undirected trees or forests.

Directed acyclic graph : A directed acyclic graph $G = (V ; R)$ is asymmetric and does not contain cycles. Every path from a vertex x to a vertex y is elementary. The closure R^+ is asymmetric and transitive. Hence it is a strict order relation. The theoretical foundations of strict order relations may therefore be applied to directed acyclic graphs.

Rank : Every vertex x of a directed acyclic graph $G = (V ; R)$ is assigned a rank r(x), which is a natural number with the following properties :

(1) A vertex x has the rank $r(x) = 0$ if it does not have any ancestors.

(2) A vertex x has the rank $r(x) = k > 0$ if it has a k-th ancestor and no $(k + 1)$-th ancestors.

It is only possible to assign ranks if the directed graph G is acyclic. If there is a cycle through the vertex x, then for every k-th ancestor of x in the cycle there is a predecessor in the cycle, and hence also a $(k + 1)$-th ancestor of x. The directed graph must therefore be free of cycles.

If the rank r(x) of a vertex x is k, then by definition the vertex x has a k-th ancestor but no $(k + 1)$-th ancestor. Thus there must be a path of length k but no path of length $k + 1$ from a vertex without predecessor in G to x. Hence the rank r(x) is the length k of a longest path from a vertex without predecessor in G to x.

Topological sorting : The determination of the ranks of the vertices of a directed graph $G = (V ; R)$ is called topological sorting. The vertex set $V = V_0$ is topologically sorted by iteratively reducing it to the empty vertex set $\emptyset$. In step k, the vertex set V_k is determined whose vertices $x \in V_k$ have a k-th ancestor in G and are therefore of rank $r(x) \geq k$. The vertex set V_k contains all predecessors of the vertices in the vertex set V_{k-1}. This iterative reduction is formulated as follows using unary relations :

initial values	:	v_0	= e	all relation
reduction	:	v_k	= $R^T\ v_{k-1}$	$k = 1,...,n$
termination	:	v_n	= $\emptyset$	null relation

A vertex x of the vertex set V_k is of degree $r(x) = k$ if it does not belong to the vertex set V_{k+1}. The set W_k of all vertices of rank k is therefore the difference $V_k - V_{k+1}$, which is calculated as the intersection of V_k and the complement of V_{k+1}. It is called the k-th vertex class and is determined as a unary relation as follows :

k-th vertex class :	w_k	=	$v_k \sqcap \overline{v}_{k+1}$	$k = 0,...,n-1$

Order structure : Topologically sorting a directed acyclic graph $G = (V ; R)$ yields a partition of the vertex set into disjoint vertex classes W_k with $k = 0,...,n-1$. The partition has the following ordinal properties :

– The vertex class W_0 contains all vertices of the lowest rank 0. These vertices have no ancestors in G, and hence no predecessors. They are therefore minimal. Since there are no other vertices without predecessors, W_0 contains all minimal vertices.

– The vertex class W_{n-1} contains all vertices of the highest rank $n-1$. These vertices have no descendants in G, and hence no successors. They are therefore maximal. Since there may generally also be other vertices without successor, W_{n-1} generally does not contain all maximal vertices.

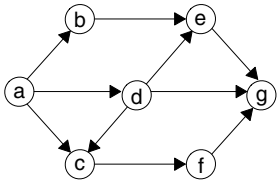
– Every vertex x in the vertex class W_k with $k > 0$ has at least one predecessor y in the vertex class W_{k-1}. If $x \in W_k$ did not have a predecessor $y \in W_{k-1}$, then x would not have any k-th ancestors, and would therefore not belong to W_k.

– A vertex has neither a predecessor nor a successor in its own vertex class. If y were a predecessor of x and hence x a successor of y, then the rank of y would have to be less than the rank of x and x, y could not belong to the same vertex class.

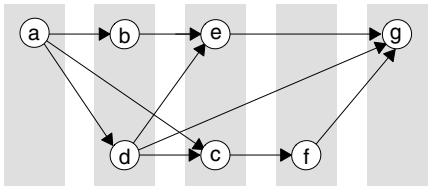
Example 1 : Topological sorting

Let a directed acyclic graph be given. The calculation steps for sorting this graph topologically are shown. The sorting leads to the formation of classes in the vertex set of the graph. The sorted graph and its classes are represented graphically.

acyclic graph



vertex classes of the graph



topological sorting $\mathbf{v}_k = \mathbf{R}^T \mathbf{v}_{k-1}$

	a	b	c	d	e	f	g
a	0	0	0	0	0	0	0
b	1	0	0	0	0	0	0
c	1	0	0	1	0	0	0
d	1	0	0	0	0	0	0
e	0	1	0	1	0	0	0
f	0	0	1	0	0	0	0
g	0	0	0	1	1	1	0

$\mathbf{R}^T$

$\mathbf{v}_0$	$\mathbf{v}_1$	$\mathbf{v}_2$	$\mathbf{v}_3$	$\mathbf{v}_4$	$\mathbf{v}_5$
1	0	0	0	0	0
1	1	0	0	0	0
1	1	1	0	0	0
1	1	0	0	0	0
1	1	1	0	0	0
1	1	1	1	0	0
1	1	1	1	1	0

vertex classes

$\mathbf{w}_k = \mathbf{v}_k \sqcap \overline{\mathbf{v}}_{k+1}$

$\mathbf{w}_0$	$\mathbf{w}_1$	$\mathbf{w}_2$	$\mathbf{w}_3$	$\mathbf{w}_4$
1	0	0	0	0
0	1	0	0	0
0	0	1	0	0
0	1	0	0	0
0	0	1	0	0
0	0	0	1	0
0	0	0	0	1

Basic edges and chords : A directed acyclic graph $G = (V ; R)$ has basic edges and chords. An edge from x to y is called a basic edge if y is reachable from x only via this edge. Otherwise it is called a chord. Since a directed acyclic graph does not contain cycles, an edge from x to y is a chord if and only if there is a path of length $n > 1$ from x to y.

path from x to y with $n > 1$ $\quad\Leftrightarrow\quad x y^T \sqsubseteq \bigsqcup_{n>1} R^n = R \bigsqcup_{n>0} R^n = RR^+$

chord (x,y) $\quad\Leftrightarrow\quad x y^T \sqsubseteq R \sqcap RR^+$

basic edge (x,y) $\quad\Leftrightarrow\quad x y^T \sqsubseteq R \sqcap \overline{RR^+}$

Basic path : A directed acyclic graph $G = (V ; R)$ does not contain cycles. If there are one or more paths from x to y, then there is at least one path of maximal length. A path of maximal length is called a basic path. A basic path contains only basic edges.

Proof : A basic path contains only basic edges.

Consider a path from x to y of maximal length m which contains an edge from a to b. If the edge from a to b were a chord, there would have to be a path from a to b of length greater than 1, and hence also a path from x to y of length greater than m. But this contradicts the hypothesis. It follows that all edges of a path from x to y of maximal length are basic edges.

Basic graph : The graph $B = (V ; Q)$ is a basic graph of a directed acyclic graph $G = (V ; R)$ if Q contains only the basic edges in R. The basic graph B is constructed by removing all chords from R. The basic graph B is unique. The transitive closures R^+ and Q^+ coincide.

$$\text{basic graph } B = (V ; Q) \qquad \text{with} \qquad Q = R \sqcap \overline{R R^+}$$

Proof : The transitive closures R^+ and Q^+ coincide.

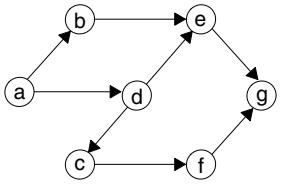
For every chord $(x, y) \in R$ there is by definition a path from x to y of length $n > 1$. Thus there is also a path of maximal length from x to y which is a basic path and consists only of basic edges. Hence y is still reachable from x if the chord (x, y) is removed from R, so that the chord (x, y) yields no additional contribution to the closure R^+. Hence the closures R^+ and Q^+ coincide.

Order diagram : In the topological sorting of a directed acyclic graph $G = (V ; R)$, the rank r(x) of a vertex $x \in V$ is equal to the length of a longest path from a vertex without predecessor to x. This path is a basic path consisting only of basic edges. Hence removing chords from R does not change the rank r(x) of a vertex x, so that topologically sorting the graph $G = (V ; R)$ and its basic graph $B = (V ; Q)$ leads to the same result. The representation of the order structure of the basic graph with its vertex classes is an order diagram according to Section 4.2.

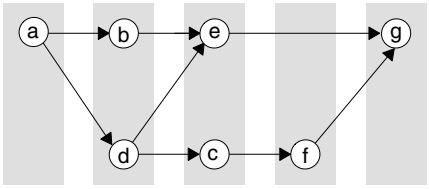
Example 2 : Basic graph and order diagram

Let the directed acyclic graph G = (V ; R) from Example 1 be given. The edges (a, c) and (d, g) are chords, since there are basic paths < a, d, c > and < d, e, g >. The basic graph B = (V ; Q) is constructed from the graph G by removing these chords from G. The edge set Q of the basic graph is calculated using the formula specified above. The basic graph B and the order diagram are represented graphically. The directed acyclic graph G and the basic graph B possess the same vertex classes.

basic graph



vertex classes of the basic graph



Simple acyclic graph : A simple acyclic graph G = (V ; Γ) does not contain any simple cycles. All undirected edges of the graph G are bridges. Removing an edge destroys the original connectedness of the graph G.

Tree : A simple acyclic graph which is simply connected is called a tree. A tree with n vertices has exactly n − 1 undirected edges.

tree	:	n − k = 1
number of vertices	:	n
number of edges	:	k

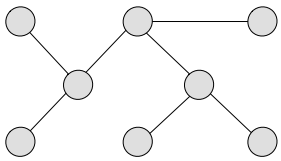
A tree is constructed as follows : A simple graph with only one vertex and no undirected edges is simply connected, does not contain simple cycles and is therefore a tree. Simple connectedness and absence of simple cycles are preserved if the tree is iteratively extended by adding a new vertex with a new undirected edge to an existing vertex in each step. For n vertices, this construction leads to n − 1 edges.

In a tree, the path between two different vertices x and y is unique. If there were several different paths between x and y, there would be cycles, but this is ruled out by the definition of a tree.

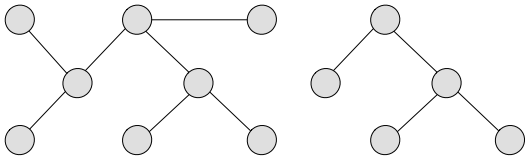
Forest : A simple acyclic graph with several simply connected components is called a forest. Every simply connected component is a tree. By the definition of trees, a forest with n vertices and k undirected edges contains exactly n − k trees.

forest	:	n − k = c
number of vertices	:	n
number of edges	:	k
number of components	:	c

Example 3 : Trees and forests



tree : n = 8 k = 7 c = 1



forest : n = 13 k = 11 c = 2

8.4.9 ROOTED GRAPHS AND ROOTED TREES

Introduction : A vertex of a graph from which all remaining vertices are reachable is called a root of the graph. Rooted graphs and rooted trees are of fundamental importance in computer science. For example, finite automata, syntax diagrams and flow diagrams are treated as rooted graphs. All hierarchical structures are regarded as rooted trees. Searching for all vertices of a graph which are reachable from a given vertex leads to a search tree which corresponds to a rooted tree and forms a skeleton of the graph. The fundamentals for rooted graphs, rooted trees and search trees are treated in the following.

Root : A vertex w is called a root (root vertex) of a directed graph $G = (V ; R)$ if all vertices of the graph are reachable from the vertex w. If a directed graph is not weakly connected, then it has no root. If it is strongly connected, then every vertex of the graph is a root.

$$w \text{ is a root} \quad :\Leftrightarrow \quad w\, e^{T} \sqsubseteq R^{*}$$

Rooted graph : A directed graph $G = (V ; R)$ is called a rooted graph if it contains at least one root. In a rooted graph, there is a special form of connectedness between pairs of vertices, called quasi-strong connectedness. Two vertices x and y are quasi-strongly connected if there is a vertex z from which the vertices x and y are both reachable. In this case, there is a path from x to z in the dual graph G^{T} and a path from z to y in the graph G, so that $(x,z) \in R^{*T}$ and $(z,y) \in R^{*}$, and hence $(x,y) \in R^{*T}R^{*}$. In a rooted graph, all vertices are pairwise quasi-strongly connected via a root, so that $R^{*T}R = E$ holds.

$$x \text{ and } y \text{ are quasi-strongly connected} \quad :\Leftrightarrow \quad x\, y^{T} \sqsubseteq R^{*T}R^{*}$$
$$G = (V ; R) \text{ is a rooted graph} \quad :\Leftrightarrow \quad R^{*T}R^{*} = E$$

Acyclic rooted graph : A directed graph $G = (V ; R)$ is acyclic if $R^{+} \sqcap R^{+T} = \emptyset$ holds. It is a rooted graph if $R^{*T}R^{*} = E$ holds. An acyclic rooted graph has exactly one root. The existence of several roots would contradict the absence of cycles.

$$G = (V ; R) \text{ is an acyclic rooted graph} \;\Leftrightarrow\; R^{+} \sqcap R^{+T} = \emptyset \;\wedge\; R^{*T}R^{*} = E$$

Rooted tree : An acyclic rooted graph $G = (V ; R)$ is called a rooted tree if R is left-unique, so that $RR^{T} \sqsubseteq I$ holds.

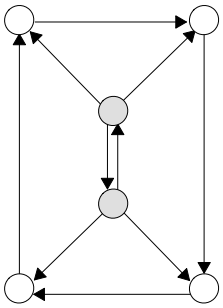
$$G = (V ; R) \text{ is a rooted tree} \;:\Leftrightarrow\; RR^{T} \sqsubseteq I \;\wedge\; R^{+} \sqcap R^{+T} = \emptyset \;\wedge\; R^{*T}R^{*} = E$$

A rooted tree with the root w has the following properties :

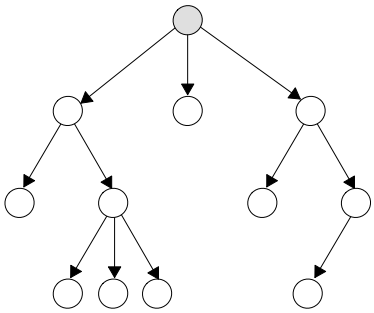
- The root w has no predecessor.
- Every vertex $x \neq w$ has exactly one predecessor.
- Every vertex $x \neq w$ is reachable along exactly one path from w to x.
- A rooted tree with n vertices has exactly n – 1 edges.

Forest of rooted trees : A directed graph is called a forest of rooted trees if every weakly connected component is a rooted tree.

Example 1 : Rooted graphs and rooted trees



rooted graph
with 2 roots



rooted tree
with 12 vertices and 11 edges

Search tree : Let a vertex a in a directed graph G be given. A rooted tree with root a which contains all descendants of a in G is called a search tree at the vertex a. A search tree is constructed by an iterative search, starting from the vertex a. Breadth-first search and depth-first search are distinguished.

Breadth-first search : In a breadth-first search, a vertex sequence F is maintained, which at first contains only the root a. As long as the vertex sequence F is not empty, the following steps are carried out in a loop :

– If the vertex at the beginning of F has a successor which has not been visited yet, such a successor is appended to the end of the sequence F.
– If the vertex at the beginning of F has no successor which has not been visited yet, it is removed from the sequence F.

The vertices visited and the edges used in the course of the breadth-first search form the breadth-first search tree. For every visited vertex x, the search tree contains a path of minimal length from a to x. This property is of fundamental importance for determining paths of minimal length between the vertices of a directed graph.

Depth-first search : In a depth-first search, a vertex sequence F is maintained, which at first contains only the root a. As long as the vertex sequence F is not empty, the following steps are carried out in a loop :

– If the vertex at the end of F has a successor which has not been visited yet, such a successor is appended to the end of the sequence F.
– If the vertex at the end of F has no successor which has not been visited yet, it is removed from the sequence F.

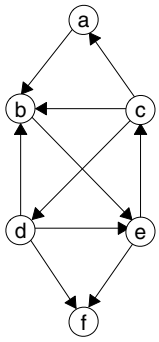
The vertices visited and the edges used in the course of the depth-first search form the depth-first search tree.

Properties : Breadth-first search and depth-first search lead to different search trees. The depth of a search tree is the length of a longest path from the root a to a visited vertex without successor. The breadth of a search tree is the maximal number of visited vertices without successor. Among all search trees, a breadth-first search tree has maximal breadth and minimal depth. A depth-first search tree generally has small breadth and great depth.

Example 2 : Breadth-first search and depth-first search

Let the directed graph shown below be given. The descendants of the vertex c are to be determined by breadth-first search and by depth-first search. The iterative construction of the vertex sequence F for the breadth-first search and the depth-first search is shown.

directed graph



breadth-first search sequence F

c					
c	a				
c	a	b			
c	a	b	d		
	a	b	d		
		b	d		
		b	d	e	
			d	e	
			d	e	f
				e	f
					f

depth-first search sequence F

c					
c	a				
c	a	b			
c	a	b	e		
c	a	b	e	f	
c	a	b	e		
c	a	b			
c	a				
c					
c	d				
c					

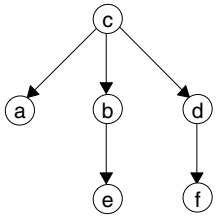
The vertex c is the root of the breadth-first search tree and the depth-first search tree. The breadth-first search tree is constructed according to the following rule, starting from the root c :

– If a new vertex y is appended to the end of the sequence F with start vertex x, the new vertex y and the edge from x to y are added to the breadth-first search tree.

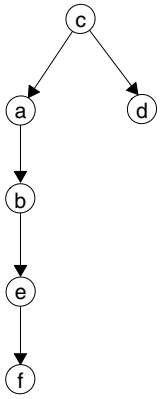
The depth-first search tree is constructed according to the following rule, starting from the root c :

– If a new vertex y is appended to the end of the sequence F with end vertex x, the new vertex y and the edge from x to y are added to the depth-first search tree.

breadth-first search tree



depth-first search tree



http://www.springer.com/978-3-540-67995-0

Mathematical Foundations of Computational Engineering
A Handbook
Pahl, P.J.; Damrath, R.
2001, XXXVI, 1007 p. In 2 volumes, not available separately., Hardcover
ISBN: 978-3-540-67995-0