10. Geometry and Topology

Gerhard Gröger, Betsy George

The representation of geometrical properties of spatial objects as well as of their structural aspects (topology) is crucial for GIS operations, analyses and visualizations. This chapter introduces the most important geometrical and topological concepts, considering the two dimensional as well as the three dimensional case. Particularly, the concepts of the standard ISO 19107 Spatial schema are introduced.

10.1 Geometry

The main purpose of geometrical characterizations in GIS is to represent the shape and metric properties of spatial objects (features), in order to compute distances between objects, to derive areas of surface objects or volumes of solid objects, to perform spatial analyses (Sect. 10.2) like viewshed calculation, spatial planning and simulation, e.g. for noise emission, or to serve as base for geovisualization.

Regarding the dimension of geometry, we have to distinguish between the *dimension of the geometry object* and the *dimension of the embedding space*. In this section, the embedding space in general is 3-D; embedding in 2-D planes are considered only in Sect. 10.1.2. This section is structured according to the dimension of the geometry objects: we start with 0-D to 2-D, consider 2.5-D as special case, and finally discuss solid 3-D objects.

An overview of geometrical 3-D models can be found in [10.1–3]. This section presents models which are relevant for 3-D-GIS, mainly boundary representations (Sect. 10.1.3), and gives a rough survey of other concepts. The focus is on the geometry model provided by the standard ISO 19107 *Spatial schema* [10.4], which is implemented particularly in the representation and exchange language *Geography Markup Language (GML 3)* [10.5–7].

The 2-D coordinates $(x, y) \in \mathbb{R}^2$ or 3-D coordinates $(x, y, z) \in \mathbb{R}^3$ of the geometry objects are represented according to any of the coordinate reference systems introduced in Sect. 10.2.2.

10.1.1 0-D, 1-D, and 2-D Geometries

A *point* as a 0-D geometry is simply represented by a 3-D coordinate (x, y, z). One-dimensional geometries are *curves* or *line segments* which have start coordinates and end coordinates. The shape of a curve between the start and the end point is specified by an *interpolation method*. The list of interpolation methods provided by the ISO 19107 *Spatial schema* or by *GML*, for example, is *linear*, *geodesic*, *circular*, *elliptical*, *clothoid*, *conic*, *polynomialSpline*, *cubicSpline*, or *rationalSpline*. If the

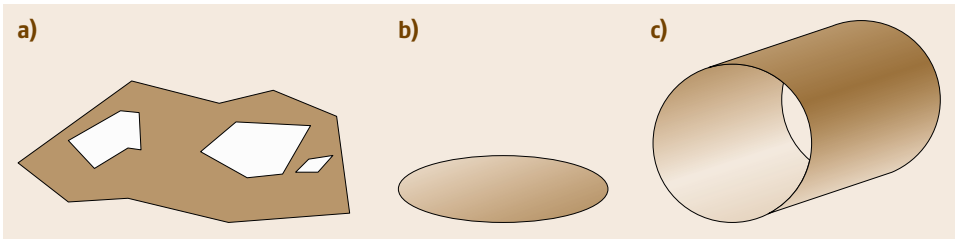

Fig. 10.1a–c Three surfaces. **(a)** Planar polygon delimited by for rings. **(b)** 2-D disk delimited by one ring. **(c)** Cylinder surface delimited by two rings

interpolation is linear, start and end points are connected by a straight line. The other interpolation methods require some more parameter values. A circular line segment is represented by three control points, and an elliptical by four control points, for example. More details on the interpolation methods are provided in [10.3, 4].

In general, curves or line segments are connected, non-branching (i. e., have at most two start/end points) and are non-self-intersecting. If the positions of the start and the end points are identical, the curve is *closed*.

2-D geometries embedded in 3-D space, which are typically called *regions*, *polygons* or *surfaces*, are continuous, connected 2-D point sets which are delimited by curves. These curves have to be closed and form so-called *rings* [10.4]. A *ring* is a closed sequence of curves, where a curve starts where the predecessor in the curve ends. The curves in a ring are non-intersecting. A region is bounded by one exterior ring and by optional interior rings, which define enclaves or holes in the region. Figure 10.1a depicts as an example a region with four rings, one exterior and three interior. Rings may be composed of curves of any interpolation method mentioned above in this section.

The shape of the surface, i. e., of the 2-D point set delimited by its rings, is defined by interpolation methods, similar to the case of line segments. ISO 19107 *Spatial schema* and GML, for example, provide the following interpolation methods: *planar*, *spherical*, *elliptical*, *conic*, *tin*, *parametricCurve*, *polynomialSpline*, *rationalSpline*, *triangulatedSpline*. In planar surfaces, all points of the surface are in the same plane. This interpolation method is common in GIS and 3-D city models; surfaces provided by commercial tools like ArcGIS or Oracle Spatial are planar. For details of the representations of other interpolation methods, the reader is referred to [10.3, 4].

Surfaces are purely areal two-dimensional point sets, without penetrations, T-shaped or X-shaped touches. Mathematically, this property is captured by the notion of a *2-manifold*. A *2-manifold* is a 2-D point set where each point has a neighborhood in the set which is topologically equivalent to an open two-dimensional disk. Intuitively, for each point on a 2-manifold, a small circular neighborhood centered at that point can be deformed to a disk.

Another important property of surfaces is the number of boundaries. A disk (Fig. 10.1b) has one boundary, whereas the number of boundaries of a cylinder surface (Fig. 10.1c) is two. A sphere has no boundaries at all. Such surfaces are called *closed*; they enclose a volume completely and hence are used to define solids (Sect. 10.1.3).

A further relevant characteristic of surfaces embedded in 3-D space and an essential precondition for defining solid objects is *orientability* [10.2]. A surface is *orientable*, if two opposite sides of the surface can be distinguished. For the general case a more formal definition of orientability is given in [10.8]. Well-known examples for non-orientable surfaces are the Möbius strip and the Klein bottle; both are depicted in Fig. 10.2.

An orientable surface can be given an orientation, by labeling exactly one of the two sides as *top* or +. When surfaces are used to define solids, the surface orientation is chosen such that the top side points outward relative to the solid's interior. If one of the rings delimiting a surface can be distinguished as an outer ring, as is the case for planar polygons, the *right-hand rule* can be applied to define an orientation. If the curve segments in the exterior ring are oriented consistently, from start to end point, then the side of the surface where the direction of the ring appears counterclockwise is the top side of the surface (Fig. 10.3).

Surfaces as well as line segments can be aggregated to larger units, which again have the same properties as the surfaces. The standard ISO 19107 defines *surface patches* on the lowest level, which can be aggregated to surfaces (class *GM_Surface*). Surfaces similarly can be aggregated to *composite surfaces*, which recursively can again be part of a larger composite surface. Rules for building composite surfaces from parts are discussed in Sect. 10.2.2, where topological data models are reviewed. In fact, a composite surface is both a topological cell complex as well as a surface. A similar

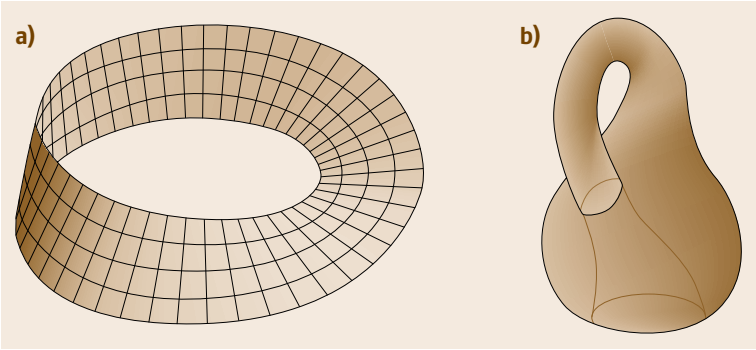

Fig. 10.2a,b Non-orientable surfaces. **(a)** Möbius strip; **(b)** Klein bottle

aggregation concept is defined for 1-D objects: line segments are aggregated to *curves*, and curves recursively to *composite curves*.

In ISO 19107 and in GML, there is another type of aggregation, which is defined less strictly and does not provide the properties that composites have. This type is called *aggregate*, and its subtype for 2-D objects *multisurface*. The surfaces that are part of a multisurface may overlap or penetrate, and the aggregates may be unconnected. A similar concept is defined for curves. The corresponding subtype of aggregates is called multicurve, which may be unconnected and branching, and the curves in a multicurve may intersect.

10.1.2 Special Cases: 2-D as Embedding Space and 2.5-D

In most commercial 2-D GIS, geometries are embedded in 2-D space, i. e., the third coordinate is omitted or set to zero. Obviously, the interpolation of regions has to be planar. An example for a 2-D geometry model is *GML 2* [10.5].

In the so-called *2.5-D geometry model*, the embedding space is 3-D, and the geometries are 0-D to 2-D, but there is an important restriction: for each geometry, the height value z is a function of each x/y-point, i. e., at each x/y-point, there is at most one height value. Typically, a 2.5-D model is used to represent the terrain surface; in that case it is called a *digital terrain model*. Due to the functional dependency between the planar and the height values, vertical walls and overhangs, e.g., the wall of a building or a balcony, are outside the scope of a 2.5-D model.

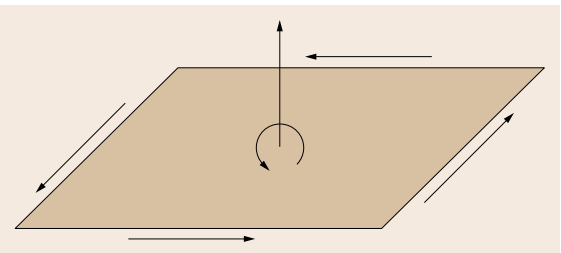

Fig. 10.3 Orientation of a planar surface by applying the right-hand rule

10.1.3 3-D Geometries

Spatial objects like buildings, rooms, or other volume objects are represented by *solids*. In geometrical modeling, solids are described mathematically by *rigid bodies* [10.9]. A rigid body is a bounded, regular, and semianalytical subset of $\mathbb{R}^3$. Regularity excludes non-volume elements like point or line enclaves, while semi-analytical sets are constructed by combining analytical sets – which are the range of analytical functions, particularly polynomials – by the set operations difference, intersection and union. In boundary representation schemes, which are widely used in geometrical modeling, CAD, and GIS, solids are represented by their bounding surfaces. Rigid bodies are exactly those bodies which are bounded by a single, closed 2-manifold [10.9].

In geometrical modeling, there are different schemas to represent solids. The most important are reviewed in the following sections: the boundary representation, constructive solid geometry, raster based or enumeration methods, sweep representations, and primitive instancing. For a comparison of these representations on the basis of several criteria (accuracy, domain, uniqueness, validity, closure, compactness, and efficiency) the reader is referred to *Foley* et al. [10.1].

Boundary Representations

The most common representation schema for solids in GIS is the *boundary representation* [10.1–4], which defines solids by its bounding surfaces. The saddle roof building in Fig. 10.4, for example, is modeled by a solid, which is bounded by seven planar surfaces: a ground surface, four wall surfaces, and two roof surfaces.

The composite surface or the set of surfaces bounding a solid must obey the following three conditions.

1. The surfaces have to enclose the solid completely, without gaps. This requirement is particularly met by closed (composite) surfaces.
2. The surfaces have to be purely areal and non-overlapping, i. e., must be a 2-manifold.
3. The surfaces have to be orientable, and must be oriented in such a way that the top side of all surfaces points outward the solid's interior (Sect. 10.1.1). However, each closed surface embedded in 3-D space without penetrations is orientable. This important theorem is implied by a well-known proposition for closed surfaces, which states that each closed surface embedded in 3-D space without penetrations is homeomorphic to a sphere with $n \geq 0$ handles, which is orientable [10.3, 9]. Hence, orientability does not need to be checked.

The solids provided by the standard ISO 19107 may have enclaves, which define volume voids inside the solid. To represent enclaves, the surfaces bounding a solid are grouped in so-called *shells* (class *GM_Shell*). Each solid is delimited by exactly one exterior shell (which has already been defined by the three conditions above) and optional interior shells, each bounding a void (Fig. 10.5). The interior shells have to fulfill the three conditions given above; particularly, the top sides of all surfaces defining the shell have to point towards the enclave.

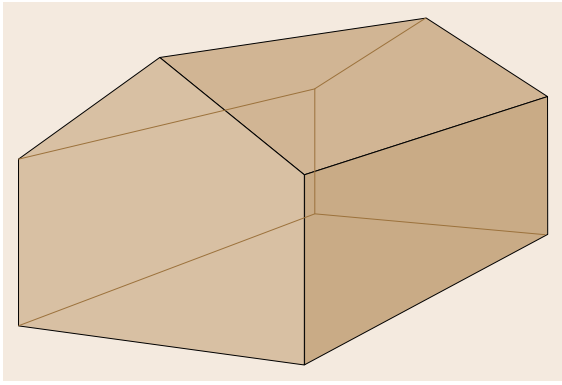

Fig. 10.4 Boundary representation of a saddle roof building

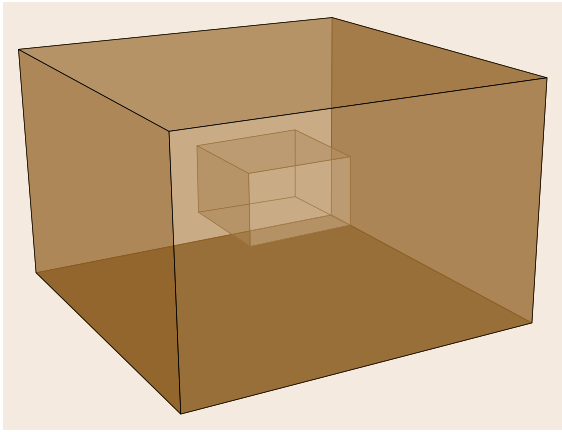

Fig. 10.5 Solid bounded by one exterior shell and one interior shell, forming an enclave

The aggregation concepts which were already introduced for curves and surfaces are provided for solids as well. A *composite solid* (class *GM_CompositeSolid*) is a topological cell complex consisting of solids, which is again a solid, i. e., the exterior and all interior shells of which each fulfills the three conditions given above. A *GM_MultiSolid* is an aggregation of solids which does not obey any restrictions, i. e., which may penetrate each other or which may be unconnected.

Some models define special solids which are not bounded by a shell completely; such solids are used to define 3-D tessellations, i. e., complete coverage of 3-D space by solids.

Constructive Solid Geometry (CSG)

Constructive solid geometry (CSG) [10.2, 3] is a procedural modeling technique which allows to create solid objects by using *Boolean operators* to combine primitive objects. As primitives, boxes or cylinders are typically used. Operations are transformations (rotation, scaling, and translation) and set operations (union, intersection, and difference). Figure 10.6 depicts, as example, the representation of a solid saddle-roof building as a CSG tree. Figure 10.6 gives an example of a representation as a solid saddle roof building as a CSG tree. From one box another box is subtracted (operator /) after transformation (translation and rotation), and from the resulting box yet another box is subtracted, also after transformation (translation and rotation). To avoid generating non-solid parts, *regularized* Boolean opera-

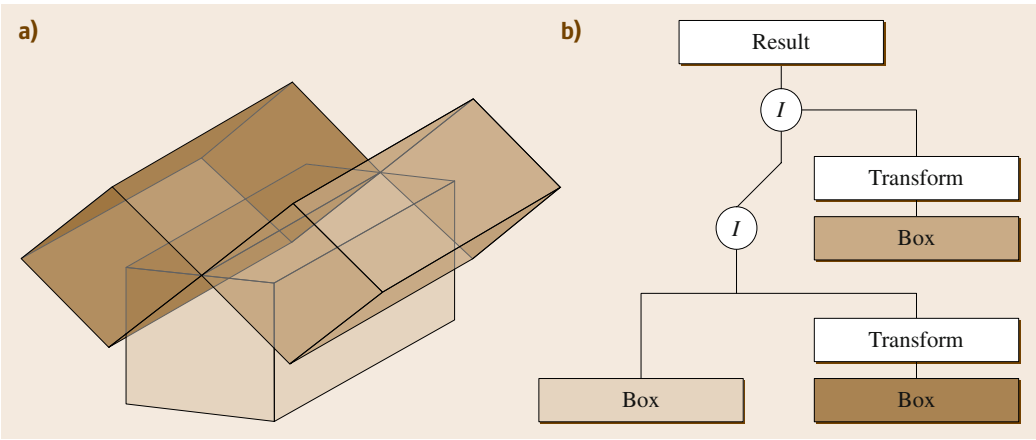

Fig. 10.6a,b CSG model of a simple saddle roof building. **(a)** Transformed primitives; **(b)** CSG tree

tors are employed, which remove all purely areal, linear, or point objects.

CSG is an implicit representation: not the result of the derivation is represented, but the CSG tree containing the sequence of operations and the corresponding parameters and primitives.

Sweep Representations

In *sweep representations*, a solid is generated by sweeping a surface along a curve. All points in space that are touched by sweeping the surface constitute the solid. Three kinds of curves are typically used: lines (translation sweep), circles (rotation sweep), and combinations of both.

A special case of sweeping is *extrusion*, where a planar, horizontal polygon is swept along a line that is perpendicular to that surface (Fig. 10.7). This method is often used to construct buildings in a 3-D city model in the less detailed level of detail 1 (blocks models, see [10.10]) from cadastral footprints. However, extrusion in that case is used as the method for constructing the model; for storing the result of the extrusion, mostly a boundary representation is used.

Raster Based/Enumeration Methods

In *enumeration methods* [10.2], space is partitioned in regular cells (cuboids), which are called *voxels*. A solid, for example, can be represented by listing all voxels contained (completely or partially) in the solid (Fig. 10.8). This method generalizes the partitioning of 2-D planes in rectangular raster cells.

Primitive Instancing

The modeling schema of *primitive instancing* [10.1, 2] enables the representation of predefined, parameterized geometrical primitive object types. To yield a geometrical instance of that type, the variable parameters have to be instantiated. An example is depicted in Fig. 10.9, where a t-brick primitive with five parameters w_1, w_2, h_1, h_2 and l is constructed by instantiating the corresponding parameter values. An important application of primitive instancing in GIS is model-

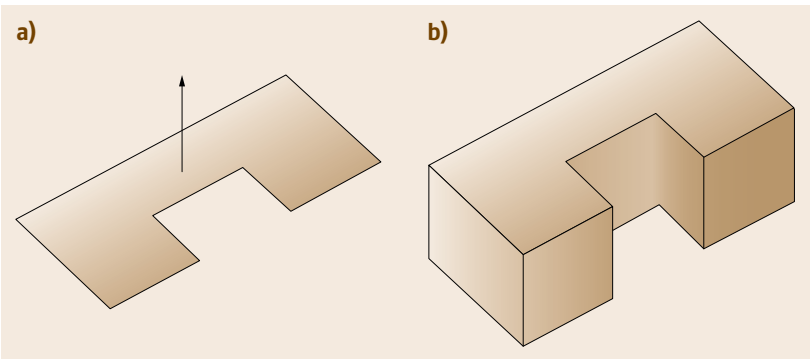

Fig. 10.7 Sweeping a polygon yielding a solid (after [10.2])

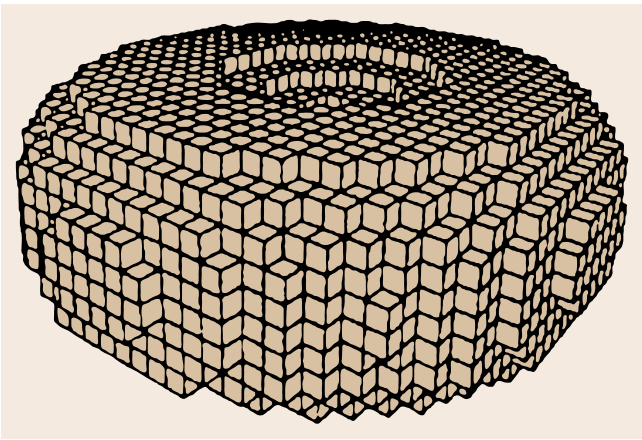

Fig. 10.8 Representation of a solid by voxels (after [10.2])

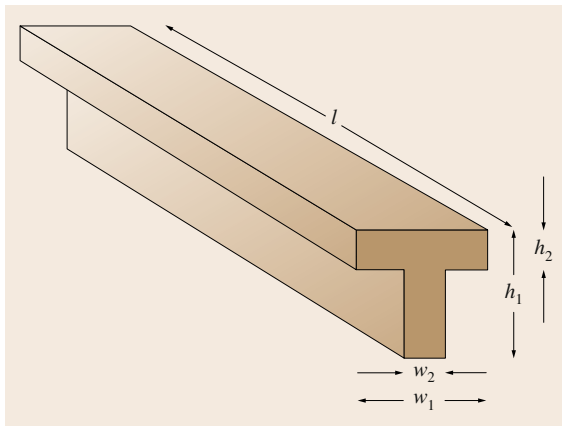

Fig. 10.9 t-brick constructed by primitive instancing (after [10.2])

based building classification and 3-D reconstruction from aerial laser scanning or imagery. The predefined primitive object types are roof types (flat, mono-pitch, saddle, hipped). A saddle roof type, for example, has four parameters: ridge height, eaves height, length, and width, under the assumption that the building is symmetrical.

10.2 Topology

Whereas geometry deals with the shape and metric properties of spatial objects, *topology* focuses on the structure of and qualitative relations between spatial objects, i. e., whether two objects overlap or whether one object is contained in another. In mathematical topology, two branches relevant for 3-D-GIS are differentiated between: *point set topology* and *algebraic topology*. Point set topology aims at defining and classifying qualitative relations between spatial objects. These relations, also called *topological predicates*, provide essential elements of *spatial query languages*, e.g., to formalize a query retrieving all municipalities inside North Rhine Westphalia, or to obtain all highways passing through California. Topological predicates are used in *OGC Filter encoding* (ISO 19143) [10.11], which are employed in the context of spatial data infrastructures, in the query language of the commercial database *Oracle Spatial* [10.12], in query languages provided by the commercial GIS ArcGIS, and in standards like ISO 19107 [10.4].

Algebraic topology [10.13, 14] is the basis of many data models in GIS, CAD, and computer graphics, which are called *topological data models*. This branch of mathematics provides formal rules to construct complex objects from primitive ones, avoiding penetrations and modeling touches of spatial objects explicitly. Topological data models aim at providing efficient navigational access without considering geometry and serve as a base for the definition of consistent models. In this section, we first discuss topological relations in different dimensions and then address 3-D topological data models.

10.2.1 Topological Relations

The specification of topological relations in general is defined for arbitrary *topological spaces*. A *topological space* [10.8],

which is a fundamental notion in topology, is a set M together with a set N of subsets of M, called *neighborhoods*, where the following conditions hold.

1. Each element $m \in M$ is in a neighborhood $n \in N$.
2. The intersection of two neighborhoods of $m \in M$ is or contains a neighborhood of m.

Let M be a topological space and X a subset of M. An element $p \in M$ is *near* X, if each neighborhood of p contains an element in X. The *interior* of X, denoted X°, is the set of all elements in X, which are not elements near the complement of X. The *boundary* of X,

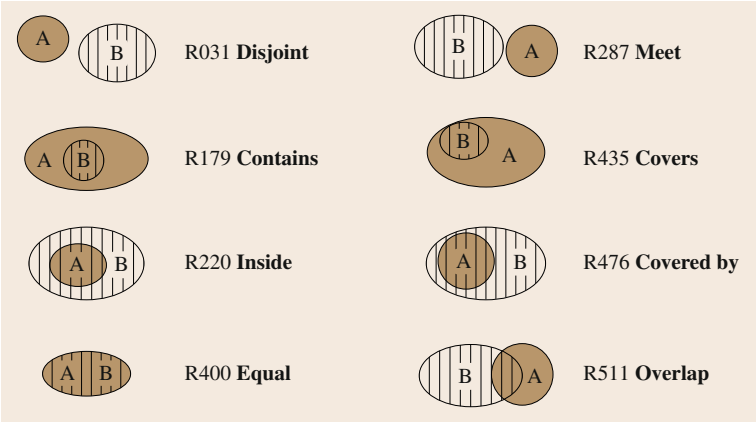

Fig. 10.10 Eight relations for simple regions distinguishable by Egenhofer's 4-intersection model (after [10.15])

denoted ∂X, is the set of all elements which are both near X and to the complement of X. The *exterior* X^- of X is the complement of the union of the boundary and the interior.

To illustrate these concepts, let M be the set $\mathbb{R}^3$ and the neighborhoods $n \in N$ be defined by open balls. Then the interior and the boundary of spatial objects (point sets) has an intuitive meaning: X may, for example, be a box. The interior of the box (X°) is the point set bounded by the six rectangles defining the box, and the boundary (∂X) is defined by the six rectangles. The exterior of the box (X^-) is the space outside the box.

Topological relations can be defined based purely on the notions of interior, boundary, and exterior. We now focus on the 4-intersection and the 9-intersection model and its extensions in 2-D and 3-D. Another similar approach for defining topological relations is region connection calculus [10.16].

2-D

The well-known *4-intersection model* introduced by *Egenhofer and Franzosa* [10.15] defines binary topological relations in 2-D, i. e., relations between two objects. For two regions A and B, the intersections ($\cap$) of the interiors (A°, B°) and the boundaries (∂A, ∂B) are considered systematically, in which it is only relevant whether the intersection is empty or not. The result is represented by a Boolean 2×2 matrix (an empty intersection $\emptyset$ is denoted by *false*, a non-empty $\emptyset$ by *true*)

$$\begin{vmatrix} A^\circ \cap B^\circ & A^\circ \cap \partial B \\ \partial A \cap B^\circ & \partial A \cap \partial B \end{vmatrix} .$$

The regions considered in that model are restricted topologically: a region must be bounded by a single connected, closed curve, which is non-self-intersecting (i. e., a *ring* as defined in Sect. 10.1.3) and hence, does not have holes. Not all of the $2^4 = 32$ relations have a spatial realization in that model; only eight relations are possible (Fig. 10.10). The other 24 relations cannot occur due to dependencies between the values of matrix elements. For example, if a boundary–interior or interior–boundary intersection is non-empty, then the interior–interior intersection is non-empty as well.

This model can also be applied to points and curves, where the boundary of a curve is defined as the union of both end points, the interior is the curve without the end points. The boundary of a point is empty, and the interior is the point itself. A line object must be connected, non-branching, and non-self-intersecting.

Two extensions of the 4-intersection model have been developed, which both provide a more fine-grained differentiation of spatial arrangements. The first extension is to consider the *exterior* A^- of a point set A as well. In this *9-intersection model* [10.17], a 3×3 matrix denotes the Boolean intersection values. The relations of the 9-intersection model are also called *Egenhofer operators* [10.4]

$$\begin{vmatrix} A^\circ \cap B^\circ & A^\circ \cap \partial B & A^\circ \cap B^- \\ \partial A \cap B^\circ & \partial A \cap \partial B & \partial A \cap B^- \\ A^- \cap B^\circ & A^- \cap \partial B & A^- \cap B^- \end{vmatrix} .$$

Figure 10.11 depicts an example demonstrating that the 9-intersection model is more powerful than the 4-intersection model. In Fig. 10.11a,b, the region A and a line segment B share boundaries, but in Fig. 10.11b, both enclose a region completely. Hence, both situations

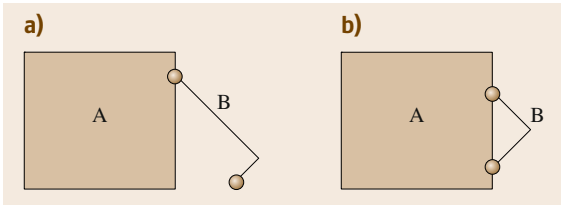

Fig. 10.11 Two topological different situations which can be distinguished by the 9-intersection model but not by the 4-intersection model

are different topologically. In the 4-intersection model, both situations are represented by the same relation: the intersection of the interiors and of the boundaries/interiors is empty, whereas the intersecting of the boundaries is not ($A^\circ \cap B^\circ = A^\circ \cap \partial B = \partial A \cap B^\circ = \emptyset$; $\partial A \cap \partial B = \neg\emptyset$). In the 9-intersection model, the intersection between the exterior of A and the boundary of B is non-empty in Fig. 10.11a ($A^- \cap \partial B = \neg\emptyset$), but empty in Fig. 10.11b ($A^- \cap \partial B = \emptyset$). Hence, both situations can be distinguished.

However, if 2-D objects embedded in 2-D space are considered (Fig. 10.10), the 4-intersection and the 9-intersection models yield identical results. In general, if the *co-dimension* – the difference between the dimension of the embedding space and the dimension of the objects – is zero, both models are identical.

A second extension of the 4-intersection model, which also can be applied to the 9-intersection model, is to consider the *dimension* of the intersection. The situations in Fig. 10.12 cannot be differentiated by the 4-intersection model or the 9-intersection model, since the relation *meet* in both cases, but both differ in the dimension of the intersection, which is 0-D or 1-D. The extension of the 4-intersection model to the 9-intersection model and the inclusion of the dimension of the intersection is called the *dimensionally extended 9-intersection model* or *DE-9IM* [10.18].

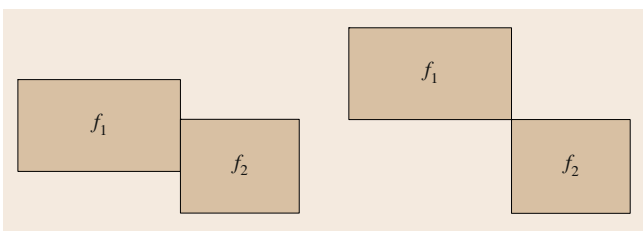

Fig. 10.12 Two situations which cannot be differentiated by the 4- or 9-intersection model but by considering the dimension of the intersection

3-D Relations

Zlatanova [10.19] extended the 4-intersection model to 3-D by using $\mathbb{R}^3$ as the embedding space for points, lines, and surfaces, and by considering solids. Solids must have a single, connected, 2-manifold boundary. Figure 10.13 depicts all possible relations between two solids, which are identical to the relations between two surfaces in 2-D (Fig. 10.10).

If the co-dimension is strictly greater than zero, a variety of topological relations is observed between two objects embedded in 3-D. As an example, Fig. 10.14 enumerates all 38 relations between two surfaces. These relations cannot be named meaningfully; they are denoted by a decimal code preceded by the character R, where the code is equivalent to the binary representation of the 9-intersection matrix. The order of the matrix elements in the binary code is as follows (the second row denotes the decimal and the third the binary representation of the summand corresponding to the relation in the first row)

$\partial A \cap \partial B$	$A^\circ \cap B^\circ$	$\partial A \cap B^\circ$
$2^8 = 256$	$2^7 = 128$	$2^6 = 64$
100 000 000	10 000 000	1 000 000

$A^\circ \cap \partial B$	$A^- \cap B^-$	$A^- \cap \partial B$	$A^- \cap B^\circ$
$2^5 = 32$	$2^4 = 16$	$2^3 = 8$	$2^2 = 4$
100 000	10 000	1000	100

$\partial A \cap B^-$	$A^\circ \cap B^-$
$2^1 = 2$	$2^0 = 1$
10	1

For example, the code R287 ($= 256+16+8+4+2+1$) is equivalent to the binary code 100011111 ($= 100\,000\,000+10\,000+1000+100+10+1$); in that relation, both boundaries intersect and all other intersections not involving an exterior are empty ($\partial A \cap \partial B = A^- \cap B^- = A^- \cap \partial B = A^- \cap B^\circ = \partial A \cap B^- = A^\circ \cap B^- = \neg\emptyset$/true; $A^\circ \cap B^\circ = \partial A \cap B^\circ = A^\circ \cap \partial B = \emptyset$/false).

The usage of the topological relations in query languages is by referencing the name of the relation. In *Oracle Spatial 11g*, a combination of relation names, connected by a logical *OR*, may be used. In the *simple features model* [10.20], ISO 19107 *Spatial schema* [10.4] and Oracle, a more flexible mechanism is also employed: a method *relate* receives the complete pattern of the 9-intersection matrix in row major form as input, containing the values F (empty intersection), the dimension 0, 1, 2, 3 of the intersection, or the wildcard symbol N (the value does not matter).

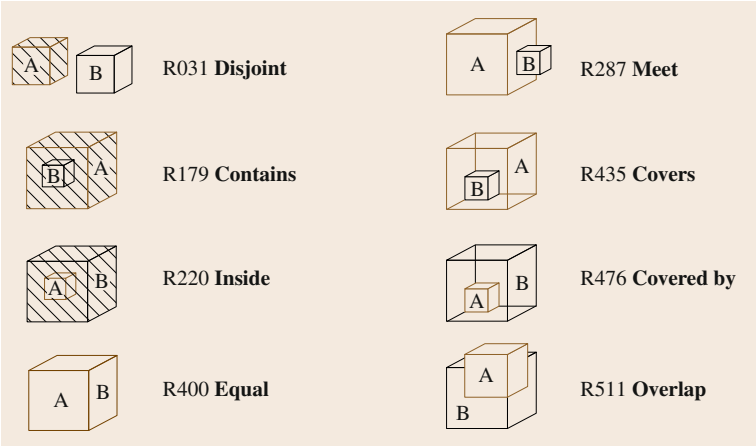

Fig. 10.13 Topological relations between two solids in $\mathbb{R}^3$ (after [10.19])

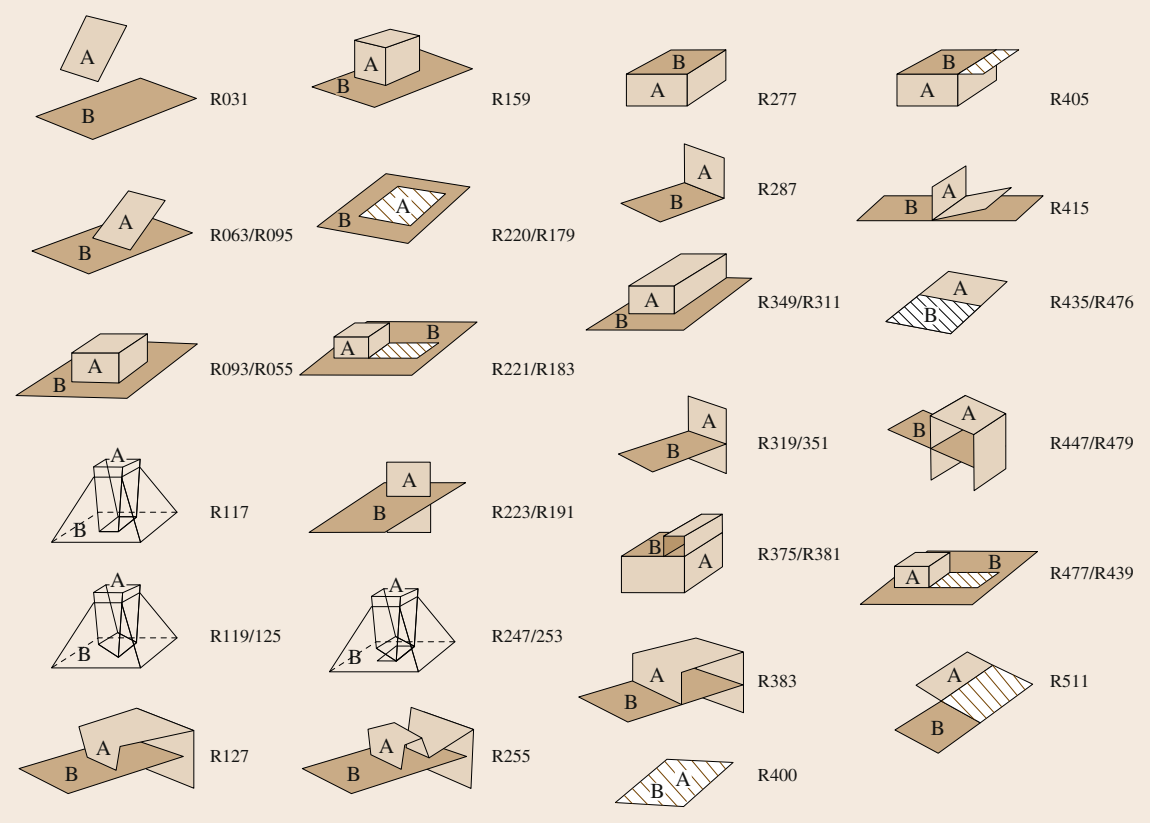

Fig. 10.14 Topological relations between two surfaces in $\mathbb{R}^3$ (after [10.19])

10.2.2 Topological Data Models

The theoretical foundation of all topological data models in GIS or CAD is the mathematical concept of *simplicial complexes* and its generalization, the concept of *cell complexes* [10.13, 14]. A cell complex consists of four types of primitives: 0-cells, also called nodes, *1-cell*, also called edges, *2-cells* (faces), and *3-cells*

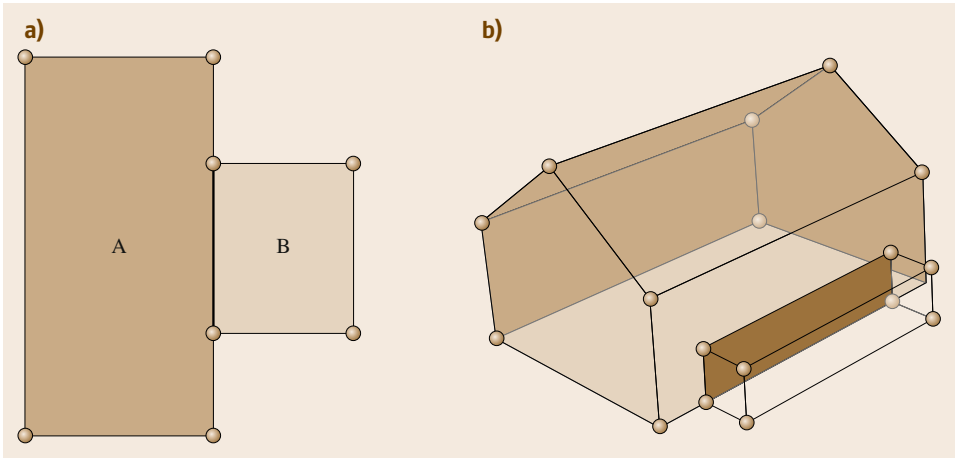

Fig. 10.15a,b Cell complexes, **(a)** consisting of two 2-cells A and B touching in a common edge (*depicted bold*), and **(b)** of two 3-cells touching in a common face (*depicted dark*)

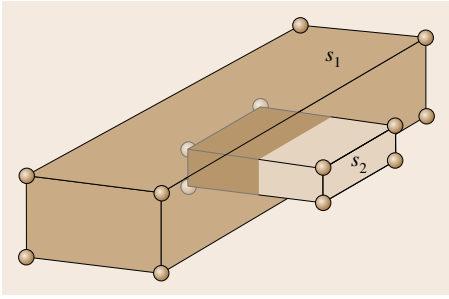

Fig. 10.16 0-cells, 1-cells, 2-cells, and 3-cells not constituting a cell complex: the intersection of solids s_1 and s_2 is not a cell in the boundary of both cells

(topological solids). Each n-cell is topologically equivalent to a manifold of the corresponding dimension. For example, a 3-cell is topologically equivalent to a sphere, and a 2-cell to a 2-D-disk. Each n-cell c is bounded by $(n-1)$-cells $c_1, c_2, \ldots, c_k$, which are the *boundary* of the cell. Vice versa, c is in the *co-boundary* of $c_1 \ldots c_k$. A *cell complex* is an aggregation of n-cells, where the following condition holds.

- The intersection of two cells in the cell complex is either empty or a cell which is part of the boundaries of both cells.

Figure 10.15 gives two examples of cell complexes. In Fig. 10.15a, the intersection of the 2-cells A and B is the 1-cell depicted by thick lines. It is part of the boundary of both A and B. The intersection of the two 3-cells in Fig. 10.15b is given by the dark colored 2-cell and the 0- and 1-cells in its boundary. This structure is part of the boundary of both 3-cells. A counterexample is depicted in Fig. 10.16. Two solids penetrate. Hence, the intersection of both is not a common boundary; the structure is not a cell complex.

The advantages of representing GIS data as a cell complex are as follows.

- The model implies that there are no penetrations or overlaps of the interiors of cells; cells touch at least in common boundaries. This is an essential consistency constraint for many GIS applications; for example, two parcels do not overlap, and two buildings do not penetrate.
- The explicit representation of any touching between objects, i. e., the boundary and co-boundary relations, facilitates navigational access to all neighboring objects, without the need to consider geometry. This supports the efficient processing of queries involving topological predicates, e.g., inside or touch. These predicates were discussed in the last section.

All topological data models reviewed in the next sections are based on the concept of cell complexes. They differ with regard to the dimension of the embedding space (2-D or 3-D), the dimension of the cells, the geometric shape of the cells (triangles/tetrahedrons, arbitrary shape, enclaves), whether cells are explicitly or implicitly modeled, and to which degree the boundary and co-boundary relations are modeled explicitly.

2-D Models

Realizations of 2-D topological models are the *maps* defined by *Plümer and Gröger* [10.21], which define a complete coverage of the plane by faces and are characterized axiomatically, i. e., provide an efficient and effective method to check whether a dataset has the map properties. In *Gröger* [10.22], the concept is extended by allowing holes in faces. *Molenaar's* [10.23, 24] *sin-*

gle and multivalued vector maps are a special case of the 3-D-version, which will be described in the next section. The cells of the model presented by *Egenhofer* et al. [10.25] are restricted to triangles geometrically, whereas the *coverages* data type of Esri's GIS tools *ArcGIS* allow for faces of arbitrary shape, which may contain holes.

Topological Networks

A *topological network* is a cell complex consisting of 0- and 1-cells, which is embedded in 3-D space. Another term for a topological network is a *graph* embedded in 3-D space. The focus of topological networks is on explicit modeling of the connectivity between line objects (edges) and junctions (nodes), not on surface topology. The main application area of topological networks is the modeling of transportation or utility networks. The third dimension is often not represented explicitly, but due to overpasses and underpasses, 2-D or 2.5-D models are not sufficient. A prominent and widely used example is the standard *Geographic Data Files* (GDF) [10.26], which are the base of all data models for commercial vehicle navigation. Level 1 in GDF, which is used for path finding, is a topological network. Another example for a topological network is the graph representing reachability inside buildings, which is used for indoor path finding [10.27]. The representation of topological networks in data bases is discussed in Chap. 3.

3-D Models

A survey of 3-D topological models for GIS can be found in *Zlatanova* et al. [10.28], whereas in *Ellul* and *Haklay* [10.29] the requirements and benefits of such models for GIS applications are identified. The first topological data model in GIS from a historical perspective was the *Formal Data Structure* (*FDS*) presented by *Molenaar* [10.30]. He distinguishes the primitive *nodes*, *arcs/edges*, and *faces*. Volumes are called *bodies* and exist on a feature level. Faces are bounded by edges/arcs, and each arc has a start and an end node. Each face has a body on the left and a body on the right side (Fig. 10.17). Edges are straight lines geometrically, and faces are planar and may contain holes. *Flick* [10.31] extends the FDS by introducing *bodies* as topological primitives. The *urban data model* (*UDM*) developed by *Coors* [10.32] and the *simplified spatial schema* [10.19] modify this model by omitting the explicit representation of edges, facilitating efficient visualization. For the same reason, faces in the UDM are restricted to triangles. A topological model based on *simplicial complexes* – the restriction of cell complexes to triangles and tetrahedrons – is the *TEN* (tetrahedral network) structure [10.33]. It can be implemented in relational databases very efficiently [10.34].

The topological model introduced by *Pigot* [10.35] provides a full implementation of the concept of cell complexes, including all boundary and co-

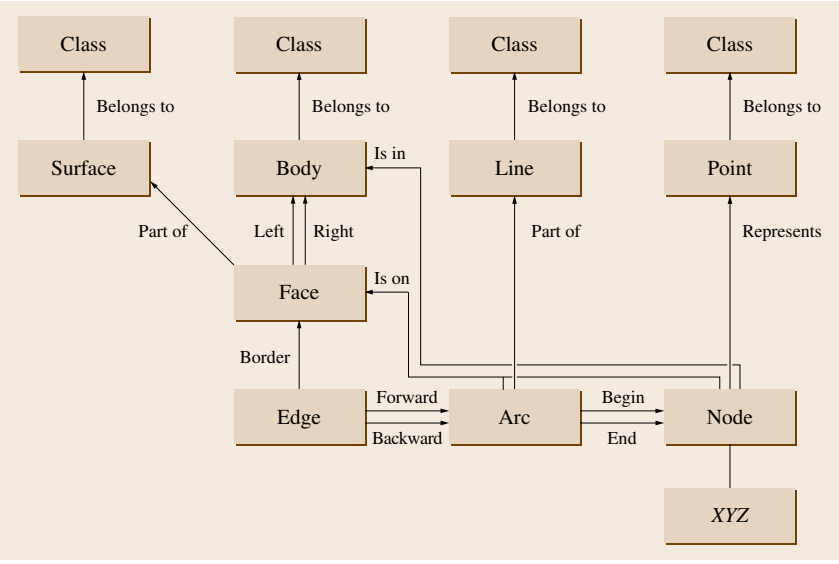

Fig. 10.17 Diagram of the 3-D FDS by *Molenaar* (after [10.23])

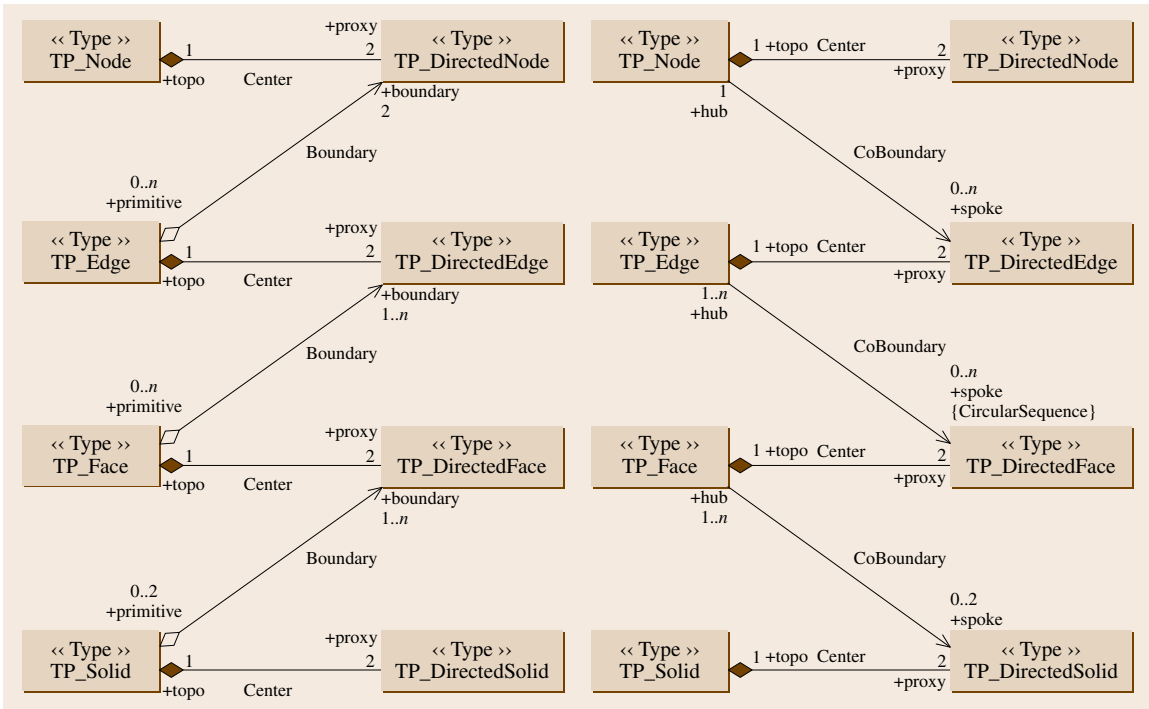

Fig. 10.18 Boundary (*left side*) and co-boundary (*right-hand side*) relations (after [10.4])

boundary relations. The model defined by *Gröger* and *Plümer* [10.36] extends cell complexes in two respects: connectivity is considered as an additional requirement, prohibiting floating buildings, for example, and two special solids are introduced: a solid representing the air space and one representing the Earth' mass. Both are bounded only partially. Hence, this model defines a 3-D tessellation of space by solids: each point in 3-D space is in the boundary of a solid or in the interior of exactly one solid. The declarative definition of the model is accompanied by *axioms*, which are used to check effectively and efficiently whether datasets are consistent, i. e., meet the requirements of the model. Transaction rules for updating datasets while preserving consistency are sketched in *Gröger and Plümer* [10.36].

A further topological model is provided by the standard ISO 19107 *Spatial schema* [10.4]. The model defines topological primitives for all dimensions (nodes, edges, faces, topological solids) and fully realizes the boundary and co-boundary relations. The properties of the topological primitive are defined by its geometrical counterparts, which were described in Sect. 10.1, faces (class *TP_Face*) must be connected and may have holes delimited by interior rings, and topological solids (class *TP_Solid*) must also be connected and may have enclaves bounded by interior shells. All boundary and co-boundary relations are represented explicitly (see the UML diagram in Fig. 10.18). In analogy to the orientable primitives on the geometry level, directed topology objects interconnecting the topological primitives are used to define consistently oriented boundaries and co-boundaries (Fig. 10.18). For example, the boundary of a *TP_Solid* consists of a set of directed faces (class *TP_DirectedFace*); each directed face is assigned to exactly one *TP_Face* by the *topo* role of the *center* association. This face is related to exactly one other directed face, which represents the face's role in the boundary of another topological solid that is a neighbor of the first one. Vice versa, the co-boundary relations are defined by using directed topology objects: a face, for example, has a co-boundary relation to two directed solids, each of them relating to a solid that is bounded by the face (Fig. 10.19).

In addition to the boundary and co-boundary relations, where the difference of dimensions of the cells is 1, there is a relation called *isolated*, which associates a node (0-cell) with a face (2-cell) or a topological

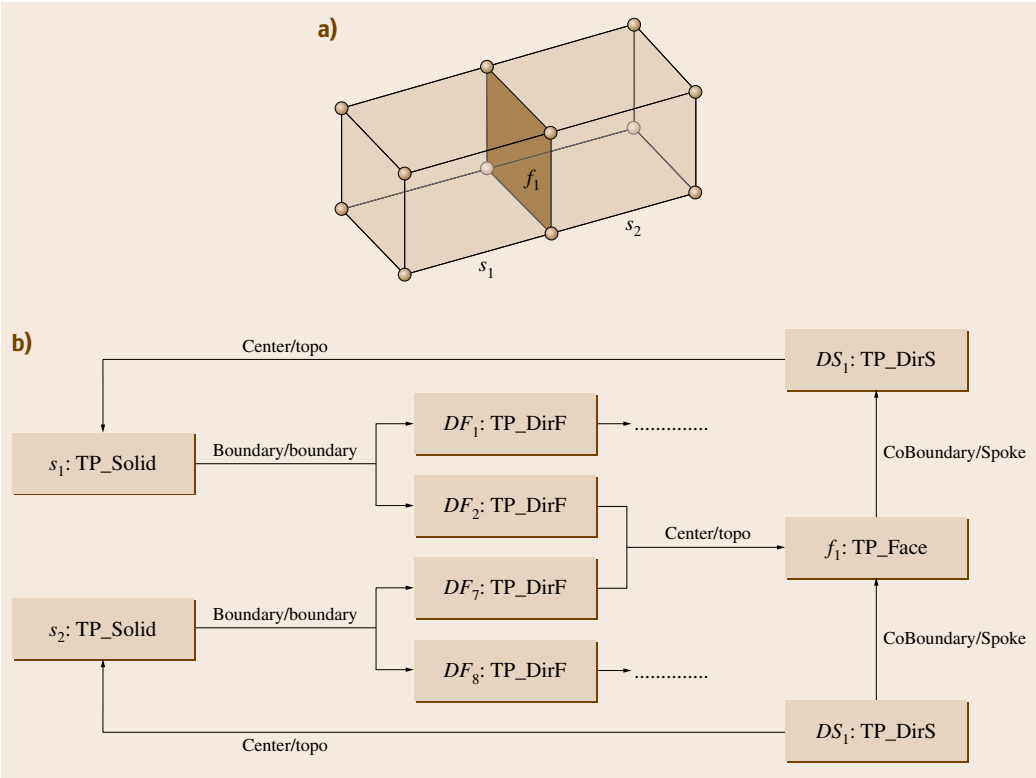

Fig. 10.19
(a) A 3-D scene with two solids s_1 and s_2 sharing a face f_1 and **(b)** an extract from the corresponding UML instance diagram (the class names *TP_DirectedFace* and *TP_DirectedSolid* are abbreviated to *TP_DirF* and *TP_DirS*)

solid (3-cell), when this node is inside the interior of the face or of the topological solid. Likewise, an edge is related to a topological solid by that association, when the edge is completely in the interior of the solid.

ISO 19107 provides classes for defining topologies (prefix *TP_*) that are independent of its geometrical counterparts (prefix *GM_*), but are related by associations. The advantage of this approach is flexibility; there are three options to use topology.

1. Topology is omitted, i. e., only the geometrical aspects are represented.
2. Both geometry and topology are modeled and linked, combining the advantages of both representations.
3. Only topology is represented, i. e., a scene is represented purely structural by topological primitives and its boundary and co-boundary relations, without any (geo)metrical information like shape, size, or location.

If the purely topological representation in case 3 is restricted to point and line primitives and the corresponding boundary and co-boundary relations, one obtains the well-known *graph* structure. This structure and corresponding algorithms which are crucial for GIS are the topic of Sect. 10.3.

10.3 Graph Theory (Königsberg Bridge Problem)

10.3.1 The Problem Introduction

In GIS, concepts from graph theory are extremely useful in expressing the spatial structure of entities seen as points, lines, areas, and solids, after the geometrical details of these entities are removed. For example, in transportation and river networks, the topological properties of their structures can be represented using graphs. This article describes the origins of graph theory and the impact it has

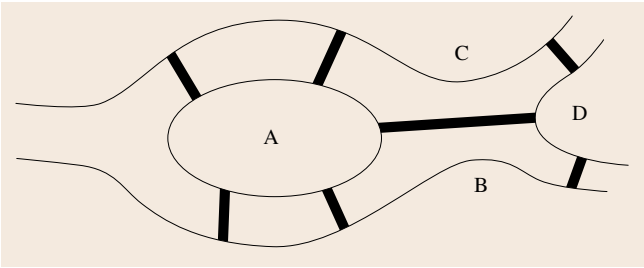

Fig. 10.20 Layout of the city of Königsberg showing the river, bridges, and land areas

on various fields ranging from geography to economics.

The Königsberg bridge problem is a classic problem, based on the topography of the city of Königsberg, formerly in Germany but now known as Kaliningrad and part of Russia. The river Pregel divides the city into two islands and two banks as shown in Fig. 10.20.

The city had seven bridges connecting the mainland and the islands (represented by thick lines in the figure) [10.37–40]. The problem asks whether there is a walk that starts at any island, traverses every bridge exactly once, and returns to the start point. The solution proposed by a Swiss Mathematician, Leonhard Euler, led to the birth of a branch of mathematics called graph theory, which finds applications in areas ranging from engineering to the social sciences. Euler proved that there is no solution to the problem based on the number of bridges connecting each land area.

The results from the solution of the Königsberg problem have been extended to various concepts in graph theory. In graph theory a path that starts and ends at the same node and traverses every edge exactly once is called a Eulerian circuit. The result obtained in the Königsberg bridge problem has been generalized as Euler's theorem, which states that a graph has a Eulerian circuit if and only if there are no nodes of odd degree. A node is a *node of odd degree* if the number of edges incident to the node is odd. Since the graph corresponding to Königsberg has four nodes of odd degree, it cannot have a Eulerian circuit. Subsequently the concept of Eulerian paths was introduced, which deals with paths that traverse every edge exactly once. It was proved that such a path exists in a graph if and only if the number of nodes of odd degree is 2 [10.39–43].

While studying the Königsberg bridge problem, Euler also observed that the number of bridges at every land area would add up to twice the number of bridges. This result came to be known as the hand-shaking lemma in graph theory, which states that the sum of node-degrees in a graph is equal to twice the number of edges. This result is the first formulation of a frequently used result in graph theory that states that the sum of node degrees in a graph is always even [10.42, 43].

10.3.2 Abstraction

The Königsberg bridge problem was formulated based on the layout of the city of Königsberg around the river Pregel. The problem was to find a tour that starts at any point in the city, crosses each bridge exactly once, and returns to the starting point. No one succeeded in doing this.

Leonhard Euler formulated the problem as finding a sequence of letters A, B, C, D (that represent the land areas) such that the pairs (A,B) and (A,C) appear twice (thus representing the two bridges between A and B, and A and C) and the pairs (A,D), (B,D), (C,D) appear only once (these pairs would represent the bridges between A and D, B and D, and C and D). Euler used a counting argument to prove that no such sequence exists, thus proving that the Königsberg bridge problem has no solution. Euler presented this result in the paper *The Solution of Problem Relating to the Geometry of Position* at the Academy of Sciences of St. Petersburg in 1735. This paper, in addition to proving the non-existence of a solution to the Königsberg bridge problem, gave some general insights into arrangements of bridges and land areas [10.41, 42, 44].

Euler summarized his main conclusions in three points.

1. If there is any land area that is connected by an odd number of bridges, then a cyclic journey that crosses each bridge exactly once is impossible.
2. If the number of bridges is odd for exactly two land areas, then there is a journey that crosses each bridge exactly once is possible, if it starts at one of these areas and ends in the other.
3. If there are no land areas that are connected by an odd number of bridges, the journey can start and end at any land area [10.42].

Euler gave heuristic reasons for the correctness of the first conclusion. To complete a cyclic journey around the land areas, crossing each bridge exactly once, there must be a bridge to leave the area for every bridge to enter it. This argument was generalized to the conclusion that a cyclic journey is possible if every island is connected by an even number of bridges.

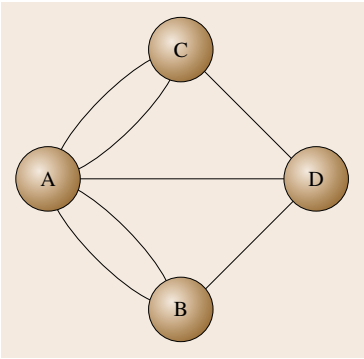

Fig. 10.21 Graph representation of the city of Königsberg

Formal proofs for the conclusions were not proposed until the year 1871, in a posthumous paper by Hierholzer [10.38, 41].

The paper presented by Euler on the Königsberg bridge problem can be considered to mark the birth of graph theory in general. Later, a diagrammatic representation evolved, which involved nodes or vertices and the connecting lines that are called edges. Using this representation, the Königsberg problem is modeled as shown in Fig. 10.21.

Circles, called nodes, represent the islands and the banks and connecting lines called edges represent the bridges. The number of edges that are incident on a node is called the degree of the node [10.42]. In the Königsberg bridge problem, the number of bridges connecting a land area would be the degree of the node representing the land area.

In an undirected graph, a cycle that traverses every edge exactly once is called a Euler tour or Euler cycle. Any graph that possesses a Euler cycle is called a Eulerian graph. A path that traverses each edge exactly once with different starting point and end point is called a Eulerian path. An undirected multigraph has a Eulerian circuit (path) if and only if it is connected and the number of vertices with odd degree is zero (two).

Figure 10.22 illustrates the Eulerian path and the Eulerian cycle in a graph. In Fig. 10.22a, a Eulerian path exists and it can be observed that the graph has exactly two, odd degree vertices, which would be the start and end vertices of the Eulerian path, A-B-C-D-A-C. Figure 10.22b does not have vertices with odd degree and has a Eulerian cycle, whereas Fig. 10.22c has neither a Eulerian path nor a Eulerian cycle.

10.3.3 Finding a Eulerian Circuit in a Graph

The method successively finds cycles in the graph. At each step the edges that are in the already discovered cycles are removed and the cycle is spliced with the one discovered in the previous step. This process is continued until all edges are exhausted. These basic ideas were formalized into an algorithm in [10.45]. The algorithm maintains a list L with each vertex x such that the kth entry in the list indicates the vertex to visit when vertex x is reached the kth time.

Algorithm

Step 1 Select any vertex $v1$. $v = v1$; set $kv = 0$. Label all edges as unvisited.

Step 2 Select an unvisited edge e incident to v. Mark this edge *visited*. Let w be the other end vertex of e. Increment kv by 1 and $Lv[kv] = w$. If w has an unvisited incident edge, go to step 3. If not, y will be $v1$. Then, go to Step 4.

Step 3 Set $v = w$ and go to Step 2.

Step 4 Find a vertex $v1$ such that there is at least one visited edge and one unvisited edge incident at $v1$. Set $v = v1$ and go to Step 2. If no such vertex exists, go to Step 5.

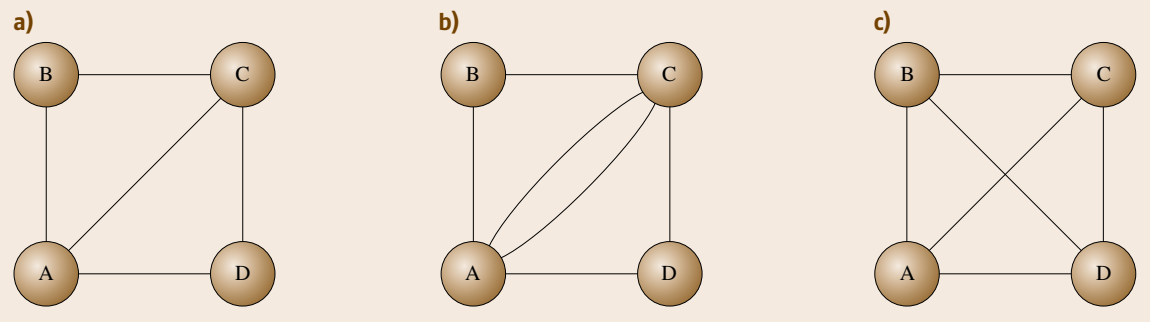

Fig. 10.22a–c Illustration of a Eulerian path and a Eulerian cycle. **(a)** Eulerian path A-B-C-D-A-C; **(b)** Eulerian cycle A-B-C-D-A-C-A; **(c)** Neither Eulerian path nor cycle exist

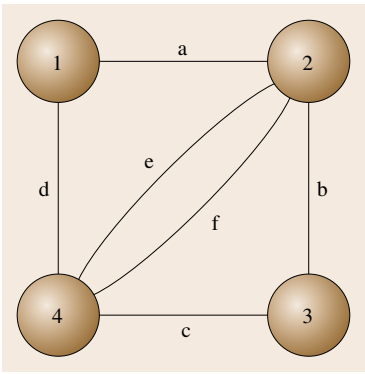

Fig. 10.23 Illustration of the Eulerian algorithm

Step 5 To construct the Eulerian circuit, start at $v1$. The first time a vertex u is reached, proceed to the vertex $Lu[ku]$. Decrement ku and continue.

Trace of the Algorithm for Fig. 10.23

Step 1 $v1 = 1 = v$; $kx = 0$ for $x = 1, 2, 3, 4$.
Step 2 Select edge a. $w = 2$; $k2 = 1$; visited $(a) = 1$.
Step 3 $v = 2$; Select edge b. $w = 3$; $k3 = 1$; visited $(b) = 1$.
Step 4 $v = 3$; Select edge c. $w = 4$; $k4 = 1$; visited $(c) = 1$.
Step 5 $v = 4$; Select edge d. $w = 1$; $k1 = 1$; visited $(d) = 1$.
Step 6 $v = 2$;
Step 7 Select edge e; $w = 4$; $k4 = 2$; visited $(e) = 1$
Step 8 $v = 4$;
Step 9 Select edge f; $w = 2$; $k2 = 2$; visited $(f) = 1$
Step 10 Construct the cycle as 1–2–4–2–3–4–1.

10.3.4 Key Applications

Eulerian cycles find applications in problems where paths or cycles need to be found that traverse a set of edges in a graph. Such problems are generally called edge routing problems.

Snow Plow Problem

This problem requires finding the least distance route in the road network that starts and ends at the station so that snow can be cleared from the streets at minimum cost. The minimum distance route is obviously the Eulerian cycle, since this cycle traverses each edge exactly once. However, it is unlikely that any real road network would happen to satisfy the necessary conditions that make it Eulerian. In that case, the problem moves to the realm of *the Chinese postman problem* [10.45–47].

Chinese Postman Problem

A postman delivers mail everyday in a network of streets. It is useful to know whether or not the postman can traverse the network and return to the mail station without driving the length of any street more than once. If the network is not Eulerian, the problem is modified to the one where it is required to find the shortest path, which visits each edge at least once. This problem statement requires a parameter to be associated with each edge that represents the cost of traversing that edge. For example, cost can be the represented in terms of the length of the street, which the edge represents.

In a non-Eulerian graph, the postman's circuit, shortest or otherwise, will repeat one or more edges. Every vertex is entered the same number of times that it is left so that any vertex of odd degree has at least one incident edge that is traversed at least twice. The

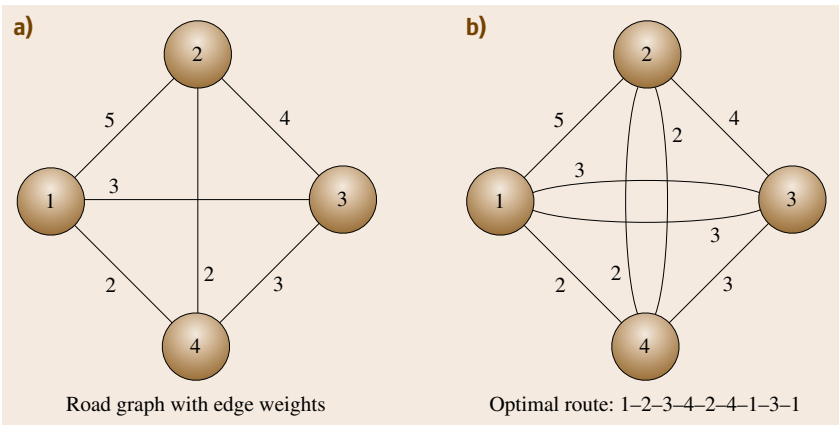

Fig. 10.24a,b Illustration of the Chinese postman problem algorithm. **(a)** Road graph with edge weights; **(b)** Optimal route: 1-2-3-4-2-4-1-3-1

Table 10.1 Shortest path cost between the pairs

	1	2	3	4
1	0	4	3	2
2	4	0	4	2
3	3	4	0	3
4	2	2	3	0

Table 10.2 Matching and costs

Matching	Cost
(1,2),(3,4)	$4+3=7$
(1,4),(2,3)	$2+4=6$
(1,3),(2,4)	$3+2=5$

Chinese postman problem is formulated as an optimization problem where the total cost of repeated edges in minimized.

Algorithm

Step 1 Find the shortest path between each pair of odd degree.

Step 2 Find the subgraph G' with the odd degree vertices.

Step 3 Find the minimum weight matching of all the edges in G'. The edges in the shortest path connecting a matched pair of odd degree vertices should be repeated.

Figure 10.24 shows a sample graph with edge weights and the Chinese postman algorithm finds the least cost (minimum edge weight) path in the graph such that every edge is traversed at least once. Table 10.1 shows the shortest path costs between every pair of vertices, which is used by the algorithm to find the minimum weight matchings on edges. Matching of a graph is a set of edges without common vertices. The three possible matchings and the corresponding costs are provided in Table 10.2. The algorithm finds that the paths from vertex 1 to vertex 3, and the path from 2 to 4 must be repeated, since this is the minimum cost matching (the cost is 5). The algorithm finds the optimal route to be 1–2–3–4–2–4–1–3–1 in the graph shown in Fig. 10.24.

Capacitated Chinese Postman Problem

This problem arises where each edge has a demand and vehicles to be routed have finite capacities. For example, in applications involving road salting in the winter season, there is a limit on the maximum amount of salt that a truck can carry. The amount of salt required is fixed for a road segment. The capacitated Chinese postman problem finds the sets of routes from a single station that service all the road segments in the network at a minimal cost and are subject to the constraint that the total demand of each route does not exceed the capacity of each truck. Christofides proposed an algorithm to solve this problem.

Capacitated Arc Routing Problem

This problem is different from the capacitated Chinese postman problem in that demands of some of the road segments can be zero. This situation can arise in road salting scenarios where state highways can be used for traveling, but need not be salted. These edges can be used to traverse between the edges that require the service.

Both the capacitated Chinese postman problem and capacitated arc routing problem are NP-hard [10.47], and heuristic methods are normally used to obtain solutions.

10.3.5 Graph Theory

The Königsberg problem had a powerful impact on mathematics, paving the way for the creation of a new modeling theory called graph theory. The applications of graph theory are numerous in science and engineering. A few are listed below.

Graph Theory in Spatial Networks

The very fact that graph theory was born when Euler solved a problem based on the bridge network of the city of Königsberg points to the apparent connection between spatial networks (e.g., transportation networks) and graphs. In modeling spatial networks, in addition to nodes and edges, the edges are usually qualified by adding weights that encode information like the length of the road segment that the edge represents. Connectivity and shortest paths in spatial networks have been extensively studied using graphs [10.48].

Graph Theory in Geography

Graphs are also widely applied in geography in the modeling of stream systems. Streams have been modeled as hierarchical graphs and random graphs in the literature [10.49].

In addition to the applications described above, graphs find other wide applications, including modeling of social networks, molecular structures in chemistry, computer networks, electrical networks, and syntax structures in linguistics.

10.3.6 Future Directions

Relationships between Eulerian graphs and other graph properties such as the Hamiltonian property are being studied [10.50]. Graphs, the mathematical model which owes its origin to the Königsberg bridge problem, are being increasingly applied to several evolving domains such as spatio-temporal networks, which has necessitated the incorporation of temporal dimension in graphs.

References

10.1 J.D. Foley, A. van Dam, S.K. Feiner, J.F. Hughes: *Computer Graphics: Principles and Practice*, 2nd edn. (Addison Wesley, Boston 1995)

10.2 M. Mäntylä: *An Introduction to Solid Modeling. Principles of Computer Science* (Computer Science, New York 1988)

10.3 M.E. Mortenson: *Geometric Modelling* (Wiley, Chichester 1997)

10.4 J. Herring: The OpenGIS Abstract Specification, Topic 1: Feature Geometry (ISO 19107 Spatial schema): OGC Document Number 01-101 (2001)

10.5 S. Cox, A. Cuthbert, R. Lake, R. Martell: OpenGIS Geography Markup Language (GML): Open GIS Doc. No. 02-009 (2002)

10.6 R. Lake, D. Burggraf, M. Trninic, L. Rae: *Geography Mark-Up Language: Foundation for the Geo Web* (Wiley, Chichester 2004)

10.7 C. Portele: OpenGIS Geography Markup Language (GML) encoding standard, version 3.2.1: OGC Doc. No. 07-036 (2007)

10.8 M.A. Armstrong: *Basic Topology* (Springer, New York 1997)

10.9 H.-J. Bungartz, M. Griebel, C. Zenger: *Einführung in die Computergraphik. Grundlagen, Geometrische Modellierung, Algorithmen* (Vieweg, Wiesbaden 2002), in German

10.10 G. Gröger, T.H. Kolbe, A. Czerwinski, C. Nagel: OpenGIS City Geography Markup Language (CityGML) Encoding Standard, Version 1.0.0. OGC Doc. No. 08-007r1 (2008)

10.11 P.A. Vretanos: OpenGIS Filter Encoding Implementation Specification (2005)

10.12 Oracle Cooperation: *Oracle Spatial Developers Guide 11g release 1 (11.1)* (2009)

10.13 P. Alexandroff: *Elementary Concepts of Topology* (Dover, New York 1961)

10.14 A. Hatcher: *Algebraic Topology*, 10th edn. (Cambridge Univ. Press, Cambridge 2001)

10.15 M.J. Egenhofer, R.D. Franzosa: Point-set topological spatial relations, Int. J. Geogr. Inf. Sci. **5**(2), 161–174 (1991)

10.16 A.G. Cohn, B. Bennet, J. Gooday, M.M. Gotts: Qualitative spatial representation and reasoning with the region connection calculus, Geoinformatica **1**, 275–316 (1997)

10.17 M.J. Egenhofer, J.R. Herring: Categorizing binary topological relationships between regions, lines and points in geographic databases, Technical Report, Department of Surveying Engineering, University of Maine, Orono (1991)

10.18 E. Clementini, P. Di Felice: A comparison of methods for representing topological relationships, Inform. Sci. **3**(3), 149–178 (1995)

10.19 S. Zlatanova: *3D GIS for Urban Development*. International Institute for Aerospace Survey and Earth Sciences; Institute for Computer Graphics and Vision Graz University of Technology (Enschede, Graz 2000)

10.20 J.R. Herring: OpenGIS Implementation Specification for Geographic information – Simple feature access – Part 1: Common architecture: Version: 1.2.0, OGC Doc. No OGC 06-103r3 (2006)

10.21 L. Plümer, G. Gröger: Achieving integrity in geographic information systems-maps and nested maps, Geoinformatica **1**(4), 345–367 (1997)

10.22 G. Gröger: *Modellierung raumbezogener Objekte und Datenintegrität in GIS* (Wichmann, Heidelberg 2000), in German

10.23 M. Molenaar: Formal data structures, object dynamics and consistency rules. In: *Digital Photogrammetric Systems*, ed. by C. Ebner, D. Fritsch, C. Heipke (Wichmann, Heidelberg 1991)

10.24 M. Molenaar: *An Introduction to the Theory of Spatial Object Modelling for GIS* (Taylor & Francis, London 1998)

10.25 M.J. Egenhofer, A.U. Frank, J.P. Jackson: A topological data model for spatial databases. In: *Design and Implementation of Large Spatial Databases, First Symposium SSD '89 Santa Barbara* 1989, Proceedings (Springer, Berlin Heidelberg 1990)

10.26 ISO: ISO 14825:2004 Intelligent transport systems – Geographic Data Files (GDF) – Overall data specification (ISO, Geneva 2004)

10.27 T. Becker, C. Nagel, T.H. Kolbe: A multilayered space-event model for navigation in indoor spaces. In: *3D Geo-Information Sciences*, ed. by J. Lee, S. Zlatanova (Springer, Berlin Heidelberg 2009) pp. 61–78

10.28 S. Zlatanova, A. Abdul Rahman, W. Shi: Topological models and frameworks for 3D spatial objects, Comput. Geosci. **30**, 419–428 (2004)

10.29 C. Ellul, M. Haklay: Requirements for topology in 3D GIS, Transactions GIS **10**(2), 157–175 (2004)

10.30 M. Molenaar: A topology for 3D vector maps, ITC Journal **1**, 25–33 (1992)

10.31 S. Flick: *Konzeption eines adaptiven Frameworks für 3D-Geo-Informationssysteme* (Fraunhofer IRB, Stuttgart 1999), in German

10.32 V. Coors: 3D-GIS in networking environments, Comput. Environ. Urban Syst. **27**, 45–357 (2003)

10.33 M. Pilouk: Integrated modelling for 3D GIS. Dissertation. ITC Publication **40** (ITC, Enschede 1994)

10.34 F. Penninga, P. Oosterom: A compact topological DBMS data structure for 3D topography. In: *The European Information Society: Leading the Way with Geo-information*, ed. by S.I. Fabrikant, M. Wachowicz (Springer, Berlin Heidelberg 2007) pp. 455–471

10.35 S. Pigot: A topological model for a 3D spatial information system. In: Proc. 5th Int. Symp. Spat. Data Handl., Charleston (1992), pp. 344–360

10.36 G. Gröger, L. Plümer: Provably correct and complete transaction rules for updating 3D city models, Geoinformatica, online first, doi://10.1007/s10707-011-0127-6 (2011)

10.37 G. Chartrand: The Königsberg bridge problem: An introduction to Eulerian graphs. In: *Introductory Graph Theory*, ed. by G. Chartrand (Dover, New York 1985)

10.38 F. Harary: *Graph Theory* (Addison-Wesley, Reading 1994) pp. 1–2

10.39 N.L. Biggs, E.K. Lloyd, R.J. Wilson: *Graph Theory 1736–1936* (Oxford Univ. Press, Oxford 1976)

10.40 B. Bollobás: *Graph Theory: An Introductory Course* (Springer, Berlin Heidelberg 1979) p. 12

10.41 H. Steinhaus: *Mathematical Snapshots*, 3rd edn. (Dover, New York 1999) pp. 256–259

10.42 N.L. Biggs, E.K. Lloyd, R.J. Wilson: *Graph Theory* (Oxford Univ. Press, Oxford 1998)

10.43 R.J. Wilson: *Introduction to Graph Theory* (Longman, New York 1985)

10.44 R. Hartsfield: *Pearls in Graph Theory* (Academic, San Diego 1990)

10.45 J. Edmonds, E.L. Johnson: Matching, Euler tours and Chinese postman, Math. Program. **5**, 88–124 (1973)

10.46 N. Christofides: The optimum traversal of a graph, Int. J. Manag. Sci. **1**(6), 12 (1973)

10.47 J.R. Evans, E. Minieka: *Optimization Algorithms for Networks and Graphs* (Dekker, New York 1992)

10.48 S. Shekhar, S. Chawla: *Spatial Databases: A Tour* (Prentice Hall, Englewood Cliffs 2003)

10.49 R.J. Wilson, L.M. Beineke (Eds.): *Applications of Graph Theory* (Academic, London New York 1979)

10.50 H. Fleischner: (Some of) the many uses of Eulerian graphs in graph theory, Discret. Math. **230**, 23–43 (2001)