

Original Lecture #13: 7 November 1991
Topics: Boundary Representations of Solids
Scribe: Jim Stewart

1 Solid Modeling

In this class we will concentrate on *boundary representations* of solids. Certain volume-based methods such as BSP trees will be presented; others, such as oct-tree decompositions, will not be discussed. The study of boundary representations involves focusing on how the boundary elements are interconnected, i.e., the *topology*, and also the separation of the topology from the geometry. Fig. 1 shows two different planar geometries that are topologically equivalent.

The boundaries of solids are, by definition, subdivisions of manifolds into three finite collections of disjoint parts, the *vertices*, the *edges* and the *faces*; these are the *elements* of a subdivision. Topologically, vertices are 0-dimensional, edges are 1-dimensional, and faces are 2-dimensional. A cube, for example, can be thought of as a collection of 8 vertices, 12 edges and 6 faces. Fig. 2 shows a subdivision which is homeomorphic to a cube. Two subdivisions which are *not* homeomorphic are shown in Fig. 3 — G2 has a face with 6 edges while G1 does not (even though as graphs on the vertices and edges, they are isomorphic).

The collection of all edges and vertices of a subdivision S constitutes an undirected graph, the *graph* of S . In addition to a simple graph, the *ordering* of edges around a face, edges incident to a vertex, etc., must be incorporated for a complete description of the topology of a manifold (see Fig. 3).

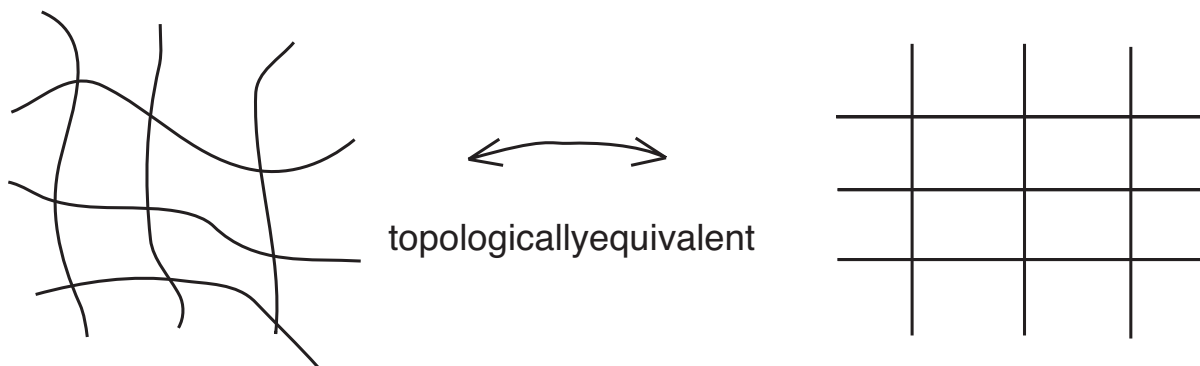


Figure 1: Topologically equivalent planar geometries

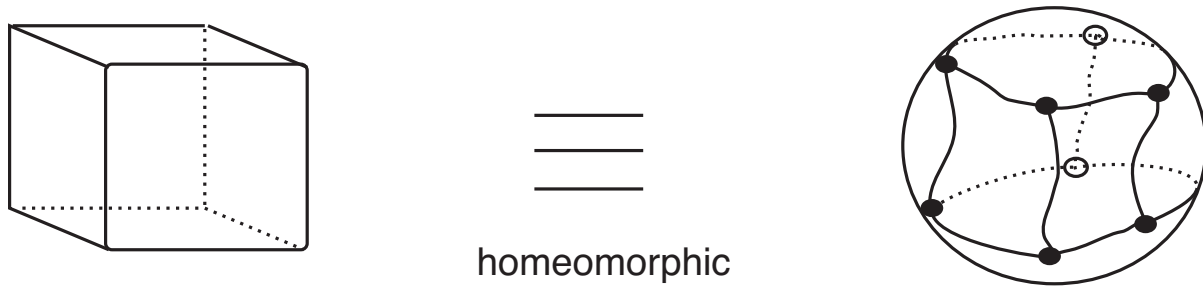


Figure 2: Subdivision of a sphere which is homeomorphic to a cube

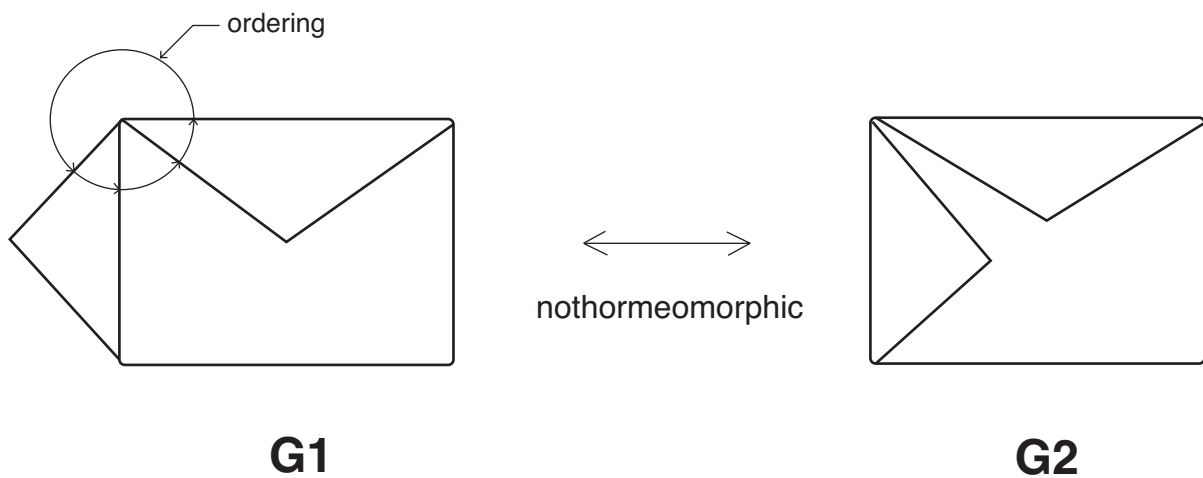


Figure 3: Subdivisions which are *not* homeomorphic. G1 shows the counterclockwise ordering of edges incident to a vertex.

2 Type of Surfaces Considered

In this class we consider only compact orientable 2-manifolds with no boundary. A *2-manifold* (i.e., a two-dimensional manifold) is a topological space with the property that every point has an open neighborhood which is a disk, i.e., a subspace homeomorphic to the open circle of unit radius.

On any disk D of a manifold there are exactly two ways of defining a local “clockwise” sense of rotation; these are called the two possible *orientations on D* . An *oriented element* of a subdivision P is an element x of P together with an orientation of a disk containing x .

A *compact 2-manifold* is one that is *not* infinite; more formally, it is one in which every infinite sequence has an accumulation point in the manifold. Thus the manifolds we will consider will also be surfaces which close upon themselves, i.e., have no boundary. The sphere, the torus and the projective plane are all compact 2-manifolds with no boundary; the disk, the line segment and the whole plane are not. In addition, we insist that every face be a simple disk, without “handles” or “holes”. Finally, as we will see, combinatorial isomorphism of the

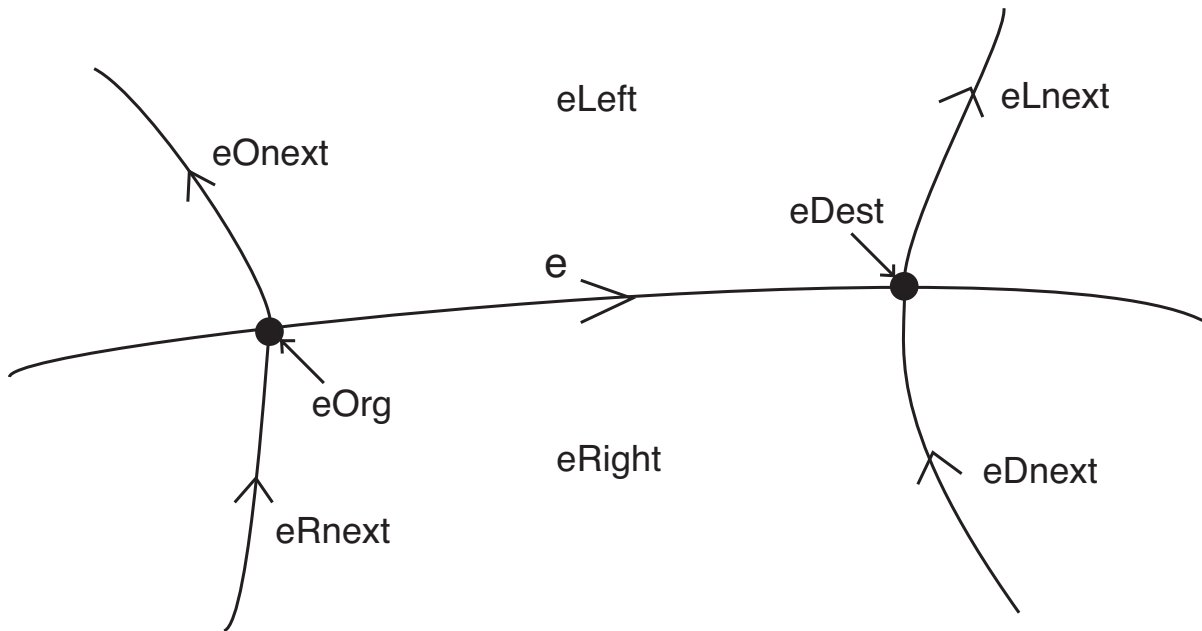


Figure 4: The edge functions

vertex-edge-face incidence structures is equivalent to homeomorphism of manifolds for two dimensions only; this equivalence does not extend to 3-manifolds and above.

3 Edge Representations

A *directed edge* of a subdivision P is an edge of P together with a direction along it. The *dual* of a planar graph G is a graph G^* obtained from G by interchanging vertices and faces while preserving the incidence relationships. Since directions and orientations can be chosen independently, for every edge of a subdivision there are four directed, oriented edges, giving a total of eight variants of an edge — four for each edge in G and four for the corresponding dual edges in G^* . In this course we will deal only with orientable manifolds. Because of this we will fix an orientation for our manifold and therefore ignore the possibility of changing the orientation on an edge. This will give us a total of four variants for an edge.

For any oriented and directed edge e we can define unambiguously its vertex of origin, $eOrg$, its vertex of destination, $eDest$, its left face, $eLeft$, and its right face, $eRight$. We can also define the next edge with the same origin, $eOnext$, as the one immediately following e (in a counterclockwise ordering). Similarly, we can define the next counterclockwise edge with the same left face, denoted by $eLnext$, as being the first edge encountered after e when moving along the boundary of the face $F = eLnext$ in the sense determined by the orientation of F . In an analogous fashion we can define the edges $eDnext$ and $eRnext$. These are all shown in Fig. 4.

A given edge is maintained in four natural orderings — around the origin and destination

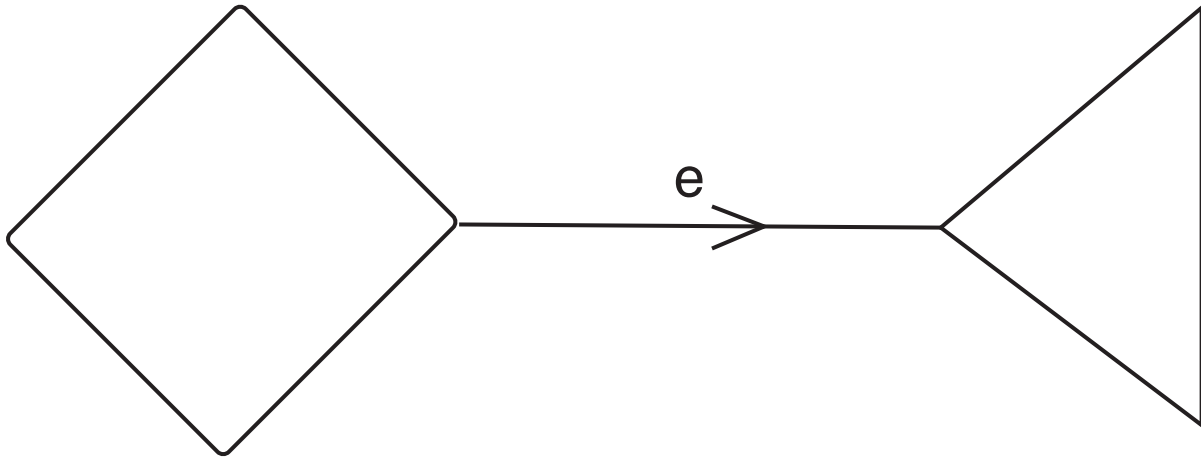


Figure 5: The edge e is used twice in both the left face ordering and the right face ordering

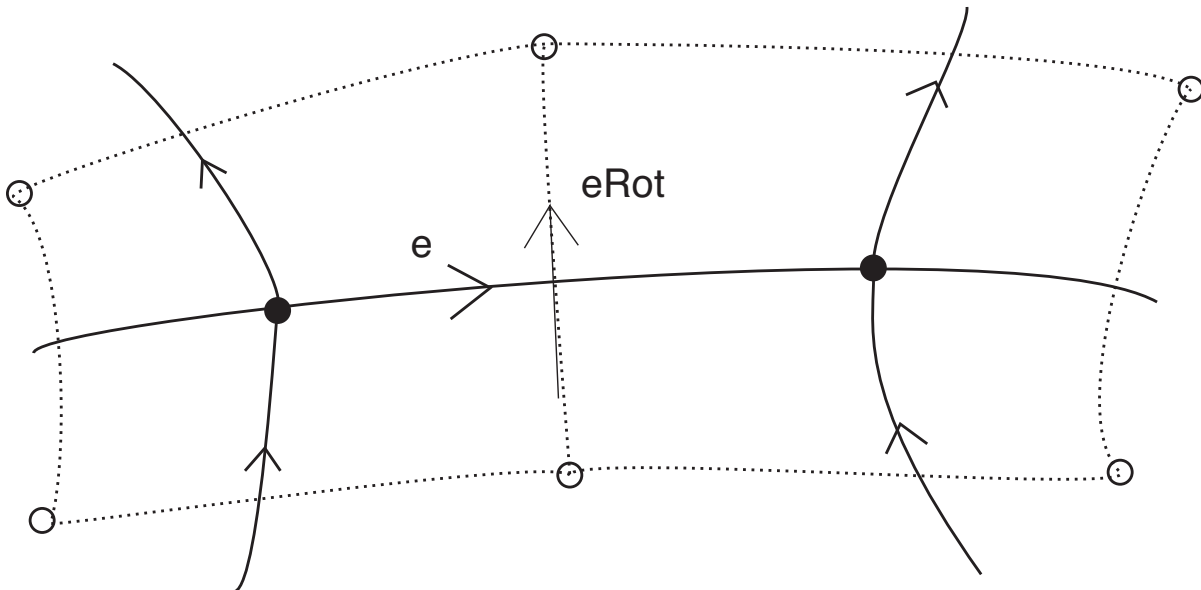


Figure 6: The graph of Fig. 4 with its strict dual (dotted lines)

vertices, and around the left and right faces. Fig. 5 shows a situation in which the edge labeled e is used twice in each face ordering.

The edge $eRot$ is called the rotated version of e ; it is the dual of e , directed from $eRight$ to $eLeft$ and oriented so that moving counterclockwise around the right face of e corresponds to moving counterclockwise around the origin of $eRot$. If the primal and dual subdivisions are superimposed, then $eRot$ can be viewed as e “rotated 90° counterclockwise” around the crossing point. The graph of Fig. 4 along with its strict dual is shown in Fig. 6.

4 Edge Algebra

The effect of the function Rot on a canonical edge e pointing to the right is seen as follows:

$$\begin{array}{c}
 e \rightarrow \\
 eRot \uparrow \\
 eRot^2 \leftarrow \\
 eRot^3 \downarrow \\
 eRot^4 = e
 \end{array}$$

The Rot and $Onext$ functions can be used to find the following edges:

$$\begin{aligned}
 eLnext &= eRot^{-1}OnextRot \\
 eRnext &= eRotOnextRot^{-1} \\
 eDnext &= eRot^2OnextRot^2
 \end{aligned}$$

In 2- d the following abstract combinatorial structure topologically represents manifolds: we have a set of edges ε partitioned into two corresponding groups, the primal and the dual edges. We also have two operators, $Onext$ and Rot , satisfying the following axioms.

1. $eRot^4 = e$
2. $eRotOnextRotOnext = e$
3. $eRot^2 \neq e$
4. e is primal if and only if $eRot$ is dual
5. e is primal if and only if $eOnext$ is primal

5 Quad-Edge Data Structure

The *quad-edge data structure* is a natural computer implementation of the corresponding edge algebra. An edge e is represented in the data structure by one *edge record* e , divided into four parts $e[0]$, $e[1]$, $e[2]$, $e[3]$. Part $e[r]$ corresponds to the edge e_0Rot^r , where e_0 is an arbitrary *canonical representation* (i.e., specified direction and orientation) of the edge. See Fig. 7a.

Each part $e[r]$ of an edge record contains two fields, `Data` and `Next`. The `Data` field is used to hold geometrical and other non-topological information about the edge e_0Rot^r . This field neither affects nor is affected by topological operations, therefore its contents and format are entirely dependent on the application. The `Next` field of $e[r]$ contains pointers to the edge $e[r]Onext$; its four variants correspond to $eOnext$, $eRotOnext$, $eRot^2Onext$ and $eRot^3Onext$. Fig. 7 illustrates a portion of a subdivision and its quad-edge data structure.

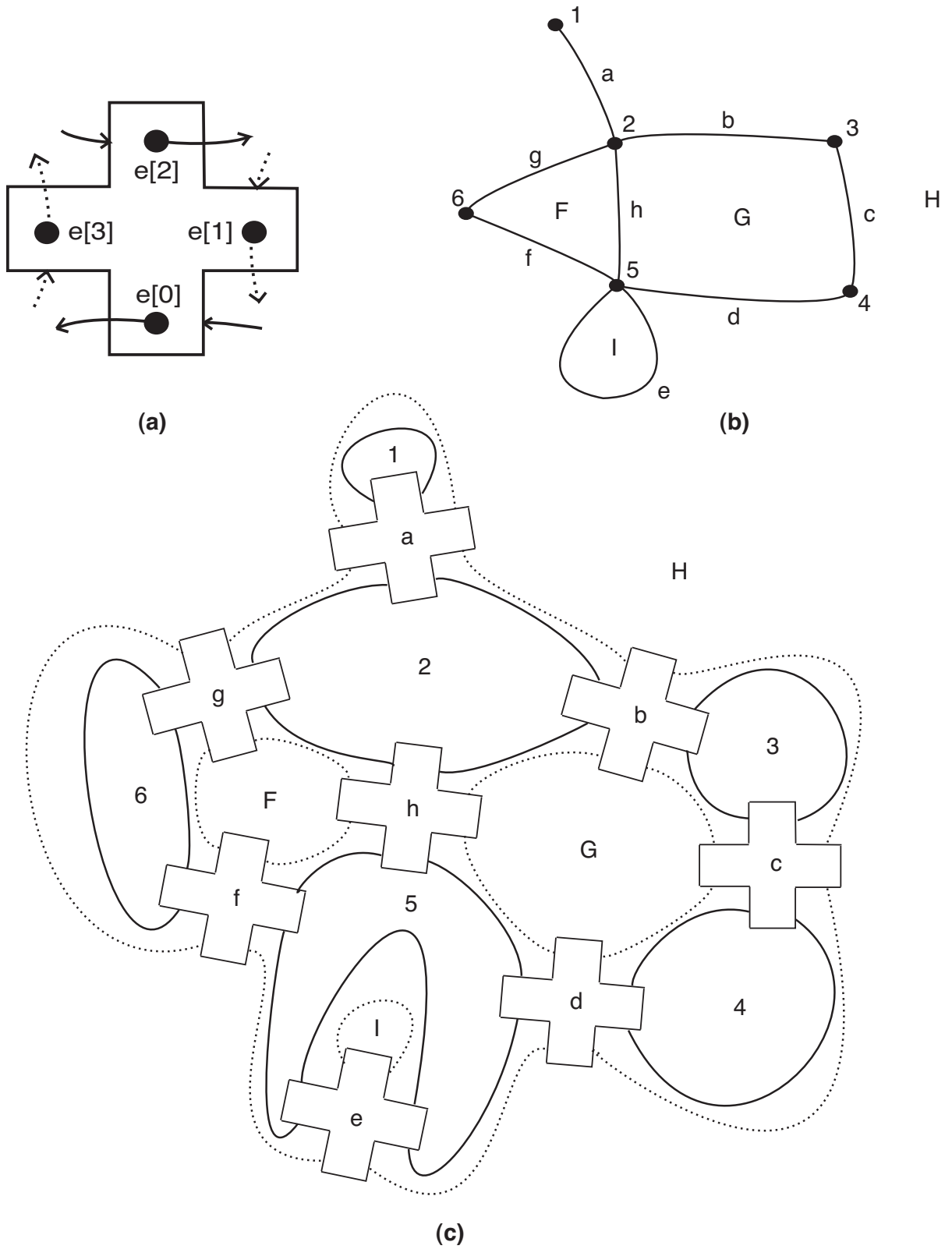


Figure 7: (a) Edge record showing Next links. (b) A subdivision of the sphere. (c) The data structure for the subdivision (b).

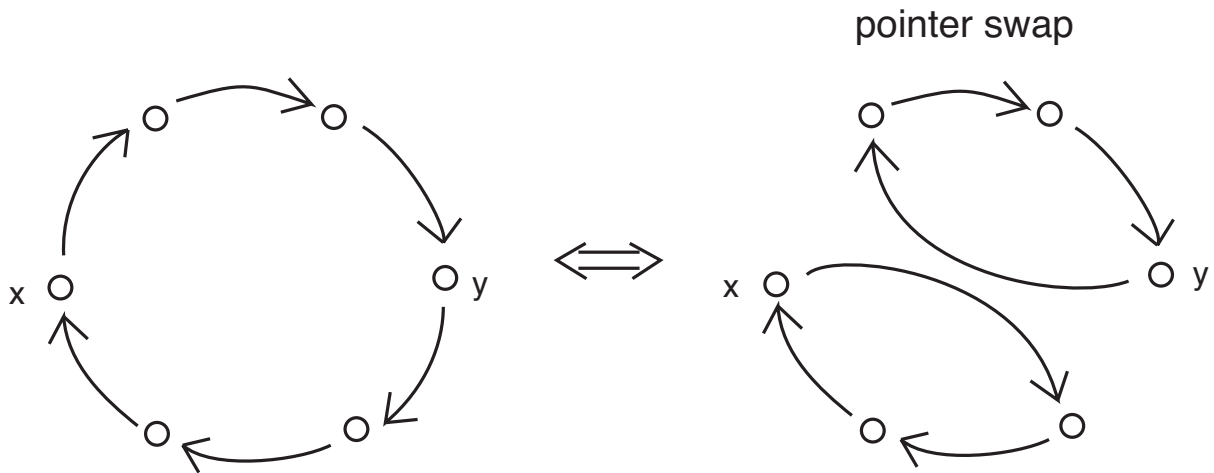


Figure 8: Circular pointer structure

6 Topological Operators

The construction and modification of arbitrary diagrams can be accomplished by two basic topological operators. The first operator is denoted by $e \leftarrow \text{MakeEdge}[\]$. It takes no parameters, and returns an edge e of a newly created data structure representing a subdivision of the sphere. Apart from orientation and direction, e will be the only edge of the subdivision and will not be a loop. To construct a loop, we may use $e \leftarrow \text{MakeEdge}[\].\text{Rot}$.

The second operator is denoted by $\text{Splice}[a, b]$; This operator relies on the following useful property of collections of circular lists (see Fig. 8): if two pointers that lie in the same cycle are swapped, then two cycles result; likewise, if two pointers that lie in different cycles are swapped, one cycle is formed (i.e., the cycles are connected). $\text{Splice}[a, b]$ will be further described in the next lecture.