A Dualizable Representation for Imbedded Graphs

Linda L. Deneen
Gary M. Shute
Clark D. Thomborson (a.k.a. Thompson)*

Computer Science Department
University of Minnesota

ABSTRACT

A new representation is described for graphs imbedded in pseudomanifolds. The graphs may have loops and parallel edges, and they may be disconnected. It is shown that our representation has a well–defined dual that naturally extends the definition of surface duality. This representation is easily transformed to a data structure with a set of operators for modifying a graph, a set of query operators, and a set of navigation operators. Three variants of the data structure are described. For a graph with n vertices, the first provides O(1)–time updates and O(n)–time queries. The second provides O(n)–time updates and O(1)–time queries. The third, a tradeoff between the first two, provides O(log n)–time updates and queries. The space used in all variants is linear in the size of the graph.

1. Introduction

In this paper we extend several known representations for imbedded graphs [1, 4, 16, 18] to allow for disconnected components, loops, parallel edges, and isolated vertices. Our representation allows for natural imbeddings into pseudomanifolds. Most noteworthy is that our representation gives rise naturally to a well–defined dual graph. Our dual can be obtained directly from the representation, and it has a natural imbedding that generalizes the usual notion of a dual imbedding. Our presentation of a graph in terms of half–edges is succinct and more general than previous presentations of imbedded graphs.

* Supported by the National Science Foundation, through its Design, Tools and Test Program under grant number MIP 84–06139.

We also develop a set of primitive operators and three data structure implementations for use in representing and manipulating graphs. This work extends previous work as well. Isolated vertices are not allowed in a quadedge structure [7], and they are tolerated as special cases at best in a DCEL or winged–edge representation [11]. The quadedge structure has a second deficiency: it does not allow one to imbed a component of a disconnected graph inside a face of some other component of that graph.

Our primitive operators are divided into three groups: modification operators, navigation operators, and query operators. Six special modification operators can be viewed as extensions of Guibas and Stolfi's splice operator [7]; we need six instead of one because of the added flexibility of our representation.

Our paper is organized as follows. Section 2 describes the basic graph theory background and gives a history of previous work that our work generalizes. Section 3 gives the mathematical description of our graph representation for the orientable case. In section 4 we describe our abstract operators, and in section 5 we describe our three implementations, again for the orientable case. The representation is extended to the nonorientable case in section 6. Conclusions and open problems are discussed in section 7.

2. Fundamentals

Graph theorists have expended much effort studying the imbeddings of graphs in geometric spaces. It is common to study imbeddings in spaces that are locally similar to $\mathcal{R}^2$. Many investigators [1, 4, 6, 12, 14, 16, 17, 18] have studied imbeddings in 2–*manifolds*, which are connected compact topological spaces with the property that each point has a neighborhood homeomorphic to the open unit disk in $\mathcal{R}^2$. These 2–manifolds can be either orientable or nonorientable. In the following we will be concerned only with spaces that are locally 2–dimensional so that we will drop the dimensional prefix from our terminology.

We will investigate imbeddings into pseudomanifolds. Our definition of a pseudomanifold will diverge from White's pseudosurface in that our pseudomanifolds need not be connected.

Definition A pseudomanifold is a compact topological space M in which each point x has a neighborhood that is homeomorphic to a finite number of open disks identified at x.

The number of disks in the neighborhood of a point x is called the *degree* of x. If the degree of x is greater than one, then x is called a *singular point* of M.

Following Biggs [1], we define a *graph* as follows:

Definition (Biggs) A graph is an ordered quadruple (E, V, λ, τ) where E and V are finite sets, λ is a surjective function from E to V, and τ is an involution on E.

We call the elements of E *half–edges* and the elements of V *vertices* of G. If the half–edge e is not fixed by τ, then we may interpret e as a directed edge with $\lambda(e)$ as the initial vertex of e and $\tau(e)$ as the edge directed opposite to e. In keeping with this interpretation, we say that e is *incident* on v if $\lambda(e) = v$. We defer the interpretation of half–edges that are fixed by τ until later. We note that this definition of a graph admits loops, parallel edges, and disconnected graphs.

An imbedding of a graph in a topological space identifies the vertices of the graph with distinct points in the space and the edges with homeomorphic images of line segments. The image of an edge must have the images of its vertices as endpoints, and images of edges may not intersect except at the endpoints. If a graph G is imbedded in a pseudomanifold M, the components of $M \setminus G$ are the *faces* of the imbedding, and the imbedding is a *2–cell imbedding* if the faces are all homeomorphic to the open unit disc.

In classifying imbeddings of graphs most authors use variants of the *permutation technique* or the *rotation system technique*. White [17] attributes this technique to Edmonds [4] and calls it *Edmonds' permutation technique*. Gross and Tucker [6] call it the method of *rotation systems* and jointly credit Heffter (1891) and Edmonds (1960), hypothesizing that Edmonds rediscovered the technique without being aware of Heffter's work. Both White and Gross and Tucker credit Youngs [18] with making Edmonds' work accessible. The central result of this work is that there is a one–to–one correspondence between the orientable 2–cell imbeddings of G in manifolds and certain permutations of the directed edges of G called *rotations*. The rotations specify the rotational order of the imbedded edges around a vertex.

Given a 2–cell imbedding of a connected graph G in a pseudomanifold M, one can define the *dual graph* G^* like this: choose one vertex for G^* for each face of G's imbedding, and for each edge e in G, add an edge e^* to G^* connecting the vertices corresponding to the two faces on either side of e. The use of the term *dual* refers to the fact that the dual graph of the dual graph of G is G again. In studying representations of graphs and their imbeddings, we consider two questions: does a given imbedded graph have a well–defined dual graph, and if so, does the particular graph representation make it easy to find the dual graph? Edmonds [5] considers only 2–cell imbeddings of connected graphs in spheres; Biggs [1] extends this to manifolds with his representation, which gives rise to the dual graph in a natural way. Tutte [16] allows imbeddings in either orientable or nonorientable manifolds, and he allows disconnected components in his graphs, but his dual graph representation requires that each separate component of the graph be imbedded in a distinct manifold; his duals are found component–wise. In our presentation, we will extend all of the above results to include graphs that are not necessarily connected and to imbeddings on pseudomanifolds.

3. An Extended Graph Representation

We present a new graph representation that extends the previously discussed work in several ways. First of all, with this new representation, we can represent graphs that may be disconnected or that may contain isolated vertices. Secondly, these graphs may be imbedded on pseudomanifolds. Thirdly, in an imbedding of a disconnected graph, we can specify clearly whether or not a component lies inside a face of another component. Finally, all of our graphs have well–defined duals, and the dual graph of a graph in our representation is easily obtained. This is a significant advance in duality theory, since duals were previously defined only for connected graphs or in the case of Tutte, disconnected graphs with separate components on separate manifolds.

Edmonds [4, 18] uses the notion of a *rotation* on a graph G to characterize 2–cell imbeddings of G in compact manifolds. Our definition of a rotation follows Biggs [1].

Definition (Biggs) A rotation on a graph $G = (E, V, \lambda, \tau)$ is a permutation O on E such that $\lambda O = \lambda$.

The definition insures that for any vertex v, O restricts to a permutation on the half–edges incident on V. We call each cycle of O an *origin* and loosely regard each vertex as a set of origins. If each vertex contains only one origin then O is said to be *smooth*. For the imbedding construction of Edmonds, rotations are required to be smooth with the cycles inducing a circular ordering on the half–edges incident on a vertex.

We eliminate the smoothness condition by considering imbeddings in pseudomanifolds. If a vertex v contains k origins, then in the imbedding v must be a point of degree k. The neighborhood of a singular point can be visualized as a set of cones sharing a common apex. When a vertex is located at a singular point, the half–edges incident on that vertex are not cyclically ordered. Instead, the ordering can be described as a not necessarily smooth rotation mapping a half–edge onto the next half–edge counterclockwise on the same cone. In the limiting case of a vertex with one half–edge per cone, the edges are completely unordered. Another interpretation of a vertex located at a singular point is that its incident edges are only partially imbedded. This notion of partial imbedding is useful when representing the intermediate results of a graph–imbedding algorithm, such as Hopcroft and Tarjan's linear–time planarity tester [8].

If a cone about v contains no edges that connect to v we say that the cone is a *blank cone*. If a half–edge e is fixed by τ then we will require that e is also fixed by

O. Then e is by itself an origin. We interpret the origin e as an *isolated* origin so that the cone corresponding to e is a blank cone.

Edmonds begins by defining a permutation P on E by $P = O\tau$. We may view P as dual to O, with cycles of P determining a cyclic ordering of half–edges around a boundary of a face in the imbedding of G. We call the cycles of P *panes*, so that panes are dual to origins. To construct a theory of imbeddings in which every imbedding has a dual, we must also dualize the notion of a vertex. Thus to specify an imbedding we must have a finite set W, which is dual to V, and a function μ from E to W, which is dual to λ. To preserve duality, we require that μ be surjective and that $\mu P = \mu$. We call the elements of W *windows*. In an imbedding of a graph G in a space M, windows correspond to the connected components of $M \setminus G$ and panes correspond to the connected components of boundaries of windows. We may loosely regard each window as a set of panes. Analogous to the terminology for vertices, we say that a half–edge e is *incident* on a window w if $\mu(e) = w$.

We are now ready to extend the concept of a rotation to allow imbeddings of graphs in orientable pseudomanifolds.

Definition A generalized rotation system on a graph $G = (E, V, \lambda, \tau)$ consists of a rotation O that fixes each half–edge that is fixed by τ, a finite set W, and a surjective function μ from E to W satisfying $\mu O\tau = \mu$.

When discussing a generalized rotation system on a graph we will use the terminology of vertices, origins, windows, and panes as introduced above.

Given a graph G and a generalized rotation system for G, we define an imbedding of G in a pseudomanifold M as follows. As in Edmonds' construction, we begin by defining a permutation P on E by $P = O\tau$. Then $\mu P = \mu$ so for any window w, P restricts to a permutation of the half–edges incident on w. Thus the concept of a pane of w is meaningful. For each window w we form a surface M_w by removing open disks with disjoint boundaries from a sphere, with one disk removed for each pane in w. Since $P = O\tau$, and O fixes the fixed half–edges of τ, it follows that if a half–edge e that is fixed by τ occurs in a pane then e must in fact be the only half–edge in that pane. In this case we retract the boundary of the hole for that pane to a point and label the point by the vertex $\lambda(e)$. Otherwise, we divide the boundary of the hole for the pane into segments, with one segment labeled by each half–edge in the origin. The segments are ordered so that the segment labeled by P(e) appears counterclockwise around the hole from the segment labeled by e, as seen from the outside of the sphere. For the segment labeled by e, label the clockwise endpoint by the vertex $\lambda(e)$.

Now, we paste the surfaces M_w together to form M' by identifying each boundary segment labeled e to the segment labeled $\tau(e)$, provided $e \neq \tau(e)$. Boundary segments are identified so that the segments labeled e and $\tau(e)$ are oppositely directed. Finally, we form M from M' by identifying all sets of points labeled by the same vertex.

It is straightforward to show that M is an orientable pseudomanifold and that the constructed imbedding satisfies the following definition:

Definition A simple imbedding of a grpah G in a pseudomanifold M is an imbedding such that

(i) Each component of M contains at least one vertex of G,

(ii) Each singular point of M is a vertex of G,

(iii) The number of blank cones about each vertex v of G is equal to the number of half–edges e that are incident on v and fixed by τ, and

(iv) Each connected component of $M \setminus G$ is a sphere with a finite number of holes whose boundaries are disjoint.

Moreover, the imbedding we have constructed is unique up to isomorphism. This follows from the fact that given a sphere S with a finite number of open holes and a permutation σ of holes, there is a homeomorphism of S onto itself that permutes the holes as does σ.

Finally, if we are given an imbedding of a graph G in an orientable pseudomanifold M, then guided by the interpretations given above, we can recover W, μ, and P. More precisely, let W be the set of components of $M \setminus G$. We associate each blank cone at a vertex v with a half–edge e that is incident on v and fixed by τ. We set $P(e) = e$ and $\mu(e) = w$, where w is the component that contains the cone. For other half–edges, e, $\mu(e)$ is the component on the right of e as a directed edge, and P(e) is the next half–edge walking in the direction of e around $\mu(e)$. Defining W, μ, and P in this way, we have $\mu P(e) = \mu(e)$ and $\lambda P\tau(e) = \lambda(e)$, for all half–edges e. See Figure 1. We now recover O by computing $O = P\tau$. Since $\lambda O = \lambda P\tau = \lambda$, we are assured that O is a rotation. Since τ is an involution, $P = O\tau$, and it follows that $\mu O\tau = \mu$. We have thus shown

Theorem 1 *Given a graph G, there is a one–to–one correspondence between generalized rotation systems on G and simple imbeddings of G in orientable pseudomanifolds.*

We can present a graph, together with a simple imbedding in an orientable pseudomanifold, in a way that makes duality more apparent. We do this by specifying the set E of half–edges, together with two sets, V and W, of permutations. There is one permutation in V for each vertex v, obtained by restricting P to the half–edges incident on v. Similarly, W is obtained from O. When writing the description of a graph, the set E may be omitted if the permutations in V and W are given in cycle notation with 1–cycles for fixed half–edges. We will use this notation in our illustrations.

The dual of an imbedded graph is then obtained by interchanging V and W. Clearly, every simple imbedding of a graph in an orientable pseudomanifold has a uniquely defined dual. The duality operator is an involution and in the case of planar imbeddings, our dual is the usual planar graph dual except for a reversal of orientation.

For example, the graphs in Figures 2 and 3 are duals of one another. A more interesting pair of dual graphs is given in Figures 4 and 5. In Figure 4 the dotted line joins two origins that belong to the same vertex. This vertex is mapped onto the singular point of a sphere whose north and south poles are identified. In the dual graph this two–origin vertex becomes a window with two panes: as drawn in Figure 5, these are the innermost pane and the exterior pane. This window is thus isomorphic to a cylinder with circular panes on opposite ends. The cylinder forms the hole through the torus on which the dual graph is imbedded.

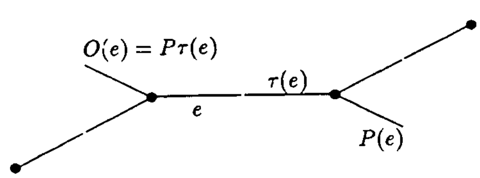
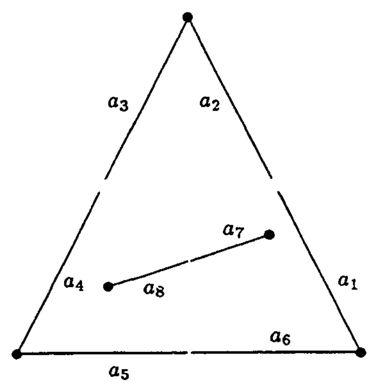


Figure 1: A portion of a graph, showing that $\lambda O(e) = \lambda(e)$.

4. Abstract Operators for the Extended Graph Representation

In this section we describe a set of abstract operators to be used to express the extended graph representation described in the previous section. These operators are part of a computer program to manipulate graphs; three implementations are described in the next section. The operators in this section can be grouped into two classes: navigation operators and modification operators.

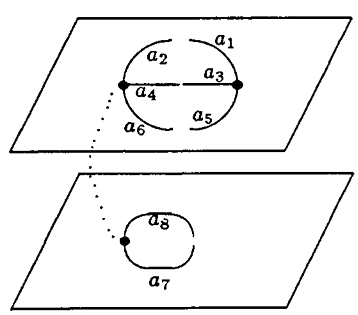
There are five navigation operators. *Onext* maps a half–edge to the next half–edge in the cyclic order around an origin. *Onext* corresponds to the permutation O of the previous section. The operator *Oprev* corresponds to O^{-1}. *Pnext* corresponds to the permutation P in the previous section. The operator *Pprev* corresponds to P^{-1}. Our *Onext* is the same as Guibas and Stolfi's *Onext* [7], while *Pnext* is the same as their *Rprev*. *Mnext* maps a half–edge a to the opposing half–edge in its edge pair if it has one; otherwise a represents an isolated vertex and *Mnext*(a) = a. *Mnext* corresponds to the involution τ in the previous section.



$$V = \{(a_1a_6), (a_2a_3), (a_5a_4), (a_7), (a_8)\}$$
$$W = \{(a_1a_3a_5), (a_6a_4a_2), (a_7a_8)\}$$

Figure 2: A graph with a 2–pane window.

There are nine modification operators. *MakeNullg* simply creates an empty graph. *Addh* returns a half–edge that is *Onext*, *Pnext*, and *Mnext* to itself. It is thus an isolated vertex imbedded in its own sphere. *Delh(a)* removes a from the graph. Before we can apply *Delh* to a, however, we must ensure that a is an isolated vertex; it must not be linked in any way to any other half–edge in the graph.



$$V = \{(a_1a_3a_5), (a_6a_4a_2), (a_7a_8)\}$$
$$W = \{(a_1a_6), (a_2a_3), (a_5a_4), (a_7), (a_8)\}$$

Figure 3: A graph with a 2–origin vertex.

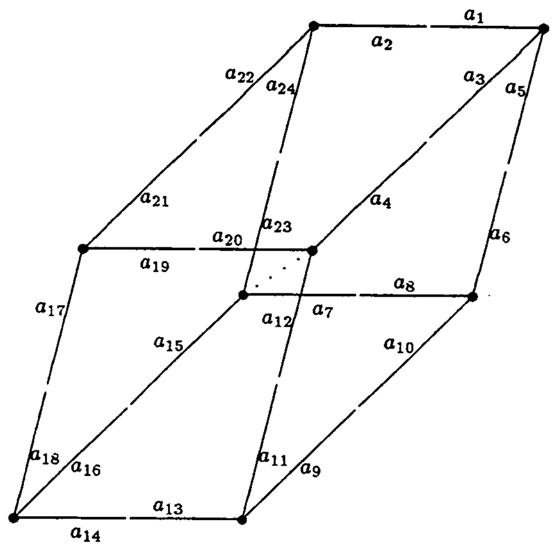
The remaining six modification operators come in pairs, corresponding to the vertex–face duality of our presentation. The first pair of operators, *Osplice* and *Psplice* affect the origin and pane cycles, respectively. Like Guibas and Stolfi's quadedge splice operator [7], our *Osplice* and *Psplice* are their own inverses. Given two half–edges a_1 and a_2 lying in distinct origins, *Osplice*(a_1, a_2) merges the two origins. This is done by writing a_1's origin as a cycle starting with a_1 and writing a_2's origin as a cycle starting with a_2 and then concatenating the two cycles. If the operands of *Osplice* are distinct half–edges with a common origin, this origin will be split in two. We require that the arguments of *Osplice* be in the same vertex. The *Psplice* operator behaves similarly, with its arguments required to be in the same window.

The remaining modification operators modify the vertex and window structures. Given two half–edges a_1 and a_2 in distinct vertices, *Vmerge* (a_1, a_2) merges a_1's vertex with a_2's vertex. *Vsplit* (a) splits the vertex containing the half–edge a into two vertices: a's origin becomes the only member of a new vertex, and the remaining origins stay with the original vertex. Similarly, *Wmerge* merges two windows and *Wsplit* splits a single window in two.

As an example, we show in Figure 6 how we can add the new edge with half–edges c and d to an already existing graph, using the operators described above. Note that at intermediate stages of the construction the medial function τ is not always an involution. We are currently developing higher level operators, built on top of our primitive operators, that will preserve the property of being a graph in the sense of Biggs' definition.

5. Three Implementations for the Orientable Extended Graph Representation

For these three implementations, we choose to store panes and origins in circular linked lists, thus fitting the order of a cycle making up a pane or origin into the data structure. The problem of storing windows and vertices is slightly more complex. A window is represented by linking together a distinguished half–edge, called a *wedged edge*, one from each pane in the window, into a circularly linked list. Similarly, a vertex is represnted by linking together a distinguished half–edge, called a *vedged edge*, one from each origin in the vertex, into a circularly linked list.



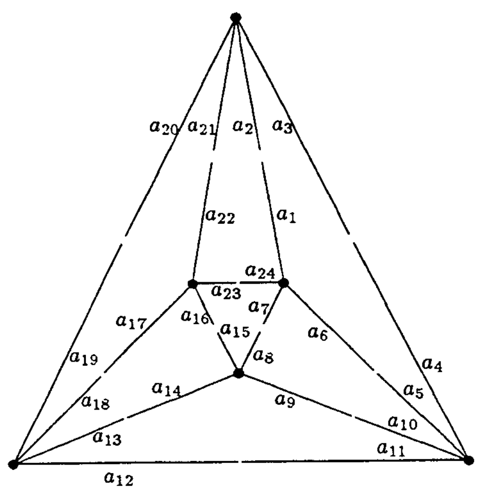
$$V = \left\{ \begin{array}{ll} (a_1a_3a_5), & (a_6a_{10}a_8), \\ (a_9a_{11}a_{13}), & (a_{14}a_{18}a_{16}), \\ (a_{17}a_{19}a_{21}), & (a_2a_{24}a_{22}), \\ (a_4a_{20}a_{12}) & (a_7a_{15}a_{23}) \end{array} \right\} \qquad W = \left\{ \begin{array}{ll} (a_1a_{24}a_7a_6), & (a_2a_3a_{20}a_{21}), \\ (a_4a_5a_{10}a_{11}), & (a_8a_{15}a_{14}a_9), \\ (a_{12}a_{13}a_{18}a_{19}), & (a_{16}a_{23}a_{22}a_{17}) \end{array} \right\}$$

Figure 4: A cube with 8 origins and 7 vertices on a pseudosphere.

In the data structure a record is allocated for each half–edge. In the first implementation we store values for *Onext, Pprev, Vnext, Vprev, Wnext,* and *Wprev*. As is common practice in pointer list implementations, we use the special value *nil* to denote the value of an undefined function such as *Wnext*(a) when a is not wedged. We need not store *Oprev, Pnext,* or *Mnext,* because these can be computed in O(1) time from the others: *Mnext*(a) = *Pprev*(*Onext*(a)), *Pnext*(a) = *Onext*(*Mnext*(a)), *Oprev*(a) = *Mnext*(*Pprev*(a)).

Finally, we add four query operators that allow the user to ask questions about the form of the graph. *Vfind* returns the vedged half–edge in th origin of its argument, and *Wfind* returns the wedged half–edge in the pane of its argument. The boolean function *Vsame* indicates whether its two arguments are in the same vertex, and *Wsame* indicates whether its two arguments are in the same window.

We now examine the run times for the first implementation and suggest two variants. The first implementation requires six pointers per half–edge, using O(n) space to store a graph of size n. The navigation and splicing operators take O(1) time. Unfortunately, queries are slow in this implementation. It takes O(n) time to execute a *Vfind, Wfind, Vsame*, or *Wsame*. This is comparable to the less general quadedge scheme [7], which uses O(n) space, O(1) time for navigation and splicing, and O(n) time to determine if two edges are in the same face or same vertex. A second implementation stores the *Vfind*(a) and *Wfind*(a) values with each half–edge a, in addition to the fields of the first implementation. This scheme allows O(1)–time queries, but now the splicing operations take O(n) time. This implementation is comparable to the DCEL representation [11]. A third scheme, of intermediate speed, maintains the *Vfind* and *Wfind* information by means of a self–adjusting binary tree on the half–edges in each vertex and window [13]. A self–adjusting tree supports O(log n)–time tree splices and root–finds, providing a graph representation with O(log n)–time queries and splices, and O(1)–time navigation operators.



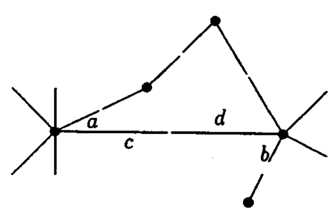
$$V = \left\{\begin{matrix} (a_1a_{24}a_7a_6), & (a_2a_3a_{20}a_{21}), \\ (a_4a_5a_{10}a_{11}), & (a_8a_{15}a_{14}a_9), \\ (a_{12}a_{13}a_{18}a_{19}), & (a_{16}a_{23}a_{22}a_{17}) \end{matrix}\right\} \qquad W = \left\{\begin{matrix} (a_1a_3a_5), & (a_6a_{10}a_8), \\ (a_9a_{11}a_{13}), & (a_{14}a_{18}a_{16}), \\ (a_{17}a_{19}a_{21}), & (a_2a_{24}a_{22}), \\ (a_4a_{20}a_{12}) & (a_7a_{15}a_{23}) \end{matrix}\right\}$$

Figure 5: The dual of Figure 4, an octahedron on a torus.

6. An Extension to the Nonorientable Case

Ringel [12], Stahl [14] and Tutte [16] have investigated imbeddings of graphs on nonorientable manifolds as well as orientable manifolds. Tutte represents each edge as a set of four *crosses*, called a *cell*. If X is a cross, the other members of its cell are θX, ϕX, and θϕX, where θ and ϕ are involutions on the finite set of all crosses making up the graph. The pair X and θX can be thought of as a *half–edge* with ϕX and θϕX making up the opposing half–edge in the edge. The involution θ is a flip, or reversal of orientation, so that X and θX represent different sides of a half–edge. The involution ϕ is a facial rotation, so that X and ϕX are adjacent elements of the boundary of the face on one side of the edge. A permutation Q is then defined on the crosses to give the rotations of the half–edges around their vertices. Breaking an edge into four crosses and defining θ, ϕ, and Q on the crosses gives enough flexibility to represent imbeddings of these graphs in nonorientable manifolds. Guibas and Stolfi [7] have developed a graph representation reminiscent of Tutte's, and they have also designed a data structure for representing and manipulating graphs with a computer.

Following Tutte, we use four records (crosses or quarter–edges) to represent an edge. An origin is then a pair of cycles of quarter–edges, one for each local orientation. As before, a vertex is a set of origins. Thus, in our presentation, we tabulate a rotation O of quarter–edges around origins, as well as a set V of sets of origins. We also tabulate Tutte's involution θ, mapping quarter–edges onto quarter–edges of opposite orientation. The involution θ is restricted by Tutte's axiom: $O\theta = \theta O^{-1}$.



Addh(c)
Vmerge(a,c)
Wmerge(a,c)
Osplice(a,c)
Psplice(a,c)
Addh(d)
Vmerge(b,d)
Wmerge(b,d)
Osplice(b,d)
Psplice(b,d)
Psplice(a,b)

Figure 6: Adding edge with half-edges *c* and *d* to an existing graph.

The facial structure of a graph is treated analogously. A pane is a pair of cycles of quarter–edges, one for each orientation of its boundary. A window is a set of panes. We thus tabulate a rotation P of quarter–edges around panes as well as a set W of sets of panes. The rotation P is restricted by Tutte's axiom: for each quarter–edge q, the orbits of P through q and θq are distinct. Our medial rotation τ as well as Tutte's involution ϕ are implict in our presentation: $\tau = P^{-1}O$ and $\phi = \theta\tau$.

In a straightforward data structure based on the above, we could represent a quarter–edge with one seven–pointer record: (*Onext*, *Pprev*, θ, *Vnext*, *Vprev*, *Wnext*, *Wprev*). Preliminary research indicates that a much more compact representation is possible. We believe we can represent each pair of oppositely oriented quarter–edges $(q, \theta q)$ with a single record containing four pointer fields and one boolean field.

7. Conclusions and Open Problems

This paper makes contributions in two areas: graph theory and data structures. We have given a succinct and general presentation for graphs imbedded on pseudomanifolds. We have also extended surface duality theory by showing how our representation gives rise to a natural dual for graph that may be disconnected or contain isolated vertices. This representation is useful in graph data structures: we have described three implementations with various query and update time preformances. Perhaps most importantly, we have defined a general set of operators for navigating, querying, and manipulating graphs.

At present, we are testing a fast–update and slow–query implementation of our data structure, in preparation for coding an algorithm for generating optimal rectilinear Steiner trees [15]. Dobkin and Laszlo have expressed interest in incorporating our ideas into their extension of the quadedge concept to three–dimensional subdivisions [3]. Relaxing the requirement that the medial function τ be involution gives rise to graphs whose edges have more than two sides and endpoints. It is possible to interpret imbeddings of such graphs. The k-sided edges may be useful in representing the output of an edge–detector in their computer vision application [2]. It may be possible to extend our results to graphs on manifolds with boundary, following Mohar [10]. We expect that many other graph–theoretic algorithms, from connectivity testing to Voronoi diagram computation, would gain clarity and conciseness if our data structures were used.

A host of open questions arise from this work. It remains to see whether our representation can be used to prove extensions of the major theorems in the area of surface imbeddings and duality. In particular, it will be interesting to see whether the notions of genus and characteristic of a graph can be extended to our generalized graphs and whether equations linking these two integers can be obtained.

Acknowledgements

The authors would like to thank T.A. McKee for a helpful discussion and a copy of a manuscript [9]. A. White sent us a number of instructive comments on an early version of this manuscript. We also appreciate critical suggestions by a referee, which have had a significant impact on the final version of this manuscript. Finally, we are grateful to Chris Jackman and Clyde Rogers for their patience in coding C language implementations of three preliminary versions of our splice primitives.

REFERENCES

[1] N. Biggs. Spanning trees of dual graphs. *Journal of Combinatorial Theory*, 11: 127 – 131, 1971.

[2] D.P. Dobkin. Private communication. September 1987.

[3] D.P. Dobkin and M.J. Laszlo. *Primitives for the Manipulation of Three–Dimensional Subdivisions*. Technical Report CS–TR–089–87, Department of Computer Science, Princeton University, April 1987.

[4] J.R. Edmonds. A combinatorial representation for polyhedral surfaces. *Notices of the American Math Society*, 7:646, 1960.

[5] J.R. Edmonds. On the surface duality of linear graphs. *Journal of Research of the National Bureau of Standards*, 69B(1 – 2):121 – 123, January–June 1965.

[6] J.L. Gross and T.W. Tucker. *Topological Graph Theory*. *Wiley Interscience Series in Discrete Mathematics and Optimization,* Wiley–Interscience, 1987.

[7] L. Guibas and J. Stolfi. Primitives for the manipulation of general subdivisions and the computation of Voronoi diagrams. *ACM Transactions on Graphics*, 4(2):74–123, April 1985.

[8] J. Hopcroft and R. Tarjan. Efficient planarity testing. *Journal of the ACM*, 21(4):549 – 568, October 1974.

[9] T.A. Mckee. Transfer principles for graph semiduality. April 1988. Submitted to *Fundamenta Mathematica.*

[10] B. Mohar. Simplicial schemes. *Journal of Combinatorial Theory, Series B*, 42:68–86, 1987.

[11] F.P. Preparata and M.I. Shamos. *Computational Geometry: An Introduction*. Springer–Verlag, New York, 1985.

[12] G. Ringel. *Map Color Theorem*. Springer–Verlag, New York, 1974.

[13] D.D. Sleator and R.E. Tarjan. Self–adjusting binary search trees. *Journal of the ACM,* 32(3):652 – 686, July 1985.

[14] S. Stahl. Generalized embedding schemes. *J. Graph Theory,* 2:41–52, 1978.

[15] C.D. Thomborson, L.L. Deneen, and G.M. Shute. Computing a rectilinear Steiner minimal tree in $n^{O(\sqrt{n})}$ time. In A. Albrecht, et. al., editor, *Parallel Algorithms and Architectures*, pages 176 – 183, Akademie–Verlag, Berlin, June 1987.

[16] W.T. Tutte. *Graph Theory*. Volume 21 of *Encyclopedia of Mathematics and its Applications*, Addison–Wesley, 1984.

[17] A.T. White. *Graphs, Groups and Surfaces*. North–Holland, Amsterdam, 1973.

[18] J.W.T. Youngs. Minimal imbeddings and the genus of a graph. *Journal of Mathematics and Mechanics,* 12(2):303 – 315, 1963.