MAU22C00 Lecture 22

John Stalker

Trinity College Dublin

Necessary and sufficient conditions for an Eulerian trail

We've now established that an undirected graph without self loops has an Eulerian trail if and only if it satisfies the following two conditions:

- At most one connected component has more than one vertex.
- At most two vertices have odd degree.

Also, if all vertices have even degree then all Eulerian trails are circuits and if two have odd degree than no Eulerian trail is a circuit. There can't be only one vertex of add degree.

So we have a complete understanding of Eulerian trails. Unfortunately we don't have, and can't have, a similar understanding of Hamiltonian paths.

Spanning trees

A path is a walk which visits each vertex at most once. It's called Hamiltonian if it visits each vertex *exactly* once.

Most graphs don't have a Hamiltonian path.

There is no good set of necessary and conditions for the existence of a Hamiltonian path.

For connected undirected graphs without self-loops we may not be able to visit every vertex exactly once with a path, but we can with a tree.

This is called a spanning tree.

A Hamiltonian path, if there is one, is a spanning tree.

A spanning tree example

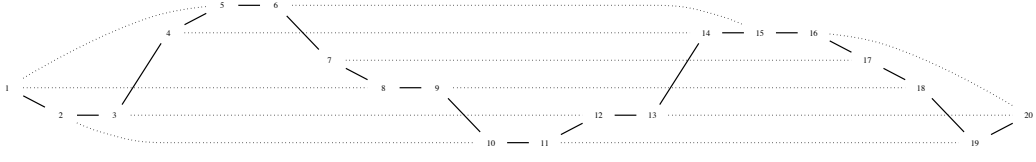

Figure 1: A spanning tree which is a Hamiltonian path

Another spanning tree example

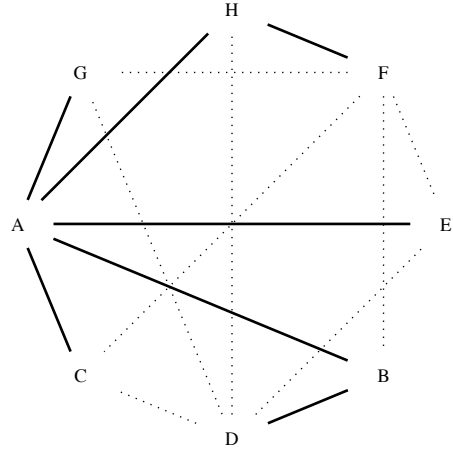

Figure 2: A spanning tree which is not a Hamiltonian path

Uses

A spanning tree has the property that there is a unique path between any two vertices.

This makes them useful for communications routing.

How do you handle broadcast packets in a network which is not a complete graph, a.k.a. bridged networks?

You need to forward packets, but don't want infinite loops.

One approach is to construct a spanning tree and forward only on edges of the tree. This is what ethernet does.

Algorithm

To construct a spanning tree on a connected graph:

- Pick any vertex as the root, and create a tree with only that vertex and no edges.
- For every other vertex there is a path from the root to that vertex, by connectedness. If the vertex is not already in the tree then adjoin to the tree the part of that path from where it *last* leaves the tree.

This must terminate, produces a tree at each step, and ultimately visits each vertex, so it's a spanning tree.

In many applications different edges have different "costs" so some spanning trees are lower cost than others. This version of the algorithm makes no effort to minimise the cost. There are (much more complicated) algorithms which do that.

An operation on a set A is a function from A^n to A, for some natural number n. We're mostly interested in binary operations ($n = 2$), but sometimes also unary operations ($n = 1$) or nullary ($n = 0$).

We'll start with a binary operation f on a set A, i.e. a function from A^2 (ordered pairs) to A.

The arithmetic operations $+$, $\cdot$, $-$ on integers are examples.

We're also interested in other examples, e.g. the concatenation operation on lists, or the union operation on subsets of a given set, or the $\wedge$ operation on Booleans.

You can find a list of more than a dozen examples in the notes.

Properties

There isn't much to be said about general binary operations, just ones with some properties.

f is said to be associative if

$$f(f(x,y),z) = f(x,f(y,z))$$

for all x, y and z.

This will look more familiar in examples, e.g.

$$(x \cdot y) \cdot z = x \cdot (y \cdot z), \quad (x + y) + z = x + (y + z), \quad (x - y) - z = x - (y - z).$$

The last of these isn't actually true. Not all binary operations are associative!

f is said to be commutative if $f(x,y) = f(y,x)$ for all x and y.

Properties, continued

The arithmetic operations $+$ and $\cdot$ are associative and commutative, while $-$ is neither associative nor commutative.

Concatenation is associative. Concatenating $(u_0, \ldots, u_i)$, $(v_0, \ldots, v_j)$, and $(w_0, \ldots, w_k)$ gives $(u_0, \ldots, u_i, v_0, \ldots, v_j, w_0, \ldots, w_k)$ regardless of whether we first concatenate u and v or v and w.

Concatenation is not commutative.

The union operation on subsets of a given set, or the $\wedge$ operation on Booleans are associative and commutative.

For many more examples see the notes.

Semigroups, monoids, groups

A set with an associative operation is called a semigroup.

If (A,f) is a semigroup then $i \in A$ is called an identity if

$$f(i,x) = x, \qquad f(x,i) = x$$

for all $x \in A$.

If f is commutative each of these equations implies the other, but that's not true in general.

There doesn't need to be an identity but there can't be more than one. If i and j are identities then

$$i = f(i,j) = j$$

The first equation holds because j is an identity and the second because i is an identity.

A semigroup with an identity is called a monoid.

If (A,f) is a monoid and i is the identity then $x \in A$ is said to be invertible if there is a $y \in A$ such that

$$f(x,y) = i, \qquad f(y,x) = i.$$

y is then said to be the inverse of x.

Again, one equation implies the other if f is commutative, but not in general.

An element of A doesn't have to have an inverse, but it can't have more than one.

A monoid where every element is invertible is called a group.

The inverse operation on a group is an example of a unary operation.

The identity on a monoid can even be thought of as a nullary operation.

Examples

$(N, \cdot)$, the natural numbers with the multiplication operation, is a monoid, with 1 as the identity. It is not a group.

$(N, +)$, the natural numbers with the addition operation, is a monoid, with 0 as the identity. It is also not a group.

$(N, \max)$, the natural numbers with the maximum operation is a monoid, with 0 as the identity. It is also not a group.

First, this is a semigroup, since $\max(\max(x, y), z) = \max(x, \max(y, z))$.

Second, 0 is an identity, since $\max(x, 0) = x$ and $\max(0, x) = x$.

Third, 0 is the only invertible element. If $x > 0$ then there is no y such that $\max(x, y) = 0$.

$(N, \min)$, the natural numbers with the minimum operation is a semigroup, but not a monoid, and therefore not a group.

There is no number i such that $\min(i, x) = x$ for all x.

Group examples

None of the examples above is a group. How do we get groups?

The set of automorphisms of a graph is a always group, with composition of the automorphisms as the operation.

The identity element is the identity function, which is always an automorphism.

The inverse is the inverse function.

The automorphism group of K_n is the group of permutations of the vertices.

The automorphism group of the Wumpus graph has 120 members. This is not obvious.

In applications groups usually arise as the symmetry group of something, e.g. the set of symmetries of dodecahedron is a group with 120 members.

Actually, it's the same group, in a sense we'll make precise later.

The group of invertible elements in a monoid

If (A,f) is a semigroup, $B \subseteq A$, and $f(x,y) \in B$ whenever $x \in B$ and $y \in B$ then (B,g) is a semigroup, where g is the restriction of f to B^2.

If (A,f) is a monoid with identity i, $B \subseteq A$, $i \in B$, and $f(x,y) \in B$ whenever $x \in B$ and $y \in B$ then (B,g) is a monoid, where g is the restriction of f to B^2. Its identity is i.

i is invertible. Its inverse is i.

If x and y are invertible then so is $f(x,y)$. Indeed the inverse of $f(x,y)$ is $f(w,v)$ where v is the inverse of x and w is the inverse of y. Note the reversal of order!

The set of invertible elements in a monoid is a group, although it might be a trivial group, i.e. might have the identity as its only member.

For example, the set of $n \times n$ matrices is a monoid, with multiplication as the operation and the identity matrix as identity. The set of invertible matrices is therefore a group, called the general linear group of order n.