Lecture 2: Eulerianity, simple graphs, and subgraphs · 1MA020

Vilhelm Agdur[1]

[1] vilhelm.agdur@math.uu.se

24 October 2023

We formalize the ideas we started with in the first exercise session, giving a proof of Euler's result on Eulerian circuits. We then make some more definitions about simple graphs and subgraphs, and we state some elementary results about these notions.

The very first definition we give in this course will actually be of a *multigraph*, not the simple graphs that were our first example.[2]

[2] These are of course a type of graph, so we will often just write or say "graph" when it is clear from context what type of graph we are referring to, or what we are saying applies to any type of graph.

Definition 1. A *multigraph* G is a tuple (V, E), consisting of a set V of *vertices*, and a multiset[3] E of *edges*. Each edge is a multiset containing two vertices from V, called its *endpoints*. We say two vertices are *adjacent* if there is an edge between them, and a vertex v is *incident* to an edge e if $v \in e$.

[3] A multiset is just like a set, except an element may occur more than once.

If the same edge occurs more than once in E, we say that these edges are *parallel*. If the two endpoints of an edge are equal, we call it a *loop*.

Unless explicitly stated, we always assume that both V and E are finite sets. Otherwise, we say the graph is *infinite*.[4]

[4] It may sometimes be the case that our proofs work without modification also for infinite graphs – thinking about whether they do may be a useful thing to do when reading the proofs, to understand them better.

Next, continuing to formalize the things we learned thinking about the bridges of Königsberg, let us define what a walk is.

Definition 2. Let $G = (V, E)$ be a multigraph. A *walk* of length k is a sequence of $k+1$ vertices $v_0 v_1 v_2 \dots v_k$ and a sequence of k edges[5] $e_1 e_2 \dots e_k$ such that $e_i = \{v_{i-1}, v_i\}$ for all i. A *trail* is a walk that uses no edge twice, and a *path* is a walk that uses no vertex twice. A *circuit* is a trail where the first and last vertices coincide, and a *cycle* is a circuit where these are the only vertices that coincide.

[5] So the length of a walk is the number of edges, not the number of vertices.

We have one example of a walk in Figure 1 – it does not repeat any of the edges, so it is a trail, but it repeats the central vertex we have labelled with both v_2 and v_6, so it is not a path.

Having introduced walks, we can give a definition of another very natural property, namely connectedness.[6]

[6] You've probably already seen the notion of connectedness of a subset of $\mathbb{R}^2$ in a calculus course, and if you've looked at other geometry it appears there as well. This is the same notion, just discretized.

Definition 3. We say that a graph G is *connected* if there is, for any two vertices $u, v \in G$, a walk from u to v. We say that two vertices in a graph are connected to each other if there is a walk between them.

Notice that there is a trivial "lazy" walk connecting every vertex to itself, so the relation of connectedness is an equivalence relation. The equivalence classes of this equivalence relation are called the *connected components* of the graph.[7]

[7] We could equivalently have defined the connected components as the maximal connected subgraphs of the graph – when we get to a formal definition of subgraph, think about why this is true.

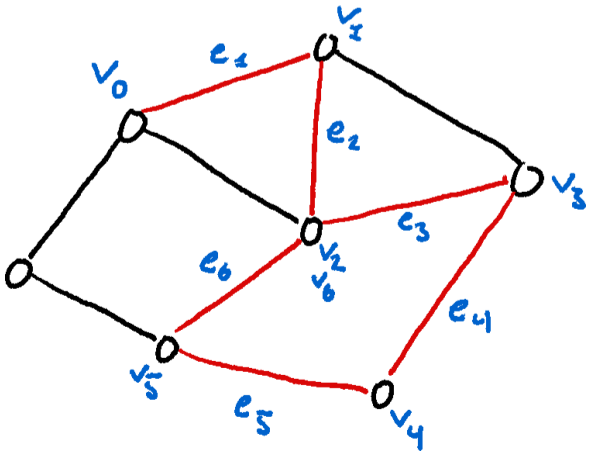

Figure 1: A walk in a graph, which is a trail but not a path.

Let us now define the thing we were studying when we thought about the bridges of Königsberg.

Definition 4. An *Eulerian trail* is a trail that uses every edge in the graph exactly once – if additionally it has the same starting and ending vertex, we call it an *Eulerian circuit*. If there is an Eulerian circuit in a graph, we call the graph *Eulerian*.

The problem we were studying was thus to find a simple condition for when a multigraph is Eulerian. The condition we found[8] involved the number of edges incident to a vertex, so let us also give this notion a name.

[8] Hopefully.

Definition 5. The *degree* of a vertex v, denoted d_v, is the number of edges a vertex is incident to, with loops counted twice.

We now have all the language we need to formally state and prove the theorem that started graph theory all those nearly three hundred years ago.

Theorem 6 (Euler (1736))**.** *A finite connected multigraph is Eulerian if and only if all its vertices have even degree.*

Proof. Let us begin with the easy direction of this statement: That a graph which is Eulerian will have only even-degree vertices. To see this, note that the Eulerian circuit gives us a way to pair up the edges incident to any vertex – the path enters through one edge and leaves through another, so we pair those up. Since the circuit uses every edge, there can't be any odd edge left over in this pairing, and so the degree of the vertex can't be odd.

So, for the harder direction, we will prove the result by induction on the number of edges.[9] The base case is $n = 0$, where the graph must just be a single vertex. That this is Eulerian is trivial – the lazy

[9] This is a different proof than the one given in the lecture notes from last year. The reason for changing the proof is that I couldn't understand what was going on in the previous proof. Feel free to look at the other proof if this one doesn't make sense to you – maybe it will.

path that does nothing uses every edge in the graph, since the graph has no edges.

Now, let $G = (V, E)$ be a finite connected multigraph with only even vertex degrees, with $n \geq 1$ edges. Pick an arbitrary vertex v, then pick an arbitrary edge $e = \{v, w\}$ going out from v.[10] Then pick another edge going out from w, making sure it hasn't already been used in our path, and so on, until you have returned to v. Let W be the set of edges we used in this walk.

[10] This edge is allowed to be a loop – think about what happens in that case.

That we will always be able to pick an unused edge to continue walking along follows from our assumption that the degrees of vertices are even – if we were able to use an edge to get to a vertex we must also be able to pick one to leave it, since we always use up edges in pairs.

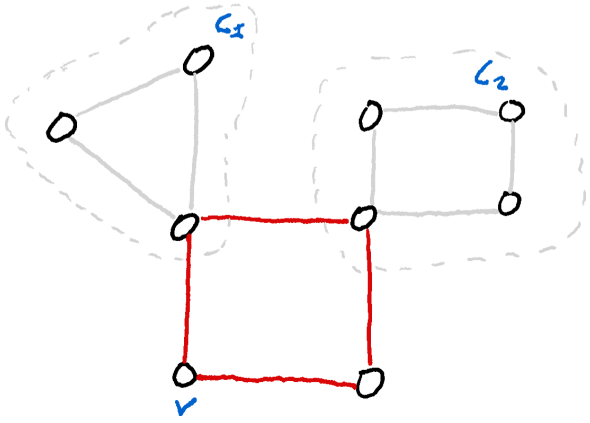

Figure 2: A graph $G = (V, E)$ with the path W highlighted in red, and the two connected components C_1 and C_2 of $H = (V, E \setminus W)$ indicated.

Now consider the graph $H = (V, E \setminus W)$, that is, G with all the edges we used in our walk removed. Let $C_1, C_2, \ldots, C_k$ be its connected components, as is illustrated in Figure 2. Now, each of these components necessarily has fewer than n edges, and is of course trivially connected, so by our induction hypothesis each contains an Eulerian circuit.

The idea now is to glue together our circuit W with the circuits on the C_i to get an Eulerian circuit on the entire graph. To do this, what we need is that each C_i contains at least one vertex that is incident to an edge in W.

To see this,[11] pick a connected component C_i. If $v \in C_i$, we are done, so assume it is not. We now pick some arbitrary vertex w in C_i – since G is by assumption connected, there exists a walk connecting w to v. Since v is not in C_i, this walk must at some point leave C_i – so consider the edge it leaves C_i by, say, $e = \{a, b\}$, walking from $a \in C_i$ to $b \notin C_i$. This edge must in fact be in W, because if it weren't, it'd be an edge of H, and a is in C_i – and thus b would also be in C_i, by definition of connected component, and this edge wouldn't be leaving

[11] This is the hardest piece of the proof – at least in the sense that it is hard to write in a way that is both rigorous and understandable. Once it "clicks" why this should be true, it is hopefully less hard. The figure, and the lecture, might help.

C_i at all. So the vertex a must both be in C_i and be incident to an edge of W.

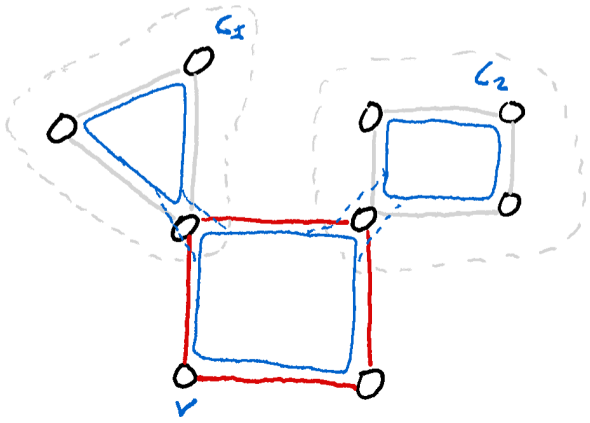

Figure 3: A graph $G = (V, E)$ with the path W highlighted in red, and the two connected components C_1 and C_2 of $H = (V, E \setminus W)$ indicated. Additionally, the Eulerian circuits on the connected components and W are drawn in blue, and their gluing together in dashed blue lines.

So, for each C_i, let a_i be a vertex in C_i incident to an edge in W. We construct our Eulerian circuit on G as follows: Start at v and walk along W. Whenever you encounter an a_i, instead follow the Eulerian circuit in C_i all the way around until you return to this a_i, and then proceed along W. That this will produce an Eulerian circuit on the entire graph should be clear, as is illustrated in Figure 3. □