Lecture 6
MATH-42021/52021 Graph Theory and Combinatorics.

Artem Zvavitch

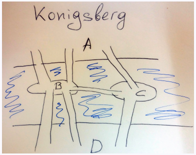
Department of Mathematical Sciences, Kent State University

June, 2016.

Euler Cycles (Example, instead of introduction).

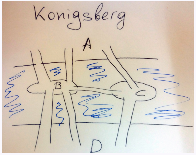
Euler spent a few years of his life in Konigsberg, were there was a very nice river with a couple of islands on it.

Euler spent a few years of his life in Konigsberg, were there was a very nice river with a couple of islands on it. There were also 7 bridges connecting islands and the banks of the river:



Euler Cycles (Example, instead of introduction).

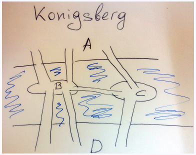
Euler spent a few years of his life in Konigsberg, were there was a very nice river with a couple of islands on it. There were also 7 bridges connecting islands and the banks of the river:



And the walk by the river and islands was amazing and well loved by locals and tourists

Euler Cycles (Example, instead of introduction).

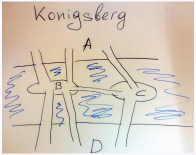
Euler spent a few years of his life in Konigsberg, were there was a very nice river with a couple of islands on it. There were also 7 bridges connecting islands and the banks of the river:



And the walk by the river and islands was amazing and well loved by locals and tourists but some wanted to figure out a walk which would go though each bridge exactly once.

Euler Cycles (Example, instead of introduction).

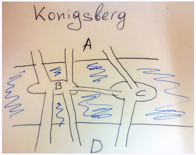
Euler spent a few years of his life in Konigsberg, were there was a very nice river with a couple of islands on it. There were also 7 bridges connecting islands and the banks of the river:



And the walk by the river and islands was amazing and well loved by locals and tourists but some wanted to figure out a walk which would go though each bridge exactly once. Euler solved this problem. The solution was a staring point for graph theory and together with the problem is called "Seven Bridges of Konigsberg" in mathematical literature.

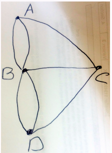
Euler Cycles (Example, instead of introduction).

Euler spent a few years of his life in Konigsberg, were there was a very nice river with a couple of islands on it. There were also 7 bridges connecting islands and the banks of the river:



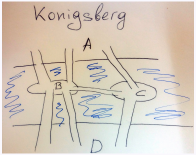
And the walk by the river and islands was amazing and well loved by locals and tourists but some wanted to figure out a walk which would go though each bridge exactly once. Euler solved this problem. The solution was a staring point for graph theory and together with the problem is called "Seven Bridges of Konigsberg" in mathematical literature.

The idea is to model this walk problem with a **multigraph** having a vertex for each body of land and edge for each bridge.



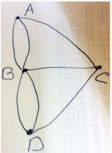
Euler Cycles (Example, instead of introduction).

Euler spent a few years of his life in Konigsberg, were there was a very nice river with a couple of islands on it. There were also 7 bridges connecting islands and the banks of the river:



And the walk by the river and islands was amazing and well loved by locals and tourists but some wanted to figure out a walk which would go though each bridge exactly once. Euler solved this problem. The solution was a staring point for graph theory and together with the problem is called "Seven Bridges of Konigsberg" in mathematical literature.

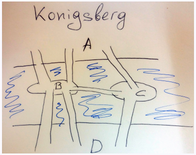
The idea is to model this walk problem with a **multigraph** having a vertex for each body of land and edge for each bridge.



The walk we are looking for is now called "Euler cycle".

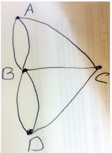
Euler Cycles (Example, instead of introduction).

Euler spent a few years of his life in Konigsberg, were there was a very nice river with a couple of islands on it. There were also 7 bridges connecting islands and the banks of the river:



And the walk by the river and islands was amazing and well loved by locals and tourists but some wanted to figure out a walk which would go though each bridge exactly once. Euler solved this problem. The solution was a staring point for graph theory and together with the problem is called "Seven Bridges of Konigsberg" in mathematical literature.

The idea is to model this walk problem with a **multigraph** having a vertex for each body of land and edge for each bridge.



The walk we are looking for is now called "Euler cycle". Look at the graph above and try to explain why the "Euler cycle" does not exist for this graph.

Euler Cycles.

We remind that by definition a circuit and a path may visit a vertex at most once.

Euler Cycles.

We remind that by definition a circuit and a path may visit a vertex at most once. To solve a previous problem we need something like circuit, BUT to allow to visit each vertex (island or river bank) as many times as we want BUT to allow to go though a given edge (bridge) no more then once.

Euler Cycles.

We remind that by definition a circuit and a path may visit a vertex at most once. To solve a previous problem we need something like circuit, BUT to allow to visit each vertex (island or river bank) as many times as we want BUT to allow to go though a given edge (bridge) no more then once.

- A **cycle** is a sequence of consecutively linked edges $((x_1, x_2), (x_2, x_3), \dots, (x_{n-1}, x_n))$ whose starting vertex is the ending vertex, i.e. $x_1 = x_n$ and which no edge can appear more then once.

Euler Cycles.

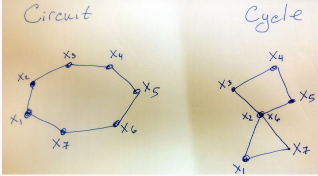
We remind that by definition a circuit and a path may visit a vertex at most once. To solve a previous problem we need something like circuit, BUT to allow to visit each vertex (island or river bank) as many times as we want BUT to allow to go though a given edge (bridge) no more then once.

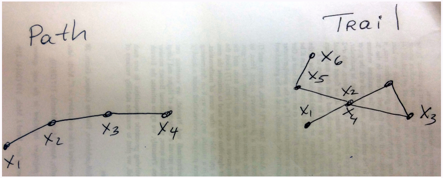
- A **cycle** is a sequence of consecutively linked edges $((x_1, x_2), (x_2, x_3), \dots, (x_{n-1}, x_n))$ whose starting vertex is the ending vertex, i.e. $x_1 = x_n$ and which no edge can appear more then once.
- A trail is a sequence of consecutively linked edges in which no edge can appear more then once.

Euler Cycles.

We remind that by definition a circuit and a path may visit a vertex at most once. To solve a previous problem we need something like circuit, BUT to allow to visit each vertex (island or river bank) as many times as we want BUT to allow to go though a given edge (bridge) no more then once.

- A **cycle** is a sequence of consecutively linked edges $((x_1, x_2), (x_2, x_3), \dots, (x_{n-1}, x_n))$ whose starting vertex is the ending vertex, i.e. $x_1 = x_n$ and which no edge can appear more then once.
- A trail is a sequence of consecutively linked edges in which no edge can appear more then once.

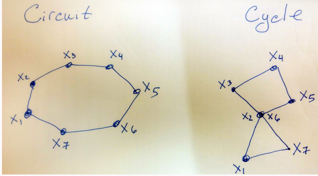


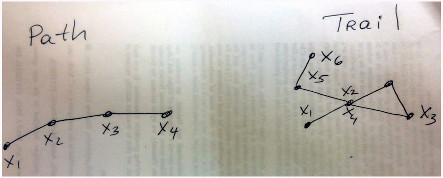


Euler Cycles.

We remind that by definition a circuit and a path may visit a vertex at most once. To solve a previous problem we need something like circuit, BUT to allow to visit each vertex (island or river bank) as many times as we want BUT to allow to go though a given edge (bridge) no more then once.

- A **cycle** is a sequence of consecutively linked edges $((x_1, x_2), (x_2, x_3), \dots, (x_{n-1}, x_n))$ whose starting vertex is the ending vertex, i.e. $x_1 = x_n$ and which no edge can appear more then once.
- A trail is a sequence of consecutively linked edges in which no edge can appear more then once.



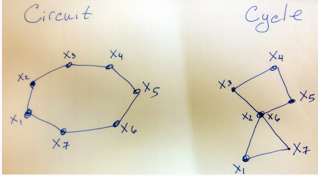


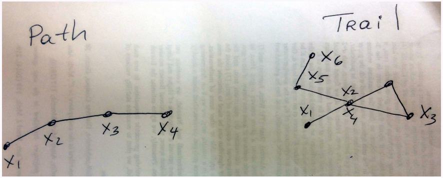
- An **Euler cycle** is a cycle that contains ALL the edges in a graph (and visits each vertex at least once).

Euler Cycles.

We remind that by definition a circuit and a path may visit a vertex at most once. To solve a previous problem we need something like circuit, BUT to allow to visit each vertex (island or river bank) as many times as we want BUT to allow to go though a given edge (bridge) no more then once.

- A **cycle** is a sequence of consecutively linked edges $((x_1, x_2), (x_2, x_3), \dots, (x_{n-1}, x_n))$ whose starting vertex is the ending vertex, i.e. $x_1 = x_n$ and which no edge can appear more then once.
- A trail is a sequence of consecutively linked edges in which no edge can appear more then once.



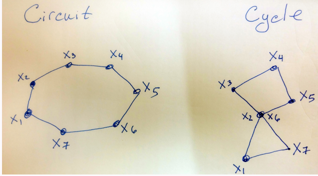


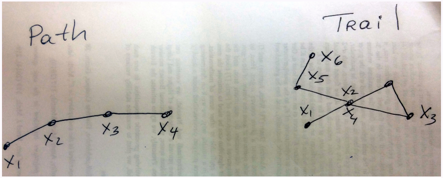
- An **Euler cycle** is a cycle that contains ALL the edges in a graph (and visits each vertex at least once).
- An **Euler trail** is a trail that contains ALL the edges in a graph (and visits each vertex at least once).

Euler Cycles.

We remind that by definition a circuit and a path may visit a vertex at most once. To solve a previous problem we need something like circuit, BUT to allow to visit each vertex (island or river bank) as many times as we want BUT to allow to go though a given edge (bridge) no more then once.

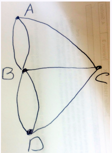
- A **cycle** is a sequence of consecutively linked edges $((x_1, x_2), (x_2, x_3), \dots, (x_{n-1}, x_n))$ whose starting vertex is the ending vertex, i.e. $x_1 = x_n$ and which no edge can appear more then once.
- A trail is a sequence of consecutively linked edges in which no edge can appear more then once.



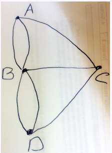


- An **Euler cycle** is a cycle that contains ALL the edges in a graph (and visits each vertex at least once).
- An **Euler trail** is a trail that contains ALL the edges in a graph (and visits each vertex at least once).
- For some applications of Euler cycles we will need to allow a multiple edges between vertices as well a loops (and edge of the form (x, x)) – we will call such generalization of a graph — **"multigraphs"**.

So why there is no Euler cycle in Konigsberg?

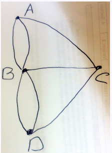


So why there is no Euler cycle in Konigsberg?



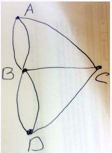
This come from a quite logical observation: a multigraph with Euler cycle must have even degree at each vertex,

So why there is no Euler cycle in Konigsberg?



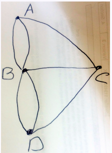
This come from a quite logical observation: a multigraph with Euler cycle must have even degree at each vertex, indeed each time the cycle passes through a vertex it will use two (new) edges (the same about "start" and "end" vertex).

So why there is no Euler cycle in Konigsberg?



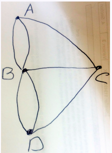
This come from a quite logical observation: a multigraph with Euler cycle must have even degree at each vertex, indeed each time the cycle passes through a vertex it will use two (new) edges (the same about "start" and "end" vertex). But would this property be enough?

So why there is no Euler cycle in Konigsberg?



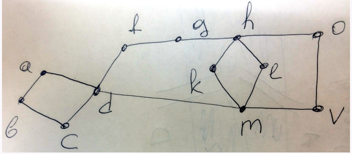
This come from a quite logical observation: a multigraph with Euler cycle must have even degree at each vertex, indeed each time the cycle passes through a vertex it will use two (new) edges (the same about "start" and "end" vertex). But would this property be enough? Yes, we also need to assume that the graph is connected.

So why there is no Euler cycle in Konigsberg?



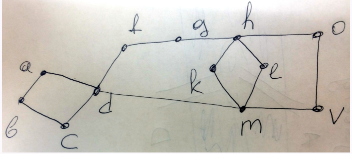
This come from a quite logical observation: a multigraph with Euler cycle must have even degree at each vertex, indeed each time the cycle passes through a vertex it will use two (new) edges (the same about "start" and "end" vertex). But would this property be enough? Yes, we also need to assume that the graph is connected. So is it true that any connected multigraph with vertices of even degree has and Euler cycle? Let try an example!

Build an Euler cycle for



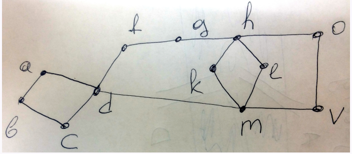
Notice that what we are trying to build is a cycle so if it exist the start/end can be any vertex.

Build an Euler cycle for



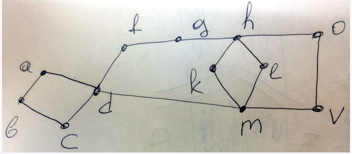
Notice that what we are trying to build is a cycle so if it exist the start/end can be any vertex. Lets pick o.

Build an Euler cycle for



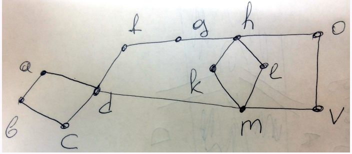
Notice that what we are trying to build is a cycle so if it exist the start/end can be any vertex. Lets pick o. Now start "almost" a random walk over your graph, just play by the rules – never use the same edge twice.

Build an Euler cycle for



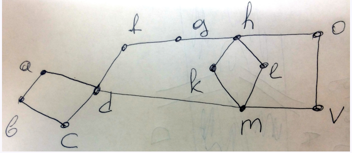
Notice that what we are trying to build is a cycle so if it exist the start/end can be any vertex. Lets pick o. Now start "almost" a random walk over your graph, just play by the rules – never use the same edge twice. Note you will ALWAYS return to o,

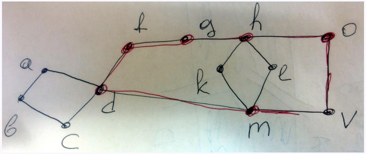
Build an Euler cycle for



Notice that what we are trying to build is a cycle so if it exist the start/end can be any vertex. Lets pick o. Now start "almost" a random walk over your graph, just play by the rules – never use the same edge twice. Note you will ALWAYS return to o, may be you will not use all edges, but you will not stuck in any other vertices - here we use "even" degree property.

Build an Euler cycle for

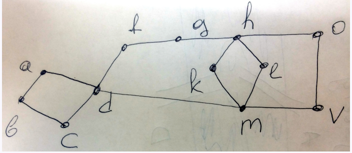


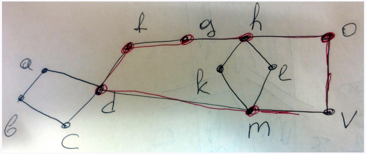


Notice that what we are trying to build is a cycle SO if it exist the start/end can be any vertex. Lets pick o. Now start "almost" a random walk over your graph, just play by the rules – never use the same edge twice. Note you will ALWAYS return to o, may be you will not use all edges, but you will not stuck in any other vertices - here we use "even" degree property. So say you walked through $o - v - m - d - f - g - h - o$.

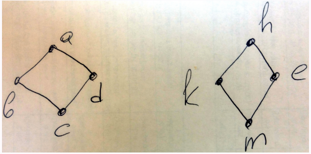
Euler Cycles.

Build an Euler cycle for



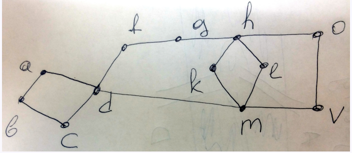


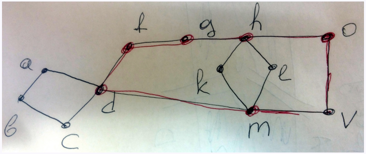
Notice that what we are trying to build is a cycle SO if it exist the start/end can be any vertex. Lets pick o. Now start "almost" a random walk over your graph, just play by the rules – never use the same edge twice. Note you will ALWAYS return to o, may be you will not use all edges, but you will not stuck in any other vertices - here we use "even" degree property. So say you walked through $o - v - m - d - f - g - h - o.$. Next consider a subgraph of edges we have not used:



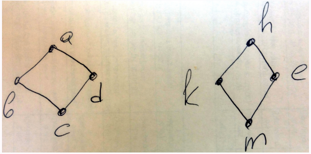
Euler Cycles.

Build an Euler cycle for



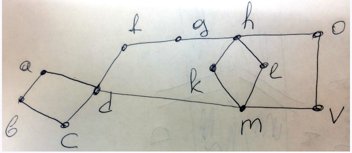


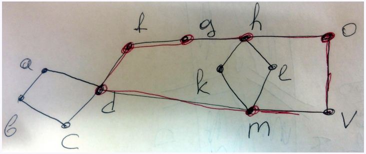
Notice that what we are trying to build is a cycle SO if it exist the start/end can be any vertex. Lets pick o. Now start "almost" a random walk over your graph, just play by the rules – never use the same edge twice. Note you will ALWAYS return to o, may be you will not use all edges, but you will not stuck in any other vertices - here we use "even" degree property. So say you walked through $o - v - m - d - f - g - h - o$.. Next consider a subgraph of edges we have not used:



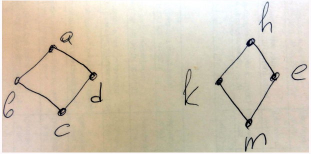
Yes, it is no longer connected but this will not a be a problem for us.

Build an Euler cycle for





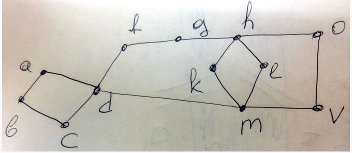
Notice that what we are trying to build is a cycle SO if it exist the start/end can be any vertex. Lets pick o. Now start "almost" a random walk over your graph, just play by the rules – never use the same edge twice. Note you will ALWAYS return to o, may be you will not use all edges, but you will not stuck in any other vertices - here we use "even" degree property. So say you walked through $o - v - m - d - f - g - h - o$.. Next consider a subgraph of edges we have not used:

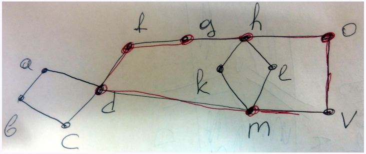


Yes, it is no longer connected but this will not a be a problem for us. An essential observation is that all vertices in above graph have even degrees

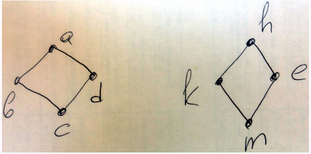
Euler Cycles.

Build an Euler cycle for





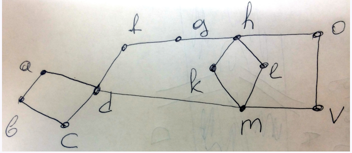
Notice that what we are trying to build is a cycle SO if it exist the start/end can be any vertex. Lets pick o. Now start "almost" a random walk over your graph, just play by the rules – never use the same edge twice. Note you will ALWAYS return to o, may be you will not use all edges, but you will not stuck in any other vertices - here we use "even" degree property. So say you walked through $o - v - m - d - f - g - h - o$.. Next consider a subgraph of edges we have not used:

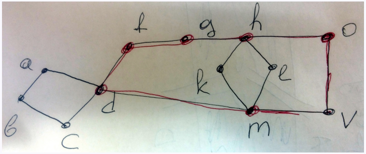


Yes, it is no longer connected but this will not a be a problem for us. An essential observation is that all vertices in above graph have even degrees (removing the cycle from a graph reduces the degree of a vertex by an even number).

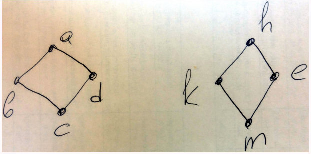
Euler Cycles.

Build an Euler cycle for





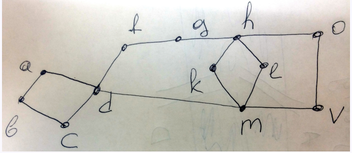
Notice that what we are trying to build is a cycle SO if it exist the start/end can be any vertex. Lets pick o. Now start "almost" a random walk over your graph, just play by the rules – never use the same edge twice. Note you will ALWAYS return to o, may be you will not use all edges, but you will not stuck in any other vertices - here we use "even" degree property. So say you walked through $o - v - m - d - f - g - h - o$.. Next consider a subgraph of edges we have not used:

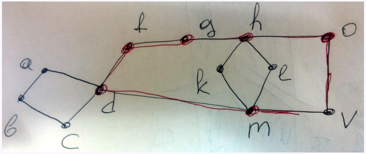


Yes, it is no longer connected but this will not a be a problem for us. An essential observation is that all vertices in above graph have even degrees (removing the cycle from a graph reduces the degree of a vertex by an even number). Note that each connected part in this graph have an Euler cycle so we get two additional cycles $h - e - m - k - h$ and $d - c - b - a - d$.

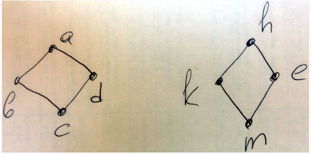
Euler Cycles.

Build an Euler cycle for





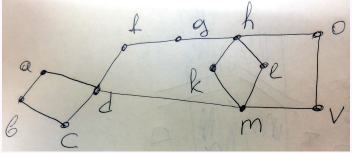
Notice that what we are trying to build is a cycle SO if it exist the start/end can be any vertex. Lets pick o. Now start "almost" a random walk over your graph, just play by the rules – never use the same edge twice. Note you will ALWAYS return to o, may be you will not use all edges, but you will not stuck in any other vertices - here we use "even" degree property. So say you walked through $o - v - m - d - f - g - h - o$.. Next consider a subgraph of edges we have not used:

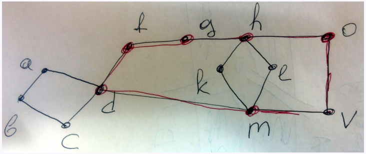


Yes, it is no longer connected but this will not a be a problem for us. An essential observation is that all vertices in above graph have even degrees (removing the cycle from a graph reduces the degree of a vertex by an even number). Note that each connected part in this graph have an Euler cycle so we get two additional cycles $h - e - m - k - h$ and $d - c - b - a - d$. The idea is that we not can "insert those two cycles into original cycle:

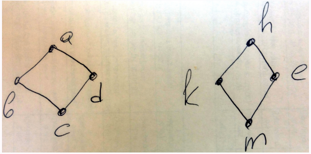
Euler Cycles.

Build an Euler cycle for





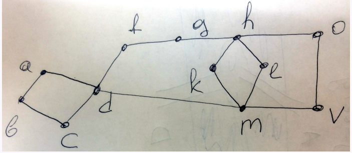
Notice that what we are trying to build is a cycle SO if it exist the start/end can be any vertex. Lets pick o. Now start "almost" a random walk over your graph, just play by the rules – never use the same edge twice. Note you will ALWAYS return to o, may be you will not use all edges, but you will not stuck in any other vertices - here we use "even" degree property. So say you walked through $o - v - m - d - f - g - h - o$.. Next consider a subgraph of edges we have not used:

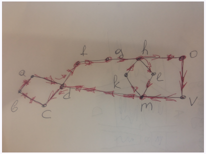


Yes, it is no longer connected but this will not a be a problem for us. An essential observation is that all vertices in above graph have even degrees (removing the cycle from a graph reduces the degree of a vertex by an even number). Note that each connected part in this graph have an Euler cycle so we get two additional cycles $h - e - m - k - h$ and $d - c - b - a - d$. The idea is that we not can "insert those two cycles into original cycle:

$o - v - m - \mathbf{d} - \mathbf{c} - \mathbf{b} - \mathbf{a} - \mathbf{d} - f - g - \mathbf{h} - \mathbf{e} - \mathbf{m} - \mathbf{k} - \mathbf{h} - o$

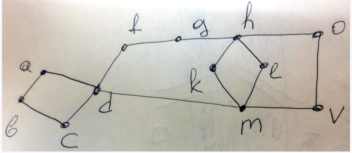
Build an Euler cycle for

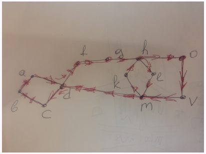




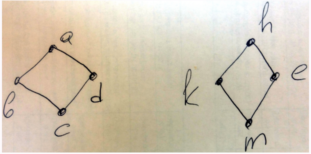
Notice that what we are trying to build is a cycle SO if it exist the start/end can be any vertex. Lets pick o. Now start "almost" a random walk over your graph, just play by the rules – never use the same edge twice. Note you will ALWAYS return to o, may be you will not use all edges, but you will not stuck in any other vertices - here we use "even" degree property. So say you walked through $o - v - m - d - f - g - h - o$.

Build an Euler cycle for





Notice that what we are trying to build is a cycle SO if it exist the start/end can be any vertex. Lets pick o. Now start "almost" a random walk over your graph, just play by the rules – never use the same edge twice. Note you will ALWAYS return to o, may be you will not use all edges, but you will not stuck in any other vertices - here we use "even" degree property. So say you walked through $o - v - m - d - f - g - h - o$.. Next consider a subgraph of edges we have not used:



Yes, it is no longer connected but this will not a be a problem for us. An essential observation that all vertices in above graph have even degrees (removing the cycle from a graph reduces the degree of a vertex by an even number). Note that each connected part in this graph have an Euler cycle so we get two additional cycles $h - e - m - k - h$ and $d - c - b - a - d$. The idea is that we not can "insert those two cycles into original cycle:

$o - v - m - \mathbf{d} - \mathbf{c} - \mathbf{b} - \mathbf{a} - \mathbf{d} - f - g - \mathbf{h} - \mathbf{e} - \mathbf{m} - \mathbf{k} - \mathbf{h} - o$.

Theorem:

An undirected multigraph has an Euler cycle if and only if it is connected and has all vertices of even degree.

Proof :

Theorem:

An undirected multigraph has an Euler cycle if and only if it is connected and has all vertices of even degree.

Proof : As we discussed: if Euler cycle exists, then the graph must be connected and have all vertices of even degree.

Theorem:

An undirected multigraph has an Euler cycle if and only if it is connected and has all vertices of even degree.

Proof : As we discussed: if Euler cycle exists, then the graph must be connected and have all vertices of even degree.
Now to prove the existence of Euler cycle we will follow the algorithm we used in the previous example. Suppose our multigraph G is connected and all vertices have even degrees.

Theorem:

An undirected multigraph has an Euler cycle if and only if it is connected and has all vertices of even degree.

Proof : As we discussed: if Euler cycle exists, then the graph must be connected and have all vertices of even degree.
Now to prove the existence of Euler cycle we will follow the algorithm we used in the previous example. Suppose our multigraph G is connected and all vertices have even degrees. Pick any vertex a and trace a trail, i.e. start walking over the graph edge by edge (do not forget that they must be connected) and never use the same edge twice,

Theorem:

An undirected multigraph has an Euler cycle if and only if it is connected and has all vertices of even degree.

Proof : As we discussed: if Euler cycle exists, then the graph must be connected and have all vertices of even degree.
Now to prove the existence of Euler cycle we will follow the algorithm we used in the previous example. Suppose our multigraph G is connected and all vertices have even degrees. Pick any vertex a and trace a trail, i.e. start walking over the graph edge by edge (do not forget that they must be connected) and never use the same edge twice, because all edges are of even degree we will never be forced to stop at any vertex other then a (we may pass though a a few times!).

Theorem:

An undirected multigraph has an Euler cycle if and only if it is connected and has all vertices of even degree.

Proof : As we discussed: if Euler cycle exists, then the graph must be connected and have all vertices of even degree.
Now to prove the existence of Euler cycle we will follow the algorithm we used in the previous example. Suppose our multigraph G is connected and all vertices have even degrees. Pick any vertex a and trace a trail, i.e. start walking over the graph edge by edge (do not forget that they must be connected) and never use the same edge twice, because all edges are of even degree we will never be forced to stop at any vertex other then a (we may pass though a a few times!). Let C be the cycle we generated and let G' be a multigraph consisting of remaining edges of G after we remove C.

Theorem:

An undirected multigraph has an Euler cycle if and only if it is connected and has all vertices of even degree.

Proof : As we discussed: if Euler cycle exists, then the graph must be connected and have all vertices of even degree.
Now to prove the existence of Euler cycle we will follow the algorithm we used in the previous example. Suppose our multigraph G is connected and all vertices have even degrees. Pick any vertex a and trace a trail, i.e. start walking over the graph edge by edge (do not forget that they must be connected) and never use the same edge twice, because all edges are of even degree we will never be forced to stop at any vertex other then a (we may pass though a a few times!). Let C be the cycle we generated and let G' be a multigraph consisting of remaining edges of G after we remove C. Yes, G' may not be connected, but it will have all vertices of even degree.

Theorem:

An undirected multigraph has an Euler cycle if and only if it is connected and has all vertices of even degree.

Proof : As we discussed: if Euler cycle exists, then the graph must be connected and have all vertices of even degree.
Now to prove the existence of Euler cycle we will follow the algorithm we used in the previous example. Suppose our multigraph G is connected and all vertices have even degrees. Pick any vertex a and trace a trail, i.e. start walking over the graph edge by edge (do not forget that they must be connected) and never use the same edge twice, because all edges are of even degree we will never be forced to stop at any vertex other then a (we may pass though a a few times!). Let C be the cycle we generated and let G' be a multigraph consisting of remaining edges of G after we remove C. Yes, G' may not be connected, but it will have all vertices of even degree.
Since the original graph was connected G' and C must have a common vertex - call it a', repeat the construction of a cycle in G' from a' call this cycle C',

Theorem:

An undirected multigraph has an Euler cycle if and only if it is connected and has all vertices of even degree.

Proof : As we discussed: if Euler cycle exists, then the graph must be connected and have all vertices of even degree.
Now to prove the existence of Euler cycle we will follow the algorithm we used in the previous example. Suppose our multigraph G is connected and all vertices have even degrees. Pick any vertex a and trace a trail, i.e. start walking over the graph edge by edge (do not forget that they must be connected) and never use the same edge twice, because all edges are of even degree we will never be forced to stop at any vertex other then a (we may pass though a a few times!). Let C be the cycle we generated and let G' be a multigraph consisting of remaining edges of G after we remove C. Yes, G' may not be connected, but it will have all vertices of even degree.
Since the original graph was connected G' and C must have a common vertex - call it a', repeat the construction of a cycle in G' from a' call this cycle C', glue it to C the same way as we done in our previous example.

Theorem:

An undirected multigraph has an Euler cycle if and only if it is connected and has all vertices of even degree.

Proof : As we discussed: if Euler cycle exists, then the graph must be connected and have all vertices of even degree.
Now to prove the existence of Euler cycle we will follow the algorithm we used in the previous example. Suppose our multigraph G is connected and all vertices have even degrees. Pick any vertex a and trace a trail, i.e. start walking over the graph edge by edge (do not forget that they must be connected) and never use the same edge twice, because all edges are of even degree we will never be forced to stop at any vertex other then a (we may pass though a a few times!). Let C be the cycle we generated and let G' be a multigraph consisting of remaining edges of G after we remove C. Yes, G' may not be connected, but it will have all vertices of even degree.
Since the original graph was connected G' and C must have a common vertex - call it a', repeat the construction of a cycle in G' from a' call this cycle C', glue it to C the same way as we done in our previous example. Now consider graph G'' which is G' after removal of C' and repeat the procedure until we use all edges.

□

Corollary:

A multigraph has an Euler trail, but not an Euler cycle, if and only if it is connected and has exactly two vertices of odd degree.

Euler Trail.

Corollary:

A multigraph has an Euler trail, but not an Euler cycle, if and only if it is connected and has exactly two vertices of odd degree.

Proof : Suppose a multigraph G has an Euler trail but not an Euler cycle.

Corollary:

A multigraph has an Euler trail, but not an Euler cycle, if and only if it is connected and has exactly two vertices of odd degree.

Proof : Suppose a multigraph G has an Euler trail but not an Euler cycle. Call this trail T. Then the starting and ending points are different (it is not a cycle!) and they must have an odd degree (of not you would be able to continue your trail).

Corollary:

A multigraph has an Euler trail, but not an Euler cycle, if and only if it is connected and has exactly two vertices of odd degree.

Proof : Suppose a multigraph G has an Euler trail but not an Euler cycle. Call this trail T. Then the starting and ending points are different (it is not a cycle!) and they must have an odd degree (of not you would be able to continue your trail). All other points must have an even degree and, clearly, the graph must be connected.

Corollary:

A multigraph has an Euler trail, but not an Euler cycle, if and only if it is connected and has exactly two vertices of odd degree.

Proof : Suppose a multigraph G has an Euler trail but not an Euler cycle. Call this trail T. Then the starting and ending points are different (it is not a cycle!) and they must have an odd degree (of not you would be able to continue your trail). All other points must have an even degree and, clearly, the graph must be connected.
Now suppose the graph G is connected and have exactly two vertices of odd degree (say p and q).

Corollary:

A multigraph has an Euler trail, but not an Euler cycle, if and only if it is connected and has exactly two vertices of odd degree.

Proof : Suppose a multigraph G has an Euler trail but not an Euler cycle. Call this trail T. Then the starting and ending points are different (it is not a cycle!) and they must have an odd degree (of not you would be able to continue your trail). All other points must have an even degree and, clearly, the graph must be connected.
Now suppose the graph G is connected and have exactly two vertices of odd degree (say p and q).
Ready for a cool trick?

Corollary:

A multigraph has an Euler trail, but not an Euler cycle, if and only if it is connected and has exactly two vertices of odd degree.

Proof : Suppose a multigraph G has an Euler trail but not an Euler cycle. Call this trail T. Then the starting and ending points are different (it is not a cycle!) and they must have an odd degree (of not you would be able to continue your trail). All other points must have an even degree and, clearly, the graph must be connected.
Now suppose the graph G is connected and have exactly two vertices of odd degree (say p and q).
Ready for a cool trick? Add to graph G a supplementary edge (p, q) and call the new graph G'.

Euler Trail.

Corollary:

A multigraph has an Euler trail, but not an Euler cycle, if and only if it is connected and has exactly two vertices of odd degree.

Proof : Suppose a multigraph G has an Euler trail but not an Euler cycle. Call this trail T. Then the starting and ending points are different (it is not a cycle!) and they must have an odd degree (of not you would be able to continue your trail). All other points must have an even degree and, clearly, the graph must be connected.
Now suppose the graph G is connected and have exactly two vertices of odd degree (say p and q).
Ready for a cool trick? Add to graph G a supplementary edge (p, q) and call the new graph G'. Then G' is connected and has all vertices of even degree.

Corollary:

A multigraph has an Euler trail, but not an Euler cycle, if and only if it is connected and has exactly two vertices of odd degree.

Proof : Suppose a multigraph G has an Euler trail but not an Euler cycle. Call this trail T. Then the starting and ending points are different (it is not a cycle!) and they must have an odd degree (of not you would be able to continue your trail). All other points must have an even degree and, clearly, the graph must be connected.
Now suppose the graph G is connected and have exactly two vertices of odd degree (say p and q). **Ready for a cool trick?** Add to graph G a supplementary edge (p, q) and call the new graph G'. Then G' is connected and has all vertices of even degree. Then there is an Euler cycle C' in G'.

Corollary:

A multigraph has an Euler trail, but not an Euler cycle, if and only if it is connected and has exactly two vertices of odd degree.

Proof : Suppose a multigraph G has an Euler trail but not an Euler cycle. Call this trail T. Then the starting and ending points are different (it is not a cycle!) and they must have an odd degree (of not you would be able to continue your trail). All other points must have an even degree and, clearly, the graph must be connected.
Now suppose the graph G is connected and have exactly two vertices of odd degree (say p and q).
Ready for a cool trick? Add to graph G a supplementary edge (p, q) and call the new graph G'. Then G' is connected and has all vertices of even degree. Then there is an Euler cycle C' in G'. Now REMOVE edge (p, q) from this cycle to get the required trail.

□