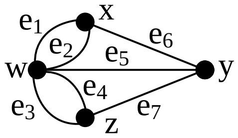
1.2. **Paths, Cycles, and Trails**

1.2.2. **Definition**. Let G be a graph.
A **walk** is a list v_0, e_1, v_1,…, e_k, v_k of vertices and edges such that, for $1 \le i \le k$, the edge e_i has end points v_{i-1} and v_i.
A **trail** is a walk with no repeated edge.
A **path** is a subgraph of G that is a path (a path can be considered as a walk with no repeated vertices).

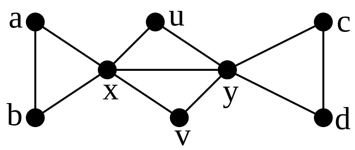
A **u,v-walk** or **u,v-trail** has first vertex u and last vertex v; these are **endpoints**.
A **u,v-path** is a path whose vertices of degree 1(its **endpoints**) are u and v; the others are **internal vertices**.
The **length** of a walk, trail, path, or cycle is its number of edges.
A single vertex u = v is considered as a u,v-walk(trail, path) of length 0.
A loop is a cycle of length 1. 
Also two distinct edges with the same endpoints form a cycle of length 2. 
A walk or trail is **closed** if its endpoints are the same.
A walk is **odd** or **even** as its length is odd or even.

1.2.3. **Example**. In the Konigsburg graph, the list x, e_2, w, e_5, y, e_6, x, e_1, w, e_2, x is a closed walk of length 5; it repeats edges e_2 and hence it is not a trail.



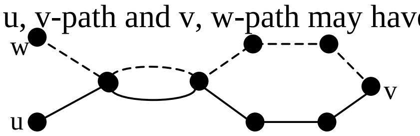
While x, e_2, w, e_5, y, e_6, x, e_1, w is a trail of length 4, it repeats vertices but not edges.
The subgraph consisting of edges e_1, e_5, e_6 and vertices x, w, y is a cycle of length 3,; deleting one of its edges yields a path.
Two edges with the same endpoints such as e_1 and e_2 form a cycle of length 2. #

1.2.4. **Example**. In the simple graph below, the list a, x, a, x, u, y, c, d, y, v, x, b, a specifies a closed walk of length 12, while a, x, u, y, c, d, y, v, x, b, a specifies a closed trail.

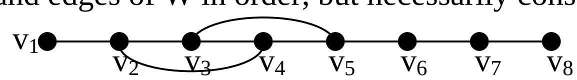


This graph has 5 cycles: (a, b, x), (c, y, d), (u, x, y), (x, y, v), (u, x, v, y).
The u, v-trail u, y, c, d, y, x, v contains the edges of the u, v-path u, y, x, v, but not of the u, v-path u, y, v. #

Remark.1. If we follow a path from u to v in a graph and then follow a path from v to w, the result need not be a u, w-path, because the u, v-path and v, w-path may have a common internal vertex, in fact the result is a u, w-walk.

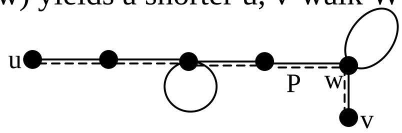


2. Saying that a walk W **contains** a path P means that the vertices and edges of P occur as a sublist of the vertices and edges of W in order, but necessarily consecutive.

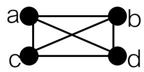


A walk W : v_2, v_4, v_3, v_2, v_3, v_5, v_6, v_5 contain a path P : v_2, v_3, v_5, v_6.

1.2.5 **Lemma**. In a graph G, every u, v-walk contains a u, v-path.
Proof: We prove statement by induction on the length ℓ of a u, v-walk W.
Basis step: $\ell = 0$. Having no edge, W consists of a single vertex (u= v).
This vertex is a u, v-path of length 0.
Induction step: $\ell \geq 1$. We suppose that the claim holds for walks of length less than ℓ.
Let W be a u, v-walk of length ℓ.
If W has no repeated vertex, then its vertices and edges form a u, v-path.
If W has a repeated vertex w, then deleting the edges and vertices between appearances of w (leaving one copy of w) yields a shorter u, v-walk W′ contained in W.



By induction hypothesis, W′ contains a u, v-path P, and this path P is contained in W. #

Exercise1.2.2. Consider K_4: 

has a walk: a, b, c, a, b that is not a trail.

K_4 has a trail: a, b, d, a, c that is not closed and is not a path.
K_4 has no closed trail that is not a cycle. Since a closed trail has even vertex degrees, in K_4 this requires degrees 2 or 0, which do not permit a connected graph that is not a cycle.
In fact, K_4 has a cycle of length 4: a, b, c, d, a and has a cycle of length 3: a, b, c, a. #

1.2.6. **Definition**. A graph G is **connected** if it has a u, v-path whenever u, v $\in$ V(G), otherwise, G is **disconnected**.
If G has a u, v-path, then u is **connected to** v in G.
The **connection relation** on V(G) consists of the ordered pairs (u, v) such that u is connected to v.
That is, $\sim = \{(u, v) \in V(G) \times V(G) : u \text{ is connected to } v\}$ is a relation on V(G).

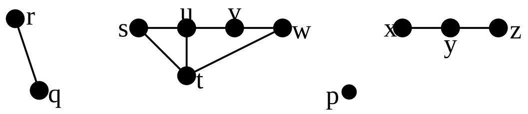
Remark.1. "connected" is an adjective, we apply only to graphs and to pairs of vertices(we never say " v is disconnected" when v is a vertex).
2. The phrase "u is connected to v" is convenient when writing proofs, but in adopting it we must clarify the distinction between connection and adjacency:

G has a u, v-path	uv $\in$ E(G)
u and v are connected u is connected to v	u and v are adjacent u is joined to v u is adjacent to v

3. The connection relation is an equivalence relation on V(G), verify!.
4. A *maximal* connected subgraph of G is a subgraph that is connected and is not contained in any other connected graph of G.

1.2.8. **Definition**. The **components** of a graph G are its maximal connected subgraphs, i.e. the subgraph that is connected and is not contained in any other connected subgraph of G.
The component (or graph) is **trivial** if it has no edges; otherwise it is **nontrivial**.
An **isolated vertex** is a vertex of degree 0.

Remark. The equivalence classes of the connection relation on V(G) are the vertex sets of components of G. An isolated vertex forms a trivial component, consisting of one vertex and no edge.
1.2.9. **Example**. The graph below has 4 components, one being an isolated vertex.



The vertex sets of the components are {p}, {q, r}, {s, t, u, v, w}, and {x, y, z}; these are the equivalence classes of the connection relation. #

Exercise1.2.1. Prove that if some vertex of a graph G is connected to all other vertices of G then G is connected.
Proof. Let u and v be any vertex in V(G). By the assumption, there exists x in V(G) such that u is connected to x and x is connected to v.
Because a u, x-path and x, v-path together contain a u, v-path, so u is connected to v.
Then every vertex is connected to every other, and G is connected. #

1.2.10. **Remark**. Components are pairwise disjoint; no two share a vertex.
Adding an edge with endpoints in distinct components combines them into one component. Thus adding an edge decreases the number of components by 0 or 1, and deleting an edge increases the number of components by 0 or 1.

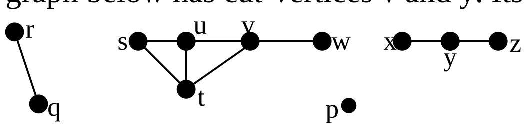
1.2.11. **Proposition**. Every graph with n vertices and k edges has at least n – k components.
Proof: An n-vertex graph with no edges has n components. By Remark 1.2.10, each edge added reduces this by at most 1, so when k edges have been added the number of components is still at least n – k. #

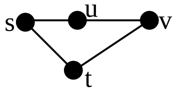
Remark. 1.When we obtain a subgraph by deleting a vertex, it must be a graph, so deleting the vertex also delete all edges incident to it.
2. Deleting a vertex or an edge can increase the number of components. In fact, deleting an edge can only increase the number of components by 1, deleting a vertex can increase it by many (consider the biclique $K_{1,m}$).

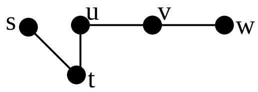
1.2.12. **Definition**. A **cut-edge** or **cut-vertex** of a graph is an edge or vertex whose deletion increases the number of components.
We write G – e or G – M for the subgraph of G obtained by deleting an edge e or a set of edges M.
We write G – v or G – S for the subgraph of G obtained by deleting a vertex v or a set of vertices S.
An **induced subgraph** is a subgraph obtained by deleting a set of vertices.
We write G[T] for $G - \overline{T}$,where $\overline{T} = V(G) - T$; this is the subgraph of G induced by T.

Remark. 1. When $T \subseteq V(G)$, the induced subgraph G[T] consists of T and all edges whose endpoints are contained in T.
2. The full graph is itself an induced subgraph, as are individual vertices.
3. A set S of vertices is an independent set if and only if the subgraph induced by it has no edges.

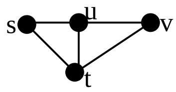
1.2.13. **Example**. The graph below has cut-vertices v and y. Its cut-edges are qr, vw, xy, and yz.



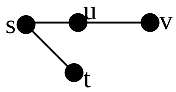
This graph has C_4 : 

and P_5 : 

as subgraphs but not as induced subgraphs.

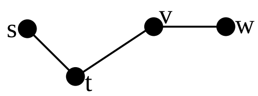
The subgraph induced by {s, t, u, v} is a kite 

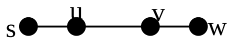
.

This graph 

is not an induced subgraph by {s, t, u, v}.

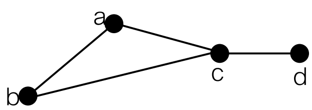
The graph P_4 does occur as an induced subgraph; it is the subgraph induced by {s, t, v, w}, also by {s, u, v, w}. #





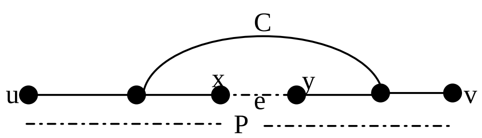
Exercise1.2.5. Let v be a vertex of a connected simple graph G.
Prove that v has a neighbor in every component of G – v.
Conclude that no graph has a cut-vertex of degree 1.
Proof. Since G is connected, the vertices in one component of G – v must have paths in G to every other component of G – v, and a path can only leave a component of G – v via v.
Thus v has a neighbor in every component of G – v.
If G – v has k components, $k \geq 2$, then $d_G(v) \geq k$, which implies there is no cut-vertex of degree 1.#

Exercise1.2.6. Consider the paw:



Maximal paths are a, c, b; a, b, c, d; b, a, c, d(two are maximum paths).
Maximal cliques are {c, d, {a, b, c} (one is a maximum clique).
Maximal independent sets are: {c}; {b, d}; {a, d}(two are maximum independent sets). #

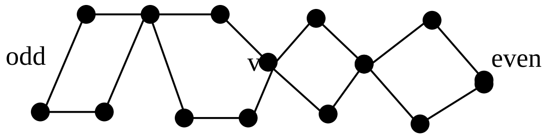
1.2.14.**Theorem**. An edge in a graph G is a cut-edge if and only if it belongs to no cycle.
Proof. Let e be an edge in a graph G(with endpoints x, y), and let H be the component containing e.
Since deletion of e affects no other component, it suffices to prove that H – e is connected if and only if e belongs to a cycle.
($\rightarrow$) Suppose that H – e is connected. This implies that H – e contains an x, y-path, and this path completes a cycle with e.
($\leftarrow$) Suppose that e lies in a cycle C. Choose $u, v \in V(H)$.
Since H is connected, H has a u, v-path P.
If P does not contain e, then P exists in H – e, so H – e is connected.
If P contains e, suppose by symmetry that x is between u and y on P.



Since H – e contains a u, x-path along P, an x, y-path along C, and a y, v-path along P.
The transitivity of connection relation implies that H – e has a u, v-path.
We did this for all $u, v \in V(H)$, so H – e is connected. #

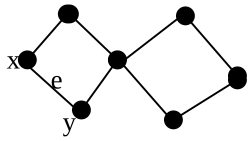
1.2.15. **Lemma**. Every closed odd walk contains an odd cycle.
Proof: We use induction on the length ℓ of a closed odd walk W.
Basis step: $\ell = 1$. A closed walk of length 1 traverses a cycle of length 1.
Induction step: $\ell > 1$. Assume the claim for closed odd walks shorter than W.

If W has no repeated vertex(other than first = last), then W itself forms a cycle of odd length.
If vertex v is repeated in W, then we view W as starting at v and break W into 2 v, v-walks.
Since W has odd length, one of these is odd and the other is even.



The odd one is shorter than W. By induction hypothesis, it contains an odd cycle, and this cycle appears in order in W. #

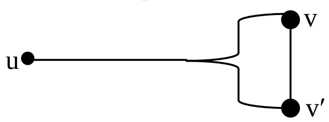
1.2.16. **Remark**. A closed even walk need not contain a cycle; it may simply repeat.
Nevertheless, if an edge e appears exactly once in a closed walk. Then W does contain a cycle through e.



Let x, y be the endpoints of e. Deleting e from W leaves an x, y-walk that avoids e.
By Lemma 1.2.5, this walk contains an x, y-path, and this path completes a cycle with e. #

1.2.17. **Definition**. A **bipartition** of G is a specification of two disjoint independent sets in G whose union is V(G).
An X, Y-**bigraph** is a bipartite graph with biparition X, Y.

1.2.18.**Theorem**. A graph is bipartite if and only if it has no odd cycle.
Proof: (→) Let G be a bipartite graph. Every walk alternates between the 2 sets of a bipartition, so every return to the original partite set happens after an even number of steps. Hence G has no odd cycle.
(←) Let G be a graph with no odd cycle. We prove that G is bipartite by constructing a bipartition of each nontrivial component.
Let u be a vertex in a nontrivial component H.
For each $v \in V(H)$, let f(v) be the minimum length of a u, v-path.
Since H is connected, f(v) is defined for each $v \in V(H)$.
Let $X = \{v \in V(H) : f(v) \text{ is even}\}$ and $Y = \{v \in V(H) : f(v) \text{ is odd}\}$.
An edge v, v′ within X or Y would create a closed odd walk using a shortest u, v-path, the edge vv′, and the reverse of a shortest u, v′-path.

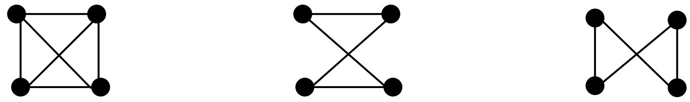


By Lemma 1.2.15, such a walk must contain an odd cycle, a contradiction.
Hence X and Y are independent sets. Also $X \cup Y = V(H)$, so H is an X, Y-bigraph. #

1.2.19. **Remark**. Theorem 1.2.18 implies that whenever a graph G is not bipartite, we can prove this statement by presenting an odd cycle in G. This is much easier than examining all possible bipartitions to prove that none work. When we want to prove that G is bipartite, we define a bipartition and prove that the 2 sets are independent; this is easier than examining all cycles.

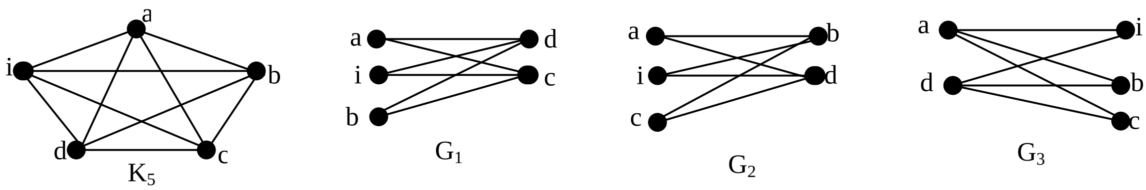
1.2.20. **Definition**. The **union** of graphs $G_1, G_2, \ldots, G_k$, written $G_1 \cup G_2 \cup \ldots \cup G_k$, is the graph with vertex set $\bigcup_{i=1}^{k} V(G_i)$ and edge set $\bigcup_{i=1}^{k} E(G_i)$.

1.2.21. **Example**. K_4 can be expressed as the union of two 4-cycles.

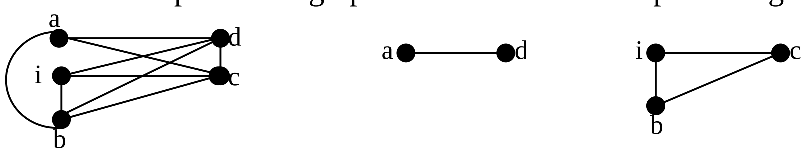


#

1.2.23. **Theorem**. K_n can be expressed as the union of k bipartite graphs if and only if $n \le 2^k$.
Proof: We use induction on k.
Basis step: k = 1. Since K_3 has an odd cycle and K_2 does not, K_n is itself a bipartite graph if and only if $n \le 2^1$.
Induction step: k > 1. We prove each implication using the induction hypothesis.
($\rightarrow$) Suppose that $K_n = G_1 \cup G_2 \cup \ldots \cup G_k$, where G_i is bipartite.
Case $n = 5 \le 2^3 = 8$.

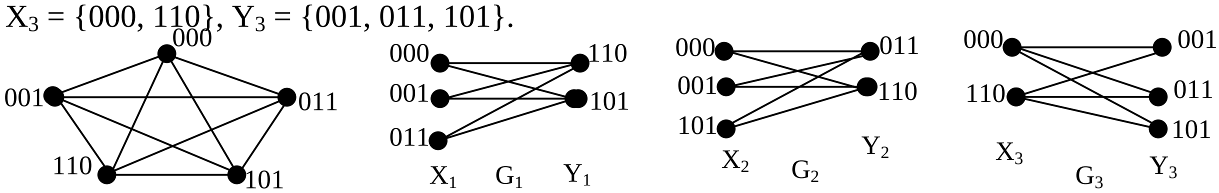


We partition the vertex set $V(K_n)$ into 2 sets X, Y such that G_k has no edge within X or within Y.
$V(K_5) = \{a, b, c, d, i\}$, $X = \{a, d\}$, $Y = \{b, c, i\}$
The union of the other k – 1 bipartite subgraphs must cover the complete subgraphs induced by X and by Y.



Applying the induction hypothesis to each yields $|X| \le 2^{k-1}$ and $|Y| \le 2^{k-1}$, so $n \le 2^{k-1} + 2^{k-1} = 2^k$.
($\leftarrow$) Suppose that $n \le 2^k$.
We partition the vertex set $V(K_n)$ into subsets X, Y, each of size at most 2^{k-1}.
By the induction hypothesis, we can cover the complete subgraph induced by either subset with k – 1 bipartite subgraphs.
The union of the i^{th} such subgraph on X with the i^{th} such subgraph on Y is a bipartite graph.
Hence we obtain k – 1 bipatite graphs whose union consists of the complete subgraphs induced by X and Y. The remaining edges are those of the biclique with bipartition X, Y.
Letting this be the k^{th} bipartite subgraph completes the construction. #

Exercise 1.2.31. Non-inductive proof of Theorem 1.2.23.
a) Given $n \le 2^k$, encode the vertices of K_n as distinct binary k-tuples.
Let G_i be the complete bipartite subgraph with bipartition X_i, Y_i, where X_i is the set of vertices whose codes have 0 in position i, and Y_i is the set of vertices whose codes have 1 in position i.
Since every two vertex codes differ in some position, thus $K_n = G_1 \cup G_2 \cup \ldots \cup G_k$.
To illustrate this: $n = 5 \le 2^3 = 8$, encode the vertices of K_5 as 000, 001, 011, 101, 110.
Let G_i be the complete bipartite subgraph with bipartition X_i, Y_i, i = 1, 2, 3 as follows:
$X_1 = \{000, 001, 011\}$, $Y_1 = \{110, 101\}$, $X_2 = \{000, 001, 101\}$, $Y_2 = \{011, 110\}$,
$X_3 = \{000, 110\}$, $Y_3 = \{001, 011, 101\}$.

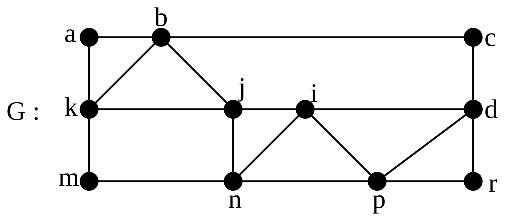


$K_5 = G_1 \cup G_2 \cup G_3$.

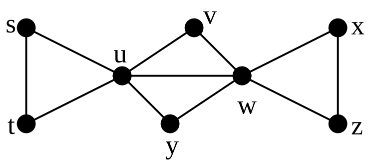
b) Given that K_n is a union of bipartite graphs $G_1, \ldots, G_k$, encode the vertices of K_n as distinct binary k-tuples.
For $1 \le i \le k$, let X_i, Y_i be a partition of G_i.
Assign vertex v the k-tuple$(a_1, \ldots, a_k)$, where $a_i = 0$ if $v \in Xi$ and $a_i = 1$ if $v \in Yi$ or $v \notin X_i \cup Y_i$.
Since every 2 vertices are adjacent and the edge joining them must be covered in the union, they lie in opposite partite sets in some G_i.
Therefore the k-tuples assigned to the vertices are distinct. Since the k-tuples are binary k-tuples, there are at most 2^k of them, so $n \le 2^k$. #

1.2.24. **Definition**. A graph is **Eulerian** if it has a closed trail containing all edges.
We call a closed trail a **circuit** when we do not specify the first vertex but keep the list in cyclic order.
An **Eulerian circuit** or **Eulerian trail** in a graph is a circuit or trail containing all edges.
An **even graph** is a graph with vertex degrees all even.
A vertex is **odd** [**even**] when its degree is odd[even].
A **maximal path** in a graph G is a path P in G that is not contained in a longer path.
When a graph is finite, no path can extend forever, so maximal(non-extendible) paths exist.

Example. The graph G below has an Eulerian circuit a, b, c, d, i, j, b, k, m, n, p, r, d, p, i, n, j, k, a.

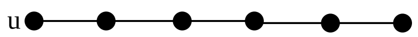


The graph H below has an Eulerian trail u, s, t, u, v, w, y, u, w, x, z, w.

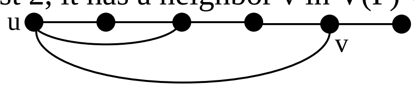


A maximal path in H : s, t, u, v, w, x, z. #

1.2.25. **Lemma**. If every vertex of a finite graph G has degree at least 2, then G contains a cycle.(The proof uses the technique called extremality.)
Proof: Let P be a maximal path in G, and let u be an endpoint of P.



Since P cannot be extended, every neighbor of u must already a vertex of P.
Since u has degree at least 2, it has a neighbor v in V(P) via an edge not in P.

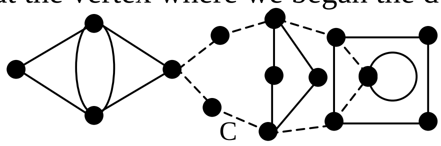


The edge uv completes a cycle with the portion of P from v to u. #

Remark. The proof of Lemma 1.2.25 is an example of an important technique of proof in graph theory that we call **extremality**. When considering structures of a given type, choosing an example that is extreme in some sense may yield useful additional information. For example, since a maximal path P cannot be extended, we obtain the extra information that every neighbor of an endpoint of P belongs to V(P).

In a sense, making an extremal choice goes directly to the important case. In Lemma 1.2.25, we could start with any path. If it is extendible, then we extend it. If not, then something important happens.

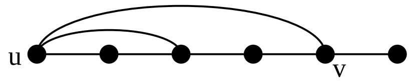
1.2.26. **Theorem**. A graph G is Eulerian if and only if it has at most one nontrivial component and its vertices all have even degree.
Proof: (→) Suppose that G has an Eulerian circuit C. Each passages of C through a vertex uses 2 incident edges, and the first edge is paired with the last at the first vertex.
Hence every vertex has even degree. Also, two edges can be in the same trail only when they lie in the same component, so there is at most one nontrivial component.
(←) Assuming that the condition holds, we obtain an Eulerian circuit using induction on the number of edges, m.
Basis step: m = 0. A closed trail consisting of one vertex suffices.
Induction step: m > 0. Assume the claim holds for a graph with k (< m) edges and has at most one nontrivial component and its vertices all have even degree.
With even degrees, each vertex in the nontrivial component of G has degree at least 2.
By Lemma 1.2.25, the nontrivial component has a cycle.
Let G′ be the graph obtained from G by deleting E(C).
Since C has 0 or 2 edges at each vertex, each component of G′ is also an even graph.
Since each component also is connected and has fewer than m edges, we can apply the induction hypothesis to conclude that each component of G′ has an Eulerian circuit.
To combine these into an Eulerian circuit of G, we traverse C, but when a component of G′ is entered for the first time we detour along an Eulerian circuit of that component.
This circuit ends at the vertex where we began the detour.



When we complete the traversal of C, we have completed an Eulerian circuit of G. #

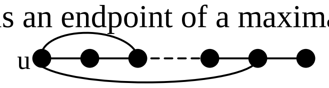
1.2.27. **Proposition**. Every even graph decomposes into cycles.
Proof: In the proof of Theorem 1.2.26, we note that every even nontrivial graph has a cycle, and that the deletion of a cycle leaves an even graph. Thus this prposition follows by induction on the number of edges. #

1.2.28. **Proposition**. If G is a simple graph in which every vertex has degree at least k, then G contains a path of length at least k. If k ≥ 2, then G also contains a cycle of length at least k+1.
Proof: Let u be an endpoint of a maximal path P in G. Since P does not extend, every neighbor of u is in V(P). Since u has at least k neighbors and G is simple, therefore P has at least k vertices other than u and has length at least k.
If k ≥ 2, then the edge from u to its farthest neighbor v along P completes a sufficiently long cycle with the portion of P from v to u.



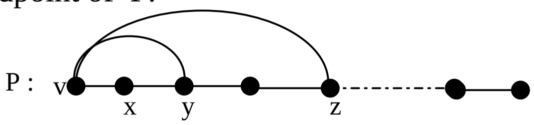
#

1.2.29. **Proposition**. Every graph with a nonloop edge has at least 2 vertices that are not cut-vertices.
Proof: If u is an endpoint of a maximal path P in G, then the neighbors of u lie on P.



Since P – u is connected in G – u, the neighbors of u belong to a single component of G – u, and u is not a cut-vertex. #

Exercise1.2.39. Suppose that every vertex of a loopless finite graph G has degree at least 3. Prove that G has a cycle of even length.
Proof. Let P be a maximal path in G with v is an endpoint of P.
Then v has at least 3 neighbors on P.
Let x, y, z be 3 such neighbors of v in order on P.
Consider 3 v, y-paths: the edge vy;
the edge vx followed by the x, y-path in P; and
the edge vz followed by the z, y-path in P.



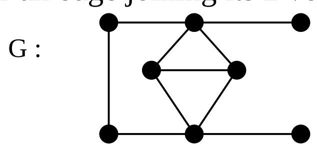
These paths share only their endpoints, so the union of any 2 is a cycle.
By the pigeonhole principle, 2 of these paths have lengths with the same parity (odd or even). The union of these 2 paths is an even cycle. #

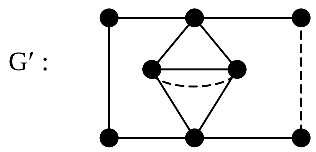
1.2.30. **Remark**. **Maximum** means "maximum-sized" and **maximal** means "no larger one contains this one".

1.2.31. **Lemma**. In an even graph, every maximal trail is closed.
Proof: Let T be a maximal trail in an even graph. Every passages of T through a vertex v uses 2 edges at v, none repeated. Thus when arriving at a vertex v other than its initial vertex, T has used an odd number of edges incident to v. Since v has even degree, there remains an edge on which T can continue. Hence T can only end at its initial vertex.
In a finite graph, T must indeed end. We conclude that a maximal trail must be closed. #

1.2.32. **Theorem** 1.2.26.-Second proof using extremality directly.
Proof: Suppose a graph G has at most one nontrivial component and its vertices all have even degree. Let T be a trail of maximum length; T must also be a maximal trail.
By Lemma 1.2.31. T is closed.
Suppose that T omits some edge e of G.
Since G has only one nontrivial component, G has a shortest path from e to the vertex set of T.
Hence some edge e′ not in T is incident to some vertex v of T.
Since T is closed, there is a trail T′ that starts and ends at v and uses the same edges as T.
We now extend T′ to obtain a longer trail than T.
This contradicts the choice of T, and hence T traverses all edges of G. #

1.2.33. **Theorem**. For a connected nontrivial graph with exactly 2k odd vertices, the minimum number of trails that decompose it is max{k, 1}.
Proof: A trail contributes even degree to every vertex, except that a non-closed trail contributes odd degree to its endpoints. Therefore, a partition of edges into trails must have some non-closed trail ending at each odd vertex.
Since each trail has only 2 ends, we must use at least k trails to satisfy 2k odd vertices.
We also need at least one trail since G has an edge, and Theorem 1.2.26 implies that one trail suffices when k = 0.
It remains to show that k trails suffice when k > 0.
Given such a graph G, we pair up the odd vertices in G (in any way) and form G′ by adding for each pair an edge joining its 2 vertices as illustrated below.

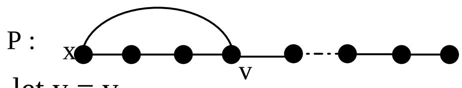




The resulting graph G′ is connected and even, so by Theorem 1.2.26, it has an Eulerian circuit C.
As we traverse C in G′, we start a new trail in G each time we traverse an edge of G′ – E(G).
This yields k trails decomposing G. #

Exercise1.2.41. Let G be a connected graph with at least 3 vertices.
Prove that G has 2 vertices x, y such that G – {x, y} is connected and x, y are adjacent or have a common neighbor.

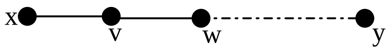
Proof. Let P be a longest path in G with x as an endpoint.

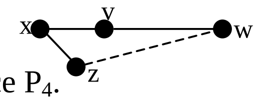


Let v be a neighbor of x on P.
Note that P has at least 3 vertices. If G – x – v is connected, let y = v.
Otherwise, a component cut off from P – x – v in G – x – v has at most one vertex; call it w.
The vertex w must be adjacent to v, since otherwise we could build a longer path.
In this case, let y = w. #

Exercise1.2.42. Let G be a connected simple graph that does not have P_4 or C_4 as an induced subgraph. Prove that G has a vertex adjacent to all other vertices.

Proof. Consider a vertex x of maximum degree in G.
If x has a nonneighbor y, let x, v, w be the beginning of a shortest path to y (w may equal y).





Since d(v) ≤ d(x), some neighbor z of x is not adjacent to v.
If z is adjacent to w, then {z, x, v, w} induce C_4; otherwise, {z, x, v, w}induce P_4.
Thus x must be adjacent to all other vertices. #

Exercise1.2.33. Prove that the edges of a connected graph with 2k odd vertices can be partitioned into k trails if k > 0 by using induction on k.

Proof. If k = 1, we add an edge between the two odd vertices, obtain an Eulerian circuit, and delete the added edge to get a trail.
If k > 1, let P be a path between 2 odd vertices. The graph G′ = G – E(P) has 2k – 2 odd vertices, since we have changed degree parity only at the ends of P. We apply the induction hypothesis to each component of G′ that has odd vertices. Any component not having odd vertices has an Eulerian circuit that contains a vertex of P; we split it into P to avoid having an additional trail.
Altogether, we have used the desired number of trails to partition E(G). #

Homework 2: 1.2.15, 1.2.18, 1.2.20, 1.2.26, 1.2.38 due on June 25.