Simple Graph and Connectivity

Since the second project is built directly based on the first, in most parts only changes are included here.

Thanks to the work of Kyle Miller, proves and constructions of some structure are now much easier.

Definitions, Corollaries and Theorems

Euler Circuit

Definition. Euler Circuit (`is_euler_circuit`).

Sometimes written as "Euler Cycle". For consistency with mathlib, "circuit" is preferred. A circuit p in a graph G is a **Euler circuit** of graph G if this circuit contains all edges in this graph. Formally,

$$\forall e, e \in G.\,\text{edge\_set} \implies e \in p.\,\text{edges} \tag{1}$$

(`is_eulerian`). It is also said that a graph G is **Eulerian** if it contains a Euler circuit. i.e., there is a p walk in G s.t. p is a Euler circuit, and all vertices in G are visited by such circuit. Thus,

```
def is_euler_circuit {v : V} (p : G.walk v v) : Prop :=
  p.is_circuit ∧ (∀ ⦃e : sym2 V⦄, e ∈ G.edge_set → (e ∈ p.edges)) ∧ ∀ ⦃u : V⦄, u ∈
p.support
```

Reachable

Definition. Reachable.

For any two vertices v w in graph G, to say v is **reachable** to w (or vice versa, since G is undirected), is to say that there is a **walk** (defined in `.simple_graph.connectivity`) from v to w,

$$\exists p : G.\,\text{walk v w} \tag{2}$$

Corollary. *Reachable* is an equivalence relation.

All Supports on a Walk are Reachable to Each Other

For any two vertices v w In graph G, v is reachable to w if there is a walk passing through both of them. Formally, we say

```
theorem reachable_if_support {u v w x : V} (p : G.walk u x) (h1 : v ∈ p.support) (h2
: w ∈ p.support) [decidable_eq V]:
  G.reachable v w
```

Existence of trail / path between reachable vertices

Theorem. Existence of trail / path between reachable vertices.

$$\begin{aligned} &\forall v\ w : V, G.\ \text{reachable v w} \implies \exists p : G.\ \text{walk v w}, p.\ \text{is\_trail} \\ &\forall v\ w : V, G.\ \text{reachable v w} \implies \exists p : G.\ \text{walk v w}, p.\ \text{is\_path} \end{aligned} \tag{4}$$

This is now a simple result from the new method of constructing a path out of an existing walk.

The Proof of Euler's Theorem

Eulerian Graph is Connected (Non-Zero Degree Vertices Only)

This is now a proven theorem.

All Vertices in Eulerian Graph have Even Degrees

> Since the graph is Eulerian, it has a circuit p passing all its edges, thus all its vertices (and since the graph is connected, no vertex will have a degree of zero).
>
> Pick an arbitrary vertex v.
>
> Suppose the Eulerian circuit p visits v k times. Each time it visits v, it must come from one other vertex via one edge, and leaves to another, thus adds up 2 degree. Hence, v will have a degree of $2k$. Note that these 2 edges must be distinct and not used before, as in a Euler circuit p, all edges are visited exactly once.

There is still an important part left incomplete. Basically, an induction over `list (sym2 V)` would be required, and inference between neighbourhood elements of a list is necessary. More lemmas would be required in this project.

Graph with All Vertices of Even Degrees is Eulerian

> omitted for now.

Reflection

I have to admit that one big issue of my project in phase 2 in the lack of time. There are too many concurrent courseworks going on and I planned so badly that I had spared too little time here.

I managed to fix most of the sorries in the previous project file, as well as trying to complete the example to the largest extent. Yet, it should still being considered that manipulating data in tactic mode is hard, especially when there is a lack of "bruteforce method" —— something that I can exemplify all the cases (with data!) and prove each of them, especially in a highly nested structure in a hypothesis.

I will try to push this project further. Good news is, I will have this project as the only coursework for more than 2 weeks until the next deadline :D