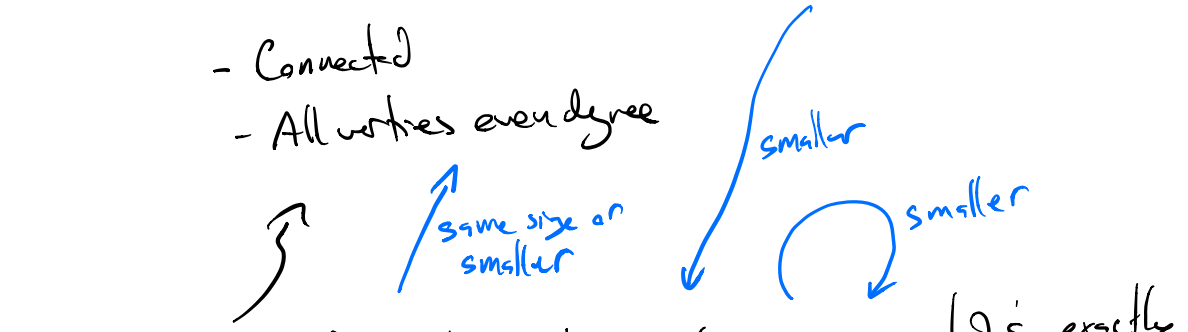
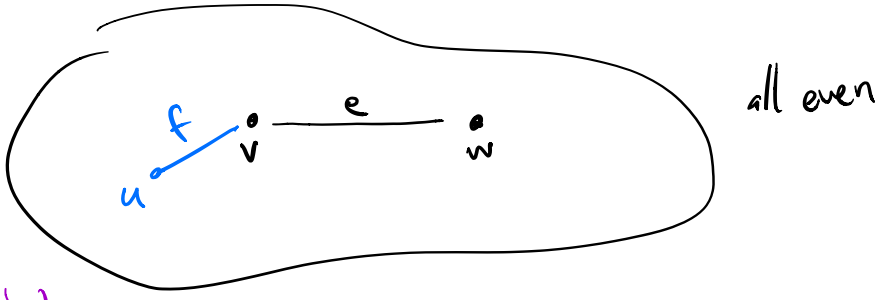
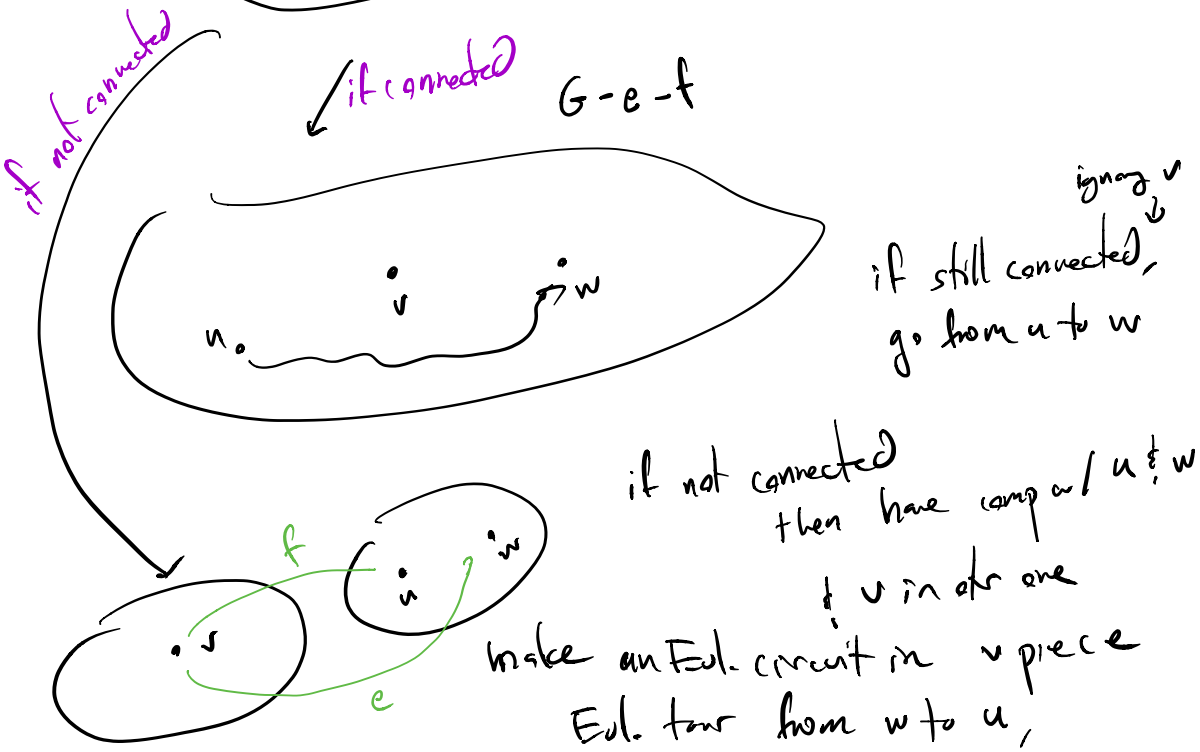
Last time:

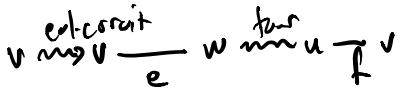
Found existence of Eulerian circuit in a graph which was

- Connected
- All vertices even degree









Minimal path problem

Setup: (most basic version)

Given a pseudograph G, vertices $v \neq w$, find a walk from v to w with minimal length.

Generalization: can assign weights ("lengths") to edges

$l : E \longrightarrow \mathbb{R}_{\geq 0}$ and use these in evaluating length of a walk.

Reduction: Can assume graph is simple.

- loops never help
- replace multiple edges by one with minimal length.

What's a "good" procedure for this?

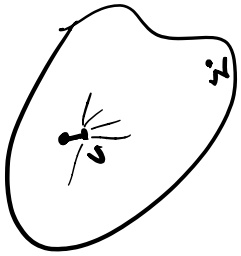
How is graph represented?

- graph as a matrix good for "small bushy graphs" (cities/airports, routes)

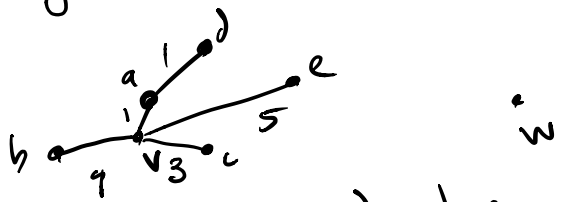
- graph as a collection of vertices and edges w/ incidence relation - good for "sparse" graphs (social networks w/ large populations)

Brief outline of Dijkstra procedure to find minimal path from v to w

1. start w/ v, put it in box Q: "vertices for which we know minimal path from v to"



2. at each step, look for a vertex which is closest to v from among all not in Q, add it to Q



v, a, d, c, b, e

For step 2, important observations:

- every vertex not yet in Q is further than any in Q
- new vertex is adjacent to one in Q

for the new vertex u, can compute distance from v to u as:

edge connecting x & u

$$d(v,u) = d(v,x) + \ell(xu)$$

distance = length of min. path from v to u

for some $x \in Q$

$$\Rightarrow d(v,u) = \min_{\substack{x \in Q, \\ x \text{ adjacent to } u}} \{ d(v,x) + \ell(xu) \}$$

Roughly, at each stage, compute $\sim n^2$ distances

$\sim n$ times $\rightarrow$ $\sim n^3$ steps.

turns out, can actually do it in $\sim n^2$ steps

Strategy:

- keep track of all distances found each iteration
- only check distances we don't already know

New procedure:

initialization {
for each vertex assign a number $b: V \to \mathbb{R}_{\geq 0} \cup \{\infty\}$ which keep track of "current best distance found from v"
these all start at "∞", except $b(v) = 0$.

{ Q: known, minimal distance vertices, $Q=\emptyset$ initially.

Procedure:

- Assumption: whenever we are in this loop, the vertex $u \notin Q$ with minimum b value, satisfies $b(u) = d(v,u)$
- At each step, $b(w)$ represents optimal dist. obtainable by a path including only w & vertices in Q.

- Choose $u \notin Q$ with $b(u)$ minimal ($u=v$ at first)
  put u in Q, assign it a distance of $d(v,u)=b(u)$
- For each vertex $w \notin Q$, replace $b(w)$ with
  $\min\{b(w), d(v,u) + \ell(uw)\}$
  $u \in Q$
  u adj to w
  ($\sim n$ steps)
- repeat $\sim n$ times