Admin:

Quiz #2 tomorrow night (Wed 11/13) 7:30-9:30 pm
Room 50-340 (Walker)
Coverage through L18 (today) (but not much on today's chapter)
If you need a conflict/makeup, see TAs (now)

Note: No lecture Thursday, or recitation on Friday.

Today:

Readings: CLRS §24.3, 24.5, Wagner-Willhelm paper (posted)

Dijkstra's algorithm:
- review
- correctness
- running time

A* search (single source s & single destination t)
- idea
- discussion
Dijkstra's Algorithm

- Assume $w(u,v) \geq 0$ (non-neg. edge wts.)
- Maintain a set $S$ of vertices whose final shortest path weights have been determined (i.e. $v \in S \Rightarrow \forall u, d = \delta(s,v))$.
- Repeatedly select $u \in V - S$ with minimum shortest path estimate $u.d$, add $u$ to $S$, and relax all edges out of $u$.

Dijkstra pseudocode:

```
Dijkstra (G, w, s)

Init (G, s)  // $v.d = \infty$ except $s.d = 0$
S ← ø
Q ← G.V  // priority queue, keyed by d

while Q ≠ ø
    u ← ExtractMin(Q)
    S ← S ∪ {u}
    for each edge (u,v)
        relax(u,v)
```

implicit DecreaseKey operation on $v$ in $Q
**Correctness of Dijkstra**:

**Lemma**: when $u$ added to $S$, $u.d = \delta(s,u)$

(Thus, along any shortest path, edges relaxed in order.)

**Proof of Lemma**:

Claim: (1) $u \in S \Rightarrow u.d = \delta(s,u)$

(2) $v \in Q \Rightarrow v.d = \min_{u \in S} (\delta(s,u) + w(u,v))$

[If claim (1) correct \(\Rightarrow\) Dijkstra correct]

To show (1): Suppose not (\&u is 1st violation)

If path to $u$ through $x$ is shorter, then subpath from $x$ to $u$ must have negative length. Contradiction! \(\blacksquare\)
Dijkstra Running Time

\[ \Theta(v) \text{ inserts into Q} \]
\[ \Theta(v) \text{ Extract Min operations} \]
\[ \Theta(E) \text{ Decrease Key operations} \]

Implement Q as array:

- Insert \( \Theta(1) \) time
- Extract Min \( \Theta(v) \) time
- Decrease Key \( \Theta(1) \) time

\[ \Rightarrow \Theta(v + E + 1) = \Theta(v^2) \]

Implement Q as binary heap:

- Insert \( \Theta(\log v) \) time
- Extract Min \( \Theta(\log v) \) time
- Decrease Key \( \Theta(\log v) \) time

\[ \Rightarrow \Theta(v \log v + E \log v) \]

Implement Q as Fibonacci Heap:

- Insert \( \Theta(1) \) time
- Extract Min \( \Theta(\log v) \) time
- Decrease Key \( \Theta(1) \) time

\[ \Rightarrow \Theta(v \log v + E) \]

\[ \uparrow \]

\[ \text{wow! (amortized)} \]

\[ \text{Best alg!} \]

not covered in 6.006!
Types of shortest-path problems:

1. SSSD - single source (s) → single destination (t)
2. SS - single source (s) to all destinations (∀t)
3. AP - all pairs: all sources (s) to all destinations (t)

- Dijkstra & Bellman-Ford are SS solvers
- Best way to solve AP is often to run SS solver from each possible source
- What about SSSD problems? How does knowing t help?
  "goal-directed search"

Idea 1: (Dijkstra with early stopping)

Run Dijkstra from source s

Stop when t is added to S.

(Then t.d = δ(s,t) as argued earlier.)

This is certainly correct

and works well when t is close to s.
**Idea 2:** Use heuristic estimate of distance to goal

Let $\lambda(u) =$ heuristic estimate of $\delta(y_t)$

*recall: this is true s.p. distance*

E.g., "crows fly distance" on a map

*How?*

**Idea 3:** Key elements in $Q$ not just by $u.d.$, but by $u.d. + \lambda(u)$

Length at best path from $s$ to $t$ path going through $u$

gives priority to $u_2$, since it is close to $t$.

Does this work?

Depends on properties of $\lambda$:

**Def:** $\lambda$ is **admissible** (aka optimistic) if

$$\forall u \lambda(u) \leq \delta(y_t) \quad [\lambda \text{ lower bound for } \delta]$$

This is helpful, but we usually want a little more:

**Def:** $\lambda$ is **consistent** (aka monotone or feasible) if

$$\forall (u,v) \in E \quad \lambda(u) \leq w(u,v) + \lambda(v)$$
Idea 4: Redefine edge weights

\[ w'(u,v) = w(u,v) - \lambda(u) + \lambda(v) \]

If \( \lambda \) is consistent, then \( w'(u,v) \geq 0 \) for all \( u, v \).

Note "telescoping":

Let \( \delta' = \) shortest path fn using \( w' \) weights
\[ \delta'(u,v) = \) shortest path from \( u \) to \( v \) with \( w' \)

length of path \( p = (v_1, v_2, \ldots, v_k) \)
\[ \lambda(p) = w(v_1, v_2) + \cdots + w(v_{k-1}, v_k) \]
\[ \lambda'(p) = w'(v_1, v_2) + \cdots + w'(v_{k-1}, v_k) \]
\[ = l(p) + \lambda(t) - \lambda(s) \]

At constant. Indep of \( p \).

\[ \therefore \] identity of shortest paths preserved
(all \( s \to t \) path lengths changed by same amount)
Behavior of Dijkstra changes from to

Reason: (example)

- in right direction
  - $w = 10 - 90 + 80 = 10$
  - $w' = 10 - 80 + 80 = 10$
  - no change

- to right direction
  - $w = 10 - 90 + 80 = 10$
  - $w' = 10 - 80 + 90 = 20$
  - doubles

Searches preferentially towards $t$.0
Note that if $G$ is embedded in $\mathbb{R}^2$ or $\mathbb{R}^3$, say, and $w(u,v) =$ (Euclidean) distance from $u$ to $v$ then $\lambda_u =$ Euclidean distance from $u$ to $t$

is consistent, by $\Delta$-inequality

![Diagram]

$$\lambda_u \leq w(u,v) + \lambda_v$$

So we can use Dijkstra with modified weights $w' (u,v) = w(u,v) - \lambda(u) + \lambda(v)$

(i.e., $A^+$)