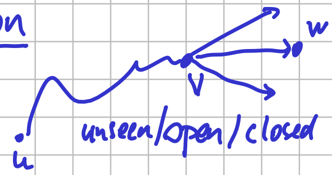
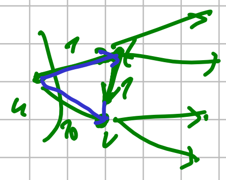
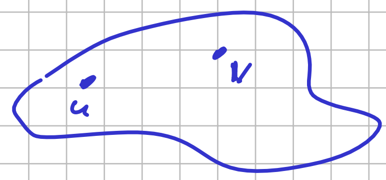
Relaxation

$h(v)$



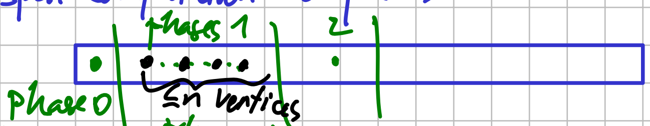
$h(w) \leftarrow \min(h(w), h(v) + \ell(vw))$





[BFM - Bellmann, Ford, Moore

- Relaxation + keep open vertices in a queue
- Split computation to phases:



ASSUMING NO NEG CYCLES

- by Lemma, BFM stops after at most n phases
  - every shortest walk is a path → at most n-1 edges
  - at end of phase n-1, $h(v) \le d(u,v)$ → $h(v)$ cannot change any longer
  - one more phase to close all open vertices
- complexity: 1 phase takes $O(m)$ time → $O(nm)$ time for all phases
  $O(\sum_v \deg(v))$

Lemma: At the end of phase i $\forall v$ $h(v) \le$ length of the shortest of uv-walks with at most i edges

Proof: By induction on i.

- $i=0$: trivially true
- $i-1 \to i$

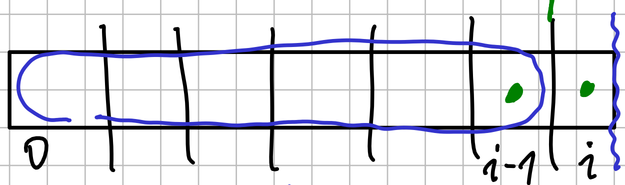
consider vertex v at the end of phase i

walks with $\le i-1$ edges: OK from previous phase (IH)

new walks with exactly i edges

$\ell(W) = \ell(W') + \ell(pv)$

by IH: at the end of phase $i-1$: $h(p) \le \ell(W')$ ☆



in phase at most $i-1$ p got the value ☆ ⇒ p was opened

In phase at most i it's closed & relaxed

p relaxed: $h(v) \le h(p) + \ell(pv) \le \ell(W)$
$\le \ell(W')$

Other:

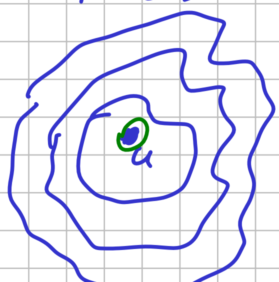
- stack (Pape) ... exponential worst case
- round-robin relax $v_1 v_2$ —— v_n & repeat → $\le n$ phases $O(nm)$

Dijkstra's alg.: select v open with $h(v)$ minimal.

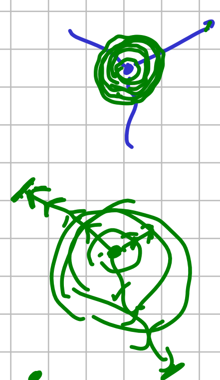
ASSUMING NO NEG. EDGES

Theorem: On graphs with no neg. edges, D.a. closes each vertex at most once] and vertices are closed in the order of increasing $d(u,-)$.]

Like BFS



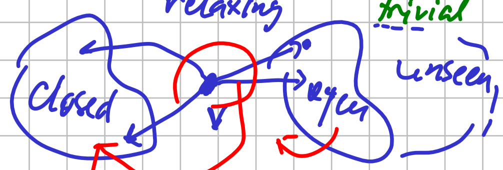
BFS layers

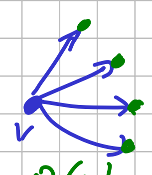


Proof: Invariant: values of closed $\le$ value of current $\le$ value of open

relaxing

trivial





$O(m)$

Corr: # relaxations $\le n$

Time Complexity: trivial: 1 step takes $O(n) \to O(n^2)$

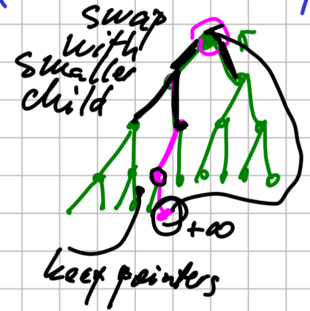
(step = relax.)

∴ all relaxations process $O(m)$ edges together

Idea: use a DS to find min. ("heap") — Insert T_I, ExtractMin T_X, Decrease T_D → D.a. runs in time $O(n \cdot T_I + n \cdot T_X + m \cdot T_D)$

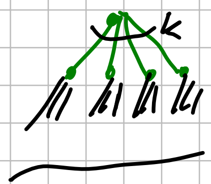
DS	T_I n	T_X n	T_D m	D.a.
array	1	n	1	n^2
binary heap	$\log n$	$\log n$	$\log n$	$m \log n$
k-ary heap	$\frac{\log n}{\log k}$	$k \cdot \frac{\log n}{\log k}$	$\frac{\log n}{\log k}$	$m \cdot \frac{\log n}{\log m/n}$
Fibonacci heap	1	$\log n$	1	$\underbrace{m}+\underbrace{n \log n}$

opt. for real lengths



$n \log n$

$\log_k n$ levels $= \frac{\log n}{\log k}$



$m \frac{\log n}{\log k} + n \cdot k \cdot \frac{\log n}{\log k}$

$\underbrace{m \log n \cdot \frac{1}{\log k}}_{\text{decreases with } k} + \underbrace{n \log n \cdot \frac{k}{\log k}}_{\text{increases with } k}$

find k: $m \cancel{\log n} \frac{1}{\cancel{\log k}} = n \cancel{\log n} \cdot \frac{k}{\cancel{\log k}}$

$m = n \cdot k \to k := \max\left(\frac{m}{n}, 2\right)$

① dense: $m \sim n^2$ $\quad \log m/n = \log n \to n^2$

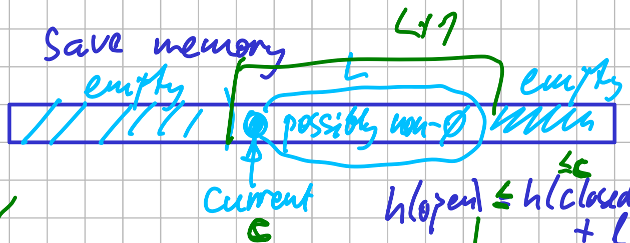
② sparse: $m \sim n$ $\quad k = \text{const.} \to m \log n$

Integers! $\forall e \; \ell(e) \in \{0 \dots L\}$

① index array by $h(v)$ $\quad \forall v \; h(v) \in \{0 \dots nL\}$ (finite)

array of buckets	1	1 + nL total	1	m + nL

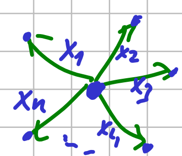
1^{++} Save memory



$h(\text{open}) \le h(\text{closed}) + \ell(\text{edge})$, $\ell(\text{edge}) \le L$, $h(\text{open}) \le c + L$

→ index the array mod $(L+1)$ → space $O(L+n)$

use D.a. for sorting



Sorting of reals → D.a.

$\Omega(n \log n)$ lower bound

$[0 \dots\dots 1]$ n random number $\in U(0,1)$

for reals: $O(n)$ expected time