Handin 9

Thomas Vinther & Jens Kristian Refsgaard Nielsen

28-11-18

1 Grid graph

Definition: We call a graph a grid graph when for some k:

$$V = \{v_{i,j} \mid 1 \leq i \leq k, 1 \leq j \leq k\} \tag{1.1}$$

$$\begin{aligned} E = &\{(v_{i,j}, v_{i,j+1}) \mid 1 \leq i \leq k, 1 \leq j < k\} \\ &\cup \{(v_{i,j}, v_{i,j-1}) \mid 1 \leq i \leq k, 1 < j \leq k\} \\ &\cup \{(v_{i,j}, v_{i+1,j}) \mid 1 \leq i < k, 1 \leq j \leq k\} \end{aligned} \tag{1.2}$$

1.1 a)

Let $n =\mid V \mid$ and $m =\mid E \mid$, we wish to calculate n and m as a function of k.
It is fairly obvious that $n(k) = k^2$.
Observe that each of the sets in the union of E contain $k(k-1)$ edges, so in total $m(k) = 3k(k-1) = 3k^2 - 3k$

1.2 b)

We wish to determine the runtime of Dijkstra(G, w, s) for $s = v_{1,1}$.
Note that if $m = o(n^2/\lg n)$ the runtime is $O(m \lg n)$. Recall that

$$o(g(n)) = \{f(n \mid \forall c > 0 \exists n_0 > 0 \forall n \geq n_0 : 0 \leq f(n) < cg(n)\}$$

$$3k^2 - 3k \stackrel{?}{=} o(k^4/\lg(k^2)) = o(k^4/(2\lg(k))$$

We prove this by induction.
Induction start: k=2

$$3 \cdot 2^2 - 3 \cdot 2 = 6$$

$$2^4/(2\lg(2)) = 2^3 > 6$$

Induction step: assume the statement holds for k.

$$3(k+1)^2 - 3(k+1) = (3k^2 - 3k) + 6k \stackrel{\text{induction}}{<} k^4/(2\lg(k)) + 6k \tag{1.3}$$

$$(k+1)^4/2\lg(k+1) = k^4/(2\lg(k+1)) + \frac{4k^3 + 6k^2 + 4k + 1}{2\lg(k+1)} \tag{1.4}$$

$$2\lg(k+1)6k \stackrel{\text{intimidation}}{<} 4k^3 + 6k^2 + 4k + 1 \tag{1.5}$$

For large k we have $k^4/\lg(k) \overset{\text{asymptotic}}{=} k^4/\lg(k+1)$ and combining (1.3) with (1.4) and (1.5) we have

$$\begin{aligned} 3(k+1)^2 - 3(k+1) &\overset{(1.3)}{<} k^4/(2\lg(k)) + 6k \overset{\text{asymptotic}}{\leq} k^4/(2\lg(k+1)) + 6k \\ &\overset{(1.5)}{<} k^4/(2\lg(k+1) + \frac{4k^3 + 6k^2 + 4k + 1}{2\lg(k+1)} \\ &\overset{(1.4)}{=} (k+1)^4/2\lg(k+1) \end{aligned}$$

Completing the induction. So Dijkstra runs in $O(k^2 \lg(k))$.

1.3 c)

We wish to compute $\delta(s,v)$ for all verticies. Consider the following dynamic algorithm. Out of bounds exceptions are treated as ∞.

Time	Line nr	Pseudocode
1	1	$\delta(s,s) = 0$
k	2	for $j = 2$ to k
1	3	$\quad\delta(s, v_{1,j}) = \delta(s, v_{1,j-1}) + w(v_{1,j-1}, v_{1,j})$
k	4	for $l = 2$ to k
k	5	$\quad$for $p = 1$ to k
1	6	$\quad\quad\delta(s, v_{l,p}) = \min\{\delta(s, v_{l-1,p}) + w(v_{l-1,p}, v_{l,p}),$
		$\quad\quad\quad\delta(s, v_{l,p-1}) + w(v_{l,p-1}, v_{l,p})\}$
k	7	$\quad$for $p = 0$ to $k-1$
1	8	$\quad\quad\delta(s, v_{l,k-p}) = \min\{\delta(s, v_{l,k-(p-1)}) + w(v_{l,k-(p-1)}, v_{l,k-p}),$
		$\quad\quad\quad\delta(s, v_{l,k-(p)})\}$

This algorithm takes k^2 extra space to store the δ values. And clearly runs in $O(k^2)$.
Correctness: In the bottom row the direct path is shortest due to the assumption that all weights are non negative. In the succeding rows we see that the shortest path to a vertex will come from directly below, to the left or to the right. Line 5 and 6 checks if it is from below or to the left, and lines 7 and 8 implicitely check left, below and right, and every possible entry is checked.