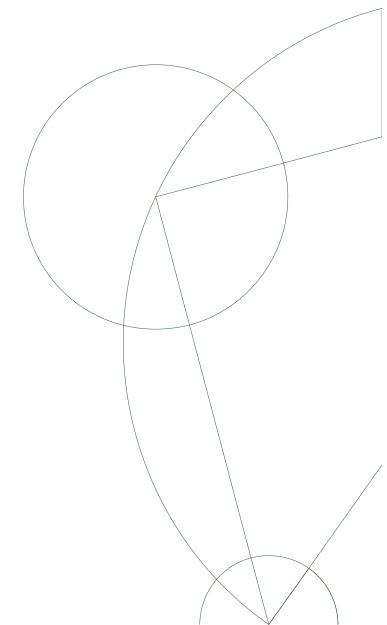


Diskret Matematik og Algoritmer Aflevering 10g

Adam Ingwersen, Aske Fjellerup, Peter Friborg

Datalogisk Institut Københavns Universitet

January 16, 2017



1

 \mathbf{a}

Ved hjælp af logaritmereglerne for et produkt, kan følgende omskrivninger foretages:

$$\begin{aligned} -log(w(e_1) \cdot w(e_2) \cdot \ldots \cdot w(e_{k-1}) \cdot w(e_k)) \\ -(log(w(e_1)) + log(w(e_2)) + \ldots + log(w(e_{k-1})) + log(w(e_k))) \\ - \sum_{i=1}^{n} log(w(e_i)) \end{aligned}$$

b)

Her tænkes det at modificere Dijkstra's algoritme via modifikation af RELAX operationen på en Edge. Det er klart, at eftersom at der arbejdes med vægte $w \in 0,1$, skal der inkoorporeres et multiplikativt element. Det observeres, at RELAX bliver kaldt for hver en Vertex nærliggende en vilkårlig Vertex, v. Hertil anskuer RELAX bagudrettet den bedste (billigste) ved fra v til u som en akkumulation af optimale vertex/edge kombinationer:

Algorithm 1 Modifikation af Dijkstra's algoritme til sandsynligheds-vægte

```
1: function Modified Relax(u, v, w)
      if \ v.d < u.d + \log(w(u,v))
         v.d = u.d + log(w(u,v))
3:
4:
         v.\pi = u
1: function DIJKSTRA(G, w, s)
      INITIALIZE-SINGLE-SOURCE (G, s)
2:
      S =
3:
      Q = G.V
4:
      while Q \neq
5:
         u = {\tt EXTRACT-MAX(Q)}
6:
7:
         S = S \cup u
         for each vertex v \in G.adj[u]
8:
             RELAX(u, v, w)
9:
```

 $\mathbf{2}$

2.1 a)

Dijkstra's algoritme og BFS i scenariet med enheds-vægte, resulterer i de samme værdier i alle kunder. Dijkstra's anvender en prioritetskø fremfor FIFO-kø (som i BFS), hvorfor Dijkstra's er asymptotisk langsommere end BFS, når w=1. Lad os se, hvorfor disse algoritmer er ækvivalente for w=1: BFS finder den korteste vej fra s til u - her menes der med den korteste vej, den vej med færrest skridt. Dijkstra's finder den mindst omkostningsfulde vej fra s til u. Hvis omkostningerne for hver edge er ens, vil den korteste vej pr. definition være den billigste.

2.2 b)

I BFS angiver farven **sort** for en node, at alle nærliggende noder er besøgt. I konteksten for Dijkstra's algoritme, for en node der indgår i sættet S, at denne er blevet tilføjet til mængden S via union af den foregående mængde S og den evaluerede node u, som via RELAX blev vurderet, at være en bedre/billigere node at besøge end den foregående bedste-vej node, v.

2.3 c)

Denne egenskab følger direkte af betingelsen om non-negative ${\tt edges}.$