Blatt 09

Gruppe Nikolic ,Nofal,Hattinger

Aufgabe Nr 28.

Algorithm for in-degree-with adjacency list

```
For each u in V do
        in-deg[u] = 0
For each u in V do
        For each v in Adj[u] do
                in-deg[v] = in-deg[v] + 1 // count # of edges to v
```

-Hier wird ein Array T, mit grösse =|V| angelegt,|V|= anzahl der knoten,dieses array wird mit felder =0 initilaisiert,für jeden knoten wird die liste von adjazentknoten durchsucht,T[u] wird um eines inkrementiert wenn u in der adjazenzliste auftaucht,am ende wenn alle adj[u] von allen knoten durchsucht wurden ,dann die integer werte in T gaben anzahl der eingangsgrad für jeden knoten an.

die laufzeit in diesem fall wäre O(|V|+ |E|)+ extra spreicherplatz für das T Array =O(|V|)

also gesamt bleibt O(|V|+|E|).

- die Berechnung des Eingangsgrad mittels Adjazenzmatrix wäre nicht optimal wenn anzahl der kanten kleiner als die Quadratische anzahl die knoten sind. da die matrixdarstellung braucht O(|V|²). fexieren wir eine spalte für jeden knoten, jede zeile wird untersucht ob eines dort steht oder nicht ,steht eines dort wird der eintrag im array um eines erhöht für jeden knoten das bedeutet dass für jede spalte müssen alle zeilen durchsucht werden also die laufzeit ist O(|V|²)+ extra speicher platz für das array T,insgesamt O(|V|²).

Augabe Nr 29

Wird der Algorithmus von Dijkstra an ein paar Stellen modifiziert, so kann er ganz einfach f¨ur die gestellte Aufgabe verwendet werden. Die Wahrschein-lichkeiten der Kanten werden einfach multipliziert und es wird die Kante mit dem h¨ochsten Wert verwendet. Außerdem muss die Ausgangsdistanz auf 1 (also 100%) gesetzt werden. All diese Modifikationen sind im folgenden Pseudocode in den Zeilen 2, 4 und 7 sowie 8 zu finden:

dijkstra_communication (G, w, s):

```
for each vertex v ∈ V do d[v]←0; π←nil
d[s]←1; S←∅; Q←V
while Q ≠ ∅ do
        u←Extract -Max(Q)
        S ←S ∪ { u}
        for each vertex v ∈ Adj[u] do
                if d[v] < d[u] · w(u, v)
                        then d[v] ← d[u] · w(u, v); π[v] ← u
```