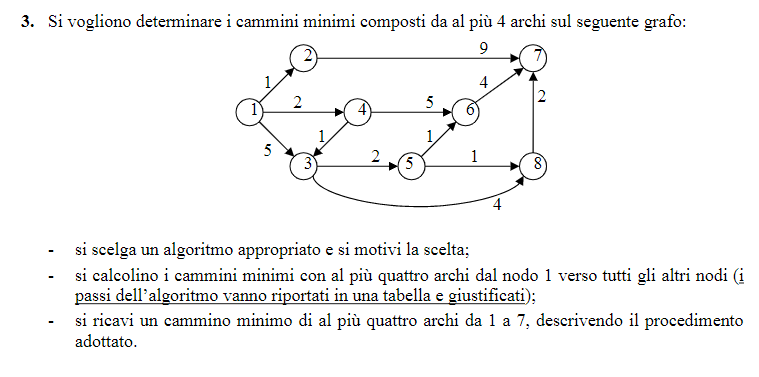
* Si discuta la complessità computazionale dell’algoritmo di Bellman-Ford.
  + L’algoritmo di Bellman-Ford ha una complessità ), in quanto viene fato un numero di iterazioni pari al numero di nodi con un ciclo che esamina le etichette duali su tutti gli archi . Qualora esistesse un ciclo negativo (abbiamo aggiornato almeno un’etichetta) oppure tutte le etichette sono stabili, l’algoritmo termina la sua esecuzione.
* Si discuta la complessità computazionale dell’algoritmo di Dijkstra per il problema del cammino minimo
  + L'algoritmo di Dijkstra è un algoritmo di grafo per trovare il percorso più breve da un nodo sorgente a tutti gli altri nodi di un grafo (single-source shortest path). È un tipo di algoritmo greedy. Funziona solo su grafi ponderati con pesi positivi. Ha una complessità temporale di usando la matrice di adiacenza che rappresenta il grafo. La complessità può essere ridotta a usando la lista di adiacenza di rappresentazione del grafo, dove è il numero di archi e è il numero di vertici.
  + In generale, dipende dalla struttura dati adottata.

Buon link di riferimento: <https://www.baeldung.com/cs/dijkstra-time-complexity>

Simulazione Esame 2018/2019 – Esercizio 3 Grafi Raccolta “Esercizi vari”



* Nella scelta dell’algoritmo, notiamo che ci viene dato un massimo numero di archi da dover rispettare, pertanto si può applicare solo l’algoritmo di Bellman – Ford, l’unico che dà la possibilità di calcolo dei cammini minimi sulla base di un massimo numero di archi. Applicheremo Bellman – Ford fermandoci alla quarta iterazioni, con un numero di archi e iterazioni pari a
* Ora, il calcolo dei cammini minimi, riportati in tabella (controllando ad ogni passo miglioramenti rispetto alla riga precedente e segnando in colonna apposita gli aggiornamenti effettuati; i predecessori sono segnati come semplici pedici senza parentesi):

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Iterazione | Nodo 1 | Nodo 2 | Nodo 3 | Nodo 4 | Nodo 5 | Nodo 6 | Nodo 7 | Nodo 8 | Aggiornamenti |
|  |  |  |  |  |  |  |  |  | 1 |
|  |  |  |  |  |  |  |  |  | 2,3,4 |
|  |  |  |  |  |  |  |  |  | 3,5,6,7,8 |
|  |  |  |  |  |  |  |  |  | 5,8 |
|  |  |  |  |  |  |  |  |  | 6,7,8 |

Le etichette di una riga sono ottenute controllando i vincoli duali su tutti gli archi uscenti dai nodi “aggiornati” della riga (iterazione) precedente secondo la dove è uno degli archi uscenti da un nodo aggiornato all’iterazione precedente, l’etichetta corrente (sulla riga corrente) del nodo , è l’etichetta del nodo all’iterazione (riga) precedente e è il costo dell’arco .

Grazie al fatto di utilizzare l’etichetta del nodo precedente, viene assicurata la scelta del cammino minimo con archi (qua si può fare un qualsiasi esempio numerico dove si va a scegliere, prendendo ad esempio l’ultima iterazione, qui un cammino che ha costo migliore considerando l’etichetta precedente e non quella corrente, qui , dimostrando la validità di quanto fatto).

Un cammino minimo con al massimo archi da viene fatta seguendo la catena dei predecessori, quindi, partendo dal nodo (ultimo nodo in generale) con si considera quindi:

Questo equivale al percorso , con costo 9.

