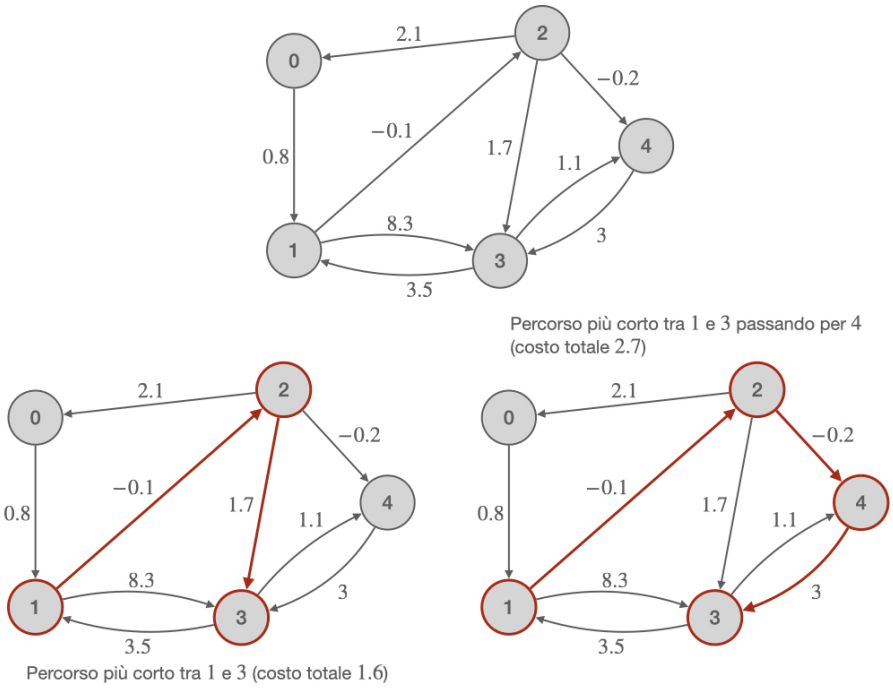
Progetto “Percorso minimo con nodo intermedio”

Appello del 25 Giugno 2020

Descrizione del progetto

Lo scopo del progetto è quello di realizzare un programma in grado di generare un grafo a partire da una sequenza di archi e, con la rappresentazione del grafo che è stata costruita, trovare un percorso di lunghezza minima tra due nodi a e b e il percorso di lunghezza minima tra a e b che passi per un terzo nodo c. Per un esempio si veda Figura 1. In essa vi è un grafo orientato con archi pesati (anche con pesi negativi) ed è evidenziato il percorso minimo tra il nodo 1 e il nodo 3. Si noti come il percorso cambia se si richiede di passare per il nodo 4.



Si richiede che questa struttura dati supporti le seguenti operazioni:

- Creazione di un oggetto di tipo `Grafo` di n nodi in cui gli archi sono dati come un array (lista Python) di triplette nella forma (u, w, v) dove u e v sono interi in $0, \ldots, n-1$ che rappresentano le due estremità dell'arco e $w \in \mathbb{R}$ è il peso dell'arco.
- Dati due nodi a e b (interi tra 0 e $n-1$), si restituisca una lista di nodi $[a, c_1, \ldots c_m, b]$ rappresentante i nodi da cui passare per ottenere un percorso di lunghezza minima tra a e b. Se vi sono più percorsi è sufficiente restituirne uno.

- Dati tre nodi a, b e c (interi tra 0 e $n-1$), si restituisca una lista di nodi $[a, c_1, \ldots c_k, c, c_{k+2}, \ldots, c_m, b]$ rappresentante un i nodi da cui passare per ottenere un percorso di lunghezza minima tra a e b passante per il nodo c (questo implica che c dovrà essere contenuto nella lista). Si noti che questo potrebbe non essere un percorso di lunghezza minima tra a e b, dato che il passaggio per c potrebbe allungarlo.

Si impongono inoltre i seguenti vincoli:

- La struttura dati usata internamente per rappresentare il grafo deve consentire di trovare il peso dell'arco `(i,j)` in tempo $O(1)$.
- La complessità temporale necessaria per individuare il percorso di lunghezza minima tra due nodi deve essere di $O(n^3)$.

Codice

Viene fornito uno scheletro del codice che deve essere implementato, con i metodi che devono essere scritti:

```
class Grafo:

    def __init__(self, n, archi):
        # Implementazione del costruttore. Il grafo di n nodi
        # dovrà venire costruito a partire dall'array (lista Python)
        # di archi fornita come argomento

    def percorso_minimo(self, inizio, fine):
        # Ritorna una lista di nodi su un percorso di lunghezza
        # minima tra inizio e fine. La lista deve avere come primo
        # elemento inizio e come ultimo elemento fine.
        # se non esiste un percorso tra inizio e fine deve
        # essere ritornata la lista vuota.

    def percorso_minimo_con_nodo_intermedio(self, inizio, fine, c):
        # Come per percorso_minimo, ma viene aggiunta la restrizione
        # che il percorso deve passare per il nodo c.
```

Nel progetto è consentito avere funzioni aggiuntive che testano il buon funzionamento delle funzionalità richieste. È anche consentito avere metodi aggiuntivi (e.g., per ricostruire il percorso o per calcolare i percorsi minimi).

Si ricorda di commentare adeguatamente il codice, approfittandone per spiegare le scelte implementative effettuate.

Esempi d'uso

Il seguente frammento di codice chiama tutti i metodi la cui implementazione è richiesta dal progetto. Come commento è indicato il valore atteso in una implementazione funzionante:

```python
def test():
    archi = [(0, 0.8, 1),  (1, -0.1, 2),  (1, 8.3, 3),
             (2, 2.1, 0), (2, 1.7, 3),  (2, -0.2, 4),
             (3, 3.5, 1),  (3, 1.1, 4),  (4, 3, 3)]
    g = Grafo(5, archi)
    print(g.percorso_minimo(1, 3))
    # stampa [1, 2, 3]
    print(g.percorso_minimo(0, 4))
    # stampa [0, 1, 2, 4]
    print(g.percorso_minimo(4, 4))
    # stampa [4]
    print(g.percorso_minimo_con_nodo_intermedio(1, 3, 4))
    # stampa [1, 2, 4, 3]
    print(g.percorso_minimo_con_nodo_intermedio(0, 4, 3))
    # stampa [0, 1, 2, 3, 4]
    print(g.percorso_minimo_con_nodo_intermedio(4, 4, 3))
    # stampa [4, 3, 4]
```

Indicazioni

Il progetto deve essere svolto **individualmente**. La consegna dovrà avvenire entro le ore 23:59 del giorno 18/06/2021 secondo le seguenti modalità:

- Invio di una email a `lmanzoni@units.it` dal vostro account email istituzionale con oggetto *[informatica] consegna progetto appello del 25/06/2021*.
- L'email deve avere come allegato il progetto in un singolo file in codice sorgente Python versione 3, dal nome `Nome_Cognome_matricola.py`, quindi, per esempio Mario Rossi di matricola 12345 consegnerà un file dal nome `Mario_Rossi_12345.py`.
- Il file deve contenere sotto forma di commento le seguenti linee indicanti nome, cognome e numero di matricola:

  ```
  # Nome: Mario
  # Cognome: Rossi
  # Matricola: 12345
  ```