

Progetto “Matrice Sparsa”

Appello del 22 Luglio 2020

Descrizione del progetto

Lo scopo del progetto è la creazione di una struttura dati in grado di rappresentare una matrice di n righe e m colonne che ha pochi valori non zero. Questa viene solitamente chiamata una *matrice sparsa* ed esistono molteplici metodi per rappresentarla in modo efficiente (i.e., con poco consumo di memoria e accesso efficiente agli elementi).

Uno dei modi di rappresentare le matrici sparse è tramite una *lista di liste*. In una lista di liste una matrice di n righe e m colonne è rappresentata tramite una lista ordinata per ogni riga con elementi non nulli. Ogni nodo della lista contiene, oltre ad un valore che indica quale sia l'indice di riga, una lista concatenata di tutti gli elementi non nulli della riga ordinati per indice di colonna, come si può vedere in Figura 1.

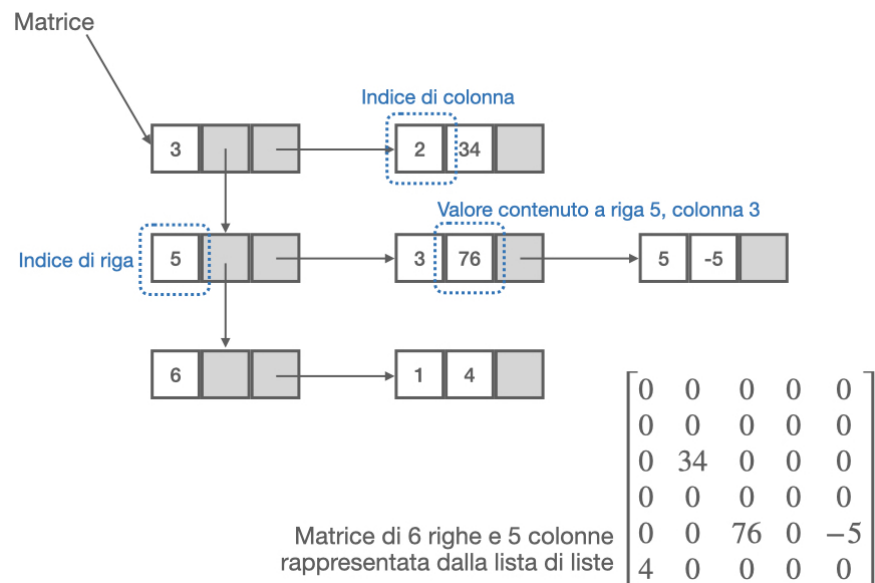


Figura 1: Rappresentazione di una matrice sparsa tramite una lista di liste.

Facendo riferimento all'esempio di Figura 1, in cui abbiamo una matrice di 6 righe e 5 colonne, per accedere all'elemento di indice (5, 3) (riga 5, contando da uno, e colonna 3, sempre contando da 1), la procedura sarà la seguente:

1. Si inizierà scorrendo la lista rappresentante le righe contenenti valori non nulli della matrice fino a raggiungere il nodo con indice di riga 5.

2. Si inizia a scorrere la lista la cui testa è contenuta nel nodo con indice di riga 5 fino a quando non si incontra l'indice di colonna 3.
3. Si ritorna quindi il valore contenuto nel nodo individuato che, nel nostro caso, è 76.

Nel caso avessimo chiesto l'elemento di indici $(4, 2)$ avremmo ritornato il valore 0, dato che pur essendo una coppia di indici valida (i.e., contenuta nella matrice), il suo valore non è presente nella lista di liste. Si noti che vi sono due modi in cui si può trovare che un valore è nullo: se il suo indice di riga non si trova nella lista delle righe, oppure se si è trovato l'indice di riga ma la lista contenente tutti gli indici di colonna per quella riga *non* contiene l'indice di colonna.

Si vuole implementare una matrice sparsa come lista di liste che consenta le seguenti operazioni:

1. Inserimento di un valore in una posizione data. Nel caso l'inserimento sia in un indice al di fuori della matrice (i.e., una posizione (i, j) con i non compreso tra 1 e n e j non compreso tra 1 e m) allora il valore non deve essere inserito.
2. Ottenere il valore presente in una posizione data, ricordando che se il valore non è stato salvato nella lista di liste esso è zero. Se si richiede un valore in una posizione che non è parte della matrice allora deve essere ritornato **None**.
3. Ottenere una stringa rappresentante l'intera matrice.

Sia indicato con q il numero di elementi inseriti nella matrice. Le operazioni per inserire ed ottenere un valore all'interno della matrice devono richiedere tempo $O(q)$, ovvero lineare nel numero di elementi inseriti nella matrice.

Codice

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

```
class MatriceSparsa:
```

```
    def __init__(self, n, m):
        # Implementazione del costruttore. La matrice dovrà
        # contenere n righe e m colonne. Inizialmente tutti
        # i valori sono 0

    def inserisci(self, i, j, valore):
        # inserisce il valore passato come argomento alla
        # riga i-esima (contando da uno) e alla colonna
        # j-esima (contando da uno).

    def valore(self, i, j):
        # Ritorna il valore contenuto alla riga i (contando da
```

```

        # uno) e colonna j (contando da uno) della matrice.
        # Si ricorda che i valori non presenti nella lista di
        # liste ma con indice di riga e colonna valido (i.e.,
        #  $1 \leq i \leq n$  e  $1 \leq j \leq m$ ) devono ritornare il valore
        # zero.

def __str__(self):
    # Ritorna una stringa rappresentante l'intera matrice
    # inclusi i valori nulli. Gli elementi sulle righe
    # devono essere separati da spazi o tab, le righe
    # da newline (\n).

```

Nel progetto è consentito avere funzioni e classi aggiuntive per l'implementazione. Per esempio, delle classi rappresentanti i nodi delle liste.

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:

```

m = MatriceSparsa(6, 6)
x = m.valore(3, 4)
print(x)    # 0
m.inserisci(3, 4, 1)
x = m.valore(3, 4)
print(x)    # 1
m.inserisci(3, 2, 7)
m.inserisci(2, 1, 4)
m.inserisci(3, 4, 9)
m.inserisci(3, 6, 2)
m.inserisci(6, 6, 8)
m.inserisci(6, 1, 4)
print(m)
# 0 0 0 0 0 0
# 4 0 0 0 0 0
# 0 7 0 9 0 2
# 0 0 0 0 0 0
# 0 0 0 0 0 0
# 4 0 0 0 0 8

```

Indicazioni

Il progetto deve essere svolto **individualmente**. La consegna dovrà avvenire entro le ore 23:59 del giorno 10/07/2020 secondo le seguenti modalità:

- Invio di una email a `lmanzoni@units.it` dal vostro account email istituzionale con oggetto *[informatica] consegna progetto appello del 22/07/2020*.
- 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
```