Presentazione algoritmo bidirezionale ShortestAugmentingPath

Filippo Magi

February 18, 2022

1 Strutture Dati

1.1 BiEdge

l'arco BiEdge contiene le informazioni dei due nodi che collega, oltre alla capacità residua di quell'arco e il flusso passante.

1.2 Node

Il nodo node contiene una lista con tutti gli archi a lui collegati, oltre a informazioni utili per l'indirizzamento durante l'invio del flusso (nextEdge e nextNode per i nodi proveniente dal nodo destinazione t e previousNode e previousEdge per i nodi provenienti da s), oltre a salvarsi le informazioni sulla distanza che intercorre tra lui e i nodi s e t.

1.3 Graph

insieme di nodi.

2 Descrizione

l'algoritmo effettua due BFS, una che parte da s, mentre l'altra da t, questo affinché ogni sappia la propria distanza da questi due nodi. invio il flusso trovato dal percorso di queste due BFS. cancello le informazioni di indirizzamento all'interno di ogni nodo. uso l'algoritmo Shortest Augmenting Path, chiamato nel codice Dfs(e SinkDfs per la parte che da sink cerca source). Trovato un percorso, restituisco dove sono arrivati i nodi (stesso nodo oppure un nodo potrebbe essere arrivato a t o s). Invio il flusso e, a seconda se ci sono incontrati a metà strada o uno è arrivato a s/t, resetto i nodi (o il nodo nel caso solo uno sia arrivato a destinazione) di partenza e cancello le informazioni di indirizzamento presenti nella coda dei nodi esplorati. ripeto ciò finché i nodi che mi restituisce Dfs sono null, in tal caso so che la distanza tra s e t è maggiore

al numero di nodi del grafo, quindi non sono più raggiungibili e l'algoritmo è terminato.

2.1 Dfs e SinkDfs

Dfs e SinkDfs hanno lo stesso funzionamento, con l'unica differenza di quali nodi vado ad esplorare, nel caso di Dfs quelli che escono dal nodo che sto esplorando, nel caso di SinkDfs quelli che entrano. Prendiamo come esempio Dfs per descriverle entrambe. Chiamiamo il nodo che sto cercando **startSource**

Esploro i nodi uscenti dal **startSource**, se sono idonei (quindi se hanno capacità positiva e la distanza da Sink del nodo esplorato è quella di **startSource**-1) in tal caso aggiorno i dati di **startSource**,li inserisco nella loro apposita coda, e nel caso il nodo esplorato sia t o già esplorato dalla parte di Sink, ritorno il nodo e il valore che mi sono salvato del flusso inviabile, altrimenti chiamo SinkDfs, con nodo esplorato come nuovo **startSource**. Nel caso non trovi nessun arco valido, procede con il retreat.

2.1.1 Retreat

cerco l'arco che esce da **startSource** con distanza minima da sink, quindi faccio diventare la distanza tra **startSource** pari a quel valore +1. nel caso in cui **startSource** = s allora salvo **startSource** in una variabile momentanea,altrimenti salvo in quella variabile il predecessore di **startSource** cancello le informazioni per l'indirizzamento di **startSource**, per poi richiamare la stessa funzione Dfs con la variabile temporanea.