More on Segment Trees

Davide Bartoli

19 novembre 2022

Data una struttura dati con inserimenti in $\mathcal{O}(T(N))$ **non ammortizzato** e query offline, vogliamo supportare anche le rimozioni in $\mathcal{O}(T(N) \log N)$.

Problema

Dato un grafo di N nodi, vuoi supportare le operazioni:

- Aggiungi un arco (a, b).
- Rimuovi un arco (a, b).
- Conta il numero di componenti connesse.

Ogni arco (a, b) esiste in un intervallo di tempo. Devo creare un segment tree dove ogni nodo rappresenta un intervallo di tempo; un arco è presente nei nodi che corrispondono all'intervallo di tempo in cui esiste (questi sono $\mathcal{O}(\log N)$).

Ogni query corrisponde a una foglia (rappresentante un istante di tempo).

Faccio una DFS nel Segment Tree; quando entro in un nodo, aggiungo gli archi salvati in quel nodo, quando esco dal nodo li rimuovo. Ogni volta che raggiungo una foglia calcolo la risposta per la query corrispondente.

Notiamo che le operazioni da "annullare" rappresentano un suffisso delle operazioni che ho eseguito. Quindi posso mantenere uno stack degli update e fare l'*undo* delle operazioni nell'ordine inverso di come sono state applicate. Questo può essere fatto in $\mathcal{O}(T(N))$ dato che mi basta riportare al valore originale ciò che era stato modificato dall'update.

In questo problema non possiamo usare la *path compression* dato che ha una complessità ammortizzata. Possiamo però utilizzare la *union by rank*, quindi la complessita della DSU è logaritmica.

La complessità totale è:

$$\mathcal{O}(N \cdot \underbrace{T(N)}_{\text{DSU}} \cdot \underbrace{\log N}_{\text{segment}})$$