Considerazioni performance

Gianluca Aguzzi

Aprile 2018

1 Descrizione del problema

Il simulatore scafi, durante le varie simulazione, avrà bisogno di accedere ai vicini di un dato nodo. Questo problema è ben noto nell'algoritmica moderna, nel nostro caso in particolare vorremo:

- accedere al vicinato di un nodo in tempo constante (o constante ammortizzato) visto che ad ogni round dovremo utilizzarlo
- spostare un nodo nello spazio dovrebbe impiegare un tempo sub lineare (per simulazioni dove i nodi si muovono nello spazio)

al momento solo il primo dei due requisiti è soddisfatto, mentre il secondo ha una complessità pari a n(visto che, ogni volta che un nodo si muove nello spazio, si dovranno fare delle operazioni atte a mantenere la mappa del vicinato sensata). Le possibili soluzioni sono:

- utilizzare un database spaziale esterno
- utilizzare strutture dati ad hoc(o indici spaziali in memory)

2 Database spaziale

In questo caso è stato analizzato principalmente il database spaziale tile38. In generale le perfomance sono decisamente buone (la maggior parte delle operazioni viene eseguita in un tempo inferiore a millisecondo) però è anche vero che vi si introduce un ritardo dovuto al protocollo di comunicazione ed il conseguente parsing del risultato (anche se come tempo è costante non è sicuramente trascurabile). I principali pro nell'utilizzo di un db spaziale sono:

- persistenza
- multi client e di conseguenza la possibilità di creare un supporto online più agevole
- salvataggio dei dati in termini di latitudine e longitudine (potrebbe quindi permettere simulazioni reali in modo più agevole) i principali contro di questo approccio sono invece:

- performance inferiore ad algoritmi ottimizzati in memory.
- utilizzo di un dbms per qualsiasi simulazione

3 Strutture dati

Le strutture dati in memory dovrebbero offrire performance superiori rispetto ad un database spaziale per via del fatto che, il tempo necessario per "impacchettare" e "spacchettare" il dato ricevuto/inviato seppur costante non è sicuramente nullo. Una possibile struttura dati ad hoc ed un algoritmo associato ad essa è descritto nel seguente link: https://algo.kaust.edu.sa/Documents/cs372l01.pdf
Se l'intenzione è quella di avere una soluzione il più possibile generale, ci si dovrebbe orientare a strutture dati (ad esempio octree) o indici(ad esempio r-tree) che garantiscono buone performance sulle operazioni principale(costante o con O(logn)) e possono essere applicate anche a più dimensioni senza alterare le performance.