Luca Rosso Matr. 887 433

Mattone Andrea Matr. 882 956

**ALGORITMI E STRUTTURE DATI 2019-20 LABORATORIO**

**Relazione Esercizio 1**

**Analisi complessità algoritmi**

-Tempo di caricamento dei dati forniti nel csv dei docenti nella struttura:

24 s

*-QuickSort*

Sul dataset fornito dai docenti i tempi di ordinamento dei record con il quicksort sono i seguenti:

field1: 58 s

field2: 42 s

field3: 42 s

Il tempo di esecuzione del quicksort analizzando i campi di tipo char[] è maggiore in quanto utilizziamo la funzione strcmp() che aumenta il tempo di esecuzione. La strcmp() impiega un maggior tempo per via della dimensione del dato da analizzare.

Gli interi impiegano tempo O(1) in quanto la compare necessita di effettuare una differenza e restituire un risultato, mentre per le stringhe la complessità dipende dalla dimensione dell’input in quanto una stringa con n caratteri avrà bisogno di n confronti.

*-InsertionSort*

L’insertion sort è veloce se si utilizzano dataset con dimensioni ridotte, in quanto il quicksort in questi casi viene appesantito dalle chiamate ricorsive.

Con il dataset fornito il tempo di ordinamento superava la soglia dei 10 minuti per tutti i campi, rendendo fallimentare l’operazione.

*Considerazioni*

Il risultato era parzialmente atteso in quanto l’insertion sort ha una complessità temporale T(n) = O(n^2) mentre il quick sort ha una complessità temporale T(n)=O(n logn).

Questa differenza temporale è, pero, solo verificabile su dataset di dimensioni molto grandi, i nostri test su dataset di piccole dimensioni non hanno prodotto risultati molto esplicativi in quanto il tempo di ordinamento è troppo basso per essere misurato.

Inoltre abbiamo notato una differenza in termini di tempo per l’ordinamento di campi di tipo Stringe e di campi di tipo intero.

Le stringhe impiegano un tempo maggiore per essere ordinate rispetto agli interi, perché la funzione utilizzata: “strcmp()” si mpiega più tempo nel confronto rispetto ad un semplice confronto tra interi.