Relazione del 1° esercizio Laboratorio di Algoritmi e Strutture Dati

Zhao Xiao matr: 913828 mail: xiao.zhao@edu.unito.it

Il progetto si compila con il commando make, e si usa il commando <./client “percorso del file csv”> per avviare.

Durante l’esecuzione, viene richiesto di inserire il valore di k, il field da ordinare e se si vuole ordinare in modo crescente. Al termine dell’esecuzione, viene chiesto se si vuole il risultato del ordinamento in .csv, nominato in <output_file.csv>

Nel mio progetto, ho fatto dei test con I valori di k da 0 fino a 10000 (fig 1, fig 2). Ho notato che quando k = 0, il tempo di esecuzione è simile al tempo di k = 100 e ho un tempo basso con k intorno al 20. Questo è perché il insertion-sort è più efficacie per ordinare gli array con dimensione piccola.

Viene dimostrato che, con k = 10~20, il tempo di esecuzione è ottimale. Per cui, si consiglia di scegliere il valore di k in questa area.

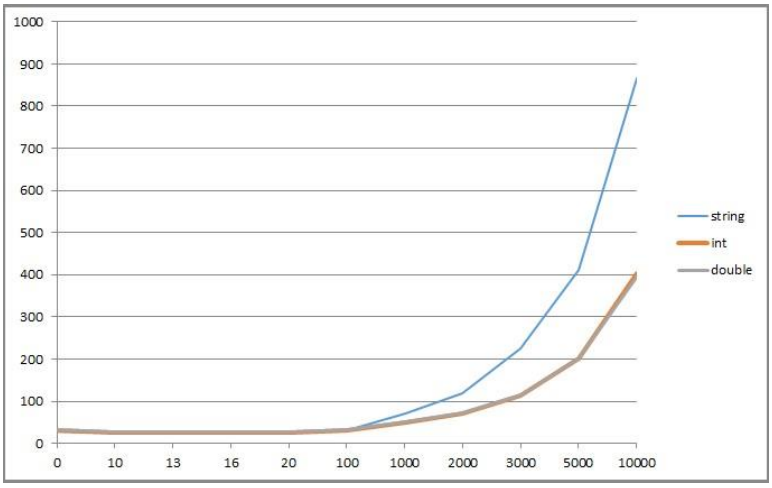
Con valori di k elevati, si rileva un tempo di esecuzione molto più lento perché insertion-sort non è adatto a gestire una dimensione così grande.

In base agli studi(fig 3), si dimostra che il tempo di esecuzione di ins-sort (O(n^2)) è minore di merge (O(n log n)) nei dimensioni piccoli. Questo succede perché la loro complessità temporale è simile, e la esecuzione dell’algoritmo ins-sort è più veloce di merge che deve fare ancora altri ricorsioni.

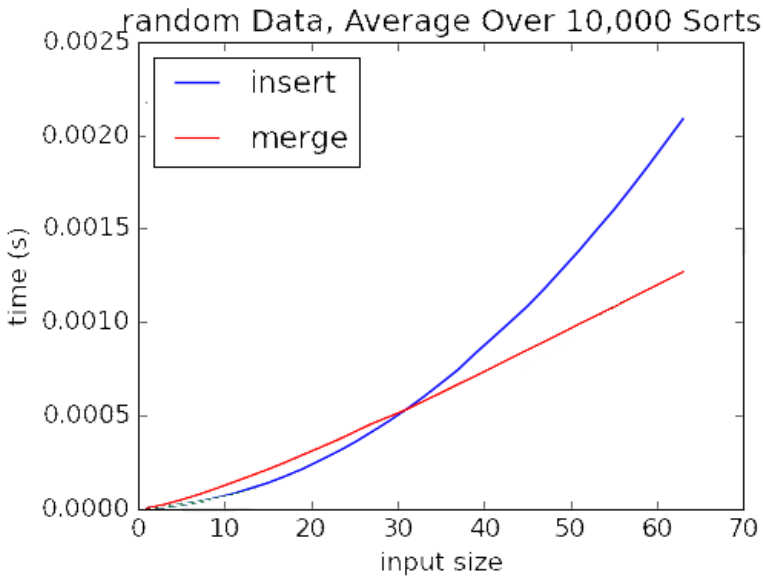
Quindi il ins-sort si dimostra migliore con una dimensione minore di circa 30, ma più la dimensione cresce, il tempo di esecuzione cresce in modo esponenziale. Questo fatto corrisponde con il risultato ottenuto del mio progetto.

	A	B	C	D	E	F
1	k		string	int	double	
2	0		28.263006	30.945268	30.947882	
3	10		24.989772	26.823827	27.297444	
4	13		24.795699	26.250095	26.709794	
5	16		24.599788	26.128016	26.66362	
6	20		25.028395	26.10765	26.619875	
7	100		30.291417	30.681131	31.034249	
8	1000		69.691173	49.046834	49.745648	
9	2000		118.28563	71.568122	72.367227	
10	3000		224.294324	112.453189	113.928573	
11	5000		412.222037	201.975124	199.972639	
12	10000		867.119151	403.611608	394.994978	
13						

(figura 1)



(figura 2)



(figura 3)