Dizionari

INSERIMENTO, complessità teorica:

	Caso Peggiore	Caso Medio	Caso Migliore
Hashtable con liste di trabocco	O(1)	O(1)	O(1)
Alberi di Ricerca	O(N)	O(h)	O(h)

Tempi reali di inserimento di 20.000.000 di record su due PC:

	Test	field1 (sec)	field2 (sec)	field3 (sec)
Hashtable	PC 1	1.162	7.57	17.957
	PC 2	2.046	6.454	17.47
Alberi di Ricerca	PC 1	3.607	63.792	78.927
	PC 2	4.298	45.517	56.585

RICERCA, complessità teorica:

	Caso Peggiore	Caso Medio	Caso Migliore
Hashtable con liste di trabocco	O(N)	O(1)	O(1)
Alberi di Ricerca	O(N)	O(h)	O(h)

Tempi reali di ricerca su due PC:

	Test	field1 (sec)	field2 (sec)	field3 (sec)
Hashtable	PC 1	0.345	0.309	0.385
	PC 2	0.204	0.359	0.438
Alberi di Ricerca	PC 1	0.528	3.675	4.165
	PC 2	0.406	2.268	2.814

CANCELLAZIONE, complessità teorica:

	Caso Peggiore	Caso Medio	Caso Migliore
Hashtable con liste di trabocco	O(N)	O(1)	O(1)
Alberi di Ricerca	O(N)	O(h)	O(h)

Tempi reali di cancellazione su due PC:

	Test	field1 (sec)	field2 (sec)	field3 (sec)
Hashtable	PC 1	0.108	1.03	1.135
	PC 2	0.093	0.547	0.594
Alberi di Ricerca	PC 1	0.11	5.193	5.11
	PC 2	0.178	3.685	4.688

Nell'accedere ad 1.000.000 di chiavi scelte a caso, le hashtable risultano più efficienti degli alberi di ricerca. Infatti, i tempi di risposta delle hashtable sono nettamente minori di quelli degli alberi sia per quanto riguarda l'inserimento che per la ricerca e cancellazione. Questo era prevedibile in quanto, l'inserimento per le hashtable ha tempo O(1); la ricerca, invece, dipende dalla funzione di hash.

Essendo la tabella dinamica, possiamo supporre l'uniformità semplice della funzione di hash e quindi ottenere complessità O(1+alfa) dove alfa = N/m, N è il numero degli elementi inseriti nella tabella e m è la dimensione della tabella. Ma, essendo m proporzionale ad N per la dinamicità della tabella, si ottiene N = O(m) e quindi alfa = O(1). Quindi, la ricerca ha tempo O(1). La cancellazione di un elemento già trovato in una lista doppiamente linkata è O(1).

In definitiva, le hashtable hanno tempo O(1) in tutte e tre le operazioni (inserimento, ricerca, cancellazione) mentre gli alberi hanno tempi O(h) per le tre operazioni, con h altezza dell'albero.

Negli alberi binari di ricerca l'inserimento delle stringhe è molto più veloce degli altri campi perché i dati contengono molti duplicati che non costituiscono nuovi nodi, l'altezza h dell'albero rimane quindi contenuta e le operazioni risultano più veloci.