Relatório
Tabela Hash

Luca Takemura Piccoli

1

Resumo. *Este meta-artigo relata a performance de diversos algoritmos de hashing de números inteiros em tabelas hash, incluindo o algoritmo de Divisão, o algoritmo de Multiplicação e o algoritmo de Dobramento. A pesquisa envolveu a análise comparativa da eficiência desses algoritmos em termos de tempo de execução para inserir em uma tabela, tempo de execução para buscar uma chave na tabela, número de colisões ocorridas e número de comparações feitas para buscar a chave. Foram usados conjuntos de dados com diferente número de chaves e tabelas de tamanhos diferentes.*

1. Introdução

A escolha da função hash certa é de extrema importancia para a eficiencia e o desempenho das tabelas hash em diversas aplicações. Neste Artigo veremos o desempenho de vários algoritmos de hashing para diferentes tamanhos de tabelas e diferentes quantidade de dados. Esse Relatorio pode ajudar em tomadas de decisões ao escolher o algoritmo de hashing mais apropriado para suas necessidades.

2. Métodologia

É essencial descrever explicitamente o metodo usado e a máquina que foi usada para obter os resultados, em particular para facilitar os pesquisadores que pretendem reverificar-los.

Para a análise dos algoritimos foram usadas tabelas de diferentes tamanhos, 1000, 10000, 100000, 1000000, 10000000, e conjuntos de chaves de diferentes tamanhos, 20000, 100000, 500000, 1000000, 5000000. As chaves usadas são números inteiros aleatorios feios com a função random do java, para facilitar a replicabilidade dos experimentos foi usada na função random a seed ”1”.

Para medir o tempo de execução das buscas nas tabelas cada busca foi executada 5 vezes, para obter a media do tempo.

Para o tratamento de colisões foi usado o metodo de encadeamento.

Os experimentos foram feitos no Visual Studio Code em um computador com um processador i7 de oitava geração.

3. Resultados

A análise comparativa dos algoritmos de hashing revelou insights valiosos sobre seus desempenhos entre tabelas de diferentes tamanhos e conjuntos de dados de diferentes tamanhos:

3.1. Tabelas de tamanho 1000

Table 1. Comparações do tempo de execução da inserção em tabelas, de cada algoritimo para cada tamanho de conjunto de dados (tempo em milisegundos)

Tamanho	Divisão	Multiplicação	Dobramento
20000	9	9	8
100000	173	205	216
500000	8577	4090	4615
1000000	15390	21231	28034
5000000	227564	317022	425494

Table 2. Comparações do tempo de execução da busca em tabelas, de cada algoritimo para cada tamanho de conjunto de dados (tempo em milisegundos)

Tamanho	Divisão	Multiplicação	Dobramento
20000	7	9	7
100000	159	172	152
500000	695	743	1167
1000000	2584	2788	3113
5000000	59956	60897	72233

Table 3. Comparações do número de colisões em tabelas, de cada algoritimo para cada tamanho de conjunto de dados

Tamanho	Divisão	Multiplicação	Dobramento
20000	19000	19000	19000
100000	99000	99000	99000
500000	499000	499000	499000
1000000	999000	999000	999000
5000000	4999000	4999000	4999000

Table 4. Comparações do número de comparações feitas em cada busca em tabelas, de cada algoritimo para cada tamanho de conjunto de dados

Tamanho	Divisão	Multiplicação	Dobramento
20000	199291	199916	200052
100000	4983798	4980977	4981066
500000	122923166	122935345	122927124
1000000	483776707	483822004	483769755
5000000	2063156924	2063352697	2063082385

3.2. Tabelas de tamanho 10000

Table 5. Comparações do tempo de execução da inserção em tabelas, de cada algoritimo para cada tamanho de conjunto de dados (tempo em milisegundos)

Tamanho	Divisão	Multiplicação	Dobramento
20000	5	9	5
100000	33	58	44
500000	1182	576	616
1000000	2083	2274	2807
5000000	30672	35573	45411

Table 6. Comparações do tempo de execução da busca em tabelas, de cada algoritimo para cada tamanho de conjunto de dados (tempo em milisegundos)

Tamanho	Divisão	Multiplicação	Dobramento
20000	10	11	5
100000	39	56	27
500000	163	276	265
1000000	665	850	2331
5000000	10215	10657	8446

Table 7. Comparações do número de colisões em tabelas, de cada algoritimo para cada tamanho de conjunto de dados

Tamanho	Divisão	Multiplicação	Dobramento
20000	11358	11357	11402
100000	90000	90000	90000
500000	490000	490000	490000
1000000	990000	990000	990000
5000000	4990000	4990000	4990000

Table 8. Comparações do número de comparações feitas em cada busca em tabelas, de cada algoritimo para cada tamanho de conjunto de dados

Tamanho	Divisão	Multiplicação	Dobramento
20000	19716	19880	20120
100000	497582	498165	498555
500000	12287869	12280770	12281908
1000000	48339928	48339261	48335981
5000000	1064407519	1064388634	1064347712

3.3. Tabelas de tamanho 100000

Table 9. Comparações do tempo de execução da inserção em tabelas, de cada algoritimo para cada tamanho de conjunto de dados (tempo em milisegundos)

Tamanho	Divisão	Multiplicação	Dobramento
20000	6	11	7
100000	25	41	30
500000	184	316	163
1000000	443	739	516
5000000	7395	8392	6703

Table 10. Comparações do tempo de execução da busca em tabelas, de cada algoritimo para cada tamanho de conjunto de dados (tempo em milisegundos)

Tamanho	Divisão	Multiplicação	Dobramento
20000	10	12	13
100000	22	40	22
500000	70	259	97
1000000	212	580	318
5000000	4299	4240	3613

Table 11. Comparações do número de colisões em tabelas, de cada algoritimo para cada tamanho de conjunto de dados

Tamanho	Divisão	Multiplicação	Dobramento
20000	1911	1815	1811
100000	36829	36685	36715
500000	400692	400656	400692
1000000	900005	900003	900005
5000000	4900000	4900000	4900000

Table 12. Comparações do número de comparações feitas em cada busca em tabelas, de cada algoritimo para cada tamanho de conjunto de dados

Tamanho	Divisão	Multiplicação	Dobramento
20000	2013	1906	1892
100000	49435	49215	49228
500000	1217693	1216781	1217597
1000000	4788809	4790732	4789629
5000000	105478434	105489517	105469229

3.4. Tabelas de tamanho 1000000

Table 13. Comparações do tempo de execução da inserção em tabelas, de cada algoritimo para cada tamanho de conjunto de dados (tempo em milisegundos)

Tamanho	Divisão	Multiplicação	Dobramento
20000	7	10	6
100000	46	51	58
500000	149	240	106
1000000	300	506	257
5000000	2486	3407	2354

Table 14. Comparações do tempo de execução da busca em tabelas, de cada algoritimo para cada tamanho de conjunto de dados (tempo em milisegundos)

Tamanho	Divisão	Multiplicação	Dobramento
20000	8	10	6
100000	17	52	23
500000	47	207	59
1000000	93	487	133
5000000	829	2901	997

Table 15. Comparações do número de colisões em tabelas, de cada algoritimo para cada tamanho de conjunto de dados

Tamanho	Divisão	Multiplicação	Dobramento
20000	237	199	182
100000	4937	4825	4894
500000	106155	107504	106455
1000000	367399	369268	368020
5000000	4006727	4007459	4006776

Table 16. Comparações do número de comparações feitas em cada busca em tabelas, de cada algoritimo para cada tamanho de conjunto de dados

Tamanho	Divisão	Multiplicação	Dobramento
20000	217	177	159
100000	4593	4453	4540
500000	110003	111813	110536
1000000	434469	439609	436686
5000000	9586028	9674961	9589741

3.5. Tabelas de tamanho 10000000

Table 17. Comparações do tempo de execução da inserção em tabelas, de cada algoritimo para cada tamanho de conjunto de dados (tempo em milisegundos)

Tamanho	Divisão	Multiplicação	Dobramento
20000	8	10	7
100000	29	54	55
500000	563	213	120
1000000	449	1262	231
5000000	3623	3695	2532

Table 18. Comparações do tempo de execução da busca em tabelas, de cada algoritimo para cada tamanho de conjunto de dados (tempo em milisegundos)

Tamanho	Divisão	Multiplicação	Dobramento
20000	9	11	7
100000	19	44	21
500000	48	194	49
1000000	77	468	82
5000000	338	2599	550

Table 19. Comparações do número de colisões em tabelas, de cada algoritimo para cada tamanho de conjunto de dados

Tamanho	Divisão	Multiplicação	Dobramento
20000	24	34	24
100000	515	801	515
500000	12377	19166	12377
1000000	48273	74221	48273
5000000	1064497	1504829	1064497

Table 20. Comparações do número de comparações feitas em cada busca em tabelas, de cada algoritimo para cada tamanho de conjunto de dados

Tamanho	Divisão	Multiplicação	Dobramento
20000	0	10	0
100000	0	288	0
500000	0	7017	0
1000000	0	27724	0
5000000	0	606066	0

4. Discução

5. Conclusão

6. Referencias

GitHub com os códigos usados: Link