Resolução de Problemas Estruturados em Computação

RA03 - Hashing - Professor Andrey Cabral Meira

[1]PUCPR- Kauany Almeida Silva – Ciência da Computação 4a

Códigos no GitHub: https://github.com/kauany-almeida/recuperacao-hashing

1. Hashing de DIVISÃO

Analisando o desempenho utilizando o hash de divisão em diferentes quantidades de elementos em um vetor. Testando ao menos 5 vezes para cada tamanho de vetor medindo seu tempo de inserção, quantidade de colisões e comparações em uma tabela; e em seguida medindo o tempo para realizar buscas e os aumentos na quantidade de comparações em outra tabela:

1.1. Hashing de Divisão — 20.000 elementos — Executando 05 vezes :

Número da execução	Tempo de Inserção	Quantidade de Colisões	Quantidade de Comparações
01	26ms	2575965	2595965
02	26ms	2575965	2595965
03	25ms	2575965	2595965
04	25ms	2575965	2595965
05	26ms	2575965	2595965

Table 1. HASHING DE DIVISÃO - DESEMPENHO COM 20.000 ELEMENTOS

O tempo para a inserção utilizando hashing de divisão para um vetor de 20.000 elementos foi rápido, com quantidade de colisões e comparações não tão altas. Notei que o número dessas colisões comparações para todas as cinco execuções permaneceu o mesmo, e isso ocorre pois estou utilizando o seeds em meu programa e estou executando todas as vezes com as mesmas sementes, e sempre para um mesmo tamanho de vetor.

ID – busca	Posição	Elemento buscado	Tempo para realizar a busca	Quantidade Comparações
01	2	451840002	0ms	2595966
02	76	194660076	0ms	2595967
03	170	500240170	0ms	2595968
04	10	980440000	0ms	2595979
05	323	114960322	0ms	2595981

Table 2. HASHING DE DIVISÃO - DESEMPENHO DE BUSCA COM 20.000 ELEMENTOS

O tempo para as buscas é instantâneo em todas as 05 buscas feitas. A quantidade de comparações aumenta a cada execução também, porém tanto que aumenta vai depender de qual é posição em que está o elemento sendo buscado.

1.2. Hashing de Divisão — 100.000 elementos — Executando 05 vezes :

Número da execução	Tempo de Inserção	Quantidade de Colisões	Quantidade de Comparações
01	131ms	15401720	15501711
02	140ms	15401720	15501711
03	142ms	15401720	15501711
04	146ms	15401720	15501711
05	145ms	15401720	15501711

Table 3. HASHING DE DIVISÃO - DESEMPENHO COM 100.000 ELEMENTOS

Utilizando hashing de divisão para um vetor com 100.000 elementos, o tempo de inserção foi um pouco mais lento, porém continua rápido. O número de colisões e comparações aumentou bastante, se comparado à inserção anterior (em um vetor de 20.000 elementos). Porém, mesmo com o número grande de colisões e comparações o código executou relativamente rápido.

ID busca	Posição	Elemento buscado	Tempo para realizar a busca	Quantidade Comparações
01	7	250700007	0ms	15503024
02	183	255098872	0ms	15503024
03	237	286600237	0ms	15503025
04	319	335200319	0ms	15503026
05	1	946400001	0ms	15503027

Table 4. HASHING DE DIVISÃO - DESEMPENHO DE BUSCA COM 100.000 ELEMENTOS

Novamente o tempo de buscas é instantâneo. A quantidade de comparações aumentou, de novo de acordo com a distância que os elementos buscados tem entre si.

1.3. Hashing de Divisão — 500.000 elementos — Executando 05 vezes :

Número da execução	Tempo de Inserção	Quantidade de Colisões	Quantidade de Comparações
01	2426ms	268754390	269254252
02	2613ms	268754390	269254252
03	2473ms	268754390	269254252
04	2690ms	268754390	269254252
05	2513ms	268754390	269254252

Table 5. HASHING DE DIVISÃO - DESEMPENHO COM 500.000 ELEMENTOS

Dessa vez o tempo de inserção foi um pouco mais lento. Mas isso ocorre pela quantidade de elementos que está sendo inserida e também pela forma que cada um dos elementos está sendo inserido, e levando em conta esses fatores + os números de colisões e comparações (que são altos), esse tempo de inserção também foi um pouco favorável.

ID busca	Posição	Elemento buscado	Tempo para realizar a busca	Quantidade Comparações
01	9	377000007	0ms	269254255
02	342	303500342	0ms	269254256
03	284	444000284	0ms	269254257
04	58	931000052	0ms	269254264
05	1	190500001	0ms	269254265

Table 6. HASHING DE DIVISÃO - DESEMPENHO DE BUSCA COM 500.000 ELEMENTOS

A busca é instantânea. Para esse tamanho de vetor e para os elementos que foram buscados, não obtivemos um grande aumento na quantidade de comparações.

1.4. Hashing de Divisão — 1.000.000 elementos — Executando 05 vezes :

Número da execução	Tempo de Inserção	Quantidade de Colisões	Quantidade de Comparações
01	3321ms	359845701	360845165
02	3443ms	359845701	360845165
03	3318ms	359845701	360845165
04	3347ms	359845701	360845165
05	2869ms	359845701	360845165

Table 7. HASHING DE DIVISÃO - DESEMPENHO COM 1.000.000 ELEMENTOS

Notei que quanto mais elementos um vetor possui, maior será o tempo de inserção de cada um deles utilizando hashing. Dessa vez houve uma média de tempo de inserção maior que das vezes anteriores, mas novamente foi em um tempo relativamente rápido se comparamos com outras formas de hashing.

ID – busca	Posição	Elemento buscado	Tempo para realizar a busca	Quantidade Comparações
01	12	661000011	0ms	360845167
02	4	868000004	0ms	360845168
03	338	160998940	0ms	360846567
04	302	443000302	0ms	360846568
05	239	949000214	0ms	360846594

Table 8. HASHING DE DIVISÃO - DESEMPENHO DE BUSCA COM 1.000.000 ELEMENTOS

O tempo de buscas é sempre instantâneo. Dessa vez, com os elementos que foram buscados e com a ”distância” que um tem do outro, em algumas buscas conseguimos obter um grande aumento de quantidade de comparações.

2. Hashing de Divisão — Média de resultados de desempenho obtida para cada um dos tamanhos de vetores:

Quantidade de Elementos em cada Vetor	Média - n* Colisões	Média - n* de Comparações
20.000	2575965	2595965
100.000	15401720	15501711
500.000	268754390	269254252
1.000.000	359845701	360845165

Table 9. Hashing de Divisão, Médias de Colisões e Médias de Compatações

Na tabela acima vemos o número de colisões e o número de comparações diferentes que obtivemos, em cada um dos vetores de tamanhos diferentes.

Comparando com os outros dois tipos de hashing, este aqui foi o segundo com melhor desempenho.

Quantidade de Vetores	Média - Tempo para INSERÇÕES	Média - Tempo para BUSCAS
20.000	25.6 ms	0ms
100.000	140.8 ms	0ms
500.000	2543.0 ms	0ms
1.000.000	3259.6 ms	0ms

Table 10. Hashing Divisão, Média de Tempo para Inserções e Buscas

A tabela acima mostra o tempo médio para inserir e também o tempo médio para buscar elementos, para cada tamanho de vetor especificado na coluna da esquerda.

Essa foi a média de resultados de desempenho que obtivemos utilizando hashing de divisão em vetores de tamanhos 20 mil, 100 mil, 500 mil, e 1 milhão.

3. Hashing de MULTIPLICAÇÃO

Agora vamos analisar o desempenho para inserção, busca, quantidade de colisões e quantidade de comparações utilizando as funções de 'insere-hashing' e 'busca-hashing', no hash de multiplicação. Novamente, iremos executar 5 vezes para cada um dos tamanhos de vetor e por fim exibiremos a média obtida para esse tipo de hash.

3.1. Hashing de Multiplicação — 20.000 elementos — Executando 05 vezes :

Número da execução	Tempo de Inserção	Quantidade de Colisões	Quantidade de Comparações
01	1625ms	199990000	200010000
02	1677ms	199990000	200010000
03	1659ms	199990000	200010000
04	1679ms	199990000	200010000
05	1607ms	199990000	200010000

Table 11. HASHING DE MULTIPLICAÇÃO - DESEMPENHO COM 20.000 ELEMENTOS

Comparado ao tempo de inserções utilizando hashing de divisão para a mesma quantidade de elementos, este aqui foi relativamente mais lento. Houve um aumento muito significativo na quantidade de comparações e colisões também.

ID busca	Posição	Elemento buscado	Tempo para realizar a busca	Quantidade Comparações
01	7	930591586	0ms	200010008
02	365	817952521	0ms	200010374
03	255	183394673	0ms	200010630
04	134	915752975	0ms	200010765
05	5	507388835	0ms	200010771

Table 12. HASHING DE MULTIPLICAÇÃO - DESEMPENHO DE BUSCA COM 20.000 ELEMENTOS

Em relação às buscas para o hashing de multiplicação em um vetor de 20.000 elementos, não notamos mudanças significativas comparado aos desempenhos de outras funções em outras tabelas. O tempo permanece instantâneo, e com aumentos consideráveis na quantidade de comparações.

3.2. Hashing de Multiplicação — 100.000 elementos — Executando 05 vezes :

Número da execução	Tempo de Inserção	Quantidade de Colisões	Quantidade de Comparações
01	39846ms	704342131	704442122
02	39631ms	704342131	704442122
03	39397ms	704342131	704442122
04	43001ms	704342131	704442122
05	37997ms	704342131	704442122

Table 13. HASHING DE MULTIPLICAÇÃO - DESEMPENHO COM 100.000 ELEMENTOS

O tempo de inserção foi novamente mais lento do que o esperado, e a quantidade de colisões e comparações aumenta muito também comparado à quantidae de comparações e colisões que obtemos na forma de hashing anterior.

ID busca	Posição	Elemento buscado	Tempo para realizar a busca	Quantidade Comparações
01	194	755382732	0ms	704442317
02	206	292989762	0ms	704442524
03	282	440326462	0ms	704442807
04	113	665143236	0ms	704516912
05	176	111137408	0ms	704517089

Table 14. HASHING DE MULTIPLICAÇÃO - DESEMPENHO DE BUSCA COM 100.000 ELEMENTOS

Tempo para realizar buscas é instantâneo, aumento na quantidade de comparações a cada busca feita.

3.3. Hashing de Multiplicação — 500.000 elementos — Executando 05 vezes :

Número da execução	Tempo de Inserção	Quantidade de Colisões	Quantidade de Comparações
01	978785ms	400691206	401191068
02	1008168ms	400691206	401191068
03	968914ms	400691206	401191068
04	1079216ms	400691206	401191068
05	1062518ms	400691206	401191068

Table 15. HASHING DE MULTIPLICAÇÃO - DESEMPENHO COM 500.000 ELEMENTOS

Comparado aos resultados obtidos anteriormente no hashing de divisão, este aqui não nos forneceu um tempo de inserção favorável para essa quantidade de elementos. A quantidade de colisões e comparações segue alta.

ID busca	Posição	Elemento buscado	Tempo para realizar a busca	Quantidade Comparações
01	56	689957519	0ms	401191125
02	202	107707277	0ms	401191328
03	237	107954619	0ms	401191566
04	307	771191446	0ms	401191874
05	347	383230408	0ms	401192222

Table 16. HASHING DE MULTIPLICAÇÃO - DESEMPENHO DE BUSCA COM 500.000 ELEMENTOS

Mesmo resultado obtido em outras buscas em relação ao tempo levado para buscar um elemento, e dessa vez obtivemos um aumento considerável olhando os resultados a cada busca que foi feita.

3.4. Hashing de Multiplicação — 1.000.000 elementos — Executando 05 vezes :

Número da execução	Tempo de Inserção	Quantidade de Colisões	Quantidade de Comparações
01	3957732ms	1427718930	1428718394
02	5669777ms	1427718930	1428718394
03	3917220ms	1427718930	1428718394
04	5590441ms	1427718930	1428718394
05	5016050ms	1427718930	1428718394

Table 17. HASHING DE MULTIPLICAÇÃO - DESEMPENHO COM 1.000.000 ELEMENTOS

Para um vetor de 1.000.000 de elementos, o tempo de inserção foi um pouco mais longo do que o esperado. Também houve um número grande de colisões e comparações.

ID busca	Posição	Elemento buscado	Tempo para realizar a busca	Quantidade Comparações
01	7	930591586	0ms	1428718402
02	111	823327382	1ms	1428718514
03	247	315846453	0ms	1428718762
04	282	440326462	0ms	1428719045
05	46	405157555	0ms	1428719092

Table 18. HASHING DE MULTIPLICAÇÃO - DESEMPENHO DE BUSCA COM 1.000.000 ELEMENTOS

No hashing de multiplicação com 1.000.000 de elementos, obtivemos pela única vez um resultado diferente de 0ms em uma das buscas realizadas. Então, isso nos dá uma média de tempo de buscas de 0.2 milissegundos para esse tipo de hashing com esse tamanho de vetor. Ainda assim, é um tempo muito bom para essa grande quantidade de elementos.

4. Hashing de Multiplicação — Média de resultados de desempenho obtida para cada um dos tamanhos de vetores:

Quantidade de Elementos em cada Vetor	Média - n* Colisões	Média - n* de Comparações
20.000	199990000	200010000
100.000	704342131	704442122
500.000	400691206	401191068
1.000.000	1427718930	1428718394

Table 19. Hashing de Multiplicação, Médias de Colisões e Médias de Comparações

Na tabela acima vemos o número de colisões e o número de comparações diferentes que obtivemos, em cada um dos vetores de tamanhos diferentes.

Quantidade de Vetores	Média - Tempo para	Média - Tempo para BUSCAS
20.000	1649.4 ms	0ms
100.000	39974.4 ms	0ms
500.000	1019520.2 ms	0ms
1.000.000	5097601.0 ms	0.2ms

Table 20. Hashing de Multiplicação, Média de Tempo para Inserções e Buscas

A tabela acima mostra o tempo médio para inserir e também o tempo médio para buscar elementos, para cada tamanho de vetor especificado na coluna da esquerda.

Essa foi a média de resultados de desempenho que obtivemos utilizando hashing de multiplicação em vetores de tamanhos 20 mil, 100 mil, 500 mil, e 1 milhão.

5. Hashing de DOBRAMENTO

Utilizando dessa vez a técnica de hashing de dobramento, vamos analisar o desempenho do código executando no mínimo 5 vezes para cada vetor, em vetores de tamanhos diferentes. Iremos armazenar os resultados de cada execução em tabelas difererentes, e em seguida exibiremos uma última tabela contendo a média de todos os resultados obtidos.

5.1. Hashing de DOBRAMENTO — 20.000 elementos — Executando 05 vezes :

Número da execução	Tempo de Inserção	Quantidade de Colisões	Quantidade de Comparações
01	12ms	1076770	1096770
02	8ms	1076770	1096770
03	11ms	1076770	1096770
04	11ms	1076770	1096770
05	10ms	1076770	1096770

Table 21. HASHING DE DOBRAMENTO - DESEMPENHO COM 20.000 ELEMENTOS

Comparando com os resultados obtidos no hashing de divisão e hashing de multiplicação, este aqui é o que tem o seu tempo de inserção mais rápido. A quantidade de comparações e colisões também é menor do que as outras que vimos anteriormente.

ID busca	Posição	Elemento buscado	Tempo para realizar a busca	Quantidade Comparações
01	2	858000187	0ms	1096771
02	211	741015829	0ms	1096772
03	145	510010818	0ms	1096774
04	132	994489825	0ms	1097043
05	360	451527001	0ms	1097044

Table 22. HASHING DE DOBRAMENTO - DESEMPENHO DE BUSCA COM 20.000 ELEMENTOS

Aqui, os resultados seguem saindo como o esperado. Tempo para realizar buscas é instantâneo, e aumentos consideráveis na quantidade de comparações a cada busca feita.

5.2. Hashing de DOBRAMENTO — 100.000 elementos — Executando 05 vezes :

Número da execução	Tempo de Inserção	Quantidade de Colisões	Quantidade de Comparações
01	160ms	19792359	19892350
02	155ms	19792359	19892350
03	164ms	19792359	19892350
04	155ms	19792359	19892350
05	161ms	19792359	19892350

Table 23. HASHING DE DOBRAMENTO - DESEMPENHO COM 100.000 ELEMENTOS

O tempo de inserção novamente foi o mais rápido até agora em comparação com os outros hashings que vimos anteriormente, e a quantidade de comparações feitas é menor que as vistas anteriormente em outras tabelas também.

ID busca	Posição	Elemento buscado	Tempo para realizar a busca	Quantidade Comparações
01	4	127500331	0ms	19892351
02	237	285017840	0ms	19892353
03	93	840007000	0ms	19892354
04	309	877440867	0ms	19893453
05	278	525020778	0ms	19893455

Table 24. HASHING DE DOBRAMENTO - DESEMPENHO DE BUSCA COM 100.000 ELEMENTOS

Aqui, os resultados continuam saindo como esperado. Tempo para realizar buscas é instantâneo, e aumentos consideráveis na quantidade de comparações a cada busca feita.

5.3. Hashing de DOBRAMENTO — 500.000 elementos — Executando 05 vezes :

Número da execução	Tempo de Inserção	Quantidade de Colisões	Quantidade de Comparações
01	1771ms	217951229	218451091
02	1813ms	217951229	218451091
03	1808ms	217951229	218451091
04	1794ms	217951229	218451091
05	1777ms	217951229	218451091

Table 25. HASHING DE DOBRAMENTO - DESEMPENHO COM 500.000 ELEMENTOS

Um tempo muito rápido para essa quantidade de elementos dentro de um vetor. Já o número de colisões e comparações, foi esperado.

ID busca	Posição	Elemento buscado	Tempo para realizar a busca	Quantidade Comparações
01	28	750002126	0ms	218451092
02	86	375006512	0ms	218451093
03	9	599999663	0ms	218451108
04	221	187516588	0ms	218451109
05	315	937493414	0ms	218451513

Table 26. HASHING DE DOBRAMENTO - DESEMPENHO DE BUSCA COM 500.000 ELEMENTOS

Novamente, obtivemos um tempo de buscas instantâneo em todas as buscas realizadas.

5.4. Hashing de DOBRAMENTO — 1.000.000 elementos — Executando 05 vezes :

Número da execução	Tempo de Inserção	Quantidade de Colisões	Quantidade de Comparações
01	5944ms	764619081	765618545
02	6228ms	764619081	765618545
03	5902ms	764619081	765618545
04	5869ms	764619081	765618545
05	6307ms	764619081	765618545

Table 27. HASHING DE DOBRAMENTO - DESEMPENHO COM 1.000.000 ELEMENTOS

Para um vetor de 1.000.000 de elementos, o tempo de inserção aqui foi muito rápido. A quantidade de colisões e comparações também foi boa para esse grande número de elementos.

ID busca	Posição	Elemento buscado	Tempo para realizar a busca	Quantidade Comparações
01	82	300006200	0ms	765618546
02	183	750013668	0ms	765618548
03	65	525003940	0ms	765618562
04	285	150020918	0ms	765618570
05	355	900026638	0ms	765618571

Table 28. HASHING DE DOBRAMENTO - DESEMPENHO DE BUSCA COM 1.000.000 ELEMENTOS

O tempo de buscas foi como o esperado (instantâneo), e a quantidade de comparações aumenta relativamente de acordo com a distância que cada um dos elementos buscados têm entre si.

6. Hashing de Dobramento — Média de resultados de desempenho obtida para cada um dos tamanhos de vetores:

Quantidade de Elementos em cada Vetor	Média - n* Colisões	Média - n* de Comparações
20.000	1076770	1096770
100.000	19792359	19892350
500.000	217951229	218451091
1.000.000	764619081	765618545

Table 29. Hashing de Dobramento, Médias de Colisões e Médias de Compatações

Na tabela acima vemos o número de colisões e o número de comparações diferentes que obtivemos, em cada um dos vetores de tamanhos diferentes.

Comparando com os dois outros tipos de hashing e os desempenhos que cada um teve, este aqui é o com o tempo de inserção mais rápido.

Quantidade de Vetores	Média - Tempo para INSERÇÕES	Média - Tempo para BUSCAS
20.000	10.4 ms	0ms
100.000	159.0 ms	0ms
500.000	1792.6 ms	0ms
1.000.000	6050.0 ms	0ms

Table 30. Hashing de Dobramento, Média de Tempo para Inserções e Buscas

A tabela acima mostra o tempo médio para inserir e também o tempo médio para buscar elementos, para cada tamanho de vetor especificado na coluna da esquerda.

Essa foi a média de resultados de desempenho que obtivemos utilizando hashing de dobramento em vetores de tamanhos 20 mil, 100 mil, 500 mil e 1 milhão.