Análise de desempenho de algoritmos para resolução de sistemas lineares

Tiago Funk[1]

[1]Universidade do Estado de Santa Catarina

tiagoff.tf@gmail.com

Abstract. *abc*

Resumo. *abc*

Introdução

Fundamentação Teórica

Um sistema linear é um conjunto de n equações lineares com m variáveis, para ser uma equação linear, os expoentes das variáveis devem ser todos iguais a 1 [Bezerra 2001]. O algoritmo implementado neste artigo é o algoritmo de Gauss, ele consiste em transformar o sistemas original em um sistema equivalente com matriz dos coeficientes triangular superior, pois estes são de resolução imediata [Ruggiero and da Rocha Lopes 1996].

A linguagem C++ é uma linguagem compilada e que suporta orientação a objetos. Como essa linguagem compila diretamente para o código nativo de uma máquina, permite que ela seja uma das linguagens mais rápidas do mundo [des a]. Python é uma linguagem interpretada e com orientação a objetos, o fato da linguagem ser interpretada permite que ela seja portável para vários sistemas diferentes [des b], é uma linguagem de programação em que o código fonte é executado por um programa de computador chamado interpretador e não diretamente pelo processador. Java é uma linguagem compilada e que suporta orientação a objetos. Ao contrário do C++, a compilação do não gera código de máquina diretamente, mas sim bytecodes, os bytecodes são executados por uma máquina virtual do java (JVM). A JVM é um aplicativo de software que simula um computador, mas oculta o sistema operacional e o hardware da máquina dos programas que interagem com ela. Os programas escritos em Java podem ser portados para outros sistemas sem compilação graças a JVM [Deitel and Deitel 2017].

Experimento

O objetivo do estudo é avaliar a execução de algoritmos de resolução de sistemas lineares em Java, C++ e Python a fim de entender qual possui melhor desempenho examinando o tempo de execução do ponto de vista de desenvolvedores no contexto de aplicações desktop.

Os experimentos que serão realizados consistem em utilizar os algoritmos implementados para resolver sistemas lineares de ordem *n*, aonde $n \geq 100$ e $n \leq 300$, essas execuções gerarão dados numéricos sobre o tempo que foi necessário para executar o algoritmo e o tempo que necessário para realizar a leitura dos sistemas a partir de arquivos em disco.

Questões:

1. Descobrir qual implementação possui melhor desempenho para a leitura do arquivo.
2. Descobrir qual implementação possuir melhor desempenho para a resolução do sistema linear.

Essas duas questões serão abordadas de forma independente, pois não adianta a linguagem oferecer um desempenho melhor no algoritmo, mas ter um desempenho muito inferior em questão da leitura de disco. Uma linguagem não ter um desempenho muito bom para leitura do arquivo vai, na prática, por remover uma possível vantagem no seu desempenho do algoritmo.

Métricas:

1. Ordem *n* do sistema linear.
2. Tempo de execução para a leitura dos sistemas lineares.
3. Tempo de execução para a resolução dos sistemas lineares.

Como o tempo de execução da leitura e execução do algoritmo pode variar muito em função do tamanho do sistema, a avaliação do desempenho será feito apenas comparando sistemas de mesma ordem, ou seja, para sistemas de ordem 100, será feito a comparação das duas questões apenas para esse ordem. Um sistema de ordem diferente, será feito uma nova comparação.

Materiais e Métodos

Os sistemas lineares que serão utilizados para o experimento foram gerados por um programa que cria o sistema $n \times n$ de forma aleatória, aonde os coeficientes das equações eram valores reais no intervalo de -10 e 10, inclusive e os valores das variáveis são valores no intervalo real de 0 e 10, inclusive. O valor de *n* é informado ao programa. Neste trabalho foram utilizados sistemas com $n \geq 100$ e $n \leq 300$, pois sistemas de *n* muito pequenos são resolvidos e lidos de forma tão rápida que mesmo considerando o tempo em milissegundos, é muito rápido para ser registrado.

Cada linguagem oferece uma função para saber quanto tempo uma determinada operação demora para ser realizada. No Java foi utilizado o *System.currentTimeMillis()*, esse método retorna um valor em milissegundos absoluto [tim a]. Para descobrir quanto tempo uma operação demorou, é preciso chamar o método antes e depois da operação e calcular a diferença. Na linguagem C++ foi utilizado a função *clock()*, ela retorna o número de interações que o relógio do sistema desde o lançamento do programa[1][clo], chamando a função antes e depois da operação e calculando a diferença entre ambos, temos um valor de interações do relógio, é utilizado a constante *CLOCKS_PER_SEC* para uma conversão para segundos [mac], depois é feita uma uma conversão para milissegundos. Em Python foi utilizado a função *time.time()*, ela retorna o número de segundos desde uma época, geralmente desde meia noite de 1 de janeiro de 1970 [tim b], o mesmo raciocínio para o calculo do tempo de execução é feito para chegar no tempo em segundos, depois só é feita uma conversão para os milissegundos.

O experimento consiste na resolução de 21 sistemas lineares pelo algoritmo de Gauss implementado em Java, C++ e Python. Foram executados 20 vezes o algoritmo de resolução em cada implementação para cada instância. As execuções ocorreram em um

[1]Geralmente é lançamento do programa, mas pode variar entre sistemas.

computador com Intel Celeron(R) CPU B820 de 1.70GHz x 2 com 3,7 GiB de memória RAM e instalado o sistema operacional Ubuntu 16.04 LTS. O java foi compilado na versão 8, C++ na versão C++98 e o python foi executado na versão 3.

Resultados e Discussão

Ordem	Média Leitura	Des pad Leitura	Maior Leitura	Menor Leitura	Média Cálculo	Des Pad Cálculo	Maior Cálculo	Menor Cálculo
100	7,49	1,00	9,03	5,87	5,25	0,60	6,29	4,58
110	8,78	1,27	11,37	7,06	6,61	0,62	7,93	6,07
120	9,72	1,32	12,75	8,38	8,17	0,36	9,04	7,87
130	11,60	1,52	14,31	9,84	10,28	0,34	10,86	9,95
140	12,80	1,61	15,83	11,37	12,55	0,21	13,02	12,40
150	15,48	1,64	18,43	13,03	15,33	0,15	15,64	15,20
160	17,31	2,16	20,28	14,81	18,50	0,07	18,74	18,41
170	19,11	1,68	21,83	16,75	22,19	0,20	22,98	22,06
180	21,28	1,94	24,47	18,73	26,30	0,25	27,32	26,13
190	23,12	2,19	28,61	20,86	31,06	0,63	33,24	30,69
200	25,15	1,87	28,43	23,14	35,85	0,07	36,05	35,76
210	28,16	2,08	33,05	25,47	41,56	0,43	43,41	41,36
220	30,08	1,71	33,40	27,91	47,69	0,15	47,98	47,52
230	33,25	1,66	35,99	30,57	54,87	1,18	59,52	54,26
240	35,53	2,03	38,95	33,20	61,87	0,15	62,29	61,62
250	39,14	2,00	41,64	36,05	69,95	0,15	70,49	69,81
260	41,25	2,02	44,59	38,99	78,72	0,23	79,35	78,40
270	43,37	1,46	46,47	42,00	88,37	0,62	90,48	87,78
280	48,58	2,75	57,08	45,15	98,31	0,49	99,58	97,79
290	50,75	1,88	54,22	48,43	109,46	1,50	115,18	108,53
300	54,21	2,02	57,92	51,82	121,19	1,17	125,19	120,26

Tabela 1. C++

Ordem	Média Leitura	Des pad Leitura	Maior Leitura	Menor Leitura	Média Cálculo	Des Pad Cálculo	Maior Cálculo	Menor Cálculo
100	44,10	7,93	69,00	37,00	16,45	5,80	33,00	9,00
110	48,40	6,56	61,00	41,00	18,05	6,43	29,00	10,00
120	55,00	6,18	65,00	46,00	21,90	5,90	36,00	13,00
130	61,45	11,15	92,00	49,00	22,10	2,98	27,00	17,00
140	68,10	8,56	87,00	56,00	27,90	10,03	69,00	19,00
150	67,70	10,94	104,00	58,00	29,70	11,37	76,00	20,00
160	76,45	12,81	124,00	65,00	29,95	5,57	47,00	24,00
170	81,10	15,13	136,00	67,00	34,25	6,19	55,00	27,00
180	89,70	13,89	115,00	69,00	36,15	5,18	49,00	31,00
190	94,55	16,93	128,00	73,00	42,35	3,92	51,00	36,00
200	91,45	8,87	107,00	79,00	43,40	2,31	48,00	38,00
210	104,35	12,34	125,00	83,00	52,40	5,30	66,00	47,00
220	99,00	9,52	123,00	88,00	53,65	4,53	64,00	43,00
230	119,90	15,55	153,00	94,00	62,10	5,62	71,00	44,00
240	108,85	11,31	128,00	93,00	59,40	5,15	71,00	51,00
250	114,45	13,93	156,00	98,00	60,65	5,17	70,00	50,00
260	120,55	12,19	141,00	105,00	65,75	7,35	79,00	52,00
270	134,45	15,64	159,00	104,00	84,40	12,65	106,00	59,00
280	124,90	13,25	147,00	106,00	71,40	6,97	87,00	61,00
290	135,75	18,02	187,00	116,00	71,55	6,09	85,00	61,00
300	141,30	16,60	173,00	120,00	75,85	9,63	92,00	62,00

Tabela 2. Java

Ordem	Média Leitura	Des pad Leitura	Maior Leitura	Menor Leitura	Média Cálculo	Des Pad Cálculo	Maior Cálculo	Menor Cálculo
100	9,17	0,43	10,29	8,77	224,25	7,85	254,20	218,11
110	11,53	0,70	13,07	10,81	293,69	4,09	306,48	288,02
120	13,33	0,63	14,84	12,59	383,99	18,81	463,69	373,04
130	15,32	0,63	17,32	14,69	482,68	11,07	518,20	473,55
140	17,61	0,80	19,51	16,80	598,88	5,88	621,03	591,48
150	20,23	1,68	26,18	18,94	748,77	48,87	959,27	723,35
160	24,88	3,56	36,77	21,73	894,14	21,32	983,55	880,98
170	27,08	2,63	33,75	24,72	1064,58	12,99	1114,50	1048,99
180	29,80	3,17	36,90	26,94	1263,71	17,39	1322,53	1246,89
190	33,59	2,99	41,91	30,40	1492,10	48,56	1699,96	1462,35
200	34,72	1,98	38,63	32,57	1726,50	11,89	1758,03	1711,44
210	36,76	1,61	43,30	35,28	2005,79	56,99	2250,17	1976,28
220	40,83	1,79	45,98	38,90	2291,76	15,12	2326,92	2272,02
230	45,15	3,06	54,41	42,21	2618,56	19,73	2673,98	2595,20
240	48,73	3,14	55,64	45,85	2970,75	13,73	3007,72	2950,50
250	52,99	3,16	59,08	49,61	3362,41	39,61	3508,89	3316,72
260	55,84	2,89	65,10	53,39	3797,14	29,07	3876,17	3761,25
270	60,38	2,69	68,19	57,45	4264,42	19,01	4299,93	4232,59
280	64,57	2,54	74,00	62,01	4789,14	103,09	5224,52	4728,24
290	71,06	4,90	83,19	66,29	5327,51	113,37	5811,04	5259,26
300	75,09	3,82	84,33	70,97	5928,81	99,84	6227,35	5859,72

Tabela 3. Python

Conclusões

Referências

C++ a brief description. http://www.cplusplus.com/info/description/. Acessado em: 20-11-2018.

General python faq. https://docs.python.org/3/faq/general.html#what-is-python. Acessado em: 20-11-2018.

Referência da linguagem c++ - função clock. http://www.cplusplus.com/reference/ctime/clock/. Acessado em: 20-11-2018.

Referência da linguagem c++ - macro clocks_per_sec. http://www.cplusplus.com/reference/ctime/CLOCKS_PER_SEC/. Acessado em: 20-11-2018.

Referência da linguagem java - funçao currenttimemillis. https://docs.oracle.com/javase/7/docs/api/java/lang/System.html#currentTimeMillis(). Acessado em: 20-11-2018.

Referência da linguagem python - funão time. https://docs.python.org/3/library/time.html#time.time. Acessado em: 20-11-2018.

Bezerra, M. J. (2001). *Matemática para o ensino médio*. Editora Scipione, 5ª edition.

Deitel, P. and Deitel, H. (2017). *Java: como programar*. Pearson Education do Brasil, 10ª edition.

Ruggiero, M. A. G. and da Rocha Lopes, V. L. (1996). *Cálculo numérico. Aspectos teóricos e computacionais*. Pearson Makron Books, 2ª edition.