MAC0219 Introdução à Computação Concorrente, Paralela e Distribuída

Mini EP: *Overhead* e *Starvation* em Algoritmos para a Seção Crítica

Nome: Julio Kenji Ueda NUSP: 9298281

Procedimento

O programa sem modificações recebe como variáveis <number_of_threads> (número de threads) e <total_time> (quantidade de acessos à seção crítica) e imprime os resultados de 30 testes de cada algoritmo, onde cada resultado exibe o tipo do algoritmo utilizado, tempo de execução, a média dos acessos à seção crítica e o desvio padrão, além da quantidade de acesso à seção crítica de cada thread.

Para a análise foram realizadas pequenas mudanças no código **main.c** de forma que agora:

- imprime a média dos desvios padrões dos 30 testes de cada algoritmo;
- imprime a média do tempo de execução dos 30 testes de cada algoritmo.

Analisar o comportamento dos algoritmos através do desvio padrão do número de acessos é mais informativo do que simplesmente observar a média de acessos, pois é possível detectar a presença de *starvation* (falta de justiça). Com o tempo médio de execução é possível detectar a presença de *overhead* (processamento excessivo).

Foi utilizado um computador com processador Intel Core i5-5200U @ 2.20GHz (2 núcleos físicos, 4 threads, 64KB L1, 256KB L2, 3072KB L3), 8GB de RAM e OS Linux Mint 18.3

Análise

Foram utilizados três escalonadores (OTHER, FIFO e RR), dez variações de <number_of_threads> e <total_time> variando de 500 mil a 3 milhões, em intervalos de 500 mil (seis variações discretas).

Um valor alto para o desvio padrão médio indica a presença de *starvation*, enquanto um valor alto de tempo de execução indica a presença de *overhead*.

Os gráficos possuem escalas similares para facilitar a comparação entre os algoritmos e escalonadores. A escala máxima é definida pelo maior valor encontrado na análise.

A remoção da primitiva *__sync_syncronize*() gera inconsistência nos resultados, portanto não foi incluída na análise.

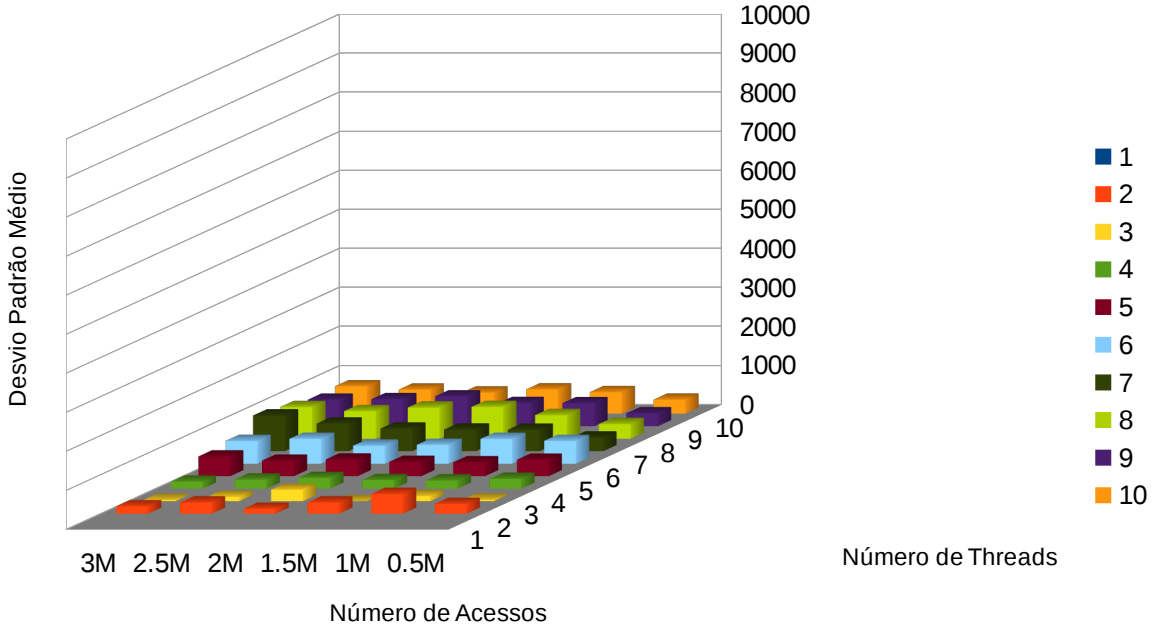
Escalonador OTHER (default)

Desvio Padrão Médio

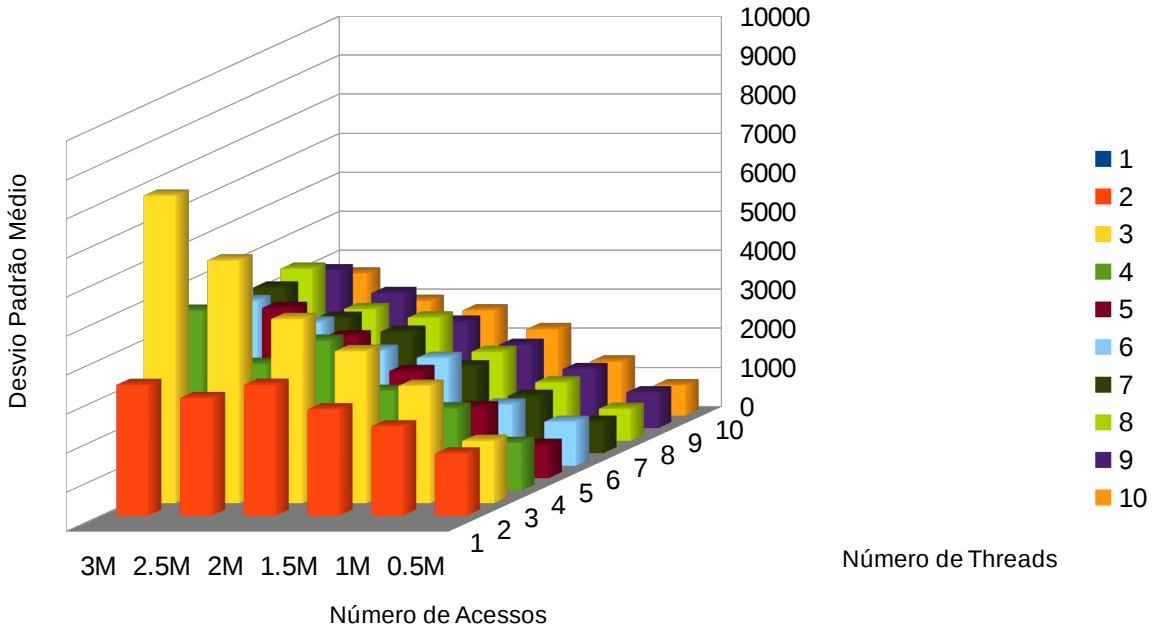
- *Bakery* apresenta pouco *starvation*, que cresce um pouco a partir quatro <number_of_threads>.
- *Gate* possui muito starvation, que cresce em função do <total_time> e é absurdamente grande com três <number_of_threads>.

Bakery	<number_of_threads>									
<total_time>	1	2	3	4	5	6	7	8	9	10
0.5M	0.0	257.4	43.0	258.7	426.8	589.4	354.8	381.0	334.4	363.8
1M	0.0	511.4	133.7	218.2	359.2	639.7	554.9	609.4	595.0	556.7
1.5M	0.0	295.8	5.1	225.0	373.7	487.6	558.5	819.3	599.9	625.5
2M	0.0	131.6	299.7	282.0	433.1	466.0	583.4	803.0	769.1	547.6
2.5M	0.0	293.4	119.6	238.0	403.3	641.7	713.3	710.5	696.8	620.5
3M	0.0	197.1	39.4	185.9	511.1	588.2	919.3	816.7	683.1	707.3

Gate	<number_of_threads>									
<total_time>	1	2	3	4	5	6	7	8	9	10
0.5M	0.0	1596.0	1601.2	1229.8	855.6	1128.0	809.4	820.7	920.4	778.7
1M	0.0	2291.6	3012.3	2112.2	1822.2	1571.7	1471.8	1497.2	1517.2	1387.3
1.5M	0.0	2726.2	3898.7	2564.4	2749.5	2765.1	2226.3	2280.8	2139.2	2216.2
2M	0.0	3354.4	4721.9	3845.2	3637.8	2950.9	3112.7	3157.4	2731.3	2700.6
2.5M	0.0	3015.6	6222.9	3250.5	4364.3	3671.0	3474.1	3376.6	3465.4	2941.8
3M	0.0	3353.3	7880.4	4611.1	4196.3	4222.7	4247.0	4407.1	4049.2	3644.9



Algoritmo Bakery



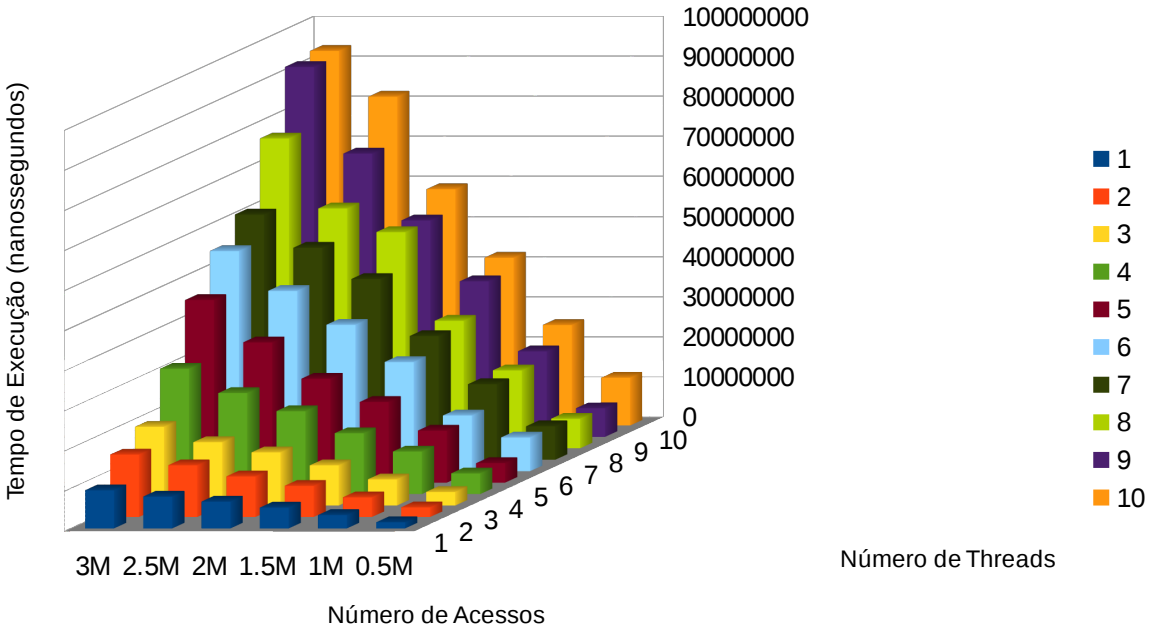
Algoritmo Gate

Tempo de Execução

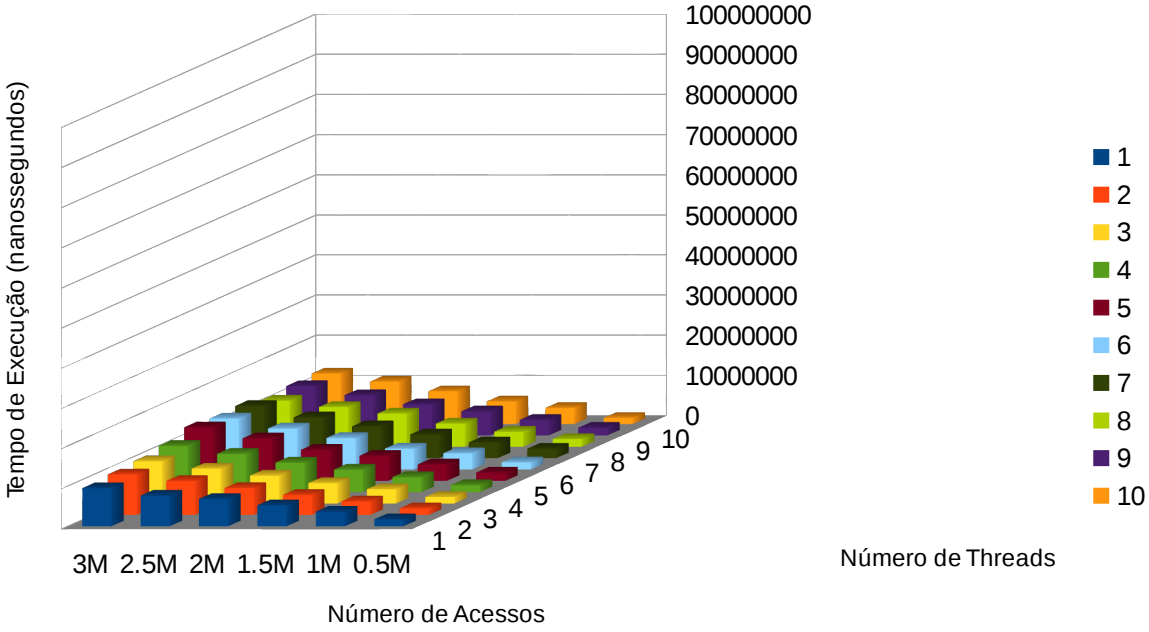
- *Bakery* apresenta *overhead*, que cresce em função do <number_of_threads> e <total_time>.
- *Gate* possui pouco *overhead*, que cresce em função do <total_time> e não depende do <number_of_threads>.

Bakery	<number_of_threads>									
<total_time>	1	2	3	4	5	6	7	8	9	10
0.5M	1583469	2448988	3440086	5203670	4921181	8480234	8407192	7435152	7202298	11963328
1M	3389039	4954817	6581097	10679303	13034992	13956134	18916382	19493966	21413252	25109901
1.5M	5250196	7814042	10075158	15230287	20184354	27281546	30915375	31872649	38852938	41895406
2M	6747430	10189812	13304591	20735541	25983120	36580283	45133132	53986084	54040277	58979049
2.5M	7996509	12958120	15862844	25256754	35040302	45034958	52935851	59874265	70701060	82045078
3M	9574847	15623391	19725945	31328913	45600020	54973179	61230416	77343823	92316649	93462771

Gate	<number_of_threads>									
<total_time>	1	2	3	4	5	6	7	8	9	10
0.5M	1750435	1680501	1679916	1798723	2097981	1831247	2388320	1970358	1943651	1599016
1M	3674070	3395639	3618985	3797920	4153682	4069400	3967959	3790900	3899936	4072965
1.5M	5254879	5017352	5145166	5666562	6285963	5237762	5936346	5809734	6023391	5702280
2M	6829655	6710982	7005554	7420480	7773435	7867421	7956493	8313045	7776194	8221433
2.5M	7660150	8478714	8744584	9584500	10609563	10243046	10183093	9941972	10053424	10641176
3M	9516984	10119198	10574413	11600192	13360931	12754905	13038790	11543456	12350634	12627040



Algoritmo Bakery



Algoritmo Gate

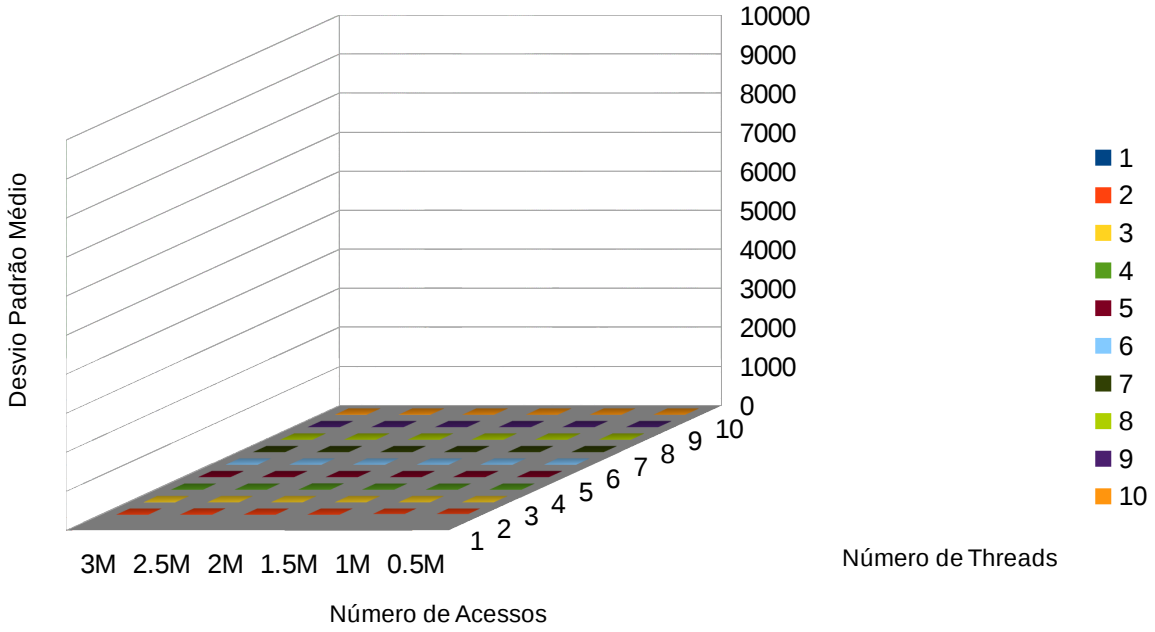
Escalonador FIFO (first-in first-out)

Desvio Padrão Médio

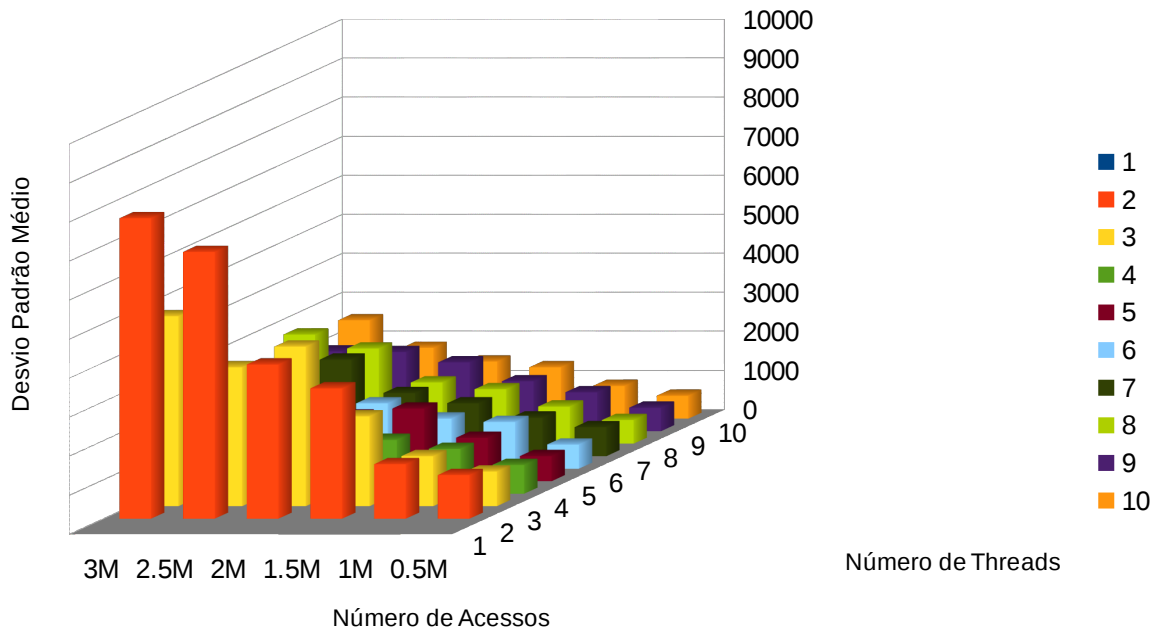
- *Bakery* praticamente não possui *starvation.*
- *Gate* possui *starvation* que cresce em função do <total_time> é absurdamente grande com dois a três <number_of_threads>.

Bakery	<number_of_threads>									
<total_time>	1	2	3	4	5	6	7	8	9	10
0.5M	0.0	0.8	1.4	2.1	3.3	4.2	3.0	3.4	3.4	2.9
1M	0.0	1.3	1.8	2.4	3.5	3.0	2.9	3.6	3.5	2.8
1.5M	0.0	2.0	1.9	2.4	2.4	3.5	3.5	3.6	3.3	3.1
2M	0.0	1.9	1.5	3.4	3.3	3.0	3.5	3.2	2.7	2.6
2.5M	0.0	4.6	1.8	3.3	3.8	2.6	3.5	2.9	3.5	3.5
3M	0.0	3.2	2.6	2.6	3.3	3.2	3.2	3.1	3.7	3.5

Gate	<number_of_threads>									
<total_time>	1	2	3	4	5	6	7	8	9	10
0.5M	0.0	1124.6	892.2	743.3	647.2	623.6	747.9	610.3	606.8	594.4
1M	0.0	1402.5	1288.4	1154.5	1117.8	1202.5	994.4	956.4	1003.6	847.5
1.5M	0.0	3344.3	2308.8	1384.5	1866.1	1288.9	1359.5	1399.2	1283.2	1324.2
2M	0.0	3955.3	4091.9	1562.6	1688.5	1678.2	1622.7	1573.5	1769.1	1469.6
2.5M	0.0	6840.5	3564.6	1761.4	2466.4	1830.4	2479.8	2443.4	2032.7	1821.5
3M	0.0	7702.2	4873.1	1973.4	2383.6	2230.2	2174.2	2802.4	2046.9	2526.4



Algoritmo Bakery



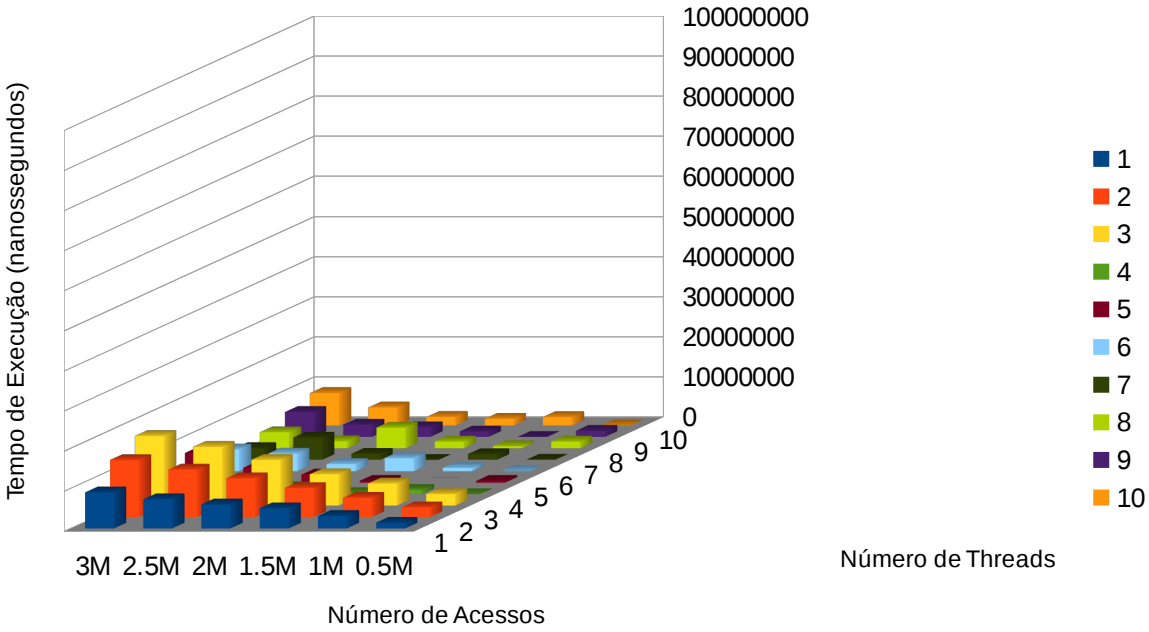
Algoritmo Gate

Tempo de Execução

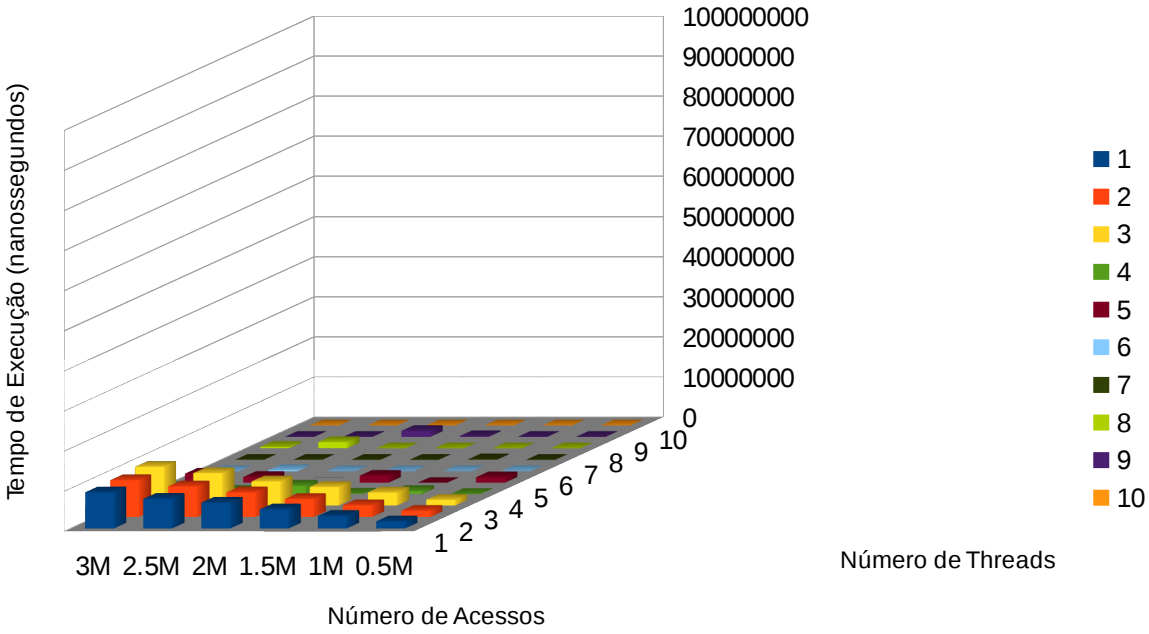
- *Bakery* possui pouco *overhead.*
- *Gate* possui pouco *overhead.*

Bakery	<number_of_threads>									
<total_time>	1	2	3	4	5	6	7	8	9	10
0.5M	1485025	2438337	2930392	806	481755	15465	26340	1715938	1567871	36454
1M	3161521	4847818	5540222	1071483	7551	884299	1451584	571717	24913	2143105
1.5M	5080514	7279641	7862232	318659	489026	3282676	18552	1732748	1395004	1698885
2M	6047054	9723864	11443496	1757510	2050941	1856482	1547898	5207668	2439711	2167403
2.5M	7326417	11866416	14679526	3232769	3760789	4434953	5566703	1845011	3084879	4519834
3M	9037597	14252357	17402021	3572668	7463321	5458398	2957217	4067151	6260333	8133965

Gate	<number_of_threads>									
<total_time>	1	2	3	4	5	6	7	8	9	10
0.5M	1812629	1575054	1465693	59328	1376857	13506	17190	21096	28447	37425
1M	3175133	2972003	3292475	815487	9578	14041	135648	26483	30725	29500
1.5M	4778504	4511807	4686187	183591	1750811	15027	19495	23141	230322	36669
2M	6472430	6179911	6060559	2141239	8398	14691	18873	23484	1516588	33430
2.5M	7425520	7669999	8130371	864811	1460378	304784	21597	1600970	34936	35817
3M	9048717	9238602	9716133	1729876	2310364	15016	20448	376451	25960	32091



Algoritmo Bakery



Algoritmo Gate

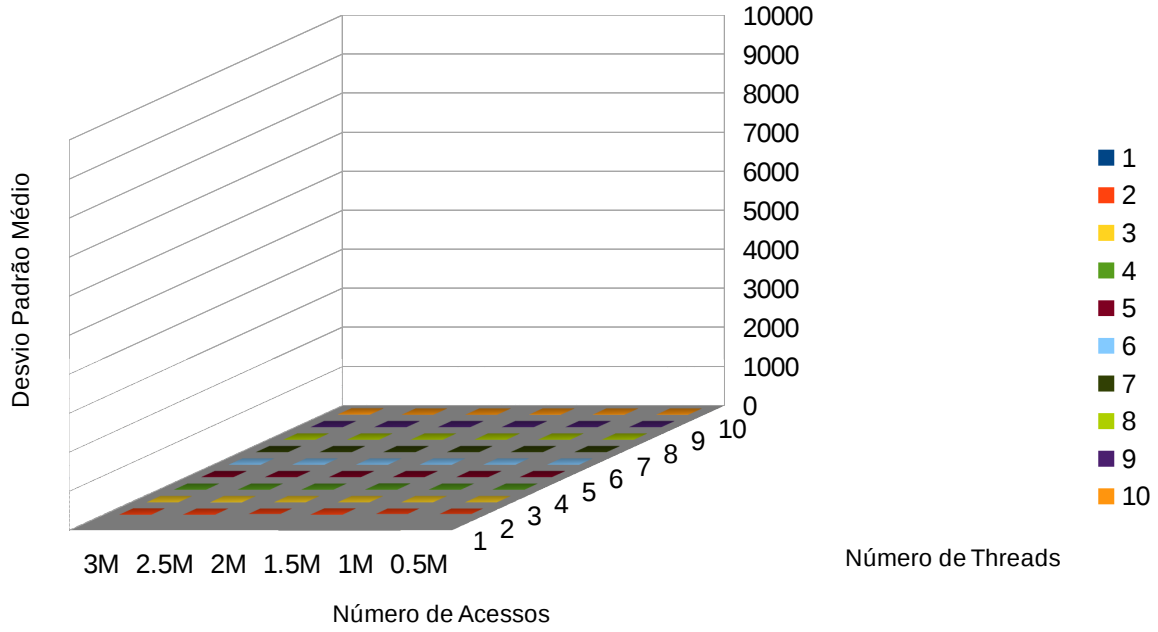
Escalonador RR (round-robin)

Desvio Padrão Médio

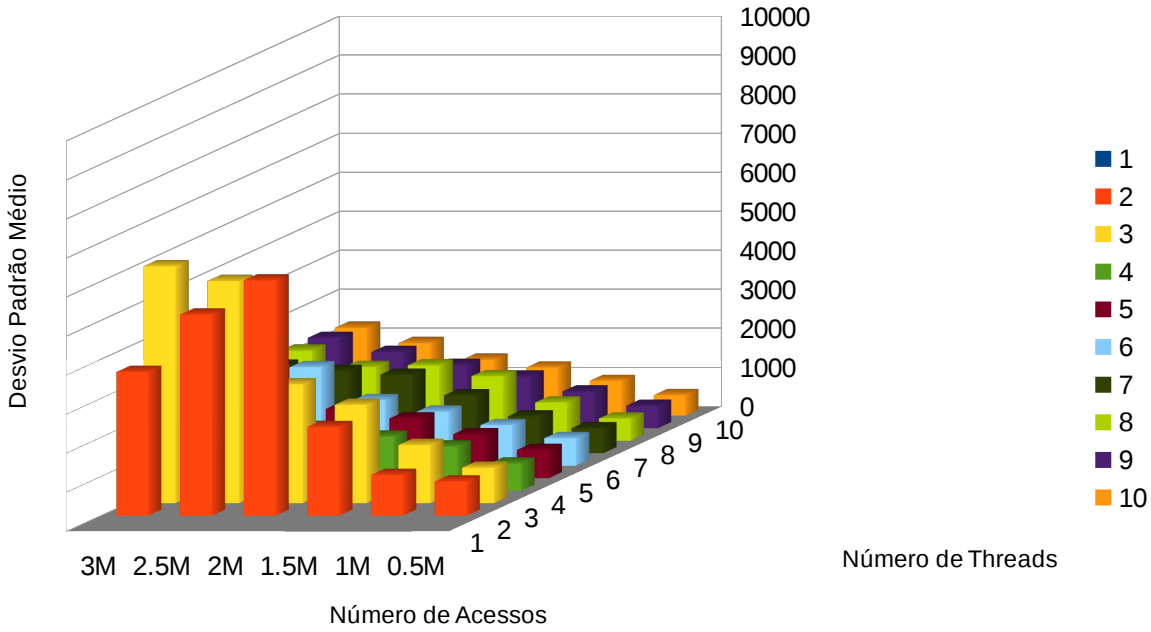
- *Bakery* praticamente não possui *starvation.*
- *Gate* possui *starvation*, que cresce em função do <total_time> é absurdamente grande com dois a três <number_of_threads>.

Bakery	<number_of_threads>									
<total_time>	1	2	3	4	5	6	7	8	9	10
0.5M	0.0	1.0	1.6	1.8	3.3	3.1	3.7	3.4	3.0	3.8
1M	0.0	1.5	1.9	2.1	2.8	3.6	3.5	2.7	2.8	3.7
1.5M	0.0	4.3	1.8	4.0	3.1	4.2	3.7	3.5	3.4	3.6
2M	0.0	1.4	1.3	3.1	2.6	3.5	3.0	2.7	3.0	2.8
2.5M	0.0	3.3	2.7	3.2	4.6	3.8	3.4	2.9	2.6	2.4
3M	0.0	4.5	1.4	3.1	3.3	3.6	3.3	3.1	2.6	11.9

Gate	<number_of_threads>									
<total_time>	1	2	3	4	5	6	7	8	9	10
0.5M	0.0	879.9	914.1	701.3	728.3	710.3	642.0	572.3	578.7	531.8
1M	0.0	1048.7	1496.1	1138.6	1131.7	1043.1	947.4	982.8	946.3	900.4
1.5M	0.0	2277.0	2523.8	1391.4	1536.6	1388.8	1488.4	1657.0	1330.3	1241.1
2M	0.0	6032.6	3049.4	1640.2	1778.2	1684.6	2015.0	1934.6	1623.3	1440.0
2.5M	0.0	5155.3	5694.7	2101.8	2160.7	2526.4	2110.5	1892.5	1957.6	1864.8
3M	0.0	3686.9	6072.9	2270.1	3014.7	2178.9	2247.4	2311.8	2317.8	2256.8



Algoritmo Bakery



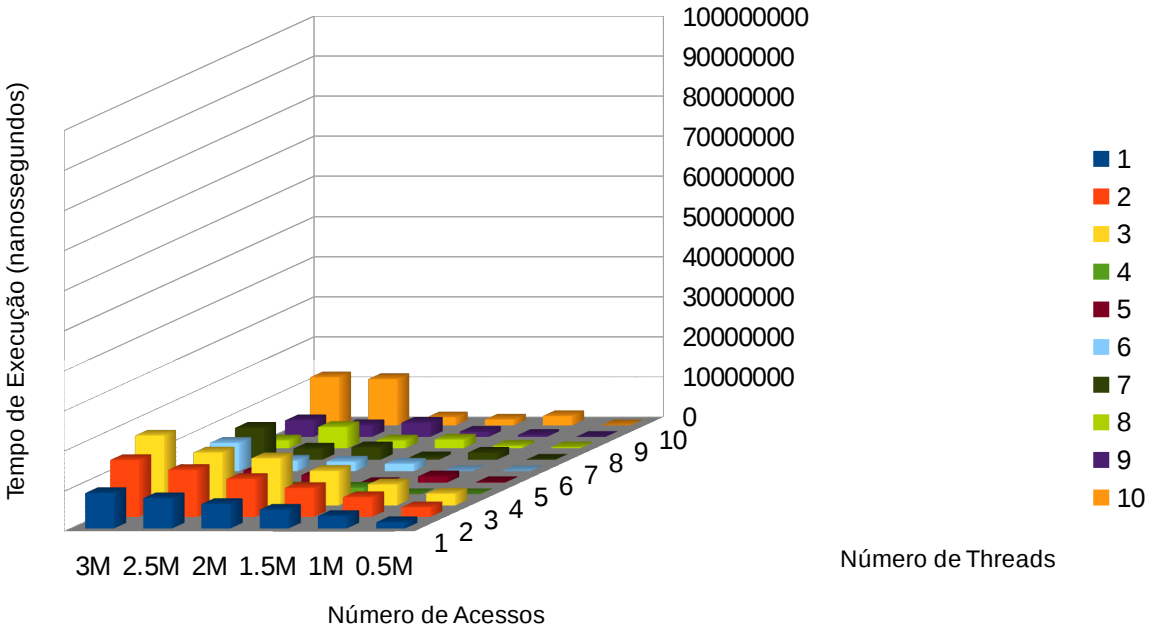
Algoritmo Gate

Tempo de Execução

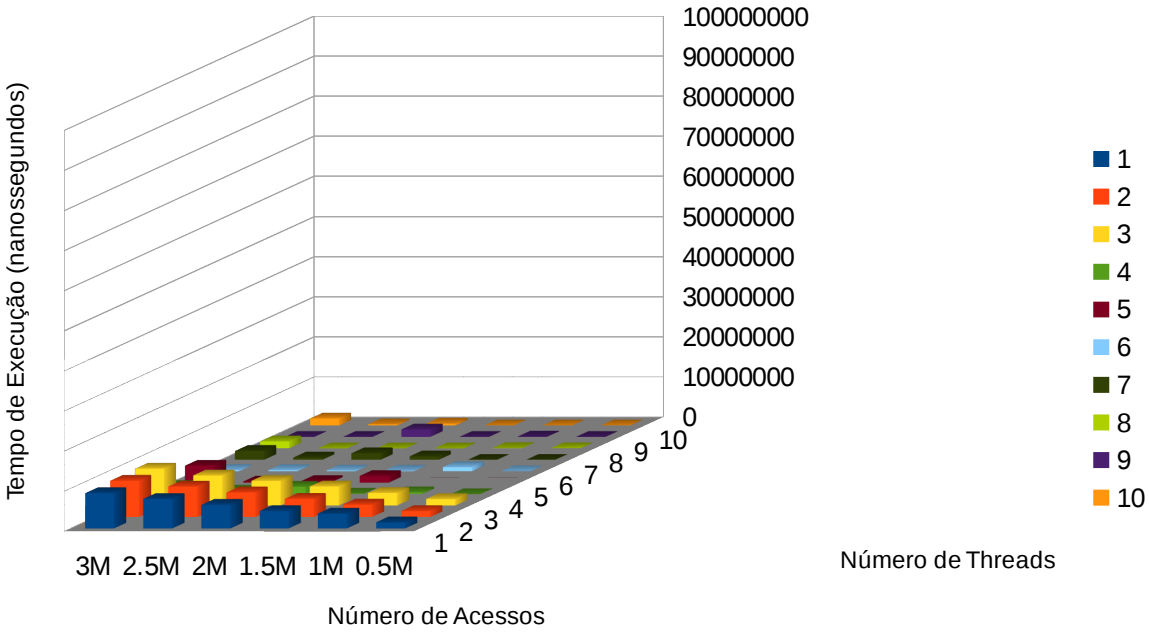
- *Bakery* possui pouco *overhead.*
- *Gate* possui pouco *overhead.*

Bakery	<number_of_threads>									
<total_time>	1	2	3	4	5	6	7	8	9	10
0.5M	1595967	2450496	3027417	760	190713	14254	33400	306218	23897	45758
1M	3133445	4971251	5327262	768	1597407	16317	1823063	674032	532674	2420698
1.5M	4722485	7271429	8679662	1621144	9796	1880897	665844	2264096	1326606	1519973
2M	6130294	9500256	11839674	1684297	1950314	2440468	3171456	2029870	3573888	2065629
2.5M	7569663	11804636	13273306	1604103	3050952	2678485	2814400	5381290	2931960	11606381
3M	8850927	14243809	17524407	3408573	2914044	7130146	8045318	2033145	4189808	12047330

Gate	<number_of_threads>									
<total_time>	1	2	3	4	5	6	7	8	9	10
0.5M	1562135	1544969	1675273	792	8090	13788	22615	86949	26881	36405
1M	3739478	3141353	3187846	465743	7293	1070076	20718	136552	144405	40025
1.5M	4388036	4553644	4801039	174565	1713101	189659	895609	29254	31875	41995
2M	5945185	6179253	6210221	1890522	472127	248237	1663596	28675	1885598	507936
2.5M	7402046	7627072	7564352	882933	312934	305121	600591	28987	30578	331452
3M	8854356	9101145	9243047	2079588	4294221	357745	2257095	1763441	27710	1783402



Algoritmo Bakery



Algoritmo Gate

Conclusões

- Com o escalonador OTHER, *Bakery* possui pouco *starvation* e muito *overhead* enquanto *Gate* é o inverso.
- O comportamento dos algoritmos utilizando os escalonadores FIFO e RR são semelhantes. Ambos os algoritmos geram pouco *overhead. Backery* não possui *starvation*, enquanto *Gate* possui bastante. Portanto ao utilizar esses escalonadores, o algoritmo *Bakery* parece ser mais recomendado.
- Do ponto de vista de corretude, ao remover a primitiva __sync_syncronize do código não gera a resposta correta, pois a atribuição das variáveis não é atômica.