Sistemas Operacionais - Segundo EP - Prof. Norton Trevisan Roman

Alexandre Farias	Anderson Kistner	Jordana Carnicelli
9761826	9761739	9894407

RELATÓRIO DE AVALIAÇÃO DO SISTEMA

1. INTRODUÇÃO

Foram implementados, neste sistema, dois algoritmos para as execuções realizadas: o *Readers/Writers* (RW) e o *Busy Wait* (BW). Para ambos, a quantidade de leitores tem início em 100 e de escritores em 0. A cada execução, um leitor é decrementado e um escritor incrementado de forma que a última execução seja na proporção de 0 leitores e 100 escritores. O tempo de cada algoritmo é calculado por meio da função *currentTimeMillis()* do Java.

São apresentados, abaixo, para as duas soluções, os resultados em milissegundos de todas as proporções executadas.

2. RESULTADOS

Proporção (R, W)	RW (ms)	BW (ms)
(100, 0)	34	213
(99, 1)	28	212
(98, 2)	29	212
(97, 3)	28	213
(96, 4)	32	213
(95, 5)	33	212
(94, 6)	35	213
(93, 7)	36	211
(92, 8)	41	208
(91, 9)	43	210
(90, 10)	46	211
(89, 11)	46	209
(88, 12)	50	213
(87, 13)	52	213
(86, 14)	50	209
(85, 15)	54	212
(84, 16)	51	210
(83, 17)	54	214

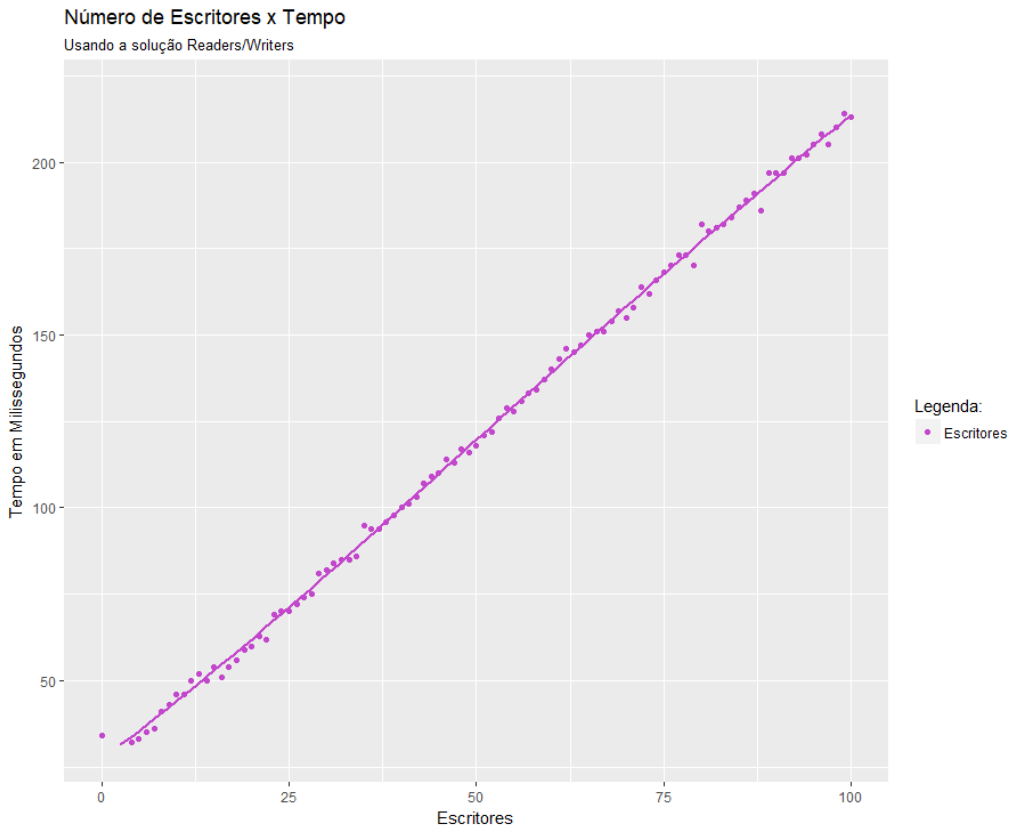
(82, 18)	56	212
(81, 19)	59	210
(80, 20)	60	212
(79, 21)	63	211
(78, 22)	62	208
(77, 23)	69	212
(76, 24)	70	213
(75, 25)	70	212
(74, 26)	72	211
(73, 27)	74	213
(72, 28)	75	211
(71, 29)	81	209
(70, 30)	82	212
(69, 31)	84	211
(68, 32)	85	213
(67, 33)	85	210
(66, 34)	86	208
(65, 35)	95	210
(64, 36)	94	211
(63, 37)	94	215
(62, 38)	96	211
(61, 39)	98	212
(60, 40)	100	213
(59, 41)	101	211
(58, 42)	103	214
(57, 43)	107	208
(56, 44)	109	213
(55, 45)	110	217
(54, 46)	114	214
(53, 47)	113	214
(52, 48)	117	208
(51, 49)	116	210
(50, 50)	118	214
(49, 51)	121	211
(48, 52)	122	209
(47, 53)	126	212
(46, 54)	129	210
(45, 55)	128	211
(44, 56)	131	213
(43, 57)	133	208
(42, 58)	134	207
(41, 59)	137	211
(40, 60)	140	210

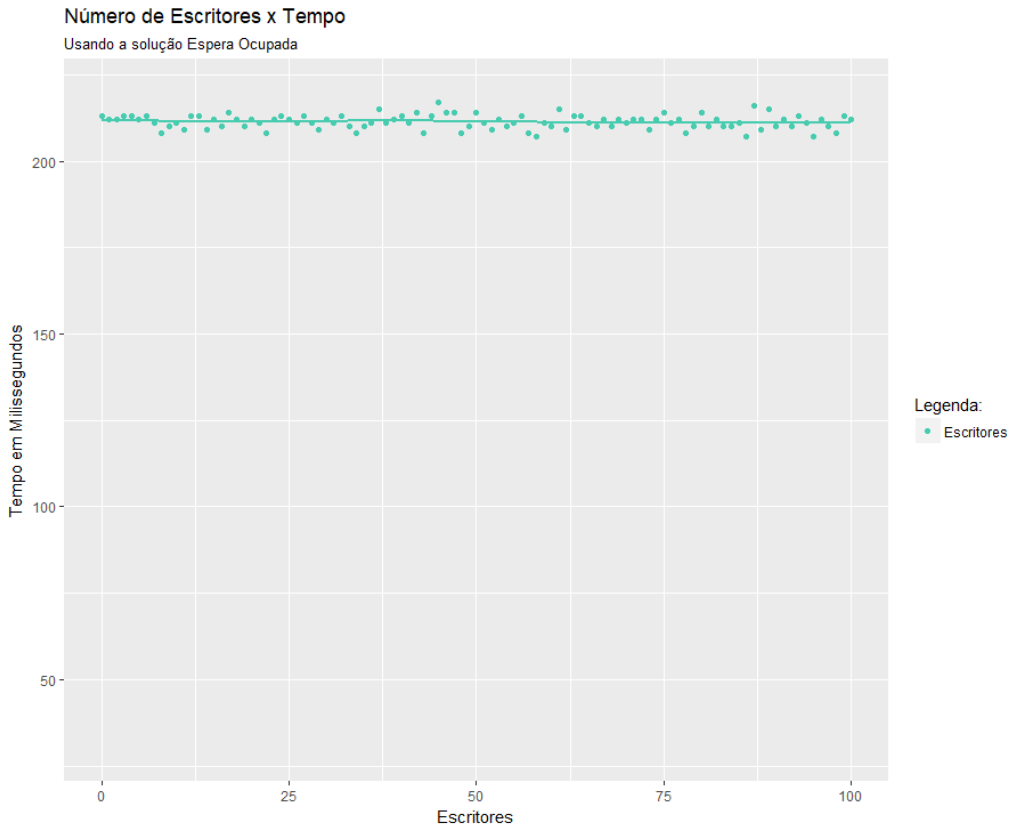
(39, 61)	143	215
(38, 62)	146	209
(37, 63)	145	213
(36, 64)	147	213
(35, 65)	150	211
(34, 66)	151	210
(33, 67)	151	212
(32, 68)	154	210
(31, 69)	157	212
(30, 70)	155	211
(29, 71)	158	212
(28, 72)	164	212
(27, 73)	162	209
(26, 74)	166	212
(25, 75)	168	214
(24, 76)	170	211
(23, 77)	173	212
(22, 78)	173	208
(21, 79)	170	210
(20, 80)	182	214
(19, 81)	180	210
(18, 82)	181	212
(17, 83)	182	210
(16, 84)	184	210
(15, 85)	187	211
(14, 86)	189	207
(13, 87)	191	216
(12, 88)	186	209
(11, 89)	197	215
(10, 90)	197	210
(9, 91)	197	212
(8, 92)	201	210
(7, 93)	201	213
(6, 94)	202	211
(5, 95)	205	207
(4, 96)	208	212
(3, 97)	205	210
(2, 98)	210	208
(1, 99)	214	213
(0, 100)	213	212

3. DISCUSSÃO

Como pode-se observar na tabela acima, quanto maior o número de escritores utilizados na proporção, para a solução *Readers/Writers*, maior o tempo médio de execução do programa. Isso ocorre pelo fato de que, nesse algoritmo, não há limite para a quantidade de leitores na região crítica. No entanto, quando há um escritor nela, nenhum outro processo pode acessá-la.

Dessa forma, na medida em que a proporção de escritores se aproxima de 100, o comportamento do programa se aproxima de uma Espera Ocupada. Justifica-se o observado, pois o número de vezes que a região crítica conterá apenas um processo aumenta. Seguem os gráficos com a visualização dos resultados.

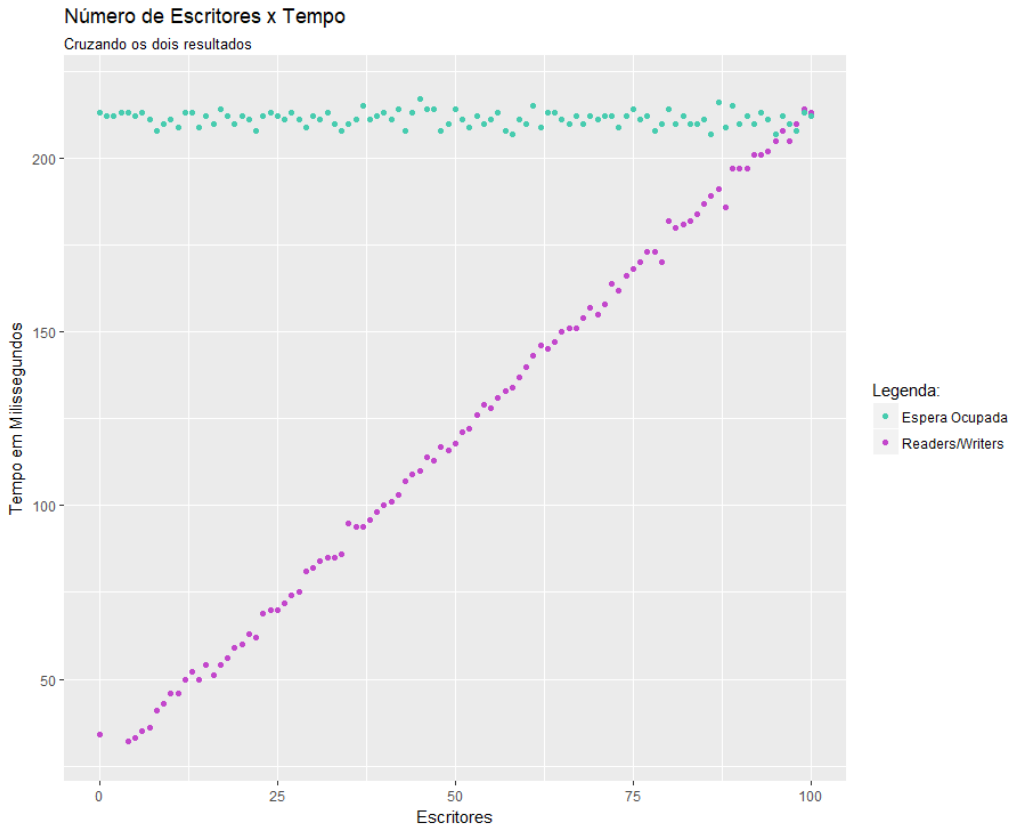




Visto que o resultado para o algoritmo *Readers/Writers*, é de 34 milissegundos para 100 leitores e 0 escritores e 213 milissegundos para 0 leitores e 100 escritores, tem-se um comportamento linear ascendente no primeiro gráfico.

Já no algoritmo Espera Ocupada, o tempo de execução se mantém constante ao se alterar as proporções de leitores e escritores. Isso é esperado uma vez que, neste último algoritmo, cada processo, independentemente da sua natureza (leitor ou escritor), deve esperar a região crítica esvaziar para ocupa-la. O gráfico deste, então, tem um comportamento constante.

O uso do algoritmo *Readers/Writers* só não é vantajoso quando há apenas escritores acessando a região crítica. Nesse caso, ele tem o mesmo desempenho que uma Espera Ocupada. O gráfico a seguir mostra isso.



Em suma, com exceção desse último caso especificado, a solução *Readers/Writers* tem tempo médio de execução menor, se mostrando, logo, mais eficiente que a solução alternativa.

REFERÊNCIA

TANENBAUM, A. S. **Sistemas Operacionais Modernos**. 2. ed. São Paulo: Pearson, 2003. 672 p.