**UNIVERSIDADE FEDERAL DE SANTA CATARINA – UFSC- CTC**

**DEPARTAMENTO DE INFORMÁTICA E ESTATÍSTICA**

**PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO**

**PROJETO E ANÁLISE DE ALGORITMO**

**Prof. Alexandre Gonçalves Silva**

**Aluno: Osmar de Oliveira Braz Junior**

# 

# Questão 1

1. Para comparação de problemas com custos conhecidos em função de n, pede-se:

(a) Suponha que estamos comparando implementações dos algoritmos de ordenação *insertion sort* e merge sort em uma mesma máquina. Para entradas de tamanho **n**, o *insertion sort* executa com custo total de 8*n*2, enquanto o *merge sort* executa com custo 64n lg n. Para quais valores de n o *insertion sort* supera o *merge sort*?

R.

Para calcularmos essa questão, devemos escrevê-la em forma de inequações e calcular seu resultado. Como queremos saber os valores de n para os quais a ordenação de inserção supera a ordenação por intercalação temos:

*insertion sort* < *merge sort*

8n2 < 64n log n

n < 8 log n

n/8 < log n

2n/8 < n

2n/8 – n < 0

2 <= n <= 43

ou

Elaborando uma tabela com três colunas, n, tempo para o algoritmo por inserção e o tempo para o algoritmo por intercalação tem-se a tabela abaixo. Onde foram simulados os valores de 1 a 50 para n.

|  |  |  |
| --- | --- | --- |
| **Valor de n** | **Tempo**  **Inserção (8n^2)** | **Tempo**  **Intercalação (64n Ig n)** |
| 1 | 8 | 0,00 |
| 2 | 32 | 128,00 |
| 3 | 72 | 304,31 |
| 4 | 128 | 512,00 |
| 5 | 200 | 743,02 |
| 6 | 288 | 992,63 |
| 7 | 392 | 1257,70 |
| 8 | 512 | 1536,00 |
| 9 | 648 | 1825,88 |
| 10 | 800 | 2126,03 |
| 11 | 968 | 2435,44 |
| 12 | 1152 | 2753,25 |
| 13 | 1352 | 3078,77 |
| 14 | 1568 | 3411,39 |
| 15 | 1800 | 3750,61 |
| 16 | 2048 | 4096,00 |
| 17 | 2312 | 4447,16 |
| 18 | 2592 | 4803,75 |
| 19 | 2888 | 5165,48 |
| 20 | 3200 | 5532,07 |
| 21 | 3528 | 5903,27 |
| 22 | 3872 | 6278,88 |
| 23 | 4232 | 6658,68 |
| 24 | 4608 | 7042,50 |
| 25 | 5000 | 7430,17 |
| 26 | 5408 | 7821,53 |
| 27 | 5832 | 8216,45 |
| 28 | 6272 | 8614,78 |
| 29 | 6728 | 9016,41 |
| 30 | 7200 | 9421,23 |
| 31 | 7688 | 9829,13 |
| 32 | 8192 | 10240,00 |
| 33 | 8712 | 10653,76 |
| 34 | 9248 | 11070,32 |
| 35 | 9800 | 11489,59 |
| 36 | 10368 | 11911,51 |
| 37 | 10952 | 12335,99 |
| 38 | 11552 | 12762,96 |
| 39 | 12168 | 13192,36 |
| 40 | 12800 | 13624,14 |
| 41 | 13448 | 14058,22 |
| 42 | 14112 | 14494,55 |
| **43** | **14792** | **14933,08** |
| 44 | 15488 | 15373,76 |
| 45 | 16200 | 15816,54 |
| 46 | 16928 | 16261,37 |
| 47 | 17672 | 16708,20 |
| 48 | 18432 | 17157,00 |
| 49 | 19208 | 17607,73 |
| 50 | 20000 | 18060,34 |

O *insertion sort* é melhor que o mergesort no intervalo de n=2 até n=43, a partir de n=44 o *merge sort* tem melhor desempenho.

(b) Qual é o menor valor de n para o qual um algoritmo, cujo tempo de execução é 100*n*2, executa mais rápido que um algoritmo cujo tempo de execução é 2n?

R.

Devemos encontrar o menor valor de n para qual a seguinte inequação é verdadeira 100n2< 2n.

Algoritmo1 < Algoritmo2

100n2< 2n.

Elaborando uma tabela com três colunas, n, tempo para o algoritmo 1 e o tempo para o algoritmo 2 tem-se a tabela abaixo. Foi simulado o intervalo de 1 a 16 para n.

|  |  |  |
| --- | --- | --- |
| **Valor de n** | **Tempo Algoritmo 1**  100n2 | **Tempo Algoritmo 1**  2n |
| 1 | 100 | 2 |
| 2 | 400 | 4 |
| 3 | 900 | 8 |
| 4 | 1600 | 16 |
| 5 | 2500 | 32 |
| 6 | 3600 | 64 |
| 7 | 4900 | 128 |
| 8 | 6400 | 256 |
| 9 | 8100 | 512 |
| 10 | 10000 | 1024 |
| 11 | 12100 | 2048 |
| 12 | 14400 | 4096 |
| 13 | 16900 | 8192 |
| 14 | 19600 | 16384 |
| **15** | **22500** | **32768** |
| 16 | 25600 | 65536 |

O algoritmo 2 com o custo 100n2,para entradas(n) de 1 até =14 tem pior desempenho, contudo a partir de n=15 o algoritmo 1 com custo 2n tem melhor desempenho. Por isso, concluímos que 15 é o menor valor de n para qual o algoritmo, cujo tempo de execução é 100n2, executa mais rápido que um algoritmo cujo temo de execução é 2n.