XIV Simpósio de Ciência,
Inovação & Tecnologia – IF Sudeste MG - Campus Rio Pomba
Ciência para redução das desigualdades
23 e 24 de novembro de 2022.

# ANALISANDO O TEMPO DE EXECUÇÃO DO RADIXSORT LSD DECIMAL AO ORDENAR MÚLTIPLOS DÍGITOS A CADA ITERAÇÃO

**VÍCTOR ROMULO CAPOBIANCO SÁ[1]; JOSÉ RUI CASTRO DE SOUSA[2]**

Algoritmos de ordenação existem para que dados possam ser ordenados o mais rápido possível em computadores. No entanto, a eficiência de cada método varia de acordo com o contexto onde será utilizado. Devido às demandas dobig data, a importância de algoritmos adequados para entradas volumosas, de objetos complexos, cresce a cada dia. É desejável que esses métodos apresentem um baixo consumo de memória, para permitir a ordenação interna das informações. Um algoritmo adequado para esse cenário é oradixsort LSDpara listas, desde que ordene-se múltiplos dígitos por iteração. Isso porque seu uso de memória é relativamente baixo e, por ser um algoritmo dehashing, apresenta grande flexibilidade nos tipos de dados que pode ordenar. Implementa-se oradixsort, obottom-up mergesorte oquicksorttrês partições em C++, na base decimal. São geradas mais de trezentos e cinquenta entradas, de cinco tipos distintos e quantidades de dígitos do maior número variando de um a vinte. É coletado o tempo de execução doradixsort, com quantidade de dígitos ordenados simultaneamente variando de um a oito. Também coleta-se dados dos outros algoritmos implementados e dostd::list::sort, método de ordenação de listas padrão do C++. Utilizar múltiplos dígitos simultaneamente melhora o desempenho doradixsortde duas a vinte vezes, dependendo do caso e do tamanho da entrada. Isso se traduz em um algoritmo várias vezes mais rápido do que os procedimentos por comparação selecionados. E a diferença cresce de acordo com a quantidade de elementos da entrada com, por exemplo, omergesortsendo 1,25 vezes mais lento na de tamanho106preenchida com inteiros de vinte dígitos? 1,57 vezes na que possui107registros e 2,15 vezes quando se tem108elementos. A única exceção ocorre ao se comparar oradixsortcom oquicksortno seu melhor caso (todas as chaves são iguais). No entanto, em todos os outros casos o algoritmo se mostrou ser a melhor escolha para entradas de tamanhos iguais e, por indução, superiores aos testados. Sendo muito eficiente para bases de dados volumosas é, portanto, compatível com as demandas dobig data.

**PALAVRAS CHAVE:** Ordenação interna, Radixsort, Multi-dígito.

**Apoio(s):** IF Sudeste MG

[1]Ex-Aluno - IFSudesteMG/Campus Rio Pomba - victorcapobianco01@yahoo.com.br
[2] Orientador - IFSudesteMG - jose.castro@ifsudestemg.edu.br