UNIVERSIDADE DE SÃO PAULO – USP
INSTITUTO DE CIÊNCIAS MATEMÁTICAS E DE COMPUTAÇÃO
DEPARTAMENTO DE CIÊNCIAS DA COMPUTAÇÃO

Projeto 03: Divisão e Conquista
SCC0218: Algoritmos Avançados e Aplicações

Professor
Gustavo Batista

Alunos:

Gil Barbosa Reis	NUSPº 8532248
Leonardo Sampaio Ferraz Ribeiro	NUSPº 8532300

1. Introdução

O objetivo deste projeto é a implementação de um algoritmo eficiente de divisão e conquista para encontrar os pontos que definem a fronteira de Pareto ótimo com dois objetivos a serem minimizados.

Para alcançar este objetivo foi utilizada a linguagem C++ (C11) na implementação e os slides de aula e conhecimentos prévios para o projeto do algoritmo.

2. Projeto

A implementação foi feita com a representação do ‘gráfico’ da função que queremos encontrar a fronteira de pareto em uma classe (‘Grafico’) onde os pontos foram representados por um vetor de pares (pair<float, float> da STL) chamados de ‘pontos’. A ‘ordenação’ e posterior identificação dos pontos de fronteira foram feitas por métodos recursivos.

Inicialmente os pontos são ordenados em relação ao ‘x’ usando o método ‘div_conq_sort’, muito parecido com o conhecido ‘merge sort’, o vetor de pontos é dividido em duas partes sucessivamente e estas partes são unidas de forma ordenada.

Com o vetor ordenado pela coordenada ‘x’ os pontos de fronteira do pareto são finalmente encontrados por um método semelhante ‘div_conq_pareto’, que aplica a mesma ideia de divisão e conquista para encontrar os pontos não-dominados por quaisquer outros pontos.

3. Complexidade

A fórmula de recorrência de cada parte do algoritmo é:

$$T(n) \begin{cases} 0 & \text{se } n = 1 \\ T(n/2) + T(n/2) + n & \text{caso contrário} \end{cases}$$

Que pode ser facilmente (como mostrado em sala) resolvida para:

$$O(n \log n)$$

Ambas as partes possuem esta mesma complexidade, nos levando portanto a:

$$2\,O(n \log n) = O(n \log n)$$

A complexidade de espaço é mais simples e semelhante àquela de outros algoritmos de divisão e conquista; para a implementação de cada parte é necessário apenas um vetor auxiliar de tamanho igual ao inicial. Completamente linear, ou seja:

$$O(n)$$