PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO GRANDE DO SUL
ESCOLA POLITÉCNICA
BACHARELADO EM ENGENHARIA DE SOFTWARE
DISCIPLINA DE ALGORITMOS AVANÇADOS – PROF. DIEGO NOBLE

**Relatório do Trabalho 1 da Disciplina**

*Gabriel Ferreira Kurtz*
*gabriel.kurtz@acad.pucrs.br*

Porto Alegre, 22 de Abril de 2018

Para o primeiro trabalho da disciplina de Algoritmos Avançados, deve ser desenvolvido um algoritmo guloso capaz de solucionar o problema do Escalonamento de Intervalos. Entende-se por algoritmo guloso um programa que procura maximizar o aproveitamento de uma determinada parte do problema, sem considerar necessariamente o problema inteiro, possuindo portanto uma visão míope (incompleta).

Em determinados casos, a abordagem do algoritmo guloso pode obter uma solução satisfatória (ótima) para o problema, enquanto em outros casos a solução pode ser errada. Por isso é importante definir corretamente o critério que será maximizado.

O problema do escalonamento prevê que hajam diferentes Intervalos, cada um com um tempo inicial (Ts) e um tempo final (Ts) que dividem um mesmo recurso. O objetivo é encaixar o maior número possível de intervalos dentro da disponibilidade de recursos, sabendo que é impossível encaixar dois Intervalos que estejam em conflito (se sobrepõem em algum momento).

Um exemplo de estratégia gulosa seria encaixar sempre o intervalo que possui o menor tempo de início após o atual, de modo a minimizar o tempo ocioso dos recursos. Embora possa parecer que esta é uma boa estratégia, ficou demonstrado no nosso estudo que é realmente uma visão falha. Os intervalos encaixados podem ser muito grandes e, embora desperdice-se pouco tempo dos recursos com esta estratégia, como o objetivo é

encaixar o maior número de intervalos no conjunto da solução, realmente não se obtem o resultado ideal com esta estratégia. Este exemplo ficou implementado no código como a estratégia Sub-ótima.

Uma alternativa melhor (e que otimiza o problema) é encaixar os intervalos pelo tempo final mais próximo. Desta forma, mesmo que o algoritmo nao enxergue em nenhum momento o problema inteiro mas sempre o passo seguinte, podemos garantir que ao encaixar sempre o tempo final mais próximo, o número máximo de intervalos possíveis será encaixado. Chamamos esta solução no código de Ótima.

O programa gera dez casos de exemplos, cada um com 10^6 intervalos onde o Tf máximo é 1000. Executa uma comparação entre nossa estratégia gulosa sub-ótima e a ótima para cada caso e também a média de todos os dez exemplos.

Em uma instância de execução do programa demostrando as médias de intervalos encaixados em cada escalonamento, tivemos os seguintes resultados:

- Estratégia Gulosa Sub-ótima: 6.5
- Estratégia Gulosa Ótima: 464.9

O que comprova a nossa explanação – realmente a estratégia ótima produz resultados muito mais satisfatórios simplesmente pelo fato de aplicar um critério mais correto para maximizar o problema.

O código em Java está sendo enviado em anexo e também pode ser verificado no GitHub:

https://github.com/gabrielkurtz/t1algoritmos