Trabalho Prático

Estruturas de Dados Avançadas (EDA)

ESI/EIM/EEC

EST-IPCA

Barcelos

4 de março de 2022

Motivação

Este trabalho prático de realização individual da Unidade Curricular (UC) *Estruturas de Dados Avançadas* (EDA) integrada no 2º semestre do 1º ano das licenciaturas ESI/EIM/EEC visa o reforço e a aplicação dos conhecimentos adquiridos ao longo do semestre.

Objetivo

Com este trabalho prático pretende-se sedimentar os conhecimentos relativos a definição e manipulação de estruturas de dados dinâmicas na linguagem de programação C. A essência deste trabalho reside no desenvolvimento de uma solução digital para o problema de escalonamento denominado *Flexible Job Shop Problem* (FJSSP). A solução a implementar deverá permitir gerar uma proposta de escalonamento para a produção de um produto envolvendo várias operações e a utilização de várias máquinas, minimizando o tempo as unidades de tempo necessário na sua produção (*makespan*). Um FJSSP pode ser formulado da seguinte forma:

1. Existe um conjunto finito de *n jobs* que têm de ser processados por um conjunto finito de *m* máquinas;
2. O conjunto de *m* máquinas é identificado por: $M = \{M_1, M_2, ..., M_n\}$;
3. Um *job* é constituído por uma sequência de *ni* operações como: $(O_{i,1}, O_{i,2}, ..., O_{i,ni})$.
4. Cada operação deve ser executada para completar o *job*. A execução de cada operação j de um *job* i ($O_{i,j}$) requer uma máquina de um conjunto de máquinas $M_{i,j}$. O tempo de uma operação $O_{i,j}$ realizada na máquina em $M_{i,j}$ é $p_{i,j,k}$. As seguintes suposições são consideradas num problema FJSSP:
    a. Todas as máquinas estão disponíveis no instante $t = 0$.

b. Todos os *jobs* estão disponíveis no tempo $t = 0$.
c. Cada operação pode ser realizada por apenas uma máquina de cada vez.
d. Não há restrições de precedência entre as operações de diferentes *jobs*; portanto os *jobs* são independentes.
e. Uma operação, uma vez iniciada, não pode ser interrompida.
f. O tempo de transporte de *jobs* entre as máquinas e tempo para configurar a máquina para realizar uma determinada operação estão incluídos no tempo de processamento.

Um *job* é um processo de produção de uma instância de um produto específico que é definido por um *process plan*. Uma operação é uma tarefa individual que é alocada a uma máquina e está associada a um *job* específico. Uma máquina é um recurso capaz de executar operações, e por fim um *process plan* é uma lista ordenada de operações necessárias para concluir um *job*.

A Tabela 1 incorpora os *process plan* com dimensão 8x7 para a produção de um produto, envolvendo a realização de 8 *jobs* (com um máximo de 7 operações) distribuídos por 8 máquinas. Cada linha da Tabela 1 apresenta a descrição da sequência das operações necessárias para cada tipo de *job* (um *job* representa a produção de um produto, por exemplo *pr1,2*). No caso do tipo de *job pr1,2* (primeira linha do *process plan*), este requer a execução de 4 operações numa predeterminada ordem, i.e. 01, 02, 03 e 04. Para cada operação, o *process plan* indica quais são as máquinas onde a mesma pode ser realizada, bem como a respetiva quantidade de unidades de tempo necessária para a sua realização. A título de exemplo, a primeira operação (01) pode ser realizada na máquina 1 com uma duração de 4 unidades de tempo ou na máquina 3 com uma duração de 5 unidades de tempo. Cada *job* de um *process plan* é composto por *n* operações que podem ser encadeadas com outras operações de outros *jobs*, mas dentro do mesmo *job* necessitam ser executadas pela sua ordem, isto é, num *job* que tenha três operações, a operação 3 não pode ser iniciada sem que a operação 2 esteja finalizada, e esta por sua vez também não pode ser iniciada sem que a operação 1 esteja finalizada. O cálculo da distribuição das operações pelas máquinas terá de se basear na capacidade das máquinas poderem executar essa operação, e na ocupação destas.

Ler isso com calma

Process Plan	Operation						
	O 1	**O 2**	**O 3**	**O 4**	**O 5**	**O 6**	**O 7**
$pr_{1,2}$	(1,3) [4,5]	(2,4) [4,5]	(3,5) [5,6]	(4,5,6,7,8) [5,5,4,5,9]			
$pr_{2,2}$	(1,3,5) [1,5,7]	(4,8) [5,4]	(4,6) [1,6]	(4,7,8) [4,4,7]	(4,6) [1,2]	(1,6,8) [5,6,4]	(4) [4]
$pr_{3,3}$	(2,3,8) [7,6,8]	(4,8) [7,7]	(3,5,7) [7,8,7]	(4,6) [7,8]	(1,2) [1,4]		
$pr_{4,2}$	(1,3,5) [4,3,7]	(2,8) [4,4]	(3,4,6,7) [4,5,6,7]	(5,6,8) [3,5,5]			
$pr_{5,1}$	(1) [3]	(2,4) [4,5]	(3,8) [4,4]	(5,6,8) [3,3,3]	(4,6) [5,4]		
$pr_{6,3}$	(1,2,3) [3,5,6]	(4,5) [7,8]	(3,6) [9,8]				
$pr_{7,2}$	(3,5,6) [4,5,4]	(4,7,8) [4,6,4]	(1,3,4,5) [3,3,4,5]	(4,6,8) [4,6,5]	(1,3) [3,3]		
$pr_{8,1}$	(1,2,6) [3,4,4]	(4,5,8) [6,5,4]	(3,7) [4,5]	(4,6) [4,6]	(7,8) [1,2]		

Tabela 1. Process plan para um problema de escalonamento com dimensão 8x7 e 8 máquinas

Fase 1

1. Definição de uma estrutura de dados dinâmica para a representação de um *job* com um conjunto finito de *n* operações;
2. Armazenamento/leitura de ficheiro de texto com representação de um *job*;
3. Inserção de uma nova operação;
4. Remoção de uma determinada operação;
5. Alteração de uma determinada operação*;*
6. Determinação da quantidade mínima de unidades de tempo necessárias para completar o *job* e listagem das respetivas operações;
7. Determinação da quantidade máxima de unidades de tempo necessárias para completar o job e listagem das respetivas operações;

8. Determinação da quantidade média de unidades de tempo necessárias para completar uma operação, considerando todas as alternativas possíveis;

Fase 2

1. Definição de uma estrutura de dados dinâmica para representação de um conjunto finito de *m jobs* associando a cada job um determinando conjunto finito de operações;
2. Armazenamento/leitura de ficheiro de texto com representação de um *process plan* (considerar obrigatoriamente para efeito de teste o *process plan* da Tabela 1);
3. Inserção de um novo *job*;
4. Remoção de um *job*;
5. Inserção de uma nova operação num *job*;
6. Remoção de uma determinada operação de um *job*;
7. Edição das operações associadas a um *job;*
8. Cálculo de uma proposta de escalonamento para o problema FJSSP (obrigatoriamente limitado a um tempo máximo de processamento configurável), apresentando a distribuição das operações pelas várias máquinas, minimizando o *makespan* (unidades de tempo necessárias para a realização de todos os *jobs*). A proposta de escalonamento deverá ser exportada para um ficheiro de texto possibilitando uma interpretação intuitiva (utilizar por exemplo um formato tabular ou representação gráfica html, ou outra);
9. Representação de diferentes *process plan* (variando a quantidade de máquinas disponíveis, quantidade de *job*, e sequência de operações, etc) associando as respetivas propostas de escalonamento.

Para cada fase, o trabalho desenvolvido deverá ser acompanhado de:

- relatório do trabalho desenvolvido;
- descrição dos resultados dos testes à aplicação desenvolvida;
- documentação de código fonte (ex. DoxyGen)
- utilização de ferramentas apropriadas para controlo de versões (Git, GitHub, outras).

O relatório deverá incorporar a estrutura seguinte:

1. Capa
2. Índice
3. Introdução
4. Propósitos e Objetivos
5. Estruturas de Dados
6. Testes realizados
7. Conclusão
8. Bibliografia

Entrega

A submissão do trabalho deverá ser efetuada na plataforma *Moodle,* através do local apropriado, de acordo com as datas seguintes:

- Fase 1: até dia 1 de abril;
- Fase 2: até dia 1 de junho.

Deverá ser submetido um ficheiro ZIP com todos os conteúdos solicitados (código desenvolvido, relatório, documentação).

Defesa do Trabalho Prático

A defesa do trabalho é individual e realizar-se-á em data a divulgar posteriormente.

Critérios de Avaliação

Os critérios de avaliação incorporam os pontos seguintes:

- Qualidade do código desenvolvido (15%);
- Qualidade da solução desenvolvida (15% **Fase 1**);
- Qualidade da proposta de escalonamento obtida (15% **Fase 2**);
- Qualidade da documentação produzida (10%);
- Qualidade da Defesa do trabalho desenvolvido (60%);