

Estruturas de Dados Avançados



Flexible Job Shop Problem

Professor:

João Carlos Silva

Luís Gonzaga Martins Ferreira

Alunos:

Bruno Dantas Aurélio Coelho Dantas: a20807

Março 26, 2022

LESI

Índice

Introdução.....	4
Propósitos e Objetivos	5
Estruturas de dados	6
Testes	8
Conclusão	14
Bibliografia	15

Índice de Figuras

Figura 1 Estrutura de dados -> Operations	6
Figura 2 Estrutura de dados -> OperationsLst.....	6
Figura 3 Estrutura de dados -> SubOperation.....	7
Figura 4 Testes -> Menu	9
Figura 5 Testes -> Leitura de um ficheiro	9
Figura 6 Testes -> Escrita de um ficheiro	10
Figura 7 Testes -> Adicionar nova operação	10
Figura 8 Testes -> Adicionar novos elementos a operação caso já exista na lista	11
Figura 9 Testes -> Apagar uma operação da lista.....	11
Figura 10 Testes -> Alterar uma operação	12
Figura 11 Testes -> Tempo mínimo possível para completar uma operação	12
Figura 12 Testes -> Tempo máximo possível para completar uma operação.....	13
Figura 13 Testes -> Tempo medio possível para completar uma operação	13

Introdução

O presente relatório pretende documentar o trabalho pratico da disciplina de Estruturas de Dados Avançados.

O trabalho pratico consiste na geração de uma proposta de escalonamento para a produção de e um produto envolvendo várias operações de várias máquinas, minimizando o tempo para sua produção. Este trabalho foi devido em 2 fases, nesta primeira fase consiste na criação e manipulação da parte da Operação.

Propósitos e Objetivos

O propósito da primeira fase deste trabalho pratico, é a criação de um programa que consiga lidar com várias **Operations** que cada terá um conjunto de **Máquinas** com determinado tempo para terminar aquela **Operation**.

O propósito da segunda fase deste trabalho e complementar a primeira fase com uma estrutura que ira guardar os dados e de uma process plan em que consiste em vários jobs com varias operations que tem varias maquinas e tempos em que podem ser realizados, e na criação de um algoritmo que proponha uma tabela de escalonamento para esse process plan.

Estruturas de dados

Para elaboração deste trabalho pratico na primeira fase foram criadas 3 estruturas em que 2 delas são Queue usando listas duplamente ligadas, uma estrutura auxiliar.

- Esta estrutura (**_Operations**) e uma auxiliar a (**_OperationsLst**), tem como função de guardar o apontador para o primeiro e ultimo elemento da Lista (**_OperationsLst**)

```
typedef struct _Operations
{
    struct _OperationsLst *first, *last;
}Operations;
```

Figura 1 Estrutura de dados -> Operations

- Na estrutura (**_OperationsLst**) serão guardados todas os identificadores de operação tendo atenção que este identificadores são únicos, também guardara dois apontadores para o primeiro e ultimo elemento da estrutura (**_SubOperations**).

```
typedef struct _OperationsLst
{
    int numOperation;
    int TotalSubOperation;

    struct _SubOperations *first, *last;
    struct _OperationsLst *prev, *next;
}OperationsLst;
```

Figura 2 Estrutura de dados -> OperationsLst

- Na estrutura (**_SubOperations**) serão guardados os dados das alternativas para a realização de uma operação e serão guardados de forma crescente relativamente a tempo que demora a realização de uma operação.

```
typedef struct _SubOperations
{
    int numMachine;
    int time;
    struct _SubOperations *prev, *next;
}SubOperations;
```

Figura 3 Estrutura de dados -> SubOperation

Para complementar e realizar a segunda fase foram adicionadas mais duas estruturas em que uma é auxiliar para a criação do escalonamento.

- Na estrutura (**_ProcessPlan**) serão guardados todos o process plan que foram inseridos pelo utilizador. O tipo de lista utilizada para esta estrutura foi um hashtable de listas ligadas em que cada posição pode n elementos.

```
/**
 * @brief Stores all process plans and pointers to their corresponding operations
 *
 */
typedef struct _ProcessPlan
{
    int ProcessPlanID; /** Number: plan process identifier */
    int totalProcesses; /** Number of */
    struct _ProcessPlan *next; /** Next process plan */
    struct _Operations *first;
}ProcessPlan;
```

Figura 4 Estrutura de dados -> ProcessPlan

- Na estrutura (**MapOperations**) serão guardados os dados do escalonamento por operations

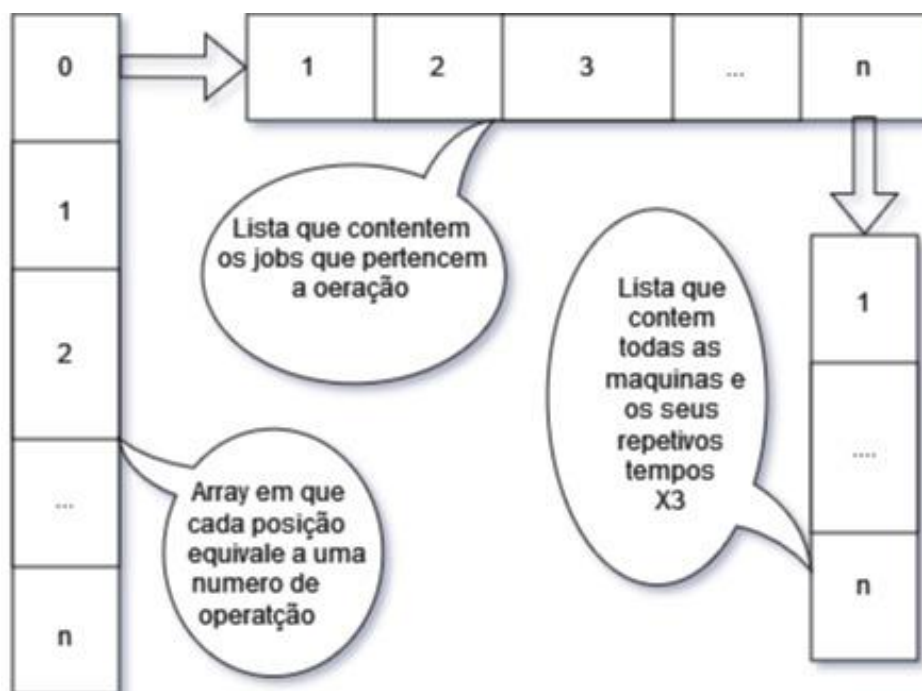
```

/**
 * @brief This structure will store the data of the calculations of a scheduling proposal.
 *
 */
typedef struct _MapOperations{
    int jobID; /** Number: Identifier of the job*/
    int OperationID; /** Number: Identifier of the operation*/
    bool tested;
    SubOperations *first; /**SubOperations: Pointer to first element of the operation*/
    SubOperations *position; /**SubOperations: Pointer to the actual element of the operation*/
    SubOperations *selected; /**SubOperations: Pointer to the selected element of the operation */
    struct _MapOperations *next; /** _MapOperations: Pointer to the next element of the list*/
} MapOp;

```

Figura 5 Estrutura de dados -> MapOperations

- JobID = Identificar do trabalho
- OperationID = identificador do número da operation
- tested = variável auxiliar para reduzir o tempo de iterações
- first = Apontador que aponta para o primeiro elemento da lista Operations
 - usada para reiniciar o position caso já tenha sido percorrido ate ao fim
- position = Apontador que aponta para a lista Operations
 - esta variável e utilizada para percorrer a lista e procurar a maquina que pode ser utilizada neste process plan
- selected = apontador para o valores da Operations selecionados pela função.
- Next = próximo job a ser testado dentro desta operation



Testes

Foram realizados dois testes a programa um com dois ficheiros diferentes.

1. Ficheiro A “job.txt”

- Um simples ficheiro que foi criado para auxiliar na criação do programa-
- Tem 3 diferentes operações cada operação tem entre 2 a 4 alternativas de escolha de tempo

```
+-----+
|                                     Menu                                     |
+-----+
| [0] -> Sair do programa                                                    |
| [1] -> Guardar dados num ficheiro txt.                                    |
| [2] -> Ler dados de um ficheiro txt com a representação de um job.        |
| [3] -> Inserção de uma nova operação.                                     |
| [4] -> Remoção de uma determinada operação.                               |
| [5] -> Editar de uma determinada operação.                               |
| [6] -> Quantidade mínima de unidades de tempo necessárias para completar o job. |
| [7] -> Quantidade máxima de unidades de tempo necessárias para completar o job. |
| [8] -> Quantidade média de unidades de tempo necessárias para completar uma operação. |
+-----+
Escolha uma opção: -> _
```

Figura 6 Testes -> Menu

```
+-----+
|                                     Menu                                     |
+-----+
| [0] -> Sair do programa                                                    |
| [1] -> Guardar dados num ficheiro txt.                                    |
| [2] -> Ler dados de um ficheiro txt com a representação de um job.        |
| [3] -> Inserção de uma nova operação.                                     |
| [4] -> Remoção de uma determinada operação.                               |
| [5] -> Editar de uma determinada operação.                               |
| [6] -> Quantidade mínima de unidades de tempo necessárias para completar o job. |
| [7] -> Quantidade máxima de unidades de tempo necessárias para completar o job. |
| [8] -> Quantidade média de unidades de tempo necessárias para completar uma operação. |
+-----+
Escolha uma opção: -> 2

+-----+
| Para o ficheiros serem encontrados                                       |
| tem que estar neste directorio [src/data/nomeficheiro.txt]              |
+-----+

Qual é o nome do ficheiro.txt: job

+-----+
| Dado lidos com sucesso. |
| Tempo medido: 0 segundos. |
+-----+

+-----+
|                                     Escolha uma opção                     |
+-----+
| [0] sair | [1] Menu |
+-----+

Escolha um opção? ->
```

Figura 7 Testes -> Leitura de um ficheiro


```
+-----+
| Escolha o número de operação que pertence alterar. |
+-----+

Operação -> 1
Operação -> 3

Qual e o elemento que pertence apagar: 1
[0]-> Máquina número [2] com uma duração de [2]
[1]-> Máquina número [1] com uma duração de [4]
[2]-> Máquina número [3] com uma duração de [10]
[3]-> Máquina número [3] com uma duração de [10]

Qual e o elemento que pertence alterar: 0

+-----+
|                               O que pertence alterar? |
+-----+
| [0] Número da máquina [1] O tempo [2] Os dois |
+-----+

Escolha um opção? -> 2

Qual é o número da Máquina: 6

Qual é a duração da operação: 1

+-----+
| A operação foi alterada com sucesso. |
+-----+

+-----+
| A operação foi alterada com sucesso. |
+-----+

+-----+
|                               Escolha uma opção |
+-----+
| [0] sair | [1] Menu |
+-----+

Escolha um opção? ->
```

Figura 12 Testes -> Alterar uma operação

```
+-----+
|                               Menu |
+-----+
| [0] -> Sair do programa |
| [1] -> Guardar dados num ficheiro txt. |
| [2] -> Ler dados de um ficheiro txt com a representação de um job. |
| [3] -> Inserção de uma nova operação. |
| [4] -> Remoção de uma determinada operação. |
| [5] -> Editar de uma determinada operação. |
| [6] -> Quantidade mínima de unidades de tempo necessárias para completar o job. |
| [7] -> Quantidade máxima de unidades de tempo necessárias para completar o job. |
| [8] -> Quantidade média de unidades de tempo necessárias para completar uma operação. |
+-----+

Escolha uma opção: -> 6

+-----+
| O tempo mínimo possível é: 5 |
+-----+

Operação -> 1 Máquina número [6] com uma duração de [1]
Operação -> 3 Máquina número [1] com uma duração de [4]

+-----+
|                               Escolha uma opção |
+-----+
| [0] sair | [1] Menu |
+-----+

Escolha um opção? -> █
```

Figura 13 Testes -> Tempo mínimo possível para completar uma operação

```

+-----+
| Menu                                     |
+-----+
| [0] -> Sair do programa                |
| [1] -> Guardar dados num ficheiro txt. |
| [2] -> Ler dados de um ficheiro txt com a representação de um job. |
| [3] -> Inserção de uma nova operação.  |
| [4] -> Remoção de uma determinada operação. |
| [5] -> Editar de uma determinada operação. |
| [6] -> Quantidade mínima de unidades de tempo necessárias para completar o job. |
| [7] -> Quantidade máxima de unidades de tempo necessárias para completar o job. |
| [8] -> Quantidade média de unidades de tempo necessárias para completar uma operação. |
+-----+
Escolha uma opção: -> 7

+-----+
| O tempo máximo possível é: 16          |
+-----+

Operação -> 1      Máquina número [3] com uma duração de [10]
Operação -> 3      Máquina número [5] com uma duração de [6]

+-----+
| Escolha uma opção                      |
+-----+
| [0] sair                               [1] Menu |
+-----+

Escolha um opção? -> _

```

Figura 14 Testes -> Tempo máximo possível para completar uma operação

```

+-----+
| Menu                                     |
+-----+
| [0] -> Sair do programa                |
| [1] -> Guardar dados num ficheiro txt. |
| [2] -> Ler dados de um ficheiro txt com a representação de um job. |
| [3] -> Inserção de uma nova operação.  |
| [4] -> Remoção de uma determinada operação. |
| [5] -> Editar de uma determinada operação. |
| [6] -> Quantidade mínima de unidades de tempo necessárias para completar o job. |
| [7] -> Quantidade máxima de unidades de tempo necessárias para completar o job. |
| [8] -> Quantidade média de unidades de tempo necessárias para completar uma operação. |
+-----+
Escolha uma opção: -> 8

+-----+
| O tempo médio possível da operação 1 é: 25/4 = 6 |
| O tempo médio possível da operação 3 é: 14/3 = 4 |
+-----+

+-----+
| Escolha uma opção                      |
+-----+
| [0] sair                               [1] Menu |
+-----+

Escolha um opção? ->

```

Figura 15 Testes -> Tempo medio possível para completar uma operação

Conclusão

Neste trabalho foi abordado as definições e manipulações de estruturas de dados dinâmicas na linguagem de programação C.

Foram cumpridos todos os objetivos propostos no enunciado.

Este trabalho foi muito importante para meu a aprofundamento de Estruturas de dados dinâmicas uma vez que elas são a base de muitas das programações mais atuais e também me permitiu aperfeiçoar a minhas competências de investigação.

Bibliografia

- MakeFile
 - <https://www.embarcados.com.br/introducao-ao-makefile/>
 - <https://makefiletutorial.com/>
- Linguagem C
 - <https://www.cplusplus.com/>
 - <https://pt.stackoverflow.com/>
 - Moodle da disciplina.