

Relatório - Simulação da propagação do vírus

Departamento de Engenharia Informática e de
Sistemas (DEIS)

Bruno Alexandre Ferreira Pinto Teixeira,
a2019100036 at isec.pt



Relatório - Simulação da propagação do vírus

DEIS

DEIS

Bruno Alexandre Ferreira Pinto Teixeira, a2019100036 at isec.pt

15 de junho de 2019

Resumo

Foi-nos proposto a implementação de um programa que simule, de forma simplificada, a propagação de um vírus entre uma população. Estes indivíduos, encontram-se num espaço constituído por locais, interligados entre si. É feita então uma simulação para perceber, mais ou menos, como é que uma infeção se espalha entre uma população e como é que as pessoas reagem, sendo tudo isto apenas uma simulação não retratando um caso real.

Conteúdo

1	Introdução	1
2	Desenvolvimento	2
2.1	Estruturas	2
2.2	Ficheiros	3
2.3	Funções	4
2.4	Funcionalidade do Programa	6
2.5	Manual do utilizador	7
3	Conclusão	15

Lista de Figuras

2.1	Estrutura Local	2
2.2	Estrutura Pessoas	3
2.3	Estrutura Iterações	3
2.4	Exemplo do ficheiro E1.bin	4
2.5	Exemplo do ficheiro pessoasA.txt	4
2.6	Menu principal	7
2.7	Menu principal - Mostrar informação das pessoas	8
2.8	Menu principal - Apresentar estatísticas	9
2.9	Menu principal - Adicionar doente	10
2.10	Menu principal - Transferir pessoa para outro local	11
2.11	Menu principal - Avançar uma iteração na simulação	12
2.12	Menu principal - Voltar atrás X iterações 1	12
2.13	Menu principal - Voltar atrás X iterações 2	13
2.14	Menu principal - Voltar atrás X iterações 3	13
2.15	Menu principal - Voltar atrás X iterações 4	13

Capítulo 1

Introdução

Este trabalho tem como objetivo criar uma aplicação em C que simule uma propagação de um vírus numa população. Existem duas fases no programa. Primeiro existe a fase da preparação, responsável por carregar todos os dados necessários a partir de ficheiros escolhidos pelo utilizador. Depois de verificada toda a informação introduzida pelo utilizador, é iniciada a fase da simulação, sendo esta a fase que vai iniciar o processo de simulação iterativo que representa a propagação do vírus ao longo de vários dias. No final da simulação, o programa apresenta um ficheiro de texto que contem o resumo dos resultados da simulação.

Capítulo 2

Desenvolvimento

Neste capítulo são apresentadas as estruturas do projeto, ficheiros e funcionalidades do mesmo.

2.1 Estruturas

Local

- Estrutura disponibilizada no enunciado do trabalho que serve para definir um espaço onde posteriormente serão inseridas pessoas.

```
typedef struct sala local, *plocal;
struct sala{
    int id; // id numérico do local
    int capacidade; // capacidade maxima
    int liga[3]; // id das ligações (-1 nos casos nao usados)
};
```

Figura 2.1: Estrutura Local

Pessoas

- Estrutura usada para a lista ligada que contém informações relativamente a uma pessoa.
- Estrutura que contem o nome da pessoa, a sua idade, o estado atual e o número de dias doente.
- O ponteiro **prox** foi criado para apontar para o próximo elemento da lista.
- Foi criado também o ponteiro **local** que aponta para a estrutura **local** referente ao espaço, para que se consiga integrar as duas estruturas.

```
typedef struct pessoas pessoas, *p_pessoa;
struct pessoas{
    char nome[100]; // Identificador único alfanumérico (1 palavra)
    int idade; // idade da pessoa
    char estado; // Estado: 'S' -> Saudavel, 'D' -> Doente, 'I' -> Imune
    int dias_doente; // caso esteja doente, mostra ha quantos dias foi infetado
    p_pessoa prox; // ponteiro que aponta para o proximo elemento da lista
    plocal local; // ponteiro que aponta para a estrutura referente ao espaço
};
```

Figura 2.2: Estrutura Pessoas

Iteracoes

- Estrutura que guarda todas as informações das pessoas relativamente a uma iteração.
- Esta estrutura vai ser usada para ser criada uma lista de listas.
- Os nós da lista principal, guardam o número da iteração em questão e contém, dois ponteiros, o **p_pessoa pessoa** que aponta para uma estrutura de pessoa e o **p_iteracoes prox** que será o ponteiro que aponta para o próximo elemento da lista principal

```
typedef struct iteracoes iteracoes, *p_iteracoes;
struct iteracoes{
    int numero_iteracoes;
    p_pessoa pessoa;
    p_iteracoes prox;
};
```

Figura 2.3: Estrutura Iterações

2.2 Ficheiros

No início do programa é pedido ao utilizador um ficheiro binário que contém a estrutura do espaço que vamos usar na simulação, carregando toda a sua informação para um **array dinâmico**.

E1

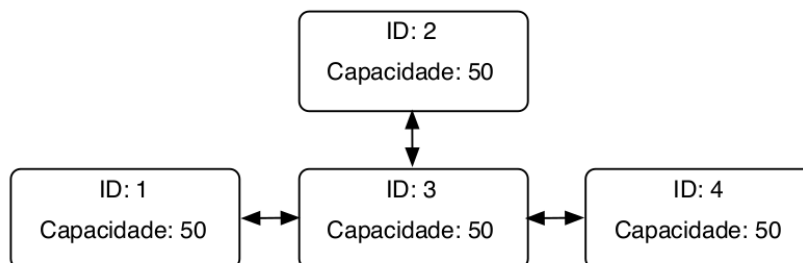


Figura 2.4: Exemplo do ficheiro E1.bin

A seguir a ser pedido ao utilizador o ficheiro binário, é pedido um ficheiro de texto que contem toda a informação das pessoas que irão participar na simulação. Esta informação será guardada numa lista ligada, assim como é pedido no enunciado.

```
PauloPires1  23  S
AnaLebre34A  55  I
Tomas111    12  S
PauloPires2   67  D  10
LuisaSantos  40  D   3
Zulmira2A    17  S
```

Figura 2.5: Exemplo do ficheiro pessoasA.txt

2.3 Funções

O programa tem várias funções, algumas destas feitas especificamente para ir resolvendo o enunciado, outras são só funções auxiliares. Ao longo da resolução do Trabalho Prático, fui criando funções auxiliares que me ajudaram a concluir o trabalho. Estas funções auxiliares foram criadas no ficheiro **utils.h** para que o código se mantivesse limpo e claro.

No ficheiro **espaco.h** estão todas as funções que manipulam o array dinâmico que contém as estruturas do espaço.

- **local *escreveArrayLocal(local *tab, int *total);**
 - Função que lê de um ficheiro binário e escreve cada estrutura num array dinâmico(chamado de "**sítio**" no **main.c**).
- **void validaDadosIniciais(local *tab, int total);**
 - Função que valida se os IDs dos espaços são válidos, únicos e se todas as ligações entre os espaços estão ou não corretos.

No ficheiro **personas.h** estão as funções que manipulam a lista ligada.

- **p_pessoa escreveNaLista(p_pessoa aux);**
 - Função que lê o ficheiro de texto e de seguida escreve numa lista ligada toda essa informação.
- **p_pessoa criaPessoa(p_pessoa p, local *tab, int total);**
 - Função que cria uma nova pessoa e insere-a na lista a pedido do utilizador.
- **void inserePessoasLocais(local *tab, p_pessoa p, int total);**
 - Função que insere pessoas em locais de maneira aleatória, tendo em conta a capacidade do total.

No ficheiro **utils.h** estão as funções auxiliares do programa. O ficheiro está devidamente comentado para que seja mais fácil e perceptível do que cada função faz. Em baixo estão alguns exemplos das funções auxiliares mais importantes presentes no ficheiro.

- **int *copy(local *tab, int total);**
 - Função que serve para armazenar num array dinâmico os IDs das salas em uso.
 - Esta função é usada noutras duas funções(**void pessoasSalas(...)** e **void taxaDi(...)**), como função auxiliar.
- **int ligaDireta(int id_origem, int id_destino, int total, local *tab);**
 - Esta função vai verificar se o **id_origem** e **id_destino** que o utilizador escolheu, para ser feita a transferência de pessoas entre locais, existe na simulação.
- **int verificaPessoas(int npessoas, int total, p_pessoa p, int id_origem, int id_destino, local *tab);**

- Esta função vai verificar várias coisas para perceber se a transferência de pessoas entre locais é válida depois de já ter validado a função **int ligaDireta(...)**.
- A primeira coisa que verifica é se ainda há espaço no **local de destino** e se o **n_pessoas** que o utilizador quer passar do **local de origem** é válido(considera-se válido uma sala de origem com 3 pessoas e sejam passadas até 3 pessoas.).
- Por fim, **soma o numero_pessoas_destino** com o **n_pessoas**, se a soma for menor que a **capacidade_destino**, então a transferência é válida.
- Depois das funções **int verificaPessoas(...)** e **int ligaDireta(...)** retornarem 1, a função **void transPesso(...)** é feita e são transferidas as pessoas.

2.4 Funcionalidade do Programa

No **main.c** são carregadas todas as informações para o array dinâmico e para a lista ligada depois de serem feitas todas as verificações nas funções vistas anteriormente. Depois disso, são chamadas duas funções(**void printIdadeDoentesFich(p_pessoa p** e **void printfTaxaSDIFich(p_pessoa p)** que escrevem no ficheiro **reports.txt** o estado e taxas das pessoas na primeira iteração.

De seguida aparece um **switch case** que representa o menu da aplicação. A primeira opção do menu é a função **void avaInt(p_pessoa p, local *tab, int total)** que está responsável por avançar uma iteração. Esta função vai chamar as funções das taxas e probabilidades que foram descritas no enunciado, aplicando assim o modelo de propagação do vírus.

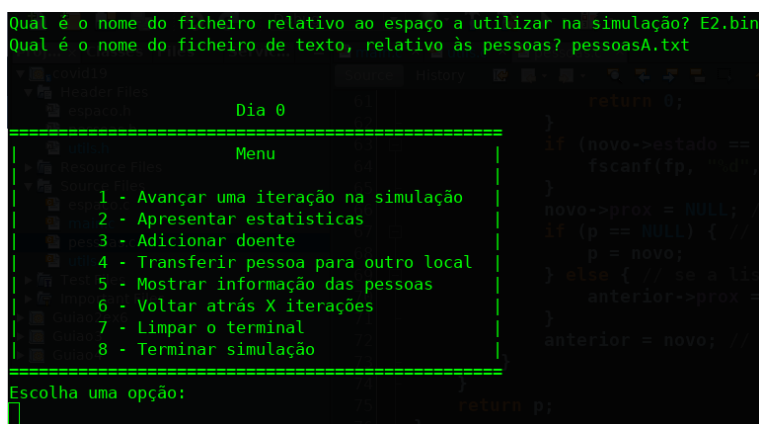
Existe também a opção de mostrar as estatísticas todas relativamente às pessoas. Isso é feito quando a função **void mostraEst(p_pessoa p, local *tab, int total)** é executada. Esta função apresenta várias médias, taxas e mostra também as pessoas por sala, tudo isto no terminal.

Sempre que uma pessoa doente passou o tempo máximo de infeção, ou recuperou usando a probabilidade de recuperação, o seu estado passa a **R**. Este estado é bastante importante porque mostra que essa pessoa já esteve doente e conseguiu recuperar. Depois de ter recuperado, ela está automaticamente sujeita à taxa de imunidade uma vez que recuperou. Caso essa taxa se aplique, o estado dela será então **I**.

Ainda é possível inserir pessoas doentes, transferir pessoas entre locais de maneira aleatória, mostrar a informação da lista das pessoas e voltar **N** iterações para trás. As funcionalidade que volta **N** iterações para trás, é feita usando uma lista de listas, em que a lista principal é o número da iteração em questão e as listas que saiem dessa lista, dizem respeito às pessoas e ao seu estado naquela iteração.

No final, quando o utilizador quiser que a aplicação termine a simulação, é pedido que o mesmo diga o nome de um ficheiro para inserir a informação relativamente às pessoas na ultima iteração e logo a seguir é guardado no ficheiro **reports.txt** as ultimas estatisticas concluindo assim um ficheiro de relatório que descreve o que aconteceu na simulação.

2.5 Manual do utilizador



```

Qual é o nome do ficheiro relativo ao espaço a utilizar na simulação? E2.bin
Qual é o nome do ficheiro de texto, relativo às pessoas? pessoasA.txt

Dia 0

Menu

1 - Avançar uma iteração na simulação
2 - Apresentar estatísticas
3 - Adicionar doente
4 - Transferir pessoa para outro local
5 - Mostrar informação das pessoas
6 - Voltar atrás X iterações
7 - Limpar o terminal
8 - Terminar simulação

Escolha uma opção: 1
  
```

Figura 2.6: Menu principal

Assim que o programa começa, é pedido que o utilizador introduza 2 ficheiros. Um ficheiro binário com o espaço a ser usado na simulação e um ficheiro de texto que contem um grupo de pessoas usadas.

```

    }
    Dia 0
=====
while (!feof(Menu)) { // percorre o fi
    1 - Avançar uma iteração na simulação));
    2 - Apresentar estatísticas
    3 - Adicionar doente na alocação de
    4 - Transferir pessoa para outro local
    5 - Mostrar informação das pessoas
    6 - Voltar atrás X iterações
    7 - Limpar o terminal "cls" ou "clear", novo-
    8 - Terminar simulação (ade < 0) {
=====
Escolha uma opção:      return 0;
5
Id: 32 Nome: PauloPires1 Idade: 23 Estado: S {
Id: 20 Nome: AnaLebre34A Idade: 55 Estado: nIvo->
Id: 32 Nome: Tomas111 Idade: 12 Estado: S
Id: 32 Nome: PauloPires2 Idade: 67 Estado: D 10
Id: 30 Nome: LuisaSantos Idade: 40 Estado: D 3
Id: 32 Nome: Zulmira2A Idade: 17 Estado: S

```

Figura 2.7: Menu principal - Mostrar informação das pessoas

Escolhendo a opção **5** do menu, conseguimos ver que as pessoas já estão inseridas nas salas de maneira aleatória.

```

Dia 0
=====
printf("Menu - Escolha a opção: ");
return 0;
1 - Avançar uma iteração na simulação
2 - Apresentar estatísticas
3 - Adicionar doente (1 - pessoa, 0 - deixar.txt até ao EOF)
4 - Transferir pessoa para outro local
5 - Mostrar informação das pessoas(as)
6 - Voltar atrás X iterações
7 - Limpar o terminal
8 - Terminar simulação
=====

Escolha uma opção:
2
if (scanf("%d", &nova-idade, &nova-idade, &nova-idade) != 0) {
    if (taxa < 0) {
        printf("Taxas não podem ser negativas!\n");
        return 0;
    }
    Taxas:
    Taxa de saudáveis: 0.50 (taxa de 0.1 - 0.9)
    Taxa de doentes: 0.33 (taxa de 0.1 - 0.9)
    Taxa de imunes: 0.17 (taxa de 0.1 - 0.9)
    Taxa de recuperados: 0.00 (taxa de 0.1 - 0.9)
    p = nova;
    anterior Médias = nova;
    }
    A média de idades de pessoas doentes é de 53.50
    A pessoa que está há mais tempo doente tem 67 anos e está há 10 dias doente
    return p;
}
Pessoas por sala
Há 0 pessoas na sala 10
Há 1 pessoas na sala 20
Há 1 pessoas na sala 30
Há 0 pessoas na sala 31
Há 4 pessoas na sala 32

```

Figura 2.8: Menu principal - Apresentar estatísticas

Na opção 2, vemos as taxas possíveis, as médias e as pessoas por sala na primeira iteração.

```

    printf(
Escolha uma opção: fclose(fp);
3      return 0;
Qual é o id: 30
Qual é o identificador da pessoa: brunoTeixeira
Qual é a idade: 23 if (novo->idade < 0) {
Há quantos dias esta doente: 3 "As idades não po
    return 0;
    } Dia 0
=====
|      Menu:inf(fp, "id", &novo->
|      )
|      1 - Avançar uma iteração na simulação ou
|      2 - Apresentar estatísticas // se a lid
|      3 - Adicionar doente novo;
|      4 - Transferir pessoa para outro local
|      5 - Mostrar informação das pessoas novo;
|      6 - Voltar atrás X iterações
|      7 - Limpar o terminal novo; // o anteri
|      8 - Terminar simulação
|
=====
Escolha uma opção:
5
Id: 30 Nome: brunoTeixeira Idade: 23 Estado: D 3
Id: 32 Nome: PauloPires1 Idade: 23 Estado: S
Id: 20 Nome: AnaLebre34A Idade: 55 Estado: I
Id: 32 Nome: Tomas111 Idade: 12 Estado: S
Id: 32 Nome: PauloPires2 Idade: 67 Estado: D 10
Id: 30 Nome: LuisaSantos Idade: 40 Estado: D 3
Id: 32 Nome: Zulmira2A Idade: 17 Estado: S
```

Figura 2.9: Menu principal - Adicionar doente

Podemos também adicionar um doente usando a opção **3**. Esta opção pede um id válido, um identificador da pessoa a sua idade e há quantos dias está doente. Como vemos na imagem, depois de usarmos novamente a opção **5** aparece o novo doente inserido no início da lista ligada.

```

Escolha uma opção: fclose(fp);
4         return 0;
Sala de origem: 32
Sala de destino: 30;scanf(fp, "%s %d %d", &novo->
Quantas pessoas da sala de origem quer passar? 1
Transferência feita com sucesso As idades não po
        return 0;
    }
    Dia 0
=====
|           Menu:mf(fp, "d", &novo-> |
|           )                         |
| 1 - Avançar uma iteração na simulação // se a li |
| 2 - Apresentar estatísticas // se a li |
| 3 - Adicionar doente;ovo; |
| 4 - Transferir pessoa para outro local |
| 5 - Mostrar informação das pessoasovo; |
| 6 - Voltar atrás X iterações |
| 7 - Limpar o terminal novo; // o anter |
| 8 - Terminar simulação |
=====
Escolha uma opção:;
5
Id: 30 Nome: brunoTeixeira Idade: 23 Estado: D 3
Id: 32 Nome: PauloPires1 Idade: 23 Estado: S
Id: 20 Nome: AnaLebre34A Idade: 55 Estado: I
Id: 32 Nome: Tomas111 Idade: 12 Estado: S
Id: 32 Nome: PauloPires2 Idade: 67 Estado: D 10
Id: 30 Nome: LuisaSantos Idade: 40 Estado: D 3
Id: 30 Nome: Zulmira2A Idade: 17 Estado: S

```

Figura 2.10: Menu principal - Transferir pessoa para outro local

Nesta opção fazemos todas as validações antes de transferirmos as pessoas entre locais **com ligação direta**. No final, se a transferência correr conforme esperado, é apresentada uma mensagem que mostra que foi feita com sucesso.


```
Escolha uma opção: if (novo->idade < 0) {
1         printf("As idades nao pod
        return 0;
    }
    Dia 1
=====
Menu
1 - Avançar uma iteração na simulação
2 - Apresentar estatísticas
3 - Adicionar doente
4 - Transferir pessoa para outro local
5 - Mostrar informação das pessoas
6 - Voltar atrás X iterações
7 - Limpar o terminal
8 - Terminar simulação
=====
Escolha uma opção: 1
5
Id: 30 Nome: brunoTeixeira Idade: 23 Estado: D 4
Id: 32 Nome: PauloPires1 Idade: 23 Estado: S
Id: 20 Nome: AnaLebre34A Idade: 55 Estado: I
Id: 32 Nome: Tomas111 Idade: 12 Estado: S
Id: 32 Nome: PauloPires2 Idade: 67 Estado: R
Id: 30 Nome: LuisaSantos Idade: 40 Estado: D 4
Id: 30 Nome: Zulmira2A Idade: 17 Estado: S
```

Figura 2.11: Menu principal - Avançar uma iteração na simulação

Usando a primeira opção avançamos uma iteração (1 dia) onde são aplicadas todas as taxas pedidas no enunciado. Como vemos na imagem, o Dia passou a ser o **Dia 1** e os dias das pessoas doentes aumentou também.

```
Id: 31 Nome: PauloPires1 Idade: 23 Estado: S
Id: 20 Nome: AnaLebre34A Idade: 55 Estado: I
Id: 32 Nome: Tomas111 Idade: 12 Estado: S
Id: 32 Nome: PauloPires2 Idade: 67 Estado: I
Id: 30 Nome: LuisaSantos Idade: 40 Estado: D 8
Id: 32 Nome: Zulmira2A Idade: 17 Estado: S
Dia 5
```

Figura 2.12: Menu principal - Voltar atrás X iterações 1

Estando no dia 5, podemos ver qual é o estado das pessoas nesse momento. O nosso objetivo é avançar para o dia 6, ver o estado das pessoas e recuar para o dia 5 novamente.

```

Id: 31 Nome: PauloPires1 Idade: 23 Estado: S
Id: 20 Nome: AnaLebre34A Idade: 55 Estado: I
Id: 32 Nome: Tomas111 Idade: 12 Estado: S
Id: 32 Nome: PauloPires2 Idade: 67 Estado: I
Id: 30 Nome: LuisaSantos Idade: 40 Estado: R
Id: 32 Nome: Zulmira2A Idade: 17 Estado: S
Dia 6

```

Figura 2.13: Menu principal - Voltar atrás X iterações 2

Podemos ver que a única pessoa que alterou o seu estado foi a LuisaSantos uma vez que estava doente há 8 dias e ficou recuperada no dia 6.

```

Dia 6
Menu
1 - Avançar uma iteração na simulação
2 - Apresentar estatísticas
3 - Adicionar doente
4 - Transferir pessoa para outro local
5 - Mostrar informação das pessoas
6 - Voltar atrás X iterações
7 - Limpar o terminal
8 - Terminar simulação
Escolha uma opção:
6
Quantas iterações quer recuar?1
Dia 5

```

Figura 2.14: Menu principal - Voltar atrás X iterações 3

Usando agora a opção para voltar X iterações atrás, como vemos na imagem.

```

Id: 31 Nome: PauloPires1 Idade: 23 Estado: S
Id: 20 Nome: AnaLebre34A Idade: 55 Estado: I
Id: 32 Nome: Tomas111 Idade: 12 Estado: S
Id: 32 Nome: PauloPires2 Idade: 67 Estado: I
Id: 30 Nome: LuisaSantos Idade: 40 Estado: D 8
Id: 32 Nome: Zulmira2A Idade: 17 Estado: S
Dia 5

```

Figura 2.15: Menu principal - Voltar atrás X iterações 4

Por fim, conseguimos perceber que o estado da LuisaSantos voltou a ser "Doente há 8 dias" uma vez que voltamos há iteração onde ela estava doente.

Capítulo 3

Conclusão

No final desta simulação, usando o modelo de propagação dado no enunciado, conseguimos perceber que se existirem muitas pessoas doentes a participar da mesma, no início o vírus irá se espalhar de maneira evidente e muito rápida, no entanto com o passar dos dias, e sendo a idade da pessoa um fator muito importante para as taxas, conseguimos chegar a um momento em que as pessoas ficam imunes.

DEIS Departamento de Engenharia Informática e de Sistemas

Bibliografia

- [1] Stack Overflow,
<https://stackoverflow.com/>
- [2] Overleaf Latex,
<https://www.overleaf.com/>
- [3] TutorialSpoinet,
<https://www.tutorialspoint.com/cprogramming/>