Licenciatura em Engenharia Informática

Relatório de trabalho prático

Trabalho Prático 1 - Simulação da Propagação de Vírus

Pedro dos Santos Alves - 2019112789

Coimbra, 7 de Junho de 2020

Índice

1. Introdução

O projeto a que este relatório se refere foi desenvolvido pelo aluno Pedro dos Santos Alves, no âmbito da disciplina de Programação da Licenciatura em Engenharia Informática do Instituto Superior de Engenharia de Coimbra.

O objetivo deste projeto é o desenvolvimento de um simulador de propagação de um vírus entre uma população.

Este relatório encontra-se organizado em 4 capítulos: Implementação, Testes efetuados, Manual de utilização e Conclusão.

Após este capítulo introdutório, no Capítulo 2, é feita uma descrição das estruturas de dados e do tipo de estruturas dinâmicas usadas no decorrer do programa e uma descrição do que o simulador é capaz de realizar.

De seguida, no Capítulo 3, é explicado os testes efetuados sobre os ficheiros contendo os dados iniciais.

No Capítulo 4, é apresentado um manual de utilização de ambos os programas usados, o simulador e programa principal e o "LocalMaker" para a criação e visualização dos ficheiros binários dos espaços.

Finalmente no último Capítulo, Conclusão , são indicadas as principais conclusões sobre o projeto e problemas encontrados durante o decorrer do projeto.

2. Implementação

2.1. Estruturas de dados

2.1.1. Espaço

O espaço em que decorre a simulação é constituído por um conjunto de locais interligados entre si. Cada local tem um identificador único, uma capacidade máxima e algumas ligações diretas a outros locais, com um máximo de 3 ligações.

Para o espaço foram criadas duas estruturas de dados. A estrutura "Place" contém o "id", que identifica cada espaço de forma única, "capacity", que indica o tamanho máximo de pessoas que consegue conter e "connections", um *array* de "id"s dos espaços adjacentes de cada espaço. A estrutura "ListPlace" contém um ponteiro "place" para a primeira posição do *array* dinâmico, que guardará cada local e um "size", onde guarda o tamanho desse *array*.

2.1.2. Pessoas

Existe um conjunto inicial de pessoas, guardadas em ficheiros de texto, que serão distribuídas pelos locais onde será realizada a simulação. Cada pessoa possui um identificador único, uma idade, um estado e eventualmente informação sobre há quantos dias foi infetado.

Para as pessoas foram criadas duas estruturas de dados e uma enumeração. Na estrutura "Person" estão os dados de cada pessoa, contendo um "id" para identificar cada pessoa, "age" para indicar a idade da pessoa, "status" para indicar o estado da pessoa, "days" para indicar há quanto tempo a pessoa está doente e um ponteiro "place" para um local onde está situada essa pessoa (NULL caso a pessoa não possa ser colocada no local). A estrutura "ListPerson" contém apenas uma estrutura "Person" e um ponteiro para a sua própria estrutura "ListPerson". Essa estrutura foi feita dessa maneira porque será usada como uma lista ligada, contendo os dados da pessoa e um ponteiro para a próxima pessoa. A enumeração "Status" contém três campos para indicar o estado da pessoa. Os estados possíveis são 'D' , 'S' e 'I', doente, saudável e imune respetivamente.

2.1.3. Configuração

Ao longo da simulação é usada uma estrutura com dados auxiliares. A estrutura "Config" contém um ponteiro "capacity" para guardar o número de pessoas situadas em cada local, "max_capacity" para guardar a soma da capacidade máxima de todos os locais, "real_capacity" que guarda a soma do número de pessoas de cada local, "days" que guarda o dia (iteração) onde a simulação se encontra, "console_mode" que guarda o modo de execução ("Normal" - Menu com números, "Consola" - Comandos), "day_0_s" onde guarda o dia em que houve zero pessoas doentes pela primeira vez, "day_peak" que guarda o dia em que houve mais doentes, "peak_sick" que guarda o número máximo de pessoas doentes atingido.

2.1.4. Snapshot

"Snapshot" é uma estrutura que guarda as alterações feitas antes de avançar uma iteração na simulação. Esta estrutura é combinada com várias estruturas do mesmo tipo para formar uma lista ligada, com um máximo de 3. A estrutura contém uma estrutura "Config" para guardar os dados da simulação e um ponteiro para "ListPerson" onde guarda o inicio da lista ligada de pessoas. Contém também um ponteiro para o próximo elemento, para poder ser feito as iterações sobre a lista ligada, assim como um ponteiro para o anterior para facilitar a remoção de elementos da lista.

2.2. Macros

Para que o modelo de propagação do vírus possa ser simulado ao longo do tempo foram definidos parâmetros, cujos valores são fixos. Esses parâmetros foram definidos através de macros (#define). O programa contém quatro macros para o modelo de propagação duas das quais aceitam parâmetros de entrada. Em relação ao "arredondamento para baixo" dos valores, poderia ter sido usado a função "double floor(double)" da livraria "<math.h>", porém foi decidido usar um *cast* para (int) para truncar o valor. Para valores positivos isto efetua o mesmo que a função "floor" e como os valores serão sempre positivos não haverá problema em usar o *cast*, sem necessidade de adicionar mais uma livraria.

Também foram usadas outras macros ao longo do programa como por exemplo:

- MAX_CONNECTIONS, para indicar o número máximo de conexões que um local pode ter;
- MAX_PERSON_ID, para indicar o número máximo de caracteres que o ID de uma pessoa pode ter;
- PERSON_MAX_AGE, para indicar o número máximo da idade de uma pessoa;

- MAX_SAVED _SNAPSHOTS, para indicar o número máximo de iterações guardadas;
- REAL_AGE, para calcular a idade de uma pessoa após N iterações;
- Etc.

2.3. Estruturas dinâmicas

2.3.1. Espaço

A estrutura dinâmica usada para guardar os espaços foi um *array* dinâmico. Cada vez que o programa lê e deteta um novo local ele é adicionado ao *array* alocando espaço e de seguida os dados são copiados para dentro desse *array*. Os dados estão todos seguidos uns aos outros, na memória, tornando possível aceder a cada valor aleatoriamente através de um índice.

2.3.2. Pessoas

A estrutura dinâmica usada para guardar as pessoas foi uma lista ligada. O início da lista é apenas um ponteiro para a primeira pessoa e essa primeira pessoa contém um ponteiro para a próxima pessoa, essa próxima pessoa possui um ponteiro para a seguinte e assim sucessivamente, até chegar ao último que não terá uma próxima pessoa.

2.3.3. Snapshot

A estrutura dinâmica usada para guardar as iterações efetuadas foi uma lista ligada. Assim como a estrutura dinâmica anteriormente referida, “Pessoas”, esta é apenas um ponteiro para a primeira iteração, com ponteiros para o próximo até chegar ao fim da lista. Porém esta estrutura tem ligações para as ligações anteriores, o que tornará a remoção de elementos mais fácil.

2.4. Programa

2.4.1. Inicialização e verificações

Antes de iniciar a simulação, o programa necessita de conter dados, cujo esses dados serão fornecidos por ficheiros. A informação sobre os locais estão guardados em ficheiros binários, com uma determinada estrutura. O programa lê o ficheiro binário que o utilizador indicar, através da própria aplicação ou através da linha de comandos. Após ler os dados contidos no ficheiro, o programa verifica se os dados estão coerentes. Caso o programa detete algum erro nos dados este avisará o utilizador sobre esse erro, e terminará o programa sem passar para o próximo passo. Caso os dados estejam todos corretos, o programa apresentará uma lista de locais que irão ser usados para a simulação.

Depois de verificar os locais e de os apresentar no ecrã, o programa tentará ler a informação que está no ficheiro com informação sobre as pessoas. Esse ficheiro será um ficheiro de texto e assim como o ficheiro dos locais este será um ficheiro indicado pelo utilizador através da própria aplicação ou através da linha de comandos. O programa lê os dados de cada pessoa e efetua as verificações antes de as adicionar à lista ligada. Tal como os locais, a aplicação irá avisar o utilizador sobre os erros encontrados e terminará o programa caso seja necessário. Finalmente, após as verificações todas o programa apresenta todas as pessoas e os seus dados.

2.4.2. Ligação das pessoas aos locais e distribuição

Existem pelo menos duas maneiras de fazer a ligação das pessoas aos locais:

- Colocar uma lista de pessoas em cada local (Cada local contém N pessoas);
- Colocar o local na pessoa (Cada pessoa está localizada num local).

Embora a primeira opção faça mais sentido, esta implicaria a criação de uma nova estrutura pelo facto dos ficheiros binários não conterem uma lista de pessoas. Também teriam que ser feitas mais verificações para certificar-se de que não existem pelo menos uma pessoa em dois ou mais locais distintos. Por isso, a opção escolhida neste trabalho foi a segunda opção. Com isto é apenas necessário um ponteiro para o local em que se encontrará a pessoa.

A distribuição das pessoas pelos locais é feita de forma aleatória. O programa verifica quantas pessoas já estão com locais associados e compara com a capacidade máxima do espaço (capacidade de todos os locais juntos). Também verifica se a capacidade do local aleatoriamente escolhido já atingiu o limite, caso não tenha atingido o limite associará a próxima pessoa da lista ao local. Assim que terminar de distribuir as pessoas pelos locais, as restantes serão “cortadas” da lista ligada e não farão parte da simulação.

Logo a seguir à distribuição das pessoas pelos locais, é inicializado as variáveis auxiliares de configuração, que ajudarão no percorrer da simulação.

2.4.3. Simulação

Na fase de simulação é apresentado um menu com várias opções. Com esse menu é possível avançar uma iteração na simulação (+1 dia), apresentar várias estatísticas, adicionar uma pessoa doente à simulação, transferir um determinado número de pessoas entre locais, mudar para o modo "consola", terminar a simulação e interromper a simulação.

Ao avançar uma iteração na simulação, o programa primeiro guarda as informações do momento numa estrutura "Snapshot", logo de seguida verifica se existe pessoas doentes no local para "espalhar" o vírus e se isso se confirmar, juntamente com outros parâmetros, N pessoas aleatórias serão infetadas, se possível. Para o vírus se propagar num local terá que pelo menos existir uma pessoa infetada entre vinte pessoas, tendo X * S * D = N em que X = número total de pessoas no local, S = probabilidade de disseminação, D = número total de pessoas doentes no local e N = número de pessoas que poderão ser infetadas nesse local, temos assim 20 * 0.05 * 1 = 1. Após efetuar a disseminação do vírus, a probabilidade de uma pessoa recuperar é calculada. Isto significa que neste programa, poderão existir pessoas que recuperam no mesmo dia e não serão mostrados ao utilizador. Para cada pessoa doente é calculada a probabilidade de recuperação, através da sua idade. Caso essa pessoa recupere, no código, os dias serão colocados ao máximo pois a probabilidade de recuperar após os dias máximos serem atingidos é de 100%, não sendo necessária outras verificações ou repetição de código. Depois é calculada a probabilidade de essa pessoa ficar imune ao vírus, o que fará que essa pessoa não seja afetada caso seja escolhida para ser infetada com o vírus.

A apresentação de várias estatísticas mostra os vários dados, como as taxas de saudáveis, imunes e doentes, assim como o número total de pessoas em cada um desses estados; distribuição de pessoas por sala, número de pessoas nessa sala, número de pessoas doentes nessa sala e o número de pessoas que podem ficar doentes; número da iteração (dia); número de locais; capacidade máxima; número de pessoas presentes na simulação; iteração (dia) em que o vírus foi erradicado; iteração (dia) em que houve um maior número de doentes e o número máximo de pessoas doentes encontradas; e por fim as salas e suas ligações.

Para adicionar uma pessoa doente à simulação o utilizador terá que introduzir uns dados primeiro, como o ID do local, o ID da pessoa, a idade, e os dias que esteve infetado. Se os dados forem todos introduzidos corretamente a pessoa é adicionada à lista ligada sendo assim adicionada à simulação.

Para transferir pessoas de um local para outro, o utilizador terá que indicar o número de pessoas a serem transferidas e os IDs dos locais associados a essa transferência. Os locais terão que ser adjacentes e o número de pessoas terão que ser menor ou igual ao número de pessoas no local de origem e menor ou igual à capacidade máxima da sala de destino mais as pessoas que já lá se encontrarem.

Ao terminar a simulação o programa pedirá para ser indicado o caminho para um ficheiro de texto onde serão guardada as pessoas após as iterações efetuadas sobre elas.

Existem duas opções extra que poderão ser convenientes para a fase de simulação, como o modo “consola” e a interrupção da simulação. A interrupção da simulação termina a simulação, porém não criará os ficheiros de texto no final. O modo “consola” é apenas um modo onde não é necessário escolher um número no menu, apenas colocar o comando e o programa fará o mesmo que as opções do menu no modo “normal”. Ou seja, em vez de andar em sub-menus a tentar transferir pessoas de um local para outro, isso poderá ser feito num só comando.

2.4.4. Relatórios

No final do programa, se o utilizador assim o desejar, serão feitos relatórios finais com informação sobre a última iteração efetuada. O utilizador indicará um caminho com o nome de um ficheiro onde serão escritos os dados das pessoas todas que participaram na simulação. Irá também ser acrescentado para um ficheiro “report.txt” os dados da última iteração da última simulação feita, assim como quem participou na simulação e quem ficou de fora e os locais. Cada simulação tem um ID único e por simplicidade foi escolhida o número de segundos após 1 de Janeiro de 1970, na altura de escrita (Unix Time).

3. Testes efetuados

3.1. Ficheiros

Para verificar se a leitura dos ficheiros está a funcionar corretamente, foram criados ficheiros extra. Alguns ficheiros com informação correta e outros com informação errada com a intenção de parar o programa antes de iniciar a simulação.

3.1.1. Espaços

Os ficheiros dos espaços são ficheiros binários, logo teve que ser criado um outro programa para auxiliar na criação desses ficheiros ou visualizar esses mesmos ficheiros. Os ficheiros que deverão terminar o programa antes iniciar a simulação são: "_EC.bin", "_EC2.bin", "_EC3.bin", "_EC4.bin" e "_pessoasE.txt". Estes ficheiros contém conexões para locais inexistentes, conexões repetidas ou conexões para o próprio local. O ficheiro "_pessoasE.txt" pode ser usado como se fosse um ficheiro binário vazio.

3.1.2. Pessoas

Os ficheiros das pessoas são ficheiros de texto, facilmente editáveis. Os ficheiros que deverão terminar o programa antes de iniciar a simulação são: "_pessoasC.txt" e "_pessoasE.txt".

O ficheiro "_pessoasC.txt" contém uma pessoa com ID repetido, fazendo com que os IDs não sejam únicos e que o programa termine mais cedo.

O ficheiro "_pessoasD.txt" armazena informação de cem pessoas com IDs únicos para testar a capacidade dos locais.

O ficheiro "_pessoasE.txt" é um ficheiro vazio. Com este ficheiro o programa irá terminar por haver falta de pessoas para iniciar a simulação.

O ficheiro "_pessoasT.txt" é igual ao ficheiro "pessoasA.txt", porém, em vez de estar separado por um ou vários espaços, estão separados por um ou vários *tabs*. O programa está programado para interpretar os *tabs* como se fossem espaços.

4. Manual de utilização

4.1. Simulador

O utilizador pode abrir o programa com argumentos da consola “programa.exe [FICHEIRO_LOCAIS] [FICHEIRO_PESSOAS]” ou clicar no executável. O primeiro argumento será sempre o ficheiro de locais e o segundo o de pessoas, podem ser usados zero, um ou dois argumentos. Caso sejam usados um argumento, o programa irá perguntar pelo ficheiro de pessoas e se usar dois argumentos, o programa irá prosseguir à leitura dos ficheiros passados por argumento.

Na fase de simulação o utilizador verá um menu com várias opções e outras informações adicionais como é mostrada na figura 1.

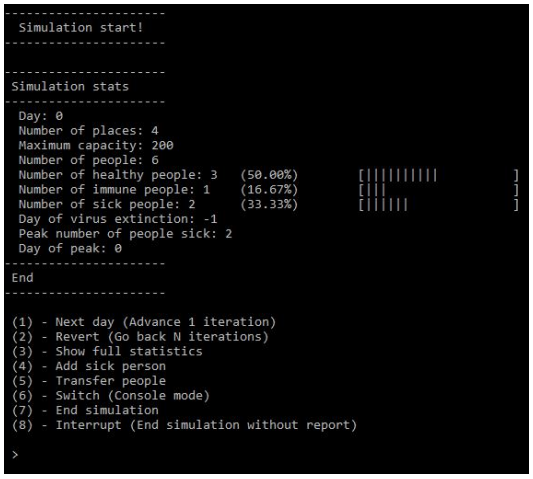

Fig. 1 - Menu principal

Para o utilizador selecionar uma opção basta colocar o número da opção a escolher e clicar na tecla “ENTER”. A figura seguinte mostra o modo “consola” em que o utilizador não introduz números mas sim comandos. Para ver a lista completa de comandos basta escrever “help” e clicar na tecla “ENTER”.

Fig. 2 - Modo "consola"

Ao selecionar a opção 1 (comando “step” ou “next” no modo “consola”), o programa irá aplicar o modelo de simulação e avançar uma iteração.

A opção 2 (comando “undo [N]” ou “prev [N]” no modo “consola”), voltará N iterações atrás na simulação.

A opção 3 (comando “show” ou “stats” no modo “consola”), mostrará mais informação da que é apresentada no menu principal.

A opção 4 (comando “add [place_id] [person_id] [person_age] [days_sick]”), irá apresentar outro menu como mostra na figura 3.

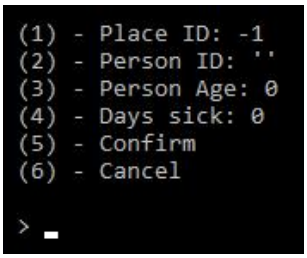

Fig. 3 - Sub-menu "Adicionar pessoa doente"

Neste sub-menu o utilizador poderá ver os dados já introduzidos e selecionar uma opção da mesma maneira que no menu principal para introduzir o dado desejado. No final para adicionar a pessoa doente basta selecionar a opção 5.

Voltando ao menu principal, ao selecionar a opção 5 (comando “move [N] [src_id] [dest_id]” no modo “consola”), irá ser mostrado um menu idêntico ao anterior, como é mostrado na figura 4.

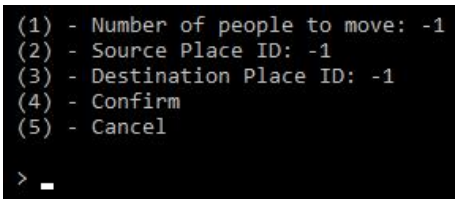

Fig. 4 - Sub-menu "Transferir pessoas"

Neste sub-menu o utilizador poderá ver os dados já introduzidos e selecionar uma opção.

Voltando novamente ao menu principal, a opção 5 deixará o utilizador usar o modo consola até que o utilizador introduza “switch” para voltar ao modo normal. Este modo de “consola” poderá ser mais rápido e fácil de usar para utilizadores mais avançados, em vez de andar entre menus e sub-menus.

A opção 7 (comando “end”, “terminate” ou “exit” no modo “consola”), irá terminar a simulação e criar dois ficheiros de texto. Um ficheiro com o mesmo formato que os ficheiros de pessoas usadas no programa. Deverá introduzir o nome desse ficheiro assim que o programa o pedir. Outro ficheiro com o nome “report.txt” será criado caso não exista e colocado toda a informação da última iteração da simulação. Caso esse ficheiro exista, o programa acrescentará ao final do ficheiro a informação.

Finalmente a opção 8 (comando “kill”, “int” ou “interrupt” no modo “consola”), irá terminar a simulação porém não irá criar relatórios finais.

4.2. LocalMaker

No inicio, o utilizador deverá introduzir o número de locais que serão criados. Depois de ser introduzido o número de locais, irá ser pedido a cada local o seu ID e a capacidade, como mostra a figura 5.

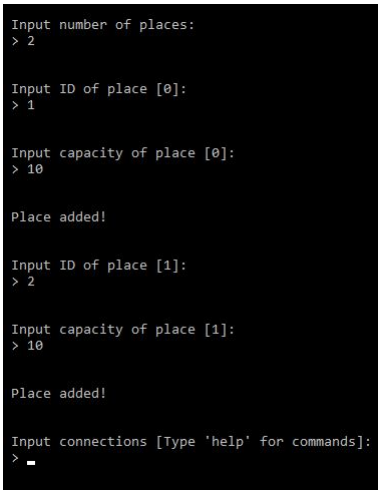

Fig. 5 - Inicio do programa "LocalMaker"

Após introduzir todos os dados iniciais, o utilizador poderá introduzir as ligações caso sejam necessárias. Este programa permite adicionar conexões de forma legítima (todas as conexões adicionada serão verificadas) ou de forma errónea (todas as conexões adicionadas não serão verificadas).

Este programa permite visualizar os ficheiros binários criados, ou já criados através do uso da linha de comandos “programa.exe [FICHEIRO]”. Irá mostrar os locais e os seus dados e conexões, assim como também se o ficheiro é válido ou inválido (mostrado na figura 6).

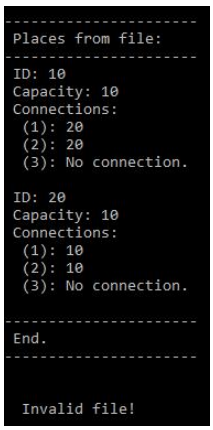

Fig. 6 - Visualizar o ficheiro "_EC4.bin"

5. Conclusão

O principal objetivo do trabalho prático da disciplina de Programação da Licenciatura em Engenharia Informática, foi a criação de um simulador de propagação de um vírus em C standard, norma C99, onde seja possível manipular as iterações, mostrar estatísticas, adicionar pessoas doentes e transferir pessoas de um local para outro. Para isso, foi necessário fazer o planeamento de algumas funcionalidades. De seguida, iniciou-se o desenvolvimento do simulador, com as funcionalidades e requisitos pedidos.

No decorrer do trabalho prático surgiram algumas dificuldades, como por exemplo: na verificação dos ficheiros dos espaços, na verificação dos ficheiros das pessoas e no uso de caracteres da língua portuguesa.

Nas fases iniciais do trabalho prático, existiam ficheiros de espaços limitados, mais concretamente três ficheiros, em que todos eles contém informação correta e coerente, logo a verificação da informação desses ficheiros não pôde ser avaliada. O algoritmo aceitava todos os ficheiros mesmo os ficheiros inválidos. Depois da criação de um programa auxiliar "LocalMaker" pôde ser criado mais ficheiros binários de espaços, alguns com informação inválida, podendo assim ser verificado o algoritmo de avaliação dos dados lidos e corrigido na versão final.

Na verificação dos ficheiros das pessoas também havia um problema que o programa aceitava os *tabs* como se fossem espaços porém não separava os dados, ou seja, dado uma linha "Pessoa1[tab]20[espaço]S" o programa ficaria com os seguintes dados: ID - "Pessoa1[tab]20"; Idade - "0"; Estado - "?"; Dias - "0" (em que "?" é "lixo"). Assim foi feito com que o programa lê-se *tabs* como se fossem espaços e separe os dados.

Finalmente o uso de caracteres da língua portuguesa como "á", "í", "ú", etc… não foi possível, mesmo com o uso do header "<locale.h>" e função setlocale(LC_ALL, "Portuguese") no inicio do programa. E por essa razão os programas foram desenvolvidos com a língua inglesa.

Concluindo, este trabalho prático terá a maioria, se não todas, as funcionalidades pedidas no enunciado como: a leitura a partir de ficheiros binários e de texto, verificação dos dados lidos, uso de estruturas dinâmicas como *array* dinâmicos e listas ligadas, parâmetros com valores fixos (macros e *function-like* macros), avançar 1 iteração na simulação aplicando o modelo de propagação, apresentação de estatísticas, adição pessoas doentes, transferência de pessoas, escrita para ficheiros de texto informações sobre pessoas e estatísticas finais.

A. Anexo I - Ficheiros dos espaços