DEEC
DEPARTAMENTO DE ENGENHARIA ELETROTÉCNICA E DE COMPUTADORES
TÉCNICO LISBOA

Licenciatura em Engenharia Electrotécnica e de Computadores (LEEC)

ALGORITMOS E ESTRUTURAS DE DADOS

Ficha de Avaliação - 2023/2024

dISTilling

Aluno Nº: 103376 Nome: Neelam Visueshcumar Nota: _____

Aluno Nº: 103681 Nome: Francisco Apolinário Nota: _____

Grupo: GR 143

Primeira fase: Número da submissão: 4 (2 do grupo) Pontuação: 300

Fase Final: Número da submissão: 50 (6 do grupo) Pontuação: 1200

Na medida da sua percepção do trabalho realizado, preencha o seguinte quadro relativamente à **organização** do seu projecto:

Organização do projecto e compilação	Sim	Não
O projecto encontra-se dividido em vários módulos `.h` e `.c`	☒	☐
Se sim, essa divisão é a mais correcta (as funções em cada módulo constituem um conjunto logicamente relacionado) (cada módulo é autónomo e tem vida para lá deste projecto)	☒	☐
Se sim, o projeto toma partido de abstração de dados	☒	☐
O projecto está correctamente documentado através de comentários no código fonte	☒	☐
Se sim, há *headers* em todos os módulos e funções	☒	☐
O projecto tem `Makefile` e compila sem erros	☒	☐
O projecto compila sem *warnings* (opção -Wall)	☒	☐
O projecto gera solução correcta em todos os problemas fornecidos	☒	☐
Se não, indique características que tenha notado nos problemas em que funciona ou em que não funciona (ex: dimensão, tipo, etc)		

Na medida da sua percepção do trabalho realizado, preencha o seguinte quadro relativamente ao **funcionamento** do seu projecto:

Funcionamento	Sim	Não	C/ falhas
O programa é robusto quando invocado com argumentos incorretos	☒	☐	☐
O programa gera sempre soluções possíveis de acordo com as regras	☒	☐	☐
O programa gera sempre a melhor solução	☒	☐	☐
O programa funciona com ficheiros com múltiplos problemas	☒	☐	☐
O programa nunca crasha em circunstâncias nenhumas	☒	☐	☐
As alocações de memória são sempre verificadas	☒	☐	☐
O programa corre sem ter fugas de memória (i.e., o relatório da execução com valgrind está limpo)	☒	☐	☐

Na medida da sua percepção do trabalho realizado, preencha os quadros abaixo, relativos a opções de **implementação** do projecto:

Representação do problema - Estruturas de dados

Leitura do ficheiro de jogo:

Quantas vezes lê o ficheiro de entrada? ☒ 1 vez ☐ Mais do que 1 vez
Se lê mais do que 1 vez, porquê? _____

Resolve problemas com ☐ apenas uma parede ☒ várias paredes
Se resolve com várias paredes, tem em memória
☐ Todas as paredes ☒ Apenas uma parede de cada vez

Representa cada parede através de:
☒ Matriz ☐ Outra? Qual: _____

Guarda alguma informação sobre as diferentes cores (para além da parede):
Se sim, em que formato ☐ Lista ☒ Tabela ☐ Hash table
☐ Outra? Qual: _____

Com que dimensão: 2 x Nº elementos distintos na parede

Porquê? A tabela apresenta uma linha com a identificação dos números/elementos e outra com a contagem dos mesmos, esta contagem vai sendo atualizada ao longo da resolução do problema, com o objetivo de a cada jogada se verificar se o caminho de jogadas adotado ainda é viável.

O armazenamento indicado atrás é mesmo quer as cores sejam números consecutivos ou é diferente caso não sejam? Explique e surpreenda-nos...
Sim, inicialmente é alocada uma tabela de dimensões 2 x Nº elementos total na parede, após isto é realizada uma leitura da parede e sempre que identificado um elemento novo este é guardado na primeira linha do primerio indice livre na tabela (cujo valor anterior era 0, forma como é inicializada a tabela) e é escrito 1 na segunda linha do mesmo indice, caso o elemento visto já tivesse sido encontrado apenas se incrementa o valor da contagem na segunda linha da tabela no indice associado ao mesmo. Após a verificação de toda a parede poderão existir ainda posições da tabela livres (com o valor na primeira linha a 0) como tal aloca-se uma tabela de dimensões 2 x Nº elementos distintos na parede, e copia-se para aí toda a informação na tabela anterior. Por fim a memória associada à primeira tabela é libertada.

Resolução do problema - Manchas e Lances

Identificação de manchas:

Em cada momento, para identificar as manchas existentes na parede de jogo usa:
☐ DFS ☐ BFS ☐ FloodFill ☒ Conetividade
☐ Outro: _____

Se usa Conetividade que algoritmo usa?
☐ QF ☐ QU ☐ WQU ☒ CWQU

Quando aplicável, a implementação é:
☐ Recursiva ☒ Iterativa

Identificação de lances:

Para efetuar uma jogada na parede atual:
☐ Identifica apenas o próximo lance ☒ Identifica e guarda todos os lances possíveis

Se identifica apenas o próximo que informação guarda para evitar voltar a repetir lances?

Se identifica todos os lances possíveis como o faz?
☒ DFS ☐ BFS ☐ Outro: _____
Como guarda essa informação: ☒ Lista ☐ Tabela ☐ Outra
Descreva sucintamente:
Após a realização da conectividade, nos modos 2 e 3 são identificadas as várias manchas e guardados os id's do vetor de conectividade de um dos elementos dessa mancha e os pontos associados à mesma numa lista. A insersão na mesma é feita na cabeça da lista, deste modo para cada parede são guardados todos os lances possiveis, para que se possa experimentar as várias hipoteses de modo a obter o resultado pretendido.

Resolução do problema - Procura

Sequência de lances:

Que algoritmo usa para testar uma sequência de lances?

☒ DFS ☐ BFS ☐ A* ☐ Outro: _____

A implementação é: ☐ Recursiva ☒ Iterativa

Como está implementada a "fila de procura"?

☐ Tabela ☒ Lista ☐ Acervo ☐ Outro: _____

Qual é a prioridade de entrada/saída na fila?

☒ LIFO (stack) ☐ FIFO (file) ☐ PFS ☐ Outro: _____

Qual é a dimensão máxima da "fila"? Nº máximo de lances testados para obter um tabuleiro sem manchas

☐ É instanciada estaticamente ☐ É alocada uma só vez

☒ É dinamicamente alocada e libertada quando necessário.

Se é alocada dinamicamente, a dimensão varia ao longo do processo de procura? Explique.

Sim, a dimensão da fila diminui e aumenta consoante se anda para a frente e para trás nos lances. Uma vez que quando se anda para trás os elementos são retirados da mesma e quando se anda para a frente estes são adicionados. Apesar disto, a memória uma vez alocada para a mesma já não o volta a ser, isto porque aquando da remoção de elementos estes são introduzidos numa lista de reciclagem e quando é necessário adicionar elementos na fila estes só são alocados se não existir nenhum restante na lista de reciclagem.

Ou é constante? Nesse caso com que dimensão? Explique

Remove **sempre** todas as manchas em variante 1? ☒ Sim ☐ Não

Resolve **sempre** correctamente paredes em variante 2? ☒ Sim ☐ Não

Encontra a pontuação máxima em variante 3 **sempre**? ☒ Sim ☐ Não

Se não, depende de: ☐ Dimensões da parede ☐ Número de cores diferentes

☐ Outra Qual? _____

Em variantes 2 e 3 possui algum mecanismo para limitar o número de caminhos explorados na totalidade? ☒ Sim ☐ Não

Se sim, descreva-o com detalhe.

O mecanismo utilizado para limitar o número de caminhos explorado baseia-se numa função que com auxilio da tabela de dimensões 2 x Nº de elementos distintos da parede (indicada na segunda página), cálcula os pontos máximos ainda atingiveis. Assumindo para tal o melhor caso possivel, isto é que todos os elementos iguais da tabela pertencem à mesma mancha. Como tal vê o número de pontos associados ao número de vezes que cada elemento existente aparece na tabela para aquela jogada especifica. Soma todos os pontos individuais dos elementos, obtendo assim o número de pontos que o tabuleiro ainda possui, no melhor caso. Por fim, soma este valor com os pontos atuais e verifica se atinge o objetivo.
No modo 2 este objetivo passa por atingir ou superar o número de pontos indicado no cabeçalho do problema.
No modo 3 este objetivo passa por superar o que se tem como máximo de pontos atual.
Em ambos os casos em caso de ainda ser possivel obter o resultado pretendido procede-se com o caminho que está a ser explorado, se se verificar logo através desta que esse resultado não é alcansavel interrompe-se esse caminho e experimenta-se uma alternativa diferente.

Verificou a eficácia do mecanismo? Como?

Sim. A eficácia do mecânismo foi verificada durante a implementação do mesmo, inicialmente o mecanismo utilizado considerava todos os números como sendo iguais e consideráva-os como sendo apenas uma mancha, contudo posteriormente por forma de tornar este mecanismo o mais seletivo possivel, no que toca a jogadas com possibilidade de sucesso alterou-se o mecanismo para o atual. Verificou-se também que com o mecanismo várias das sequências de jogadas não chegavam a ser feitas direcionando o programa para o resultado pretendido e e resutando numa redução substancial do tempo de resolução dos problemas.

No final da procura como determina a sequência de lances a imprimir:

☒ Recursivamente com as estruturas existentes ☒ Iterativamente com as estruturas existentes

☐ Com estruturas adicionais Quais e como? _____

No modo 1 e 2 esta impressão é realizada recursivamente, imprimindo uma lista e stack, respetivamente, do fim para o início, no modo 3 a impressão é realizada iterativamente, imprimindo uma lista do início para o fim.

Análise de Complexidade para uma só parede com L linhas, C colunas e T cores diferentes

Com base na sua implementação indique:

Nota: N = L x C
P = Nº lances máximo na stack
M = Nº total de lances realizados/experimentados

Memória:

Qual a complexidade espacial anterior à determinação da solução? Caso seja diferente, explicite as diferenças para cada uma das 3 variantes:

A complexidade a nível espacial anterior à determinação da solução é O(N) nos 3 modos.
No modo1 é apenas guardada a informação referente ao cabeçalho do problema e à parede a resolver (L por C), como tal, da inserção da informação na tabela surge esta complexidade.
No modo 2 e 3 o mesmo ocorre, com o acrescimo da tabela anteriormente referida, na segunda página. Este processo como visto também tem complexidade O(N).

Qual a complexidade espacial da determinação da solução? Caso seja diferente, explicite as diferenças para cada uma das 3 variantes.

A complexidade espacial da determinação da solução é O(N), no modo1, este valor está asso - ciado aos vetores da conectividade (id e sz) de dimensões iguais às da parede (N = L x C), exis - te também memória associada à lista de jogadas realizadas contudo as dimensões dos vetores sobressai.
A complexidade espacial da determinação da solução nos modos 2 e 3 é de O(P x N), sendo o N associado às dimensões dos vetores de conectividade, à semelhança do modo 1 e P associado ao número de lances máximos que existiram na stack. Existe também uma lista de jogadas pos - siveis, contudo N sobressai em relação a estas. No modo 3 existe também uma outra estrutura com uma lista com a melhor sequência de jogadas, tendo complexidade <O(N), não sendo portanto considerada.

Temporal: (se preferir, abaixo, detalhe em vários passos, justifique as respostas)
Qual a complexidade computacional anterior à determinação da solução? Muda quando muda a variante? Como?

A complexidade computacional anterior à determinação da solução no modo 1 é O(N), comple - xidade associada à leitura da parede do problema de dimensões N (L x C).
Já no modo 2 e 3 a complexidade temporal é de O(N^2), uma vez que se realiza o mesmo pro - cedimento que no modo 1, mas com o acrescimo da identificação de número diferentes na pa - rede e contagem dos mesmos, guardados na tabela abordada na segunda página. Este proce - sso no pior caso possivel pressupõe a verificação de todos os elementos do vetor (N) para todos os elementos da parede (N).

Qual a complexidade temporal da determinação de uma solução? Descreva em que é que as 3 variantes diferem e porquê.

A complexidade temporal da determinação de uma solução no modo 1 é O(N x M), onde N se de- ve aos processos de atualização de informação e M ao número de lances que se realizaram.
A complexidade temporal da determinação de uma solução no modo 2 e 3 é também O(N x M), estando estes associados às mesmas coisas que no modo 1. De notar que nestes dois modos M será aproximadamente o valor fatorial do número de lances possiveis inicialmente, uma vez que, no decorrer da sequência de jogadas assumindo que nunca se formam novas manchas, apenas se eliminando manchas existentes no tabuleiro inicial, haverá o (número de manchas iniciais)! formas de se chegar ao resultado pretendido.

Na medida da sua percepção do trabalho realizado, preencha o seguinte quadro relativamente ao **desenvolvimento** do seu projecto:

Desenvolvimento	Sim	Não
Algum dos elementos do grupo desistiu Se sim, qual Nº _____	☐	☒
O trabalho foi equitativamente desenvolvido por todos os elementos do grupo	☒	☐
Se não foi, estime a contribuição efectuada por cada elemento do grupo (total deve ser 100%) Nº _____: _____% Nº _____: _____%		
Estimativa de esforço:		
Quanto tempo dispendeu a planear o projecto	24 horas	
Quanto tempo dispendeu a codificar o projecto	160 horas	
Quanto tempo dispendeu a testar o projecto após este estar pronto	2 horas	
Quanto tempo dispendeu a preencher esta ficha	1 horas	

Ambiente de desenvolvimento usado:
☒ Linux : (distribuição) Ubuntu ☒ WSL ☐ Windows ☐ Unix ☐ MacOS

Desempenho na UC de Programação:

Número: 103376 ☒ Aprovado com: 16 ☐ Reprovado

Número: 103681 ☒ Aprovado com: 19 ☐ Reprovado

Se usou código retirado de algum local, indique que código (listas, heaps, algoritmo, etc) e de onde o retirou (acetatos, livro, net, neste caso de onde, URL, etc). Indique a proveniência para cada bloco de código separadamente.

Código: Função problems_solve baseada em código fornecido
Fonte:
☐ Acetatos ☐ Livro da UC ☒ Outras (Qual(is)?) Laboratório 6 da UC
...