TP3 - Centro de distribuição

Algoritmos I
Data de entrega: 02/07/2023

1 Objetivo do trabalho

O objetivo deste trabalho é modelar o problema computacional descrito a seguir utilizando estruturas de dados e algoritmos, em particular aqueles vistos na disciplina, que permitam resolvê-lo de forma eficiente.

Serão fornecidos alguns casos de teste bem como a resposta esperada dos casos fornecidos para que o aluno possa testar a corretude de seu algoritmo. Não obstante, recomenda-se que o aluno crie casos de teste adicionais a fim de validar sua própria implementação. A sua solução deve obrigatoriamente ser desenvolvida utilizando programação dinâmica.

O código-fonte da solução e uma documentação sucinta (relatório contendo não mais do que 5 páginas) deverão ser submetidos via Moodle até a data limite de 02/07/2023. A especificação do conteúdo do relatório e linguagens de programação aceitas serão detalhadas nas seções subsequentes.

2 Definição do problema

Parte 1 - Programação dinâmica

Você trabalha em um centro de distribuição de uma fábrica de ligas metálicas e, com seu conhecimento em algoritmos, está pensando em otimizar o processo de distribuição. Cada fábrica produz ligas metálicas de diferentes tamanhos (em metros) e os clientes também fazem a solicitação de ligas em metros (ex: 1000 metros em ligas). O papel do centro de distribuição é separar as encomendas de cada cliente para o despache da transportadora.

Para otimizar tempo e recursos, é interessante que o número de ligas usadas para suprir uma demanda seja a mínima possível, levando em conta os tamanhos de liga disponíveis no momento. Por exemplo, se um cliente precisa de 40 metros de ligas e o centro de distribuição contém os tamanhos [1, 5, 10, 20, 25], o número mínimo de ligas que podem ser usadas é 2.

Figura 1: Caminhão saindo do centro de distribuição com 2 ligas metálicas de 20 metros para atender o pedido do cliente de 40 metros de ligas.

Parte 2 - NP-Completude

Após alcançar sucesso com o seu algoritmo no centro de distribuição regional, você tornou-se amplamente reconhecido dentro da empresa, e a sua solução inovadora chegou aos ouvidos do presidente da companhia, que ficou encantado com o algoritmo desenvolvido. Agora, ele quer que você teste o mesmo algoritmo em uma escala muito maior: o centro de distribuição global, onde as demandas dos clientes são significativamente maiores e os tipos de produtos são mais diversos.

Ao receber uma amostra das demandas dos clientes para teste, você percebeu que o algoritmo estava levando consideravelmente mais tempo para ser executado. Após realizar algumas pesquisas, descobriu-se que o problema em questão é classificado como NP-completo.

Você deve escrever um relatório ressaltando a complexidade do algoritmo fornecido no item anterior, em função do tamanho da entrada (em bits), e provando que o problema em questão se trata de um problema NP-completo.

3 Exemplo do problema

3.1 Modelagem do problema

Este trabalho prático aborda a parte de programação dinâmica e NP-completude da ementa desta disciplina. Para a resolução do problema a sua modelagem **precisa** usar usar programação dinâmica e a solução deve ser descrita sucintamente no relatório apresentado.

3.2 Formato da entrada esperada

O seu programa deverá processar vários casos de teste em uma execução. A primeira linha de um cenário de teste é composta por um inteiro T, que representa o número de casos de teste no arquivo. Cada caso de teste possui duas linhas de informações. A primeira linha é composta por dois número inteiros N e W, representando respectivamente o número de tipos de ligas metálicas disponível ($1 \leq N \leq 1000$) e a demanda do cliente, em metros ($1 \leq W \leq 1000000$). A segunda linha de um caso de teste contém N inteiros representando o tamanho l_i de cada liga metálica disponível ($1 \leq l_i \leq 1000$). Você pode assumir que **sempre haverá estoque de ligas metálicas de tamanho 1.**

3.3 Formato da saída esperada

Para cada caso de teste seu programa deve o número mínimo de ligas metálicas para suprir uma demanda de tamanho W.

3.4 Casos de teste

Entrada	Saída
3 6 126 1 5 10 15 25 50 5 40 1 5 10 20 25 2 103 1 5	4 2 23

Junto com a descrição do problema, disponibilizaremos um conjunto de casos de teste e as soluções exatas para cada caso de teste.

4 Implementação

O seu programa deverá ser implementado na linguagem C ou C++, e deverá fazer uso apenas de funções da biblioteca padrão da linguagem. Não serão aceitos trabalhos que utilizem qualquer outra linguagem de programação e/ou que façam uso de bibliotecas que não a padrão.

O aluno pode implementar seu programa em qualquer ambiente (Windows, Linux, MacOS, etc...), no entanto, deve garantir que seu código compile e rode nas máquinas do DCC (tigre.dcc.ufmg.br ou jaguar.dcc.ufmg.br ou login.dcc.ufmg.br), pois será neste ambiente que o TP será corrigido. Note que essas máquinas são acessíveis a todos os alunos do DCC com seu login e senha, podendo inclusive ser realizado acesso remoto via SSH. O aluno pode buscar informações no site do CRC (Centro de Recursos Computacionais) do DCC (https://www.crc.dcc.ufmg.br/).

Para facilitar o desenvolvimento vamos fornecer uma estrutura base de arquivos com Makefile já configurado. A pasta TP03-Template-CPP.zip, disponível para download na tarefa do Moodle, contém 2 arquivos: main.cpp e Makefile. O ponto de entrada do seu programa está no arquivo main.cpp. Fique à vontade para criar novos arquivos .hpp e .cpp conforme sua preferência. Para compilar seu programa basta executar o comando "make" no mesmo diretório que o Makefile está. Ao final deste comando, se a compilação for bem sucedida, será criado um arquivo executável chamado "tp03". Esse arquivo pode ser executado pela linha de comando usando "./tp03".

O arquivo da entrada deve ser passado ao seu programa como entrada padrão, através da linha de comando (e.g., $ ./tp03 < casoTeste01.txt) e gerar o resultado também na saída padrão (não gerar saída em arquivo).

Para avaliar automaticamente sua solução em todos os casos de teste disponibilizados, disponibilizamos o comando "make eval", que avalia seu algoritmo em todos os casos de teste disponibilizados.

Nota: o comando de avaliação foi testado apenas em ambientes Linux.

5 O que deve ser entregue

Deverá ser submetido um arquivo .zip contendo apenas uma pasta chamada tp2, esta pasta deverá conter: (i) Documentação em formato PDF e (ii) Implementação.

5.1 Documentação

A documentação deve ser sucinta e não ultrapassar 5 páginas. Você deve descrever cada solução do problema de maneira clara e precisa, detalhando e justificando os algoritmos e estruturas de dados utilizados. Para tal, artifícios como pseudo-códigos, exemplos ou diagramas podem ser úteis. Note que documentar uma solução não é o mesmo que documentar seu código. Não é necessário incluir trechos de código em sua documentação nem mostrar detalhes de sua implementação, exceto quando estes influenciem o seu algoritmo principal, o que se torna interessante. É essencial que a documentação contenha ao menos:

1. **Identificação:** Nome e Matrícula.
2. **Introdução:** Breve resumo do problema com suas palavras.
3. **Modelagem:** Detalhamento da implementação do problema usando programação dinâmica
4. **Análise de Complexidade e NP-Completude:** Análise de complexidade do problema e redução de um problema NP-Completo (dica: esse é um problema pseudo-polinomial, em que o seu tempo de execução é polinomial no valor numérico da entrada, mas é exponencial no comprimento da entrada – o número de bits necessários para representá-lo.)

Observações:

- Soluções muito lentas não serão consideradas, uma vez que terão complexidades muito acima da adequada para este problema.

5.2 Implementação

O código fonte submetido deve conter todos os arquivos fonte e o Makefile usado para compilar o projeto. Lembre que seu código deve ser **legível**, então **evite variáveis com nomes não descritivos** (int a, aa, aaa;) e lembre-se de **comentar seu código.** Já estamos fornecendo uma implementação base com os arquivos necessários, então indicamos que você só o altere se for necessário.

5.3 Atrasos

Trabalhos poderão ser entregues após o prazo estabelecido, porém sujeitos a uma penalização regida pela seguinte fórmula:

$$\Delta_p = \frac{2^{(d-1)}}{0.32}\% \tag{1}$$

Nesta fórmula d representa dias de atraso. Por exemplo, se a nota dada pelo corretor for 70 e você entregou o TP com 4 dias corridos de atraso, sua penalização será de $\Delta_p = 25\%$ e, portanto, a sua nota final será: $N_f = 70(1 - \Delta_p) = 52.2$. Note que a penalização é exponencial e 6 dias de atraso resultam em uma penalização de 100%.

6 Considerações finais

Assim como em todos os trabalhos desta disciplina, é estritamente proibida a cópia parcial ou integral de códigos, seja da internet ou de colegas. Utilizaremos o algoritmo MOSS para detecção de plágio em trabalhos, seja honesto. Você não aprende nada copiando código de terceiros nem pedindo a outra pessoa que faça o trabalho por você. Se a cópia for detectada, sua nota será zerada e os professores serão informados para que as devidas providências sejam tomadas.