MC558 - Análise de Algoritmos II
Lista de Exercícios: reduções

1. Sejam P_1 e P_2 dois problemas tais que $P_1 \propto_n P_2$ e suponha que P_1 tem cota inferior $\Omega(n \log n)$, onde n é um parâmetro que mede o tamanho da entrada do problema P_1. Quais das seguintes afirmações são verdadeiras? Justifique cuidadosamente as suas respostas.
   (a) $\Omega(n \log n)$ também é cota inferior para P_2.
   (b) Todo algoritmo que resolve P_1 também pode ser usado para resolver P_2.
   (c) Todo algoritmo que resolve P_2 também pode ser usado para resolver P_1.
   (d) O problema P_2 pode ser resolvido no pior caso em tempo $O(n \log n)$.

2. Sejam P_1 e P_2 dois problemas tais que um deles tenha cota inferior $\Omega(n^k)$, para algum $k > 1$, e o outro é solúvel em tempo $O(n \log n)$. Se P_1 é redutível a P_2 em tempo linear, diga qual é qual. O parâmetro n denota o tamanho da entrada dos dois problemas.

3. A idéia deste exercício é entender porque em geral é fácil reduzir (Turing) um problema de otimização para sua versão de decisão. Considere o problema da Árvore Geradora Mínima (`AGM`):

   > Dado um grafo não orientado ponderado (G, w), encontrar uma árvore geradora de peso mínimo de G.

   Considere agora a versão de decisão (`AGMdec`) do mesmo:

   > Dados um grafo ponderado (G, w) com pesos inteiros não-negativos nas arestas e um número $W > 0$, decidir se existe uma árvore geradora em G de peso menor ou igual a W (a resposta é `SIM` ou `NÃO`.)

   Suponha que n e m denotam o número de vértices e arestas de G e que os valores absolutos dos pesoss são limitados por um valor C. Suponha que você tem à disposição um algoritmo polinomial `Alg` que dados (G, w) e W resolve `AGMdec`, isto é, responde `SIM` ou `NÃO` corretamente.

   (a) Mostre, usando `Alg`, como determinar o **peso** de uma árvore geradora mínima de G em tempo polinomial em n, m e $\log C$.

   (b) Mostre, usando `Alg` e o peso de uma árvore geradora mínima determinada em (a), como **encontrar** uma árvore geradora mínima de G em tempo polinomial.

   (c) Conclua que `AGM` $\propto_{\text{poli}}$ `AGMdec`.

4. Diz-se que um ponto $p = (x_p, y_p)$ do plano **domina** um outro ponto <u>distinto</u> $q = (x_q, y_q)$ do plano se $x_p \geq x_q$ e $y_p \geq y_q$. Um ponto p é **maximal** em relação a um conjunto de pontos P se $p \in P$ e nenhum outro ponto de $P - \{p\}$ domina p (por favor, note que isto **não** significa que p domina todos os pontos de $P - \{p\}$).

   Projete um algoritmo de complexidade $O(n \log n)$ para encontrar todos os pontos maximais de um conjunto P de n pontos no plano.

   Exemplo: suponha que $P = \{(0, n-1), (1, n-2), \ldots, (n-2, 1), (n-1, 0)\}$. Quais sãos os pontos maximais de P?

5. Considere o seguinte problema: dados n intervalos (fechados) na reta real, definidos pelos seus pontos de início e de fim, projete um algoritmo que lista todos os intervalos que estão contidos dentro de pelo menos um dos outros intervalos passados na entrada. O seu algoritmo deve ter complexidade $O(n \log n)$.

6. Denote por `MAXIMAL` o problema do exercício 4 e por `INTERVAL` o problema do exercício 5. Encontre uma redução de complexidade linear de `MAXIMAL` para `INTERVAL`.

   É possível usar o algoritmo desenvolvido no exercício anterior e a redução proposta por você para projetar um algoritmo para `MAXIMAL`? Em caso afirmativo, como se compara a complexidade deste algoritmo com a do algoritmo do exercício 4?

7. Encontre uma redução de complexidade linear de `INTERVAL` para `MAXIMAL`.

8. Usando o conceito de dominância entre pontos do exercício 4, pode-se definir os **Pareto**s de um dado conjunto não vazio de pontos $P = \{p_1, \ldots, p_n\}$ no plano da seguinte forma:

   (i) o **Pareto** 1 de P, denotado por P_1, é o conjunto de pontos maximais de P;

   (ii) para $i \geq 2$, o **Pareto** i de P, denotado por P_i, é o conjunto de pontos maximais de $P \setminus (P_1 \cup \ldots \cup P_{i-1})$.

   Chamemos de **índice de Pareto** de P o maior valor de i para o qual o **Pareto** i é não vazio. Denotemos por $i(P)$ este valor.

   Dado um conjunto P como acima, considere o problema de encontrar os $i(P)$ primeiros Paretos de P. Projete um algoritmo $O(n \log n)$ para este problema.

9. Encontre uma redução polinomial do problema de ordenação de um vetor de n elementos para o problema `PARETO` do exercício anterior. A sua redução deve ter complexidade $O(n)$.

   Pergunta-se: esta redução prova que o algoritmo do exercício anterior é ótimo (do ponto de vista de complexidade computacional)? Justifique sua resposta.

10. Uma matriz quadrada é dita ser **triangular inferior** (**superior**) se todos os seus elementos não nulos estiverem na diagonal principal ou abaixo (acima) dela.

    Seja `MMIS` o problema de multiplicar uma matriz triangular inferior por uma matriz triangular superior e `MMQ` o problema de multiplicar duas matrizes quadradas arbitrárias.

    Seja $T(n)$ a complexidade de um algoritmo ótimo para resolver `MMIS` quando as matrizes passadas na entrada tem ordem n. Suponha que $T(cn) \in O(T(n))$ para toda constante $c > 0$.

    Mostre que `MMIS` é pelo menos tão difícil quanto `MMQ` no sentido de que ambos têm a mesma cota inferior (supondo o modelo de computação usual).

11. Seja S um conjunto de n pontos distintos do plano. Seja $G = (V, E)$ o grafo completo onde cada vértice corresponde a um ponto de S (ou seja, $V = S$). Além disso, suponha que para cada aresta (u, v) em E está associado um peso $\omega(u, v)$ igual à distância euclidiana entre os pontos u e v em S. O **Problema da Árvore Geradora Mínima Geométrica** consiste em encontrar uma árvore geradora mínima em um grafo (G, ω) como descrito. Suponha que a entrada é um conjunto S e a resposta é um par (s, π) onde $s \in V$ e π é o vetor de predecessores da AGM de (G, ω) com raiz s (que você **não** pode escolher).

Mostre que o Problema da Árvore Geradora Mínima Geométrica tem cota inferior $\Omega(n \log n)$.

12. Considere dados um grafo orientado $G = (V, E)$, um vértice especial s em V e custos $c(v) \geq 0$ para cada vértice v em V. Suponha que o custo de um caminho orientado representado pela seqüência de vértices $(s, x_1, x_2, \ldots, x_k, v)$ seja dado por $\sum_{i=1}^{k} c(x_i)$, ou seja, o custo de um caminho é a soma do custo dos seus vértices internos. Assim, se (s, v) é uma aresta do grafo, o custo deste caminho é *zero*.

    Deseja-se encontrar um caminho de custo mínimo de s para todos os vértices de $V \setminus \{s\}$.

    Encontre uma redução polinomial deste problema ao problema do caminho mínimo usual (com custos nas arestas) visto em aula.

13. Dados vértices s e t em um **grafo orientado** G, dizemos que uma coleção $P_1, P_2, \ldots, P_k$ de caminhos com início em s e final em t é **aresta-disjunta** se para quaisquer caminhos P_i e P_j, P_i e P_j não têm arestas em comum. O problema dos caminhos aresta-disjuntos (`CAD`) consiste em dados um grafo G e vértices s e t em G encontrar uma coleção aresta-disjunta máxima de caminhos de s a t. Mostre que `CAD` $\propto_{\text{poli}}$ FM.

    Considere agora a versão do `CAD`, chamada `NCAD`, em que o grafo G de entrada é não-orientado. Mostre que `NCAD` $\propto_{\text{poli}}$ `CAD`.

14. (Difícil) Seja $G = (V, E)$ um grafo não orientado tal que pra cada vértice v do grafo temos associado uma função $b(v) \leq grau(v)$. Um b-emparelhamento é um subconjunto de E tal que cada vértice v não tem mais do que $b(v)$ arestas incidentes a ele. Em outras palavras, um b-emparelhamento é um subgrafo gerador de G onde cada vértice v tem grau menor ou igual a $b(v)$. Um b-emparelhamento máximo é aquele que tem o maior número de arestas possível. Reduza o problema de se achar um b-emparelhamento máximo ao problema de se achar um emparelhamento máximo em um grafo.