Análise de Algoritmos e Estruturas de Dados
Centro de Matemática, Computação e Cognição
Universidade Federal do ABC
Profa. Carla Negri Lintzmayer

Lista 5: Algoritmos gulosos e programação dinâmica

Instruções importantes

(1) Em qualquer exercício que peça para você fornecer um algoritmo/solução para um problema, a menos que explicitamente dito o contrário, você deve:

- descrever em palavras qual é a ideia do seu algoritmo (**obrigatório**);
- escrever um pseudocódigo (opcional, se a descrição do item anterior ficou clara o suficiente);
- provar ou fornecer um argumento intuitivo para sua corretude (**obrigatório**). É claro que, se o exercício explicitamente pede uma prova, então um argumento intuitivo não será o suficiente.
  - Observe que em problemas de otimização, duas coisas são importantes:
    - argumentar que seu algoritmo devolve sempre uma solução viável;
    - argumentar se o algoritmo é ótimo ou não.
- analisar o tempo do seu algoritmo (**obrigatório**).

O não cumprimento dos itens acima implica que o exercício está incorreto e o mesmo será desconsiderado.

(2) Você pode utilizar qualquer algoritmo visto em aula sem reescrevê-lo ou provar sua corretude novamente, mesmo que o algoritmo necessite de alguma pequena alteração.

- Descrever clara e sucintamente o que o algoritmo recebe, o que ele devolve e qual o seu consumo de tempo.
- Se houver alterações, descreva-as, indicando, por exemplo, quais linhas estão sendo alteradas/acrescentadas.

1. O problema da Mochila Inteira é semelhante ao da Mochila Fracionária: são dados n itens, cada item i tendo um valor v_i e um peso w_i, e é dada uma capacidade W da mochila; queremos selecionar um subconjunto $S \subseteq \{1, 2, \ldots, n\}$ dos itens cujo peso $\sum_{i \in S} w_i$ seja no máximo W e cujo valor $\sum_{i \in S} v_i$ seja máximo. Perceba que o problema não permite pegar frações dos itens: só é permitido adicionar à mochila itens por inteiro. A estratégia gulosa de escolher sempre o item com maior razão *valor/peso* encontra uma solução ótima para o problema da Mochila Inteira? Justifique.

2. Considere o problema de escalonamento de tarefas compatíveis: dado um conjunto $T = \{t_1, \ldots, t_n\}$ com n tarefas onde cada $t_i \in T$ tem um tempo inicial s_i e um tempo final f_i, encontre o maior subconjunto de tarefas mutuamente compatíveis. Considere o seguinte algoritmo guloso: enquanto houver tarefas compatíveis com as já escolhidas, escolha a tarefa que começa por último (com maior valor s_i) que seja compatível com as tarefas já escolhidas. Prove que essa estratégia é ótima.

3. Sejam n e m dois inteiros positivos. Seja $\mathcal{C} = \{C_1, \ldots, C_m\}$ uma coleção com m conjuntos de números entre 1 e n, isto é, $C_i \subseteq \{1, \ldots, n\}$ para cada $1 \leq i \leq m$.

   Uma *cobertura* $\mathcal{S}$ é um subconjunto de $\mathcal{C}$ cuja união é igual a $\{1, \ldots, n\}$, isto é, $\mathcal{S} \subseteq \mathcal{C}$ é uma cobertura se $\bigcup_{C_i \in \mathcal{S}} C_i = \{1, \ldots, n\}$. O tamanho de uma cobertura $\mathcal{S}$ é sua cardinalidade, $|\mathcal{S}|$. Uma *cobertura mínima* é uma cobertura de menor tamanho possível.

   O problema da *Cobertura de Conjuntos* consiste em, dados n, m e $\mathcal{C} = \{C_1, \ldots, C_m\}$, com cada $C_i \subseteq \{1, \ldots, n\}$, encontrar uma cobertura de cardinalidade mínima. Descreva um algoritmo guloso para o problema da cobertura de conjuntos. Você pode assumir que $\bigcup_{i=1}^{m} C_i = \{1, \ldots, n\}$, isto é, existe ao menos uma solução viável na instância de entrada. Prove que seu algoritmo não é ótimo.

4. Nesta questão, considere o problema de fazer troco para n centavos usando o menor número total de moedas. Você pode assumir que existe um número infinito de moedas de cada valor.

   (a) Forneça um algoritmo guloso ótimo para fazer troco tendo disponíveis moedas de 1, 5, 10, 25 e 50 centavos. *Dica: para provar que seu algoritmo é ótimo, assuma que ele não é. Qual é a diferença de uma solução ótima para a solução do seu algoritmo?*

   (b) O algoritmo que você deu acima continua ótimo se o conjunto de moedas disponíveis for 1, 7, 10 e 16?

   (c) Forneça um algoritmo de programação dinâmica que é ótimo para fazer troco tendo disponíveis moedas de valores $m_1, m_2, \ldots, m_t$ centavos, para $t \geq 1$, onde algum $m_k = 1$.

5. Seja G um grafo e w uma função de peso sobre as arestas de G. Suponha que todos os pesos das arestas são números inteiros entre 1 e $|V(G)|$, isto é, $w \colon E(G) \to \{1, \ldots, |V(G)|\}$. Como pode ficar o tempo de execução do Kruskal (versão *union-find*) nesse caso?

6. Mostre que se todos os pesos das arestas são distintos, então o grafo só tem uma árvore geradora mínima. *Dica:* por contradição, suponha que o grafo tem duas árvores geradoras mínimas diferentes.

7. O diâmetro de um grafo é o máximo das distâncias entre quaisquer dois vértices. Escreva um algoritmo que usa o algoritmo de Dijkstra para calcular o diâmetro de um grafo.

8. Considere o grafo G com $V(G) = \{v_1, v_2, \ldots, v_{16}\}$ e arestas v_iv_{i+1} para $1 \leq i \leq 15$, v_iv_{i+2} para $1 \leq i \leq 14$ e v_iv_{2i} para $3 \leq i \leq 8$. Considere a função w de pesos em $E(G)$ tal que $w(v_iv_{i+1}) = i$ para $1 \leq i \leq 15$, $w(v_iv_{i+2}) = 15 - i$ para $1 \leq i \leq 14$ e $w(v_iv_{2i}) = i + 1$ para $3 \leq i \leq 8$. *Dica:* desenhe os vértices ao redor de um círculo para facilitar a visualização.

   Execute o algoritmo de Kruskal nesse grafo e apresente a sequência de arestas escolhidas pelo algoritmo. Execute o algoritmo de Dijkstra no mesmo grafo a partir do vértice v_7 e apresente a distância dele para todos os outros vértices.

9. Execute o algoritmo de programação dinâmica para o problema da Mochila Inteira e mostre a matriz final preenchida pelo mesmo sobre a seguinte entrada: $W = 10$, $v_1 = 15$, $w_1 = 3$, $v_2 = 12$, $w_2 = 4$, $v_3 = 17$, $w_3 = 2$, $v_4 = 18$, $w_4 = 5$, $v_5 = 15$, $w_5 = 2$, $v_6 = 10$, $w_6 = 3$, $v_7 = 15$, $w_7 = 1$. Explique como você preencheu a entrada da linha referente ao quarto item e à capacidade 9.

10. Forneça um algoritmo que recebe a matriz totalmente preenchida pelo algoritmo de programação dinâmica para o problema do Alinhamento de Sequências e constrói um alinhamento para as sequências da entrada. Um alinhamento pode ser representado por dois vetores, um para cada sequência. Não é necessário provar a corretude deste algoritmo.

11. Um *grafo caminho* G é um grafo que consiste apenas de um caminho, isto é, $V(G) = \{v_1, v_2, \ldots, v_n\}$ e $E(G) = \{v_iv_{i+1} \colon 1 \leq i < n\}$ e seja $w \colon V(G) \to \mathbb{R}$ uma função de peso sobre **os vértices** de G. Assim, para qualquer conjunto $X \subseteq V(G)$ de vértices, o peso de X, $w(X)$, é naturalmente definido como $w(X) = \sum_{v \in X} w(v)$.

    Um conjunto S de vértices de um grafo qualquer é *independente* se para quaisquer dois vértices $u, v \in S$, não existe aresta uv no grafo.

    O problema do *Conjunto Independente de Peso Máximo* consiste em receber um grafo caminho G e uma função w de peso nos vértices e o objetivo é encontrar um subconjunto S de vértices tal que S é um conjunto independente e $\sum_{v \in S} w(v)$ é máximo. Responda:

    (a) Mostre que o algoritmo a seguir é guloso e não é ótimo.

    ```
    seja S = ∅
    Enquanto V(G) ≠ ∅ faça
        escolha v ∈ V(G) com peso w(v) máximo
        S = S ∪ {v}
        remova v e seus vizinhos de G
    Devolve S
    ```

    (b) Mostre que o algoritmo a seguir também é guloso e não é ótimo.

    ```
    seja S1 = {v_i : i é ímpar}
    seja S2 = {v_i : i é par}
    Devolve o conjunto de maior peso dentre S1 e S2
    ```

    (c) Forneça um algoritmo de programação dinâmica que resolve esse problema otimamente.

12. Temos n estações ao longo de um rio, numeradas de 1 a n na direção da correnteza. Você pode alugar um barco em qualquer estação i, descer o rio até uma estação $k > i$, devolver o barco e pagar uma taxa $c(i, k)$ pelo passeio. É possível que $c(i, k)$ seja maior que $c(i, j) + c(j, k)$, sendo j uma estação intermediária entre i e k. Nesse caso, é mais barato alugar um barco de i a j e depois outro de j até k. Dê um algoritmo eficiente que calcule o custo mínimo de uma viagem de 1 a n. Em função de n, quanto tempo o seu algoritmo consome?

13. Um *emparelhamento* em um grafo G é um conjunto de arestas $F \subseteq E(G)$ tal que nenhum par de arestas em F possui vértices em comum. Note que uma aresta $e \in E(G)$ sozinha já é um emparelhamento. É interessante, portanto, encontrar um emparelhamento de tamanho maior possível. Descreva um algoritmo guloso que devolva um emparelhamento de um grafo qualquer. Seu algoritmo é ótimo?

14. Faça um algoritmo de programação dinâmica para calcular $\binom{n}{k}$ sem usar a fórmula fechada para isso. Por definição, $\binom{n}{k} = \binom{n-1}{k-1} + \binom{n-1}{k}$ para $1 \leq k < n$ e $\binom{n}{n} = \binom{n}{0} = 1$.

1 Questões retiradas do Enade e Poscomp

QUESTÃO 22

Um país utiliza moedas de 1, 5, 10, 25 e 50 centavos. Um programador desenvolveu o método a seguir, que implementa a estratégia gulosa para o problema do troco mínimo. Esse método recebe como parâmetro um valor inteiro, em centavos, e retorna um *array* no qual cada posição indica a quantidade de moedas de cada valor.

```
public static int[] troco(int valor){
    int[] moedas = new int[5];

    moedas[4] = valor / 50;
    valor = valor % 50;
    moedas[3] = valor / 25;
    valor = valor % 25;
    moedas[2] = valor / 10;
    valor = valor % 10;
    moedas[1] = valor / 5;
    valor = valor % 5;
    moedas[0] = valor;
    return(moedas);
}
```

Considerando o método apresentado, avalie as asserções a seguir e a relação proposta entre elas.

I. O método guloso encontra o menor número de moedas para o valor de entrada, considerando as moedas do país.

PORQUE

II. Métodos gulosos sempre encontram a solução global ótima.

A respeito dessas asserções, assinale a opção correta.

A As asserções I e II são proposições verdadeiras, e a II é uma justificativa correta da I.

B As asserções I e II são proposições verdadeiras, mas a II não é uma justificativa correta da I.

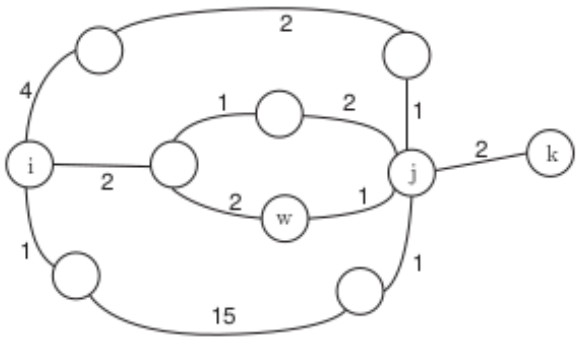
C A asserção I é uma proposição verdadeira, e a II é uma proposição falsa.

D A asserção I é uma proposição falsa, e a II é uma proposição verdadeira.

E As asserções I e II são proposições falsas.

QUESTÃO 24

A figura a seguir exibe um grafo que representa um mapa rodoviário, no qual os vértices representam cidades e as arestas representam vias. Os pesos indicam o tempo atual de deslocamento entre duas cidades.



Considerando que os tempos de ida e volta são iguais para qualquer via, avalie as afirmações a seguir acerca desse grafo.

I. Dado o vértice de origem i, o algoritmo de Dijkstra encontra o menor tempo de deslocamento entre a cidade i e todas as demais cidades do grafo.

II. Uma árvore geradora de custo mínimo gerada pelo algoritmo de Kruskal contém um caminho de custo mínimo cuja origem é i e cujo destino é k.

III. Se um caminho de custo mínimo entre os vértices i e k contém o vértice w, então o subcaminho de origem w e destino k deve também ser mínimo.

É correto o que se afirma em

A) I, apenas.
B) II, apenas.
C) I e III, apenas.
D) II e III, apenas.
E) I, II e III.

QUESTÃO 36 – Um mapa rodoviário é modelado como um grafo em que os vértices representam interseções. As arestas representam segmentos de estrada entre interseções. O peso de cada aresta representa a distância entre interseções. Agora, considere que um motorista deseja obter o caminho mais curto entre duas cidades. Dado um mapa contendo as distâncias entre cada par de interseções adjacentes, como obter o caminho mais curto entre duas cidades?

A) Caminho mais curto com destino único.
B) Caminho gerador mínimo de origem única.
C) Caminho mais curto com origem única.
D) Caminho mais curto entre todos os pares de vértices.
E) Caminho gerador mínimo de origem múltipla.