Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Projeto e Análise de Algoritmos (2026/1)
Segunda Avaliação - Parte objetiva

7 de julho de 2026

Nome: _____

Matrícula: _____

- **Instruções**:
  - Esta avaliação é **individual** e **sem consulta**;
  - Selecione apenas 1 item por questão.
    - Questões com mais de 1 item selecionado serão anuladas;
  - Tempo de duração: 10 minutos.

Exercício 1. *A* ***propriedade da escolha gulosa*** *(*greedy choice property*) que, junto com a subestrutura ótima, é condição suficiente para a garantir a correção de um algoritmo guloso, afirma que:*

(a) Para todo problema com subestrutura ótima, a escolha localmente ótima em cada etapa pertence necessariamente a alguma solução globalmente ótima.

(b) É possível chegar a uma solução globalmente ótima fazendo escolhas localmente ótimas a cada etapa, sem reconsiderar decisões anteriores nem depender das soluções de subproblemas futuros.

(c) A escolha ótima em cada etapa é única, de modo que o algoritmo guloso produz uma única solução ótima independentemente da ordem em que os elementos são processados.

(d) O problema admite sobreposição de subproblemas, o que justifica armazenar as escolhas parciais do algoritmo guloso em uma tabela.

(e) A solução globalmente ótima é sempre construída selecionando o elemento de maior valor (ou menor custo) disponível, sem nenhuma restrição sobre a estrutura do problema.

Gabarito

Alternativa **(b)**.

A propriedade da escolha gulosa afirma que é possível construir uma solução globalmente ótima realizando escolhas localmente ótimas de forma irrevogável, sem necessidade de examinar os subproblemas futuros nem reconsiderar o que foi escolhido. Essa propriedade é *mais restritiva* do que a subestrutura ótima: um problema pode ter subestrutura ótima sem ter escolha gulosa (ex.: mochila 0/1), e por isso requer programação dinâmica em vez de um algoritmo guloso.

As demais alternativas são incorretas: (a) descreve a subestrutura ótima (não a escolha gulosa) e usa "necessariamente", o que é falso sem a propriedade adicional; (c) adiciona unicidade que não faz parte da definição — pode haver múltiplas soluções ótimas; (d) descreve a sobreposição de subproblemas, conceito de programação dinâmica, não de algoritmos gulosos; (e) é uma caracterização informal incorreta que não captura a essência da propriedade.

Exercício 2. *Considere o sistema monetário com moedas de valores* $\{1, 4, 5\}$ *e o objetivo de representar o valor* 8 *com o* ***menor número possível*** *de moedas. A estratégia gulosa seleciona, a cada passo, a maior moeda cujo valor não exceda o restante a ser representado. Assinale a alternativa* ***correta****.*

(a) A estratégia gulosa retorna $\{5, 1, 1, 1\}$ *(4 moedas), que é a solução ótima; portanto, a estratégia gulosa é sempre ótima para qualquer sistema monetário com denominação unitária.*

(b) A estratégia gulosa retorna $\{5, 1, 1, 1\}$ *(4 moedas), enquanto a solução ótima é* $\{4, 4\}$ *(2 moedas); logo, a estratégia gulosa falha neste sistema pois ele não possui a propriedade da escolha gulosa.*

(c) A estratégia gulosa retorna $\{4, 4\}$ *(2 moedas), que é a solução ótima; portanto, a estratégia gulosa coincide com a programação dinâmica neste exemplo.*

(d) A estratégia gulosa falha porque o sistema $\{1, 4, 5\}$ *não possui a propriedade da subestrutura ótima, impossibilitando qualquer abordagem sistemática.*

(e) A estratégia gulosa retorna $\{4, 1, 1, 1, 1\}$ *(5 moedas), enquanto a solução ótima é* $\{5, 1, 1, 1\}$ *(4 moedas).*

Gabarito

Alternativa **(b)**.

Aplicando a estratégia gulosa ao valor 8 com moedas $\{1, 4, 5\}$: seleciona-se 5 (restam 3), depois 1 três vezes, obtendo $\{5, 1, 1, 1\}$ com 4 moedas. A solução ótima é $\{4, 4\}$ com apenas 2 moedas. A falha ocorre porque este sistema monetário não satisfaz a *propriedade da escolha gulosa*: a escolha localmente ótima (maior moeda disponível) não pode ser estendida a uma solução globalmente ótima. A propriedade da subestrutura ótima existe (o subproblema restante após cada escolha é da mesma natureza), mas ela sozinha não é suficiente para garantir a corretude da estratégia gulosa, daí a necessidade de programação dinâmica para este sistema.

Exercício 3. *Na teoria da NP-completude, um problema de decisão L é dito* ***NP-difícil*** *quando:*

(a) $L \in \text{NP}$ *e* L *pode ser resolvido em tempo polinomial por algum algoritmo não-determinístico.*

(b) $L \in \text{NP}$ *e* L *pertence à classe NPC, satisfazendo portanto ambas as condições da definição de NP-completude.*

(c) $L \notin \text{NP}$ *e não existe algoritmo determinístico polinomial para* L*.*

(d) Todo problema em NP *é redutível polinomialmente a* L*, mesmo que* L *não pertença necessariamente a* NP*.*

(e) L *possui um certificado verificável em tempo exponencial, mas não em tempo polinomial.*

Gabarito

Alternativa **(d)**.

Um problema L é NP-difícil se satisfaz a segunda condição da definição de NP-completude, ou seja, $L' \leq_p L$ para todo $L' \in \text{NP}$, mas *não necessariamente* a primeira ($L \in \text{NP}$). Em contraste, um problema NP-completo exige ambas: $L \in \text{NP}$ e NP-dificuldade.

As demais alternativas são incorretas: (a) descreve a classe NP, não NP-dificuldade; (b) confunde NP-difícil com NP-completo; (c) é uma caracterização informal equivocada (existem problemas NP-difíceis fora de NP, mas a definição formal é via reduções); (e) confunde com a noção de verificador: problemas em NP possuem verificadores polinomiais, não exponenciais.