Universidade Federal de Ouro Preto
PCC104 - Projeto e Análise de Algoritmos
Classes de Problemas e Programação Dinâmica

Prof. Rodrigo Silva

November 7, 2023

1 Classes de Problemas

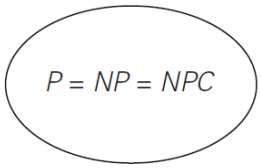
1.1 Leitura Recomendada

- Seção 11.3 - *Introduction to the Design and Analysis of Algorithms (3rd Edition)* - Anany Levitin

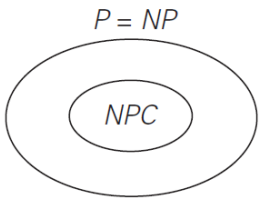
1.2 Atividades

1. O que significa dizer que um algoritmo resolve um problema em tempo polinomial?
2. Que tipo de problemas considera-se tratável?
3. Que tipo de problema considera-se intratável?
4. Em ciência da computação, o que é o conjunto ou classe de problemas P?
5. Como podemos provar que um problema pertence à classe P?
6. O que é um problema decidível? E um problema indecidível?
7. De forma geral, o que é um algoritmo determinístico?
8. De forma geral, o que é um algoritmo não determinístico?
9. Em ciência da computação, o que é o conjunto ou classe de problemas NP?
10. O que é um algoritmo polinomial não determinístico?
11. Explique por quê $P \subseteq NP$?
12. Por quê saber se $P = NP$ é interessante?
13. Como provamos que um problema é NP-Completo?
14. Como provamos que um problema é NP-Completo quando já conhecemos algum problema NP-Completo?
15. O que significaria resolver ao problema NP-Completo em $O(n^5)$?
16. Um algoritmo que faz um número polinomial de chamadas a um procedimento que executa em tempo polinomial pode ter complexidade exponencial? Explique.
17. Qual dos diagramas abaixo não contradiz o estado corrente do nosso conhecimento sobre as classes de problemas P, NP e NP-Completo.
18. Mostre que o Problema do Conjunto independente é um problema NP-Completo utilizando a redução entre problemas, considerando o 3-SAT como problema base.

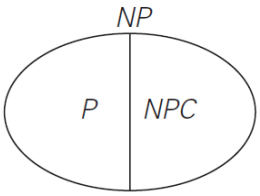
a.



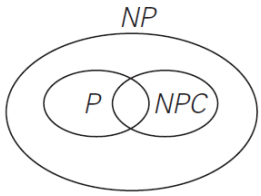
b.



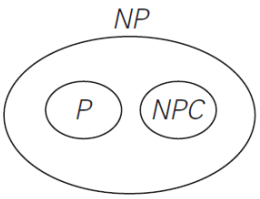
c.



d.



e.



2 Programação Dinâmica

2.1 Leitura Recomendada

- Capítulo 8 - *Introduction to the Design and Analysis of Algorithms (3rd Edition)* - Anany Levitin
- Livro - *Introdução à programação* - Alan de Freitas (disponível em `http://www.decom.ufop.br/alan/bcc702/livrocpp.pdf`)
- Livro - *Problem Solving with Algorithms and Data Structures using C++* (disponível em: `https://runestone.academy/runestone/books/published/cppds/index.html#`)

2.2 Atividades

1. Implemente um algoritmo para o cálculo do n-ésimo número de Fibonacci sem utilizar programação dinâmica.
2. Implemente um algoritmo para o cálculo do n-ésimo número de Fibonacci utilização programação dinâmica.
3. Implemente um algoritmo para o problema do troco (*Change-making problem* (Seção 8.1)) utilizando programação dinâmica.
4. Implemente um algoritmo para o problema de coleta de moedas (*Coin-collecting problem* (Seção 8.1)) utilizando programação dinâmica.
5. Implemente um algoritmo para o problema de coleta de moedas (*Coin-collecting problem* (Seção 8.1)) sem utilizar programação dinâmica.

6. Implemente o algoritmo baseado em função de memória (*memory function*) para solução do problema da mochila (*knapsack problem*).

Para cada implementação, apresentar a análise de complexidade de tempo do algoritmo. Esta análise deverá conter:

- Expressão matemática que define o custo do algoritmo (relação de recorrência para recursivos ou somatórios para iterativos)
- Uma reflexão sobre melhor caso, pior caso e caso médio.
- Cálculo da função de custo (quando possível, utilizar o teorema mestre para verificar o cálculo).
- Indicação da classe de eficiência (O ou Θ). A indicação da classe, deve ser justificada. Você pode provar pela definição, pelo limite, teorema mestre ou utilizar os resultados demonstrados em aula.