Aula 08 – Classes de Complexidade Adicionais

Prof. Eduardo Zambon

Departamento de Informática (DI)
Centro Tecnológico (CT)
Universidade Federal do Espírito Santo (Ufes)

Algoritmos e Fundamentos da Teoria de Computação (ToCE)
Engenharia de Computação

Introdução

- **Aula anterior**: Problemas NP-complete clássicos e suas reduções.
- **Estes *slides***: Quais são as demais classes existentes (mais comuns)?
- **Objetivos**: Apresentar o conceito de complexidade de espaço e a classe $\mathcal{P}$-Space.

Referências

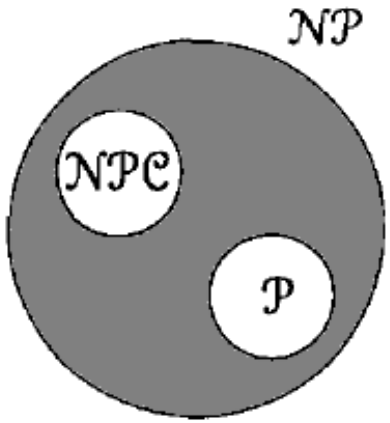
Chapter 17 – Additional Complexity Classes
T. Sudkamp
Chapter 8 – Space Complexity
M. Sipser
Chapter 7 – Summary
A. Maheshwari

Classes de Complexidade Derivadas

- Classe $\mathcal{P}$: linguagens decidíveis em tempo polinomial determinístico.
- Classe $\mathcal{NP}$: linguagens decidíveis em tempo polinomial não-determinístico.
- $\mathcal{P} = \mathcal{NP}$?: pergunta mais importante da computação teórica.
- Classes $\mathcal{P}$ e $\mathcal{NPC}$ são subconjuntos não vazios de $\mathcal{NP}$.
- Qual é a relação entre essas duas classes?
- Se $\mathcal{P}$ e $\mathcal{NPC}$ possuem uma interseção $\Rightarrow \mathcal{P} = \mathcal{NP}$.
- Se $\mathcal{P} \neq \mathcal{NP} \Rightarrow \mathcal{P}$ e $\mathcal{NPC}$ são disjuntos.

Classes de Complexidade Derivadas

Inclusões das classes $\mathcal{P}$ e $\mathcal{NPC}$ se $\mathcal{P} \neq \mathcal{NP}$:



- Pergunta imediata: existem linguagens na área cinza?
- Outra forma de se perguntar a mesma coisa: (assumindo que $\mathcal{P} \cap \mathcal{NPC} = \emptyset$,) $\mathcal{P} \cup \mathcal{NPC} = \mathcal{NP}$?
- $\mathcal{NPI}$: classe de problemas NP-intermediate (em cinza).

Classes de Complexidade Derivadas

- Qualquer problema em $\mathcal{NPI}$ não é NP-hard.
- Por quê?
- $\Rightarrow$ Porque se fosse então o problema seria NP-complete.
- A conclusão é que os problemas em $\mathcal{NPI}$ não são considerados tão difíceis como os em $\mathcal{NPC}$.
- Mas esse problemas também não estão em $\mathcal{P}$ e portanto podem ser vistos como mais difíceis do que os em $\mathcal{P}$.
- Isso justifica o nome *intermediate*.

Teorema 17.1.1 (Ladner, 1975)

Se $\mathcal{P} \neq \mathcal{NP}$, então $\mathcal{NPI}$ não é vazio.

Problemas candidatos a estarem em $\mathcal{NPI}$: fatoração de inteiros, isomorfismo de grafos.

Classes de Complexidade Derivadas

- $L \subseteq \Sigma^*$: linguagem sobre um alfabeto Σ.
- $\overline{L} = \Sigma^* - L$: complemento da linguagem L.
- Uma classe $\mathcal{C}$ é fechada sob complementação se $L \in \mathcal{C} \Rightarrow \overline{L} \in \mathcal{C}$.
- Classe $\mathcal{P}$ é fechada sob complementação.
- Qualquer DTM que decide L em tempo polinomial pode ser modificada para decidir $\overline{L}$ com o mesmo limite de tempo.
- Basta inverter os estados de aceite e rejeição da TM.

Classes de Complexidade Derivadas

- Em NTMs decidir sobre o complemento não é tão simples.
- Solução padrão por NTM: *guess-and-check*.
- SAT: uma fórmula é satisfatível?
  - Sim: existe pelo menos uma atribuição que satisfaz.
  - Não: não existe nenhuma atribuição que satisfaz.
  - NTM "adivinha" de forma não-determinística uma atribuição e testa se a fórmula é verdadeira.
  - Sabidamente em $\mathcal{NP}$.
- Complemento de SAT: uma fórmula é insatisfatível?
  - Sim: não existe nenhuma atribuição que satisfaz.
  - Não: existe pelo menos uma atribuição que satisfaz.
  - Aqui não basta "adivinhar" uma única atribuição para responder positivamente à pergunta.
  - Não se sabe se o complemento de SAT está em $\mathcal{NP}$.

Classes de Complexidade Derivadas

- co-$\mathcal{NP} = \{\overline{\mathsf{L}} \mid \mathsf{L} \in \mathcal{NP}\}$: classe dos complementos das linguagens $\mathcal{NP}$.
- Não se sabe se co-$\mathcal{NP} = \mathcal{NP}$.
- Responder não a essa pergunta é equivalente a responder que $\mathcal{P} \neq \mathcal{NP}$.

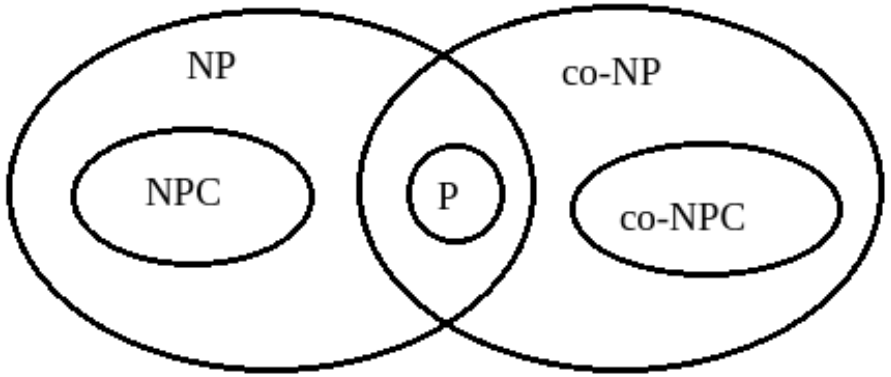
Teorema 17.1.2 (Sudkamp)

Se co-$\mathcal{NP} \neq \mathcal{NP}$, então $\mathcal{P} \neq \mathcal{NP}$.

- $\mathcal{P}$ é fechada sob complementação.
- Se $\mathcal{NP}$ não for, então as classes não podem ser iguais.

Classes de Complexidade Derivadas

- Teorema anterior provê outra forma de responder à questão se $\mathcal{P} = \mathcal{NP}$.
- Basta achar uma linguagem $\mathrm{L} \in \mathcal{NP}$ com $\overline{\mathrm{L}} \notin \mathcal{NP}$.
- Por outro lado, se co-$\mathcal{NP} = \mathcal{NP}$ nada se pode afirmar sobre a relação entre $\mathcal{P}$ e $\mathcal{NP}$.
- Inclusões se $\mathcal{P} \neq \mathcal{NP}$ e co-$\mathcal{NP} \neq \mathcal{NP}$:



Complexidade de Espaço

Definição 17.2.1 (Sudkamp)

A complexidade de espaço de uma TM M é a função $sc_M : \mathbf{N} \to \mathbf{N}$ aonde $sc_M(n)$ é o número máximo de posições da fita que são lidas por uma computação de M iniciada com uma string de entrada de tamanho n.

- Definição acima serve para qualquer tipo de máquina.
- No caso de máquinas multi-faixa ou multi-fita, basta somar todas as posições lidas.
- No caso de NTMs, o máximo é tomado sobre todas as possíveis computações para cada string de tamanho n.
- Como uma TM M pode parar antes de consumir toda a entrada, é possível que $sc_M(n) < n$.

Relações entre Complexidades de Tempo e Espaço

- A complexidade de tempo de uma TM pode ser usada para se obter um limite superior da complexidade de espaço.
- O número de posições que uma cabeça da TM pode ler durante uma computação é limitado pelo número de transições.

Teorema 17.3.1 (Sudkamp)

Seja M uma TM com complexidade de tempo $tc_M(n) = f(n)$. Então $sc_M(n) \leq f(n) + 1$.

- O número máximo de posições lidas ocorre quando M anda para a direita em todas as transições.
- Nesse caso, $sc_M(n) = f(n) + 1$.

Relações entre Complexidades de Tempo e Espaço

- Obter um limite para a complexidade de tempo a partir da complexidade de espaço é mais difícil.
- Tempo não pode ser reusado, mas espaço sim.
- Uma máquina pode ler um pedaço da fita várias vezes!

Teorema 17.3.2 (Sudkamp)

Seja M uma DTM com complexidade de espaço $sc_M(n) = s(n)$. Então $tc_M(n) \leq m \cdot s(n) \cdot t^{s(n)}$, onde m é o número de estados de M e t o tamanho do alfabeto da fita.

- Para uma entrada de tamanho n, a complexidade de espaço restringe a computação de M a no máximo $s(n)$ posições da fita.

Relações entre Complexidades de Tempo e Espaço

- Limitar a computação a um segmento finito da fita permite contar o número de configurações distintas de M.
- A fita pode ter qualquer dos t símbolos em cada posição, gerando $t^{s(n)}$ possibilidades.
- Existem $s(n) \cdot t^{s(n)}$ combinações de configurações da fita e posições da cabeça.
- Para qualquer uma das combinações acima, a TM pode estar em um dos m estados.
- Portanto, o total de configurações distintas é $m \cdot s(n) \cdot t^{s(n)}$.
- Uma repetição de uma configuração por uma DTM indica um *loop*.
- Como M precisa parar para todas as entradas (pré-condição para complexidade de tempo), a computação deve parar antes de $m \cdot s(n) \cdot t^{s(n)}$ transições.

$\mathcal{P}$-Space, $\mathcal{NP}$-Space e Teorema de Savitch

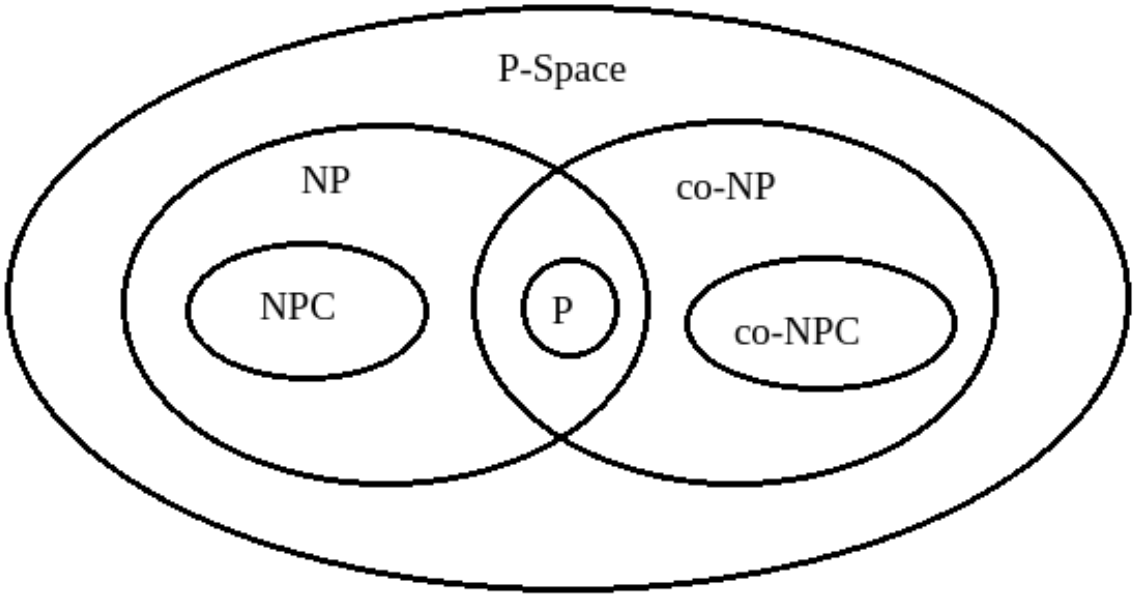
- Uma linguagem L é decidível em espaço polinomial se existe uma TM M que decide L com $sc_M \in O(n^r)$, onde r é um natural independente de n.
- $\mathcal{P}$-Space: classe de linguagens decidíveis em espaço polinomial por DTM.
- $\mathcal{NP}$-Space: classe de linguagens decidíveis em espaço polinomial por NTM.
- É imediato ver que $\mathcal{P}$-Space $\subseteq \mathcal{NP}$-Space.
- Mas e $\mathcal{NP}$-Space $\subseteq \mathcal{P}$-Space?
- Ao contrário da pergunta equivalente para tempo, sabe-se a resposta para a pergunta acima.
- Teorema a seguir mostra que $\mathcal{P}$-Space $= \mathcal{NP}$-Space.
- A diferença fundamental entre complexidade de tempo e espaço é que espaço pode ser reutilizado durante uma computação.

$\mathcal{P}$-Space, $\mathcal{NP}$-Space e Teorema de Savitch

Teorema 17.4.2 (Savitch, 1970)

Seja M uma NTM com $sc_M = s(n)$. Então L(M) é aceita por uma DTM M′ com $sc_{M'} \in O(s(n)^2)$.

Devido ao limite superior de tempo dado pelo Teorema 17.3.2, temos que $\mathcal{NP} \subseteq \mathcal{P}$-Space. Relação suposta entre as classes:



$\mathcal{P}$-Space, $\mathcal{NP}$-Space e Teorema de Savitch

Definição 17.5.1 (Sudkamp)

- Uma linguagem Q é dita $\mathcal{P}$-Space hard se para toda linguagem L $\in \mathcal{P}$-Space, L é redutível a Q em tempo polinomial.
- Uma linguagem $\mathcal{P}$-Space hard que também está em $\mathcal{P}$-Space é dita $\mathcal{P}$-Space complete.

É essencial notar que a redução da definição acima tem uma restrição de tempo e não de espaço. Isso leva a este teorema.

Teorema 17.5.2 (Sudkamp)

Seja Q uma linguagem $\mathcal{P}$-Space complete. Então

1. Se Q está em $\mathcal{P}$, $\mathcal{P}$-Space $= \mathcal{P}$.
2. Se Q está em $\mathcal{NP}$, $\mathcal{P}$-Space $= \mathcal{NP}$.

$\mathcal{P}$-Space, $\mathcal{NP}$-Space e Teorema de Savitch

REG

Input: Expressão regular α sobre alfabeto Σ.

Output: sim; se $\alpha \neq \Sigma^*$
não; caso contrário.

- O problema REG é $\mathcal{P}$-Space complete.
- Devido à inclusão de $\mathcal{NP}$ em $\mathcal{P}$-Space, todo problema $\mathcal{P}$-Space complete também é NP-hard.
- Portanto, REG é um exemplo de um problema NP-hard para o qual não existe uma solução em tempo polinomial não-determinístico.

$\mathcal{P}$-Space, $\mathcal{NP}$-Space e Teorema de Savitch

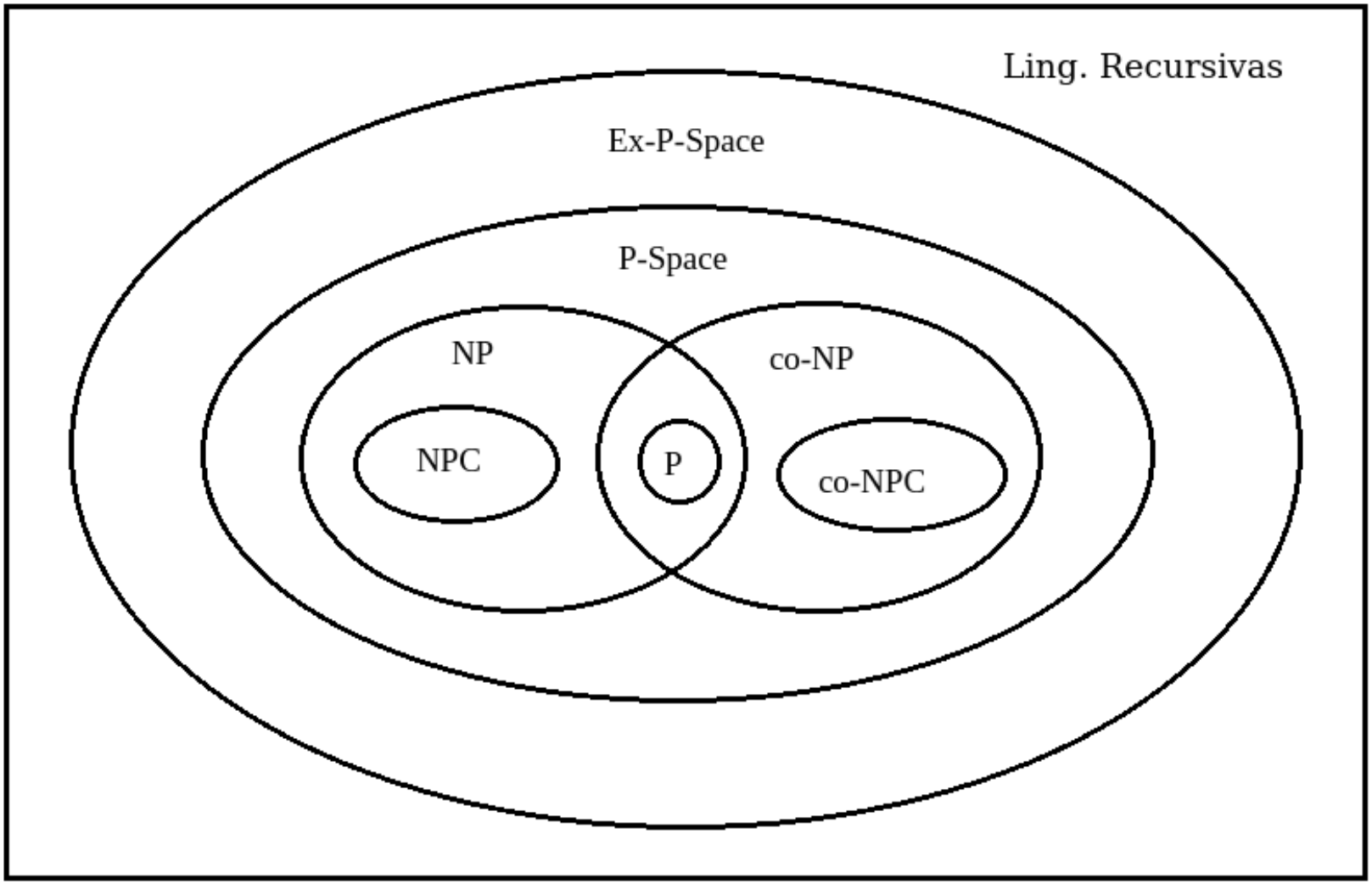
REG2

Input: Expressão regular com quadrados α sobre alfabeto Σ.

Output: sim; se $\alpha \neq \Sigma^*$
não; caso contrário.

- O problema REG2 não está em $\mathcal{P}$-Space.
- A linguagem L_{REG2} é decidível mas requer espaço que cresce exponencialmente com a entrada.
- Ex-$\mathcal{P}$-Space: classe de problemas decidíveis por TM em espaço $O(2^{p(n)})$, onde $p(n)$ é um polinômio.
- Ex-$\mathcal{P}$-Space é um superconjunto estrito de $\mathcal{P}$-Space, $\mathcal{NP}$ e $\mathcal{P}$.

Complexity Zoo



Muitas outras classes em:

`http://complexityzoo.uwaterloo.ca/`

Aula 08 – Classes de Complexidade Adicionais

Prof. Eduardo Zambon

Departamento de Informática (DI)
Centro Tecnológico (CT)
Universidade Federal do Espírito Santo (Ufes)

Algoritmos e Fundamentos da Teoria de Computação (ToCE)
Engenharia de Computação