Problemas NP-Completos

- aula sobre questão P vs. NP e Más Notícias :)
- começar c/ uma perspectiva sobre resolução de problemas nos primórdios da computação e visão do que a complexidade nos diz sobre o universo
- nos cursos de algoritmos vimos muitos problemas como ordenação e algs. eficientes p/ os mesmos, ainda que o espaço de soluções fosse exponencial ou fatorial
- será possível resolver c/ eficiência qualquer problema?
- vamos começar introduzindo o conceito de tratabilidade

Tratabilidade

- Intuição: problema é tratável se existem algs. eficientes para ele, e é intratável caso não existam

- Definição: tratável pode ser resolvido em $O(n^k)$ para k constante, intratável caso contrário

↳ não se bloqueam

- Questões: E se k for grande? Um alg. c/ pior caso exponencial $O(2^n)$ pode ser eficiente na prática?

- Em geral, definição captura bem a dificuldade do problema

Classificando Problemas

O que esperar ao tentar resolver um problema?

Será ele Natural?

- Caminho Mais Curto Sim
- Circuito Euleriano Sim
- Ordenação Sim
- Árvore geradora mínima Sim
- Fluxo máximo / Corte Mínimo Sim
- Conj. ind. em árvores Sim
- Mochila fracionária Sim

- Caminho Mais Longo Não sei
- Circuito Hamiltoriano Não sei
- Caixeiro Viajante Não sei
- Árvore de Steiner Não sei
- Corte Máximo Não sei
- Conj. ind. em grafos gerais NS
- Mochila binária Não sei

Classe P - problemas que podem ser resolvidos em tempo polinomial

- Nem todo problema está em P :c
- Existem problemas p/ os quais não existe alg.
  - Ex.: problema da parada (decidir se um programa termina)
    - ↳ indecidível (questão anterior à complexidade)
- Para outros não é conhecido alg. polinomial → não significa que não exista
  - Ex.: problema do caixeiro viajante (TSP)

- Como descrever uma classe de problemas como o TSP?

- Como descrever uma classe de problemas como o TSP?
  - Conseguimos resolver ele por força bruta → a questão é a eficiência
  - É fácil avaliar uma solução → contraste c/ checar jogada no xadrez (avalia árvore de decisão de tamanho exponencial

Classe NP - sol. tem tamanho polinomial
- e verificar se uma solução está correta leva tempo polinomial
  ↳ explicitar relação c/ ter sol. por força bruta

- Definições originais vem da computabilidade
  NP são prob. resolvíveis em tempo polinomial por um autômato não-determinístico (checa bifurcações em paralelo)

Situação dos problemas

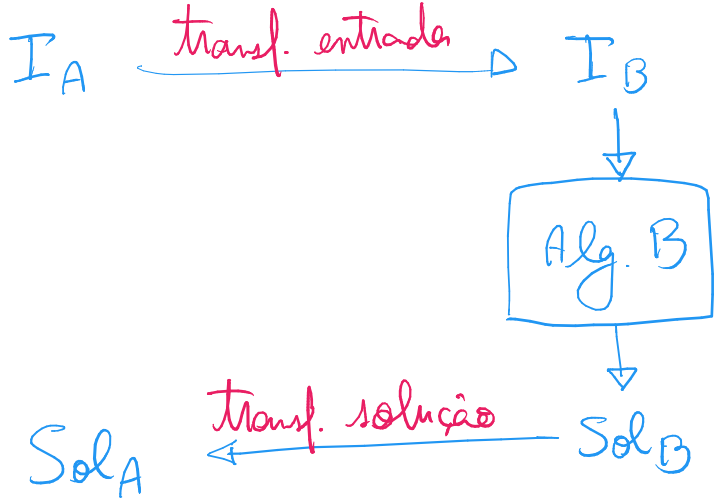
Por quê?

- temos problemas em P, que também estão em NP
- temos problemas em NP, que não sabemos se estão em P

↳ muitos problemas importantes aqui

- Como corroborar a dificuldade destes problemas?
  - Não conseguimos mostrar que não existem alg. eficientes para eles
- Vamos usar evidências relativas, através de:
  - redução
  - completude

Redução do problema A para B



- Com a redução resolvemos qualquer entrada de A usando um alg. p/ B, junto das transformações
- Se as transf. forem polinomiais, dizemos que A é polinomialmente redutível a B

Exemplos de redução

- Seleção em ordenação
- Caminhos mínimos de Todos p/ Todos em Caminho mínimo 1 p/ Todos

- Redução é uma técnica poderosa de projeto de algoritmos
  - permite usar soluções conhecidas p/ problemas novos

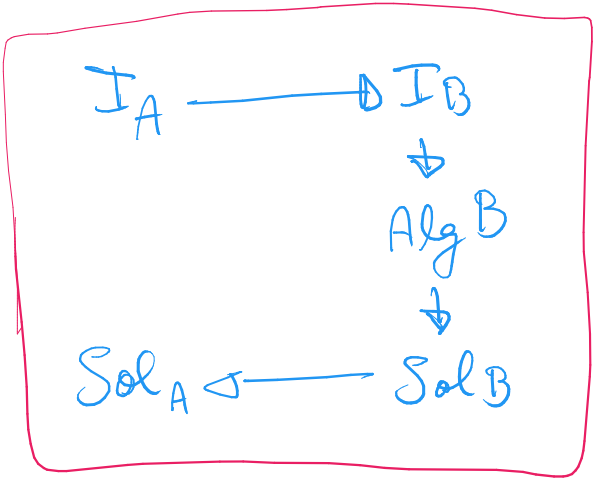
↳ Será que conheço um problema parecido com este?

- Também existe um segundo uso perspicaz das reduções

Redução p/ atestar dificuldade

- A é difícil
- Reduzo (polinomialmente) A p/ B

⇒ B é (pelo menos) tão difícil quanto A



Mostrar que:

I_A tem solução se, e somente se, I_B tem solução

Exemplos de redução

→ c/ Cam. Mínimo c/ ciclos e circuitos negativos

- Caminho Hamiltoniano em Caminho Mais Longo
- Soma de Subconjunto em Mochila Binária
- Clique em Cobertura por Vértices

Interpretações p/ A é redutível a B:

(Positiva) A é (pelo menos) tão fácil quanto B → expande o conj. dos problemas tratáveis

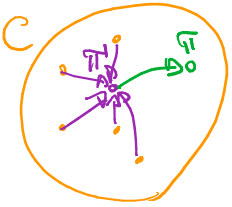
(Negativa) B é (pelo menos) tão difícil quanto A → expande o conj. dos problemas intratáveis

Completude

- Um problema Π é C-completo se:
  - $\Pi \in C$
  - todo problema em C é redutível a Π

} Π é um problema mais difícil de C

- Definição forte que parece inalcançável p/ um conjunto C interessante



- Supondo que exista um problema Π' que é C-completo
  - para mostrar que Π é C-completo basta mostrar que $\Pi \in C$ e reduzir Π' a Π (pois redução é transitiva, i.e., $A \to B$ e $B \to C \Rightarrow A \to C$)

NP-Completude

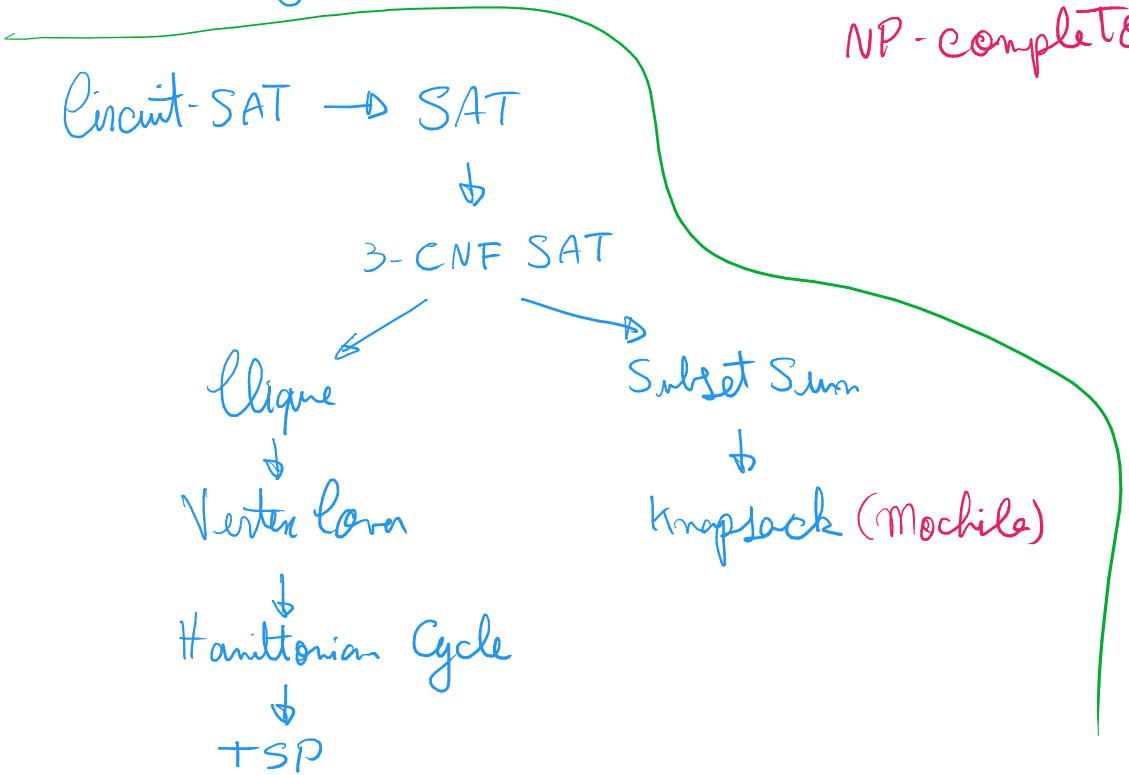
- Voltamos à questão: Como atestar a dificuldade de um problema π em NP (que não parece estar em P)?

- Podemos mostrar que π é NP-Completo
  - Mas será que a classe NP-Completa é não vazia?

- Teorema de Cook-Levin mostra uma redução de qualquer problema em NP p/ o SAT (ou p/ o circuit SAT) - provando que ele é NP-Completo

↳ prob. de satisfatibilidade

Posteriormente muitos problemas foram mostrados NP-completos via uma árvore de reduções enraizada no SAT → Destaque p/ os 21 problemas NP-completos de Karp



Questão P vs. NP

- P e NP são distintos? Maior questão aberta da computação

- Se existir alg. polinomial p/ qualquer problema NP-completo, então temos alg. pol. p/ todo problema em NP, i.e., P = NP

- Mas e daí? Por que um projetista de alg. deveria se preocupar? Ao se deparar c/ um problema,
  - é importante verificar se ele é NP-completo (via redução)
  - e tratá-lo adequadamente (contornando as dificuldades)
    - Assuntos das próximas (e últimas) aulas