Motivação e complexidade

MOTIVAÇÃO:

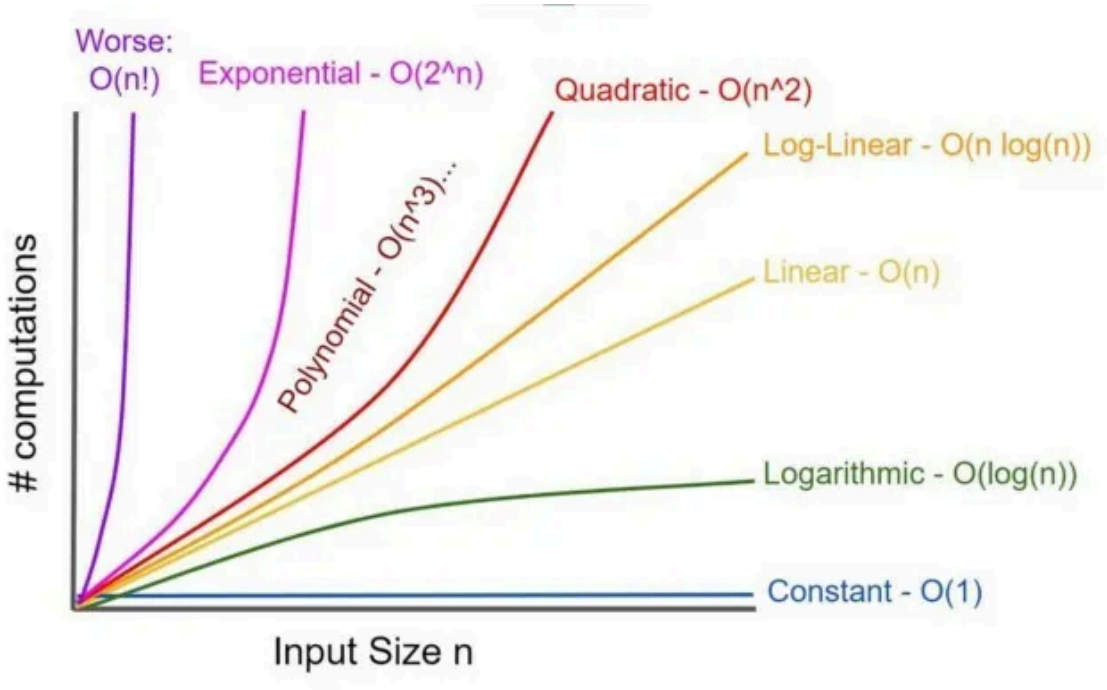
A motivação central do capítulo surge da constatação de que a computação quântica está avançando rapidamente e apresenta a capacidade de resolver problemas considerados intratáveis para computadores clássicos. Entre esses problemas estão justamente aqueles que sustentam a segurança dos algoritmos criptográficos mais utilizados no mundo, como RSA, Diffie-Hellman e curvas elípticas. A possibilidade real de que algoritmos quânticos — especialmente o de Shor — possam quebrar esses sistemas cria um cenário de risco imediato para a confidencialidade, integridade e autenticidade da informação em escala global.

COMPLEXIDADE:

Na computação clássica, algoritmos eficientes são aqueles cuja complexidade cresce de forma polinomial em relação ao tamanho da entrada, enquanto algoritmos com crescimento exponencial são considerados inviáveis na prática. Essa distinção entre o que é eficiente e o que é intratável é a base da segurança dos sistemas assimétricos: fatorar grandes inteiros ou resolver logaritmos discretos exige tempo exponencial para algoritmos clássicos conhecidos.

Parte do Itman

Coloca alguma coisa básica sobre complexidade, pode ser aquela imagem daquele gráfico, eu explico no pelo na hora



Coloca uma explicação simples:

P (Polynomial Time): Classe de problemas que podem ser **resolvidos** em tempo polinomial. Exemplos: ordenação, busca em grafos, cálculo de caminho mínimo.

NP (Nondeterministic Polynomial Time): Classe de problemas cujas **soluções podem ser verificadas** em tempo polinomial. Exemplos: caixeiro-viajante, satisfatibilidade booleana (SAT), coloração de grafos.

Relação chave:

- P ⊆ NP (todo problema em P está em NP)
- P = NP? → questão em aberto há 50+ anos, um dos Problemas do Milênio

NP-completo: Subconjunto de NP contendo os problemas "mais difíceis". Se você resolver um NP-completo em tempo polinomial, resolve todos os NP (e prova P = NP).

NP-difícil: Problemas pelo menos tão difíceis quanto NP-completos, mas não necessariamente em NP.

Na prática: P = tratável, NP-completo = intratável para instâncias grandes (força bruta explode exponencialmente).

Vamos colocar um exemplo prático para exemplificar (Ford-Fulkerson X Edmonds Karp)

Ford-Fulkerson (DFS):

- Complexidade: **O(E × f_max)** onde f_max = fluxo máximo
- Problema: depende do **valor** do fluxo, não do tamanho do grafo
- Se capacidades forem grandes (ex: 10^9), pode fazer 10^9 iterações
- Com capacidades irracionais, pode nem terminar

Edmonds-Karp (BFS):

- Complexidade: **O(V × E²)**
- BFS garante caminho aumentante **mais curto** (menor nº de arestas)
- Limita iterações a O(VE), independente dos valores das capacidades
- Sempre polinomial e termina

Por que BFS melhora? BFS força usar caminhos curtos → cada aresta satura/dessatura no máximo V vezes → máximo VE aumentações de fluxo → O(VE) iterações × O(E) por BFS = **$O(VE^2)$**

Ford-Fulkerson pode explodir com valores grandes; Edmonds-Karp é sempre polinomial porque BFS controla o número de iterações.

Slides: Motivação e Complexidade da Computação Quântica

Slide 1: Motivação e Complexidade Computacional

Por que Computação Quântica?

- Problemas intratáveis para computadores clássicos
- Exploração de fenômenos quânticos: superposição, emaranhamento e interferência
- Expansão da classe de problemas: de P (clássico) para BQP (quântico)

Análise de Complexidade:

- Complexidade temporal: consumo de recursos em função do tamanho n da entrada
- Notação Big-O: limite superior assintótico O(f(n))
- **Eficiência:** polinomial $O(n^k)$ vs exponencial $O(2^n)$
- Classes: **P** (resolúvel em tempo polinomial clássico) e **BQP** (quântico)

Slide 2: Aceleração Quântica e Algoritmo de Shor

Tipos de Aceleração:

- **Quadrática (Grover):** $O(2^n) \rightarrow O(2^{n/2})$ - busca não estruturada, impacto em AES/SHA
- **Exponencial (Shor):** $O(e^{(n)^a}) \rightarrow O(\text{poly}(n))$ - fatoração e logaritmo discreto

Algoritmo de Shor - Complexidade:

Problema	Clássico (melhor)	Quântico (Shor)	Impacto
Fatoração (IFP)	$O(e^{(n)}\text{^}(1/3)(\log n)\text{^}(2/3))$	**$O(n^2 \log n)$**	RSA quebrado
Log. Discreto (DLP)	$O(e^{(n)}\text{^}(1/3)(\log n)\text{^}(2/3))$	**$O(n^3)$**	DSA quebrado
Log. Elíptico (ECDLP)	O(e^(poly(n)))	**$O(n^3)$**	ECDSA quebrado

Como funciona: Transforma fatoração em encontrar período de função usando Transformada Quântica de Fourier (QFT)

Slide 3: Por que Não Está Sendo Executado?

Requisitos de Hardware (CRQC):

1. **Número de qubits:** milhares a milhões necessários
    - Exemplo: ~20 milhões de qubits para RSA-2048
    - Estado atual: ~100-1000 qubits ruidosos (era NISQ)
2. **Qualidade dos qubits:**
    - Correção de erros quânticos robusta
    - Taxa de erro < 10^{-4} por operação
    - Tempo de coerência suficientemente longo

Desafios Práticos:

- **Erros quânticos:** qubits extremamente sensíveis a ruído
- **Decoerência:** perda de informação ao longo do tempo
- **Escalabilidade:** construir e controlar milhões de qubits
- **Overhead:** constantes multiplicativas enormes + correção de erros
- **Último recorde clássico:** RSA-250 (829 bits) fatorado em 2020

Previsão: 10-30 anos para CRQC operacional

Slide 4: Implicações e Soluções

Urgência da Transição:

- Ataque **"Store Now, Decrypt Later" (SNDL):** dados capturados hoje podem ser decriptados no futuro
- Necessidade de migração **antes** do CRQC existir
- Todos algoritmos baseados em IFP, DLP e ECDLP devem ser substituídos

Soluções em Desenvolvimento:

Abordagem	Descrição	Status
Criptografia Pós-Quântica (PQC)	Algoritmos clássicos resistentes a ataques quânticos	Padronização NIST 2024
Criptografia Quântica (QKD)	Segurança baseada em leis da física	Implementações limitadas

Padrões NIST (2024):

- **ML-KEM** (encapsulamento de chaves) - baseado em reticulados
- **ML-DSA** (assinatura digital) - baseado em reticulados
- **SLH-DSA** (assinatura digital) - baseado em hash

Conclusão: Aceleração exponencial comprovada, mas implementação prática ainda distante. Transição já deve começar.