Theory of Computation

George Karakostas, Rm. ITB/218, karakos@mcmaster.ca

Introduction

Computability vs Complexity

- **Computability:** Problems that can be solved by algorithms (or *impossibility of an algorithm*).
- **Complexity:** Most efficient algorithm for a *computable* problem (or *impossibility of a better algorithm*).

Introduction

Computability vs Complexity

- **Computability:** Problems that can be solved by algorithms (or *impossibility of an algorithm*).
- **Complexity:** Most efficient algorithm for a *computable* problem (or *impossibility of a better algorithm*).

In this course we will focus on **Computational Complexity**.

Introduction

Basic complexity questions

Introduction

Basic complexity questions

1. Can we do better than exhaustive search? (cf. $P \neq NP$)

Introduction

Basic complexity questions

1. Can we do better than exhaustive search? (cf. $P \neq NP$)
2. Does randomness help? (cf. pseudorandom generators)

Introduction

Basic complexity questions

1. Can we do better than exhaustive search? (cf. $P \neq NP$)
2. Does randomness help? (cf. pseudorandom generators)
3. Are there efficient approximation algorithms? (cf. Independent Set)

Basic complexity questions

1. Can we do better than exhaustive search? (cf. $P \neq NP$)
2. Does randomness help? (cf. pseudorandom generators)
3. Are there efficient approximation algorithms? (cf. Independent Set)
4. Can we exploit problem hardness? (cf. cryptography)

Introduction

Basic complexity questions

1. Can we do better than exhaustive search? (cf. $P \neq NP$)
2. Does randomness help? (cf. pseudorandom generators)
3. Are there efficient approximation algorithms? (cf. Independent Set)
4. Can we exploit problem hardness? (cf. cryptography)
5. Are quantum computers more powerful than classical computers? (cf. Shor's algorithm)

Introduction

Basic complexity questions

1. Can we do better than exhaustive search? (cf. $P \neq NP$)
2. Does randomness help? (cf. pseudorandom generators)
3. Are there efficient approximation algorithms? (cf. Independent Set)
4. Can we exploit problem hardness? (cf. cryptography)
5. Are quantum computers more powerful than classical computers? (cf. Shor's algorithm)
6. Can proofs be efficiently produced automatically? (cf. $P \neq NP$)

Introduction

Basic complexity questions

1. Can we do better than exhaustive search? (cf. $P \neq NP$)
2. Does randomness help? (cf. pseudorandom generators)
3. Are there efficient approximation algorithms? (cf. Independent Set)
4. Can we exploit problem hardness? (cf. cryptography)
5. Are quantum computers more powerful than classical computers? (cf. Shor's algorithm)
6. Can proofs be efficiently produced automatically? (cf. $P \neq NP$)
7. Can check a proof reading only a few bits of it? (cf. PCP theorem)

Introduction

Basic complexity questions

1. Can we do better than exhaustive search? (cf. $P \neq NP$)
2. Does randomness help? (cf. pseudorandom generators)
3. Are there efficient approximation algorithms? (cf. Independent Set)
4. Can we exploit problem hardness? (cf. cryptography)
5. Are quantum computers more powerful than classical computers? (cf. Shor's algorithm)
6. Can proofs be efficiently produced automatically? (cf. $P \neq NP$)
7. Can check a proof reading only a few bits of it? (cf. PCP theorem)
8. Are proofs produced by prover/verifier dialogue more powerful ? (cf. interactive proofs)