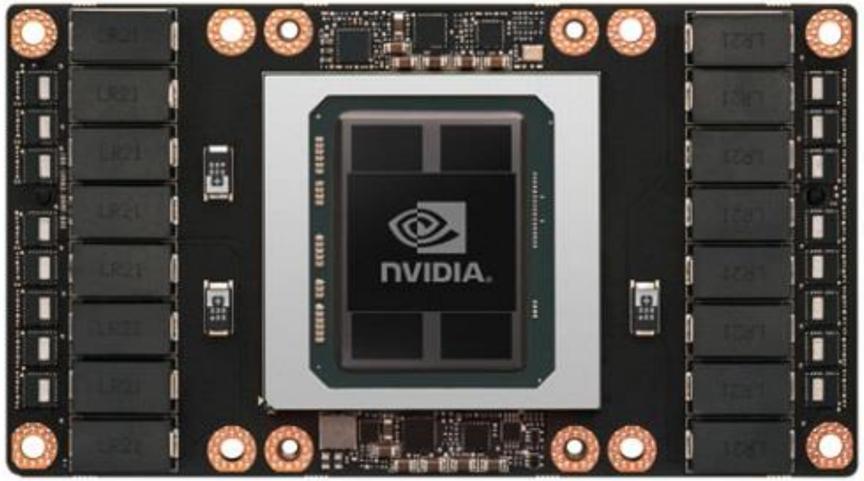

Design and Analysis of Algorithms I

Introduction

About The Course

COURSE TOPICS

- Vocabulary for design and analysis of algorithms
- Divide and conquer algorithm design paradigm
- Radomization in algorithm design
- Primitives for reasoning about graphs
- Use and implementation of data structures

COURSE TOPICS

- Vocabulary for design and analysis of algorithms
  - E.g., “Big O” notation
  - “sweet spot” for high level reasoning about algorithms

COURSE TOPICS

- Vocabulary for design and analysis of algorithms
- Divide and conquer algorithm design paradigm

COURSE TOPICS

- Vocabulary for design and analysis of algorithms
- Divide and conquer algorithm design paradigm
  - Will apply to: Integer multiplication, sorting, matrix multiplication, closest pair
  - General analysis methods (“Master Method/Theorem”)

COURSE TOPICS

- Vocabulary for design and analysis of algorithms
- Divide and conquer algorithm design paradigm
- Randomization in algorithm design
  - Will apply to: QuickSort, primality testing, graph partitioning, hashing.

COURSE TOPICS

- Vocabulary for design and analysis of algorithms
- Divide and conquer algorithm design paradigm
- Randomization in algorithm design
- Primitives for reasoning about graphs
  - Connectivity information, shortest paths, structure of information and social networks.

COURSE TOPICS

- Vocabulary for design and analysis of algorithms
- Divide and conquer algorithm design paradigm
- Randomization in algorithm design
- Primitives for reasoning about graphs
- Use and implementation of data structures
  - Heaps, balanced binary search trees, hashing and some variants (e.g., bloom filters)

TOPICS FOR LATER IN COURSE

- Greedy algorithm design paradigm
- Dynamic programming algorithm design paradigm

- Greedy algorithm design paradigm
- Dynamic programming algorithm design paradigm
- NP Complete problems and what to do about them

- Greedy algorithm design paradigm
- Dynamic programming algorithm design paradigm
- NP Complete problems and what to do about them
- Fast heuristics with provable guarantees
- Fast exact algorithms for special cases
- Exact algorithms that beat brute force search

Skills You’ll Learn

- Become a better programmer

- Become a better programmer
- Sharpen your analytical skills

- Become a better programmer
- Sharpen your analytical
- Start “thinking algorithmically”

- Become a better programmer
- Sharpen your analytical
- Start “thinking algorithmically”
- Literacy with computer science’s “greatest hits”

- Become a better programmer
- Sharpen your analytical skills
- Start “thinking algorithmically”
- Literacy with computer science’s “greatest hits”
- Ace your technical interviews

Who Are You?

- It doesn’t matter

WHO ARE YOU?

- It doesn’t matter
- Ideally, you know some programming.

WHO ARE YOU?

- It doesn’t matter
- Ideally, you know some programming.
- It doesn’t matter
  - But you should be capable of translting algorithm descript i ons into working programs in *some* programming language.

WHO ARE YOU?

- It doesn’t matter
- Ideally, you know some programming.
- It doesn’t matter which languages you kno
- Some (perhaps rusty) mathematical experience.
  - Basic discrete math, proofs by induction etc.

WHO ARE YOU?

- It doesn't matter
- Ideally, you know some programming.
- It doesn't matter which languages you know.
- Some (perhaps rusty) mathematical experience.
  - Basic discrete math, proofs by induction etc.

- *Excellent free reference:* "Mathematics for Computer Science", by Eric Lehman and Tom Leighton. (Easy to find on the Web)

Support i ng Materials

- All (annotated)slides available from course site.

SUPPORT I NG MATERIALS

- All (annotated) slides available from course site.
- No required textbook. A few of the many good ones:
  - Kleinberg/Tardos, *Algorithm Design*, 2005.
  - Dasgupta/Papadimitriou/Vazirani, *Algorithms*, 2006.
  - Cormen/Leiserson/Rivest/Stein, *Introduc8on to Algorithms*, 2009 (3rd edition)
  - Mehlhorn/Sanders, *Data Structures and Algorithms: The Basic Toolbox*, 2008.

Biggest influence on instructor

Freely available online

SUPPORT MATERIALS

- All (annotated) slides available from course site.
- No required textbook. A few of the many good ones:
  - Kleinberg/Tardos, *Algorithm Design*, 2005.
  - Dasgupta/Papadimitriou/Vazirani, *Algorithms*, 2006.
  - Cormen/Leiserson/Rivest/Stein, *Introduc8on to Algorithms*, 2009 (3rd editionti.
  - Mehlhorn/Sanders, *Data Structures and Algorithms: The Basic Toolbox*, 2008.
- No specific development environment required.
  - But you should be able to write and execute programs.