02-713 Introduction

Slides by Carl Kingsford

Jan. 14, 2013

Reading: KT Chapter 1

Objective of this Course

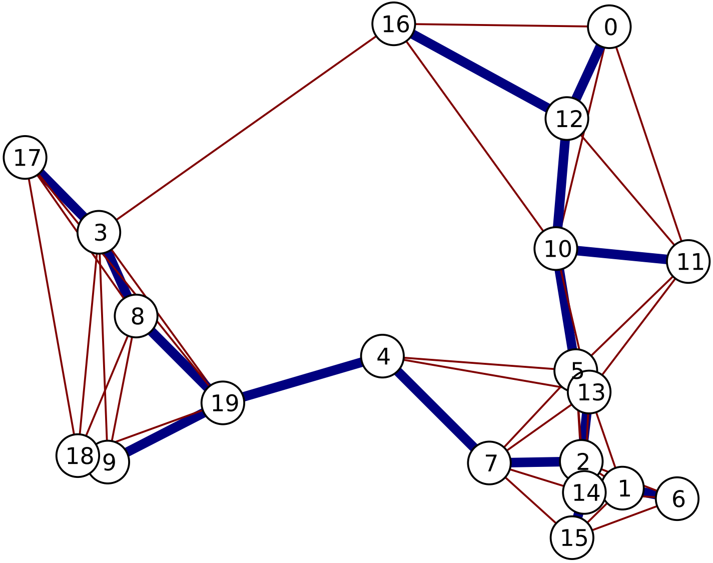
To study general computational problems and their algorithms, with a focus on the principles used to design those algorithms.

After passing this class, you should be able to:

1. Design algorithms using several common techniques
2. Prove a worst-case running time for many algorithms
3. Prove a problem is probably hard (NP-complete)

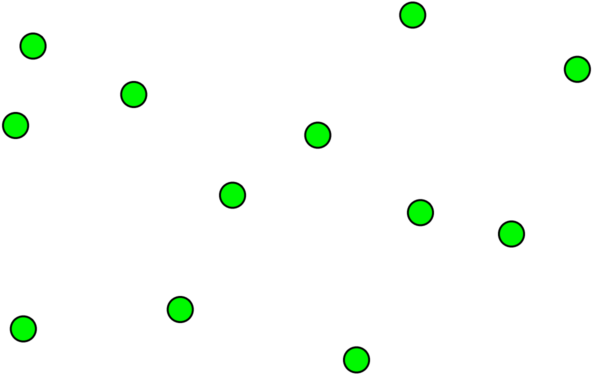
Example Problems

Example I: Low-cost network design



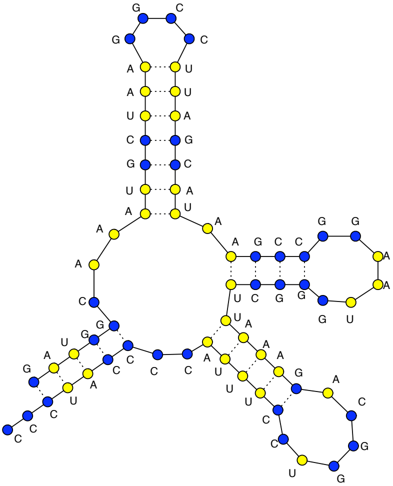
Example II: Finding closest pair of points

Given a set of points $\{p_1, \ldots, p_n\}$ find the pair of points $\{p_i, p_j\}$ that are closest together.

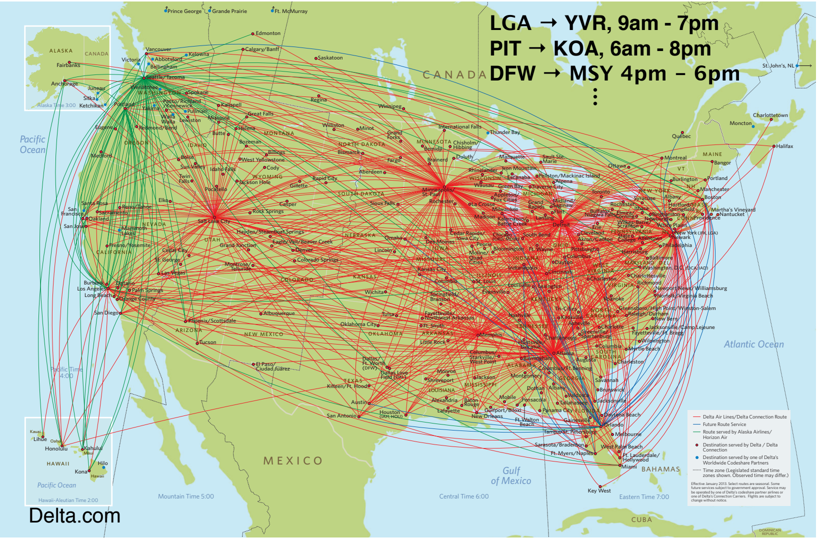


Example III: RNA folding

`GAUGGCAAAUGCUAAGGCCU...` →



Example IV: Scheduling *k* planes


CANADA
ALASKA
Pacific Ocean
Atlantic Ocean
Gulf of Mexico
MEXICO
BAHAMAS
CUBA
DOMINICAN REPUBLIC
HAWAII
Alaska Time 3:00
Pacific Time 4:00
Mountain Time 5:00
Central Time 6:00
Eastern Time 7:00
Hawaii-Aleutian Time 2:00
Seattle/Tacoma
Salt Lake City
Minneapolis/St. Paul
Detroit
Atlanta
Memphis
New York (JFK, LGA)
Los Angeles
Dallas/Ft. Worth (DFW)
Vancouver
Kona
Honolulu
Delta Air Lines/Delta Connection Route
Future Route Service
Route served by Alaska Airlines/ Horizon Air
Destination served by Delta / Delta Connection
Destination served by one of Delta's Worldwide Codeshare Partners
Time zone (Legislated standard time zones shown. Observed time may differ.)
Effective January 2013. Select routes are seasonal. Some future services subject to government approval. Service may be operated by one of Delta's codeshare partner airlines or one of Delta's Connection Carriers. Flights are subject to change without notice.

LGA → YVR, 9am - 7pm

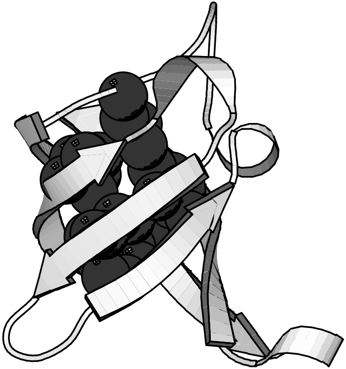
PIT → KOA, 6am - 8pm

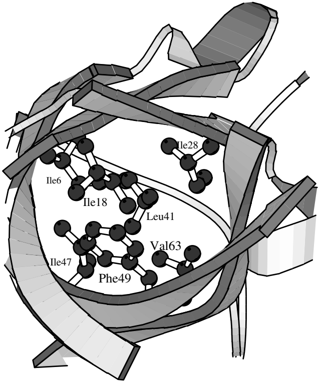
DFW → MSY 4pm – 6pm

⋮

Delta.com

Example V: Side-chain positioning





Design of algorithms

General techniques:

- Greedy . . . . (Chapter 4)
- Divide & conquer . . . . (Chapter 5)
- Dynamic programming . . . . (Chapter 6)
- Network flow . . . . (Chapter 7)
- Linear and integer programming . . . . (Sections 11.6-11.7)

Not all algorithms fit into these categories, but a very large fraction do.

Analysis of algorithms

- Prove **correctness**
  (the algorithm always returns the right answer)

- Discuss how to **implement**
  (what data structures do we need to implement the algorithm?).

- Prove **worst-case running time**
  (no matter the input, it will never run slower that we expect).

- Prove no algorithm can do better
  (theory of computational complexity).

Tentative Schedule

1. Introduction, Minimum Spanning Tree case study, and Python

2. Elementary algorithms: divide & conquer and graph algorithms
    - Asymptotic analysis
    - Closest pair of points
    - Fast Fourier Transform
    - Graph search: Breadth first, depth first, topological sorting
    - Shortest path algorithms
    - A* search

3. Advanced algorithmic design techniques
    - Dynamic programming
    - Network flow
    - Linear and integer programming
    - NP-completeness
    - Randomized algorithms

Homeworks

- Near-weekly homeworks
- 10% of your grade
- Encouraged to discuss homeworks with other students in class
- **MUST WRITE UP HOMEWORKS ON YOUR OWN**
- You must list, at the top of your homework, those people with whom you discussed the problems & any sources you used
- Homework answers must be typeset and submitted online (instructions will be on website)
- A few homeworks will consist of programming in Python

What does “on your own” mean?

You cannot, for example:

- look at another person’s homework
- have them look at yours to see if it is correct
- take notes from a discussion and edit them into your homework
- sit in a group and continue discussing the homework while writing it up

Intent: you can gather around a whiteboard with your fellow students and discuss how to solve the problems. Then you must all walk away and write the answers up separately.

Exams

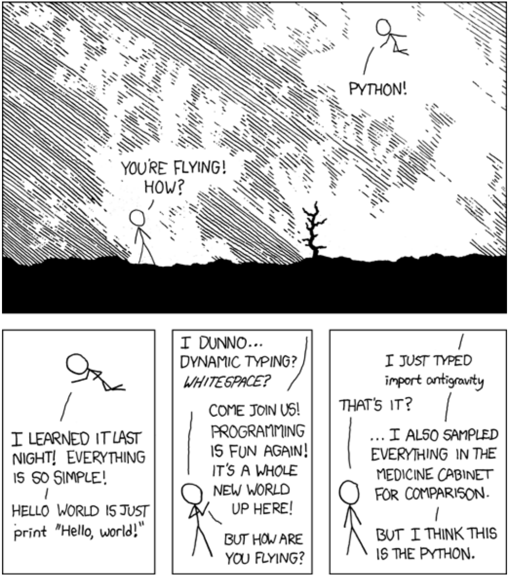
Two non-cumulative midterm exams, each 25% of grade:

- Friday, March 1st, 2013
- Friday, April 26, 2013

A cumulative final exam:

- According to the official university exam schedule.

Why Python?


PYTHON!
YOU'RE FLYING! HOW?
I LEARNED IT LAST NIGHT! EVERYTHING IS SO SIMPLE!
HELLO WORLD IS JUST print "Hello, world!"
I DUNNO... DYNAMIC TYPING? WHITESPACE?
COME JOIN US! PROGRAMMING IS FUN AGAIN! IT'S A WHOLE NEW WORLD UP HERE!
BUT HOW ARE YOU FLYING?
I JUST TYPED import antigravity
THAT'S IT?
... I ALSO SAMPLED EVERYTHING IN THE MEDICINE CABINET FOR COMPARISON.
BUT I THINK THIS IS THE PYTHON.

http://xkcd.com/353/

Why Python?

Pros:

- Expressive, math-like syntax
- Support for modern programming paradigms (object orientation, some functional programming)
- Scripting language avoids compilation
- Extensive on-line help and documentation
- Extensive libraries (graphs, matlab functions, numerical methods)
- Widely used in bioinformatics & other disciplines

Cons:

- Can be slower than other languages (especially loops)
- Less memory efficient than other languages

Homework 0: Survey

Complete the survey at

`http://www.cs.cmu.edu/~ckingsf/class/02713-s13/survey.html`

Due by 11:59pm on Tuesday, Jan 15.