Greedy Algorithms, Minimum Spanning Trees, and Dynamic Programming

Stanford University

About this Course

The primary topics in this part of the specialization are: greedy algorithms (scheduling, minimum spanning trees, clustering, Huffman codes) and dynamic programming (knapsack, sequence alignment, optimal search trees).

Taught by: Tim Roughgarden, Professor
Computer Science

Basic Info	Course 3 of 4 in the Algorithms Specialization
Level	Intermediate
Commitment	4 weeks of study, 4-8 hours/week
Language	English Volunteer to translate subtitles for this course
How To Pass	Pass all graded assignments to complete the course.
User Ratings	4.8 stars

Syllabus

WEEK 1

coursera

Week 1

Two motivating applications; selected review; introduction to greedy algorithms; a scheduling application; Prim's MST algorithm.

16 videos, 4 readings

1. **Reading:** Week 1 Overview
2. **Reading:** Overview, Resources, and Policies
3. **Reading:** Lecture slides
4. **Video:** Application: Internet Routing
5. **Video:** Application: Sequence Alignment
6. **Video:** Introduction to Greedy Algorithms
7. **Video:** Application: Optimal Caching
8. **Video:** Problem Definition
9. **Video:** A Greedy Algorithm
10. **Video:** Correctness Proof - Part I
11. **Video:** Correctness Proof - Part II
12. **Video:** Handling Ties [Advanced - Optional]
13. **Video:** MST Problem Definition
14. **Video:** Prim's MST Algorithm
15. **Video:** Correctness Proof I
16. **Video:** Correctness Proof II
17. **Video:** Proof of Cut Property [Advanced - Optional]
18. **Video:** Fast Implementation I
19. **Video:** Fast Implementation II
20. **Reading:** Optional Theory Problems (Week 1)

Show less

Graded: Problem Set #1

Graded: Programming Assignment #1

Week 2

Kruskal's MST algorithm and applications to clustering; advanced union-find (optional).

16 videos, 2 readings

1. **Reading:** Week 2 Overview
2. **Video:** Kruskal's MST Algorithm
3. **Video:** Correctness of Kruskal's Algorithm
4. **Video:** Implementing Kruskal's Algorithm via Union-Find I
5. **Video:** Implementing Kruskal's Algorithm via Union-Find II
6. **Video:** MSTs: State-of-the-Art and Open Questions [Advanced - Optional]
7. **Video:** Application to Clustering
8. **Video:** Correctness of Clustering Algorithm
9. **Video:** Lazy Unions [Advanced - Optional]
10. **Video:** Union-by-Rank [Advanced - Optional]
11. **Video:** Analysis of Union-by-Rank [Advanced - Optional]
12. **Video:** Path Compression [Advanced - Optional]
13. **Video:** Path Compression: The Hopcroft-Ullman Analysis I [Advanced - Optional]
14. **Video:** Path Compression: The Hopcroft-Ullman Analysis II [Advanced - Optional]
15. **Video:** The Ackermann Function [Advanced - Optional]
16. **Video:** Path Compression: Tarjan's Analysis I [Advanced - Optional]
17. **Video:** Path Compression: Tarjan's Analysis II [Advanced - Optional]
18. **Reading:** Optional Theory Problems (Week 2)

Show less

Graded: Problem Set #2

Graded: Programming Assignment #2

WEEK 3

Week 3

Huffman codes; introduction to dynamic programming.

11 videos, 1 reading

1. **Reading:** Week 3 Overview
2. **Video:** Introduction and Motivation
3. **Video:** Problem Definition
4. **Video:** A Greedy Algorithm
5. **Video:** A More Complex Example
6. **Video:** Correctness Proof I
7. **Video:** Correctness Proof II
8. **Video:** Introduction: Weighted Independent Sets in Path Graphs
9. **Video:** WIS in Path Graphs: Optimal Substructure
10. **Video:** WIS in Path Graphs: A Linear-Time Algorithm
11. **Video:** WIS in Path Graphs: A Reconstruction Algorithm
12. **Video:** Principles of Dynamic Programming

Show less

Graded: Problem Set #3

Graded: Programming Assignment #3

WEEK 4

Week 4

Advanced dynamic programming: the knapsack problem, sequence alignment, and optimal binary search trees.

10 videos, 3 readings

1. **Reading:** Week 4 Overview
2. **Video:** The Knapsack Problem
3. **Video:** A Dynamic Programming Algorithm
4. **Video:** Example [Review - Optional]

5. **Video:** Optimal Substructure
6. **Video:** A Dynamic Programming Algorithm
7. **Video:** Problem Definition
8. **Video:** Optimal Substructure
9. **Video:** Proof of Optimal Substructure
10. **Video:** A Dynamic Programming Algorithm I
11. **Video:** A Dynamic Programming Algorithm II
12. **Reading:** Optional Theory Problems (Week 4)
13. **Reading:** Info and FAQ for final exam

Show less

Graded: Problem Set #4

Graded: Programming Assignment #4

Graded: Final Exam

View Less

How It Works

GENERAL

How do I pass the course?

To earn your Course Certificate, you'll need to earn a passing grade on each of the required assignments—these can be quizzes, peer-graded assignments, or programming assignments. Videos, readings, and practice exercises are there to help you prepare for the graded assignments.

More

What do start dates and end dates mean?

Most courses have sessions that run multiple times a year — each with a specific start and end date. Once you enroll, you'll have access to all videos, readings, quizzes, and programming assignments (if applicable). Peer-graded assignments can only be submitted and reviewed once your session has begun. If you choose to explore the course without purchasing, you may not be able to access certain assignments. If you don't finish all graded assignments before the end of the session, you can enroll in the next session. Your progress will be saved and you'll be able to pick up where you left off when the next session begins.

Course 3 of Specialization

Learn To Think Like A Computer Scientist

Master the fundamentals of the design and analysis of algorithms.

Algorithms

Algorithms

Stanford University

View the course in catalog

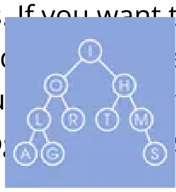
What are due dates? Is there a penalty for submitting my work after a due date?

Within each session, there are suggested due dates to help you manage your schedule and keep coursework from piling up. Quizzes and programming assignments can be submitted late without consequence. However, it is possible that you won't receive a grade if you submit your peer-graded assignment too late because classmates usually review assignment within three days of the assignment deadline.

Can I re-attempt an assignment?

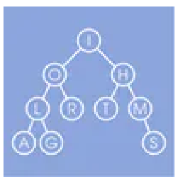
Yes. If you want to improve your grade, you can always try again. If you're re-attempting a peer-graded assignment, re-submit your work as soon as you can to make sure there's enough time for your classmates to review your work. In some cases you may need to wait before re-submitting a programming assignment or quiz. We encourage you to review course material during this delay.

Related Courses



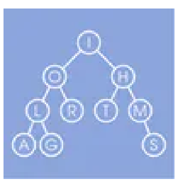
Algorithms on Strings

University of California San Diego, National Research University Higher School of Economics



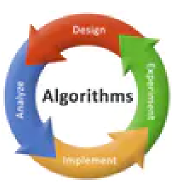
Algorithms on Graphs

University of California San Diego, National Research University Higher School of Economics



Advanced Algorithms and Complexity

University of California San Diego, National Research University Higher School of Economics



Graph Search, Shortest Paths, and Data Structures

Stanford University

Shortest Paths Revisited, NP-Complete Problems and What To Do About Them

Stanford University