Learning Guide Unit 1

Site: University of the People
Course: CS 3304-01 Analysis of Algorithms - AY2024-T5
Book: Learning Guide Unit 1
Printed by: Mejbaul Mubin
Date: Saturday, 29 June 2024, 6:43 AM

Description

Learning Guide Unit 1

Table of contents

Overview

Unit 1: Review of Data Structures and Algorithms

Topics:

- Review of Data Structures
- Review of Asymptotic Analysis
- Upper bounds analysis
- Introductions to Sequences and Recurrence Relations
- Brute Force Algorithms
- Backtracking Algorithms
- Branch and Bound Algorithms

Learning Objectives:

By the end of this Unit, students will be able to:

1. Apply asymptotic analysis including the use of the Big-Oh, Big-Omega, and Big-Theta notations
2. Understand the role of summations and recurrence relations
3. Articulate the use of recurrences as a means to analyze the cost of an algorithm
4. Explain and be able to distinguish between the variations of brute force algorithms including:
    - Brute Force Algorithms
    - Backtracking Algorithms
    - Branch and Bound Algorithms

Tasks:

- Read the Learning Guide and Reading Assignments
- Participate in the Discussion Assignment (post, comment, and rate in the Discussion Forum)
- Make entries to the Learning Journal
- Take the Self-Quiz

Introduction

Unit one should introduce some new material and a lot of material that is familiar from the CS 3303 Data Structures course. In the data structures course we looked at several basic data structures and the algorithms that were used to implement those structures. We also looked at some key algorithms such as the algorithms for searching, sorting and maintaining various tree structures.

Although we are familiar with these structures and algorithms, we may not be familiar with the fact that they follow specific Algorithm design patterns that we must be familiar with.

In this unit we will review some of the topics that we learned in CS 3303 and will add to that learning an understanding of the algorithm design pattern that each represents. The first we will look at is Brute Force Algorithms.

Brute Force Algorithms
Brute Force Algorithms are algorithms that use obvious non-sophisticated approaches to solve the problems in hand. Typically they are useful for small domains, due to large overheads in sophisticated approaches. An example of a brute force algorithm that we learned about is the sequential search (Chapter 9).

The sequential search is a search algorithm which traverses all of the elements of a given set. Sequential searches are simple to implement but as the lists become long they are not that efficient because every element in the list may potentially need to be evaluated in order to find the item you are looking for.

Backtracking Algorithms
Although we will not explore this in depth it is important to be aware of a class of algorithms known as backtracking algorithms. The backtracking algorithm is a variation of brute force. A backtracking algorithm tries to build a solution to a computational problem incrementally. Whenever the algorithm needs to decide between two alternatives to the next component of the solution, it simply tries both options recursively.

In this way the backtracking algorithm systematically considers all possible outcomes for each decision. In this sense, backtracking algorithms are like the brute-force algorithms discussed in the preceding section. However, backtracking algorithms are distinguished by the way in which the space of possible solutions is explored. Sometimes a backtracking algorithm can detect that an exhaustive search is unnecessary and, therefore, it can perform much better.

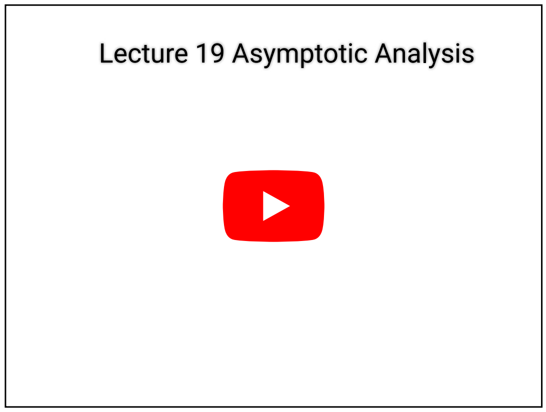
Branch and Bound Algorithms
Branch and Bound Algorithms are a further extension of the backtracking search algorithm which, in the absence of a cost criteria, the algorithm traverses a spanning tree of the solution space using the breadth-first approach. That is, a queue is used, and the nodes are processed in first-in-first-out order.

If a cost criteria is available, the node to be expanded next (i.e., the branch) is the one with the best cost within the queue. In such a case, the cost function may also be used to discard (i.e., the bound) from the queue nodes that can be determined to be expensive. A priority queue is needed here.

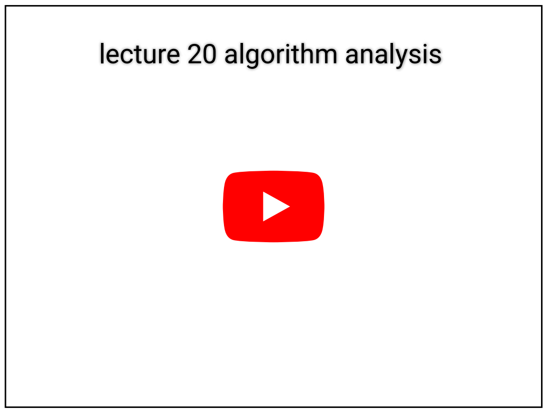
Asymptotic Analysis
Another key area from CS 3303 that we review in Unit1 is Asymptotic Analysis. If you recall Asymptotic analysis is a way of determining the complexity of an algorithm and uses the Big-Oh, Big-Alpha, and Big Theta notations.

Asymptotic analysis is an important concept that will be used extensively throughout this course so we need to refresh our knowledge of this important concept by reviewing Chapter 2 in the text.   For students who have access to computers with sufficient network speed, I would also recommend the following video lectures on Asymptotic analysis. These lectures are entirely optional and provided as a supplemental resource for those students who wish to take advantage of them.

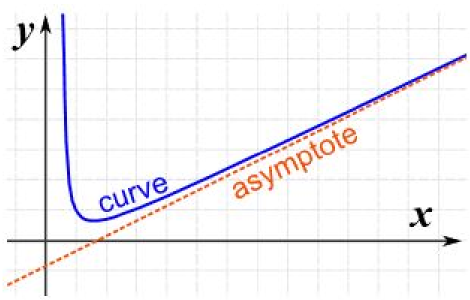
-Lecture 19: Asymptotic Analysis by Instructor Jonathan Shewchuk, University of California Berkley. Retrieved from



-Lecture 20: Algorithm Analysis by Instructor Jonathan Shewchuk, University of California Berkley. Retrieved from



If you look up what an Asymptote is in the encyclopedia, you will find that it refers to the line that a curve approaches as it heads towards infinity.

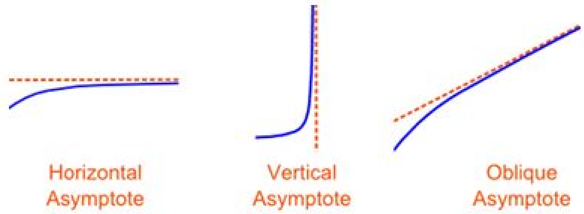


The goal of algorithm analysis is to discover the 'growth' rate in processing of an algorithm as the size of n grows large. This is typically expressed as the output of some function in the form of processing time is the output of some function with input n.

T(n) = f(n)

From the perspective of the graph think of this in terms of the y value that is associated with an x value. If you consider the graph above, for every input value of x there is a corresponding value of y. Think of the x value as the n and the y value as the T(n).

Consider the following figure:



We can see three types of asymptotes in the preceding figure. In the horizontal asymptote we see an example of a log function. The processing time quickly grows to a limit (the asymptote line) and then levels out seeing smaller and smaller growth until infinity. The vertical asymptote shows us an example of exponential growth as the size of y (or T(n) in our case) begins to experience rapid growth and as n grows large the growth the T(n) quickly grows towards infinity. The knapsack problem that we have discussed previously is a great example. The oblique asymptote, after some initial values, demonstrates a linear growth as increases in the size of n result in proportional increases in T(n).

Such asymptotes are associated with Calculus as they rely upon the theory of limits. Throughout this course we will strive to explain and explore these concepts without getting too deep into calculus, however, to fully understand these concepts a basic understanding of calculus is required. As such we will be presenting some material throughout this course that assumes such knowledge of calculus but the impact of this in terms of tests and exams will be limited.

Reading Assignment

Topic 1: Asymptotic Analysis and Complexity
-Review of Asymptotic Analysis and Complexity - http://discrete.gr/complexity/

-Chapter 3: Algorithm Analysis in A Practical Introduction to Data Structures and Algorithm Analysis by Clifford A. Shaffer. http://people.cs.vt.edu/~shaffer/Book/C++3e20100119.pdf

-Chapter 14 Sections 14.1 through 14.2 Analysis Techniques in A Practical Introduction to Data Structures and Algorithm Analysis by Clifford A. Shaffer.

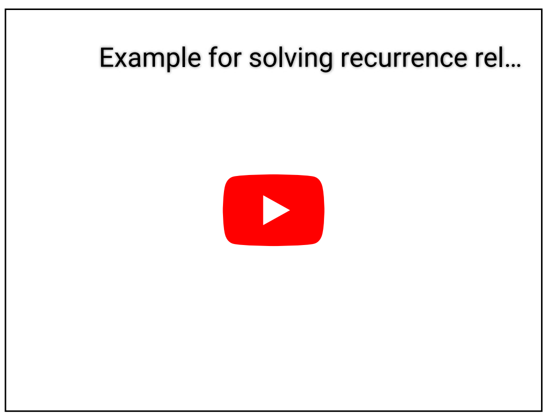
Topic 2: Sequences and Recurrence relations

Solving Recurrences
Erickson, J. (2010). Algorithms. Available under Creative Commons 3.0 at http://jeffe.cs.illinois.edu/teaching/algorithms/notes/99-recurrences.pdf

-Introduction to Recurrence Relations - algo_ch4_recurrences.pdf

-Read the following lecture notes on solving recurrence relations - http://www.cs.duke.edu/~reif/courses/alglectures/skiena.lectures/lecture3.pdf

-Video Lecture detailing how to solve a recurrence relation -



Topic 3: Brute Force Algorithms and variations
-Algorithm Strategies - Download the pdf.

-Algorithm Design Paradigms - https://cgi.csc.liv.ac.uk/~ped/teachadmin/algor/algor.html

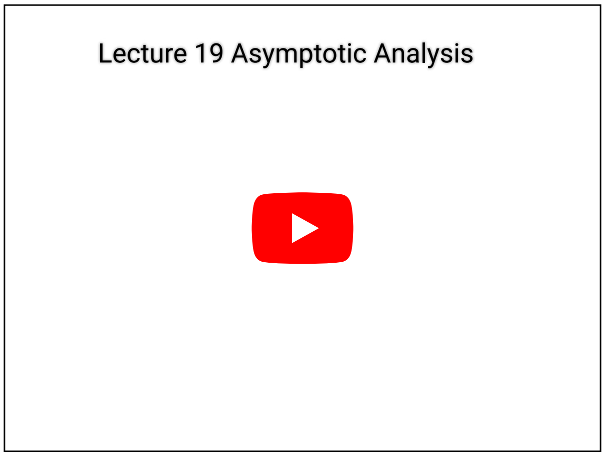
Supplemental Materials
-Chapter 7: Internal Sorting in A Practical Introduction to Data Structures and Algorithm Analysis by Clifford A. Shaffer.

-Chapter 9: Searching in A Practical Introduction to Data Structures and Algorithm Analysis by Clifford A. Shaffer.

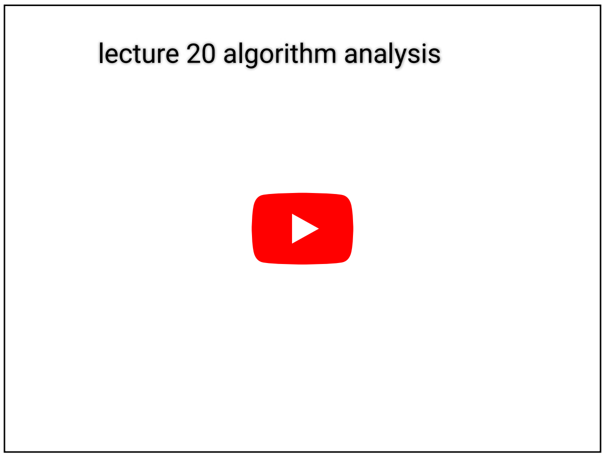
-Algorithm Animations - https://www.cs.auckland.ac.nz/software/AlgAnim/alg_anim.html

The following are video lectures that are available via YouTube and other sources that are related to the topics in the unit and can be used as a supplemental resource for students looking for more details or to be introduced to the same topic from another source. The use of these resources is not required and is entirely optional.

-Lecture 19: Asymptotic Analysis by Instructor Jonathan Shewchuk, University of California Berkley. Retrieved from



-Lecture 20: Algorithm Analysis by Instructor Jonathan Shewchuk, University of California Berkley. Retrieved from



Unit 1 Optional Video Lectures

The following video lectures are optional resources that have been made available to students who can take advantage of them. These lectures are strictly optional resources. All of the information in these lectures is available in other learning resources within the course. These lectures are provided for those students who have sufficient network bandwidth and technology capabilities to take advantage of video content. These lectures cannot be used instead of the required assigned resources and there is no information that is not contained in the assigned resources. These lectures simply present some of the information in a different format.

- Unit 1 Lecture 1: Asymptotic Analysis and Complexity
- Unit 1 Lecture 2: Sequences and Recurrence relations
- Unit 1 Lecture 3: Brute Force Algorithms
- Unit 1 Lecture 4: Backtracking and Branch and Bound

Discussion Assignment

Discussion Assignment #1
In your own words describe both brute force algorithms and branch and bound algorithms and identify the differences between the two. As part of your description you must discuss the Asymptotic behavior of each and discuss why this is important and how this might factor into the decision to use either algorithm design.

Discussion Assignment #2
Complete the following exercises and post your answers to the discussion forum. In each assignment the questions and answers are provided so that you can self-check your work. What is important is your ability to show your work or line of reasoning. After submitting your assignment you should engage with your peers to both validate your own techniques and perhaps critique the work of others. The objective of this assignment is for all of the members of the class to work collaboratively to develop a solid understanding of the Asymptotic assessment of an algorithm and the ability to solve recurrence relations

Exercise 1
$f(n) = n^6 + 3^n$
$f(n) = 2^n + 12$
$f(n) = 3^n + 2^n$
$f(n) = n^n + n$

(Write down your results first; the solution is given below)

Exercise 2:
For the following Θ complexities write down a tight and a non-tight O bound, and a tight and non-tight Ω bound of your choice, providing they exist.

Θ(1)
Θ(√n)
Θ(n)
Θ(n2)
Θ(n3)

For your discussion post, explain in your own words how you solved the exercise. Include one or two examples to explain your thought process to show what is occurring and how the methodology works. Demonstrate your understanding of how to solve each exercise. Use APA citations and references for any sources used.

<u>Discussion Assignment #2 Answers</u>:

Exercise 1 Answers:
$f(n) = n^6 + 3^n \in \Theta(3^n)$
$2^n + 12 \in \Theta(2^n)$
$3^n + 2^n \in \Theta(3^n)$
$n^n + n \in \Theta(n^n)$

Exercise 2 Answers:
This is a straight-forward application of the definitions above.

1. The tight bounds will be O(1) and Ω(1). A non-tight O-bound would be O(n). Recall that O gives us an upper bound. As n is of larger scale than 1 this is a non-tight bound and we can write it as o(n) as well. But we cannot find a non-tight bound for Ω, as we can't get lower than 1 for these functions. So we'll have to do with the tight bound.

2. The tight bounds will have to be the same as the Θ complexity, so they are O(√n) and Ω(√n) respectively. For non-tight bounds we can have O(n), as n is larger than and so it is an upper bound for √n. As we know this is a non-tight upper bound, we can also write it as o(n). For a lower bound that is not tight, we can simply use Ω(1). As we know that this bound is not tight, we can also

write it as ω(1).

3. The tight bounds are O(n) and Ω(n). Two non-tight bounds could be ω(1) and o(n^3). These are in fact pretty bad bounds, as they are far from the original complexities, but they are still valid using our definitions.
4. The tight bounds are O(n2) and Ω(n^2). For non-tight bounds we could again use ω(1) and o(n3) as in our previous example.
5. The tight bounds are O(n^3) and Ω(n^3) respectively. Two non-tight bounds could be ω(n^2) and o(n^4). Although these bounds are not tight, they're better than the ones we gave above.

Learning Journal

The Learning Journal is a tool for self-reflection on the learning process. In addition to completing directed tasks, you should use the Learning Journal to document your activities, record problems you may have encountered and to draft answers for Discussion Forums and Assignments. The Learning Journal should be updated regularly (on a weekly basis), as the learning journals will be assessed by your instructor as part of your Final Grade.

Your learning journal entry must be a reflective statement that considers the following questions:

- Describe what you did. This does not mean that you copy and paste from what you have posted or the assignments you have prepared. You need to describe what you did and how you did it.
- Describe your reactions to what you did
- Describe any feedback you received or any specific interactions you had. Discuss how they were helpful
- Describe your feelings and attitudes
- Describe what you learned

Another set of questions to consider in your learning journal statement include:

- What surprised me or caused me to wonder?
- What happened that felt particularly challenging? Why was it challenging to me?
- What skills and knowledge do I recognize that I am gaining?
- What am I realizing about myself as a learner?
- In what ways am I able to apply the ideas and concepts gained to my own experience?

Your Learning Journal should be a minimum of 500 words

Self-Quiz

The Self-Quiz gives you an opportunity to self-assess your knowledge of what you have learned so far.

The results of the Self-Quiz do not count towards your final grade, but the quiz is an important part of the University's learning process and it is expected that you will take it to ensure understanding of the materials presented. Reviewing and analyzing your results will help you perform better on future Graded Quizzes and the Final Exam.

Please access the Self-Quiz on the main course homepage; it will be listed inside the Unit.

Checklist

Read the Learning Guide and Reading Assignments

Participate in the Discussion Assignment (post, comment, and rate in the Discussion Forum)

Make entries to the Learning Journal

Take the Self-Quiz