



IT1312

Data Structures & Algorithms

Topic 02: Algorithms

version 1.0

What is an algorithm?



<https://www.youtube.com/watch?v=6hfOvs8pY1k>

What is an algorithm?

- ❑ In mathematics and computer science, an algorithm is a set of step-by-step instructions for solving problems.
- ❑ In the TED-Ed video, the author uses the scenario of “counting number of people in a room” to illustrate the concept of an algorithm.
- ❑ What are some examples of algorithms in action that you can observe in our daily lives?

Examples of algorithms

- ❑ How does live video streaming platforms (e.g. YouTube Live, Instagram Live Video, Facebook Live etc.) transmit live video across the internet so quickly?
- ❑ How does Google Maps figures out how to get from NYP to Gardens by the Bay?
- ❑ How does Pixar color a 3D model of a character based on the lighting in a virtual room?

Examples of algorithms

- How does live video streaming platforms (e.g. YouTube Live, Instagram Live Video, Facebook Live etc.) transmit live video across the internet so quickly?

Audio and Video Compression Algorithms

- How does Google Maps figures out how to get from NYP to Gardens by the Bay?

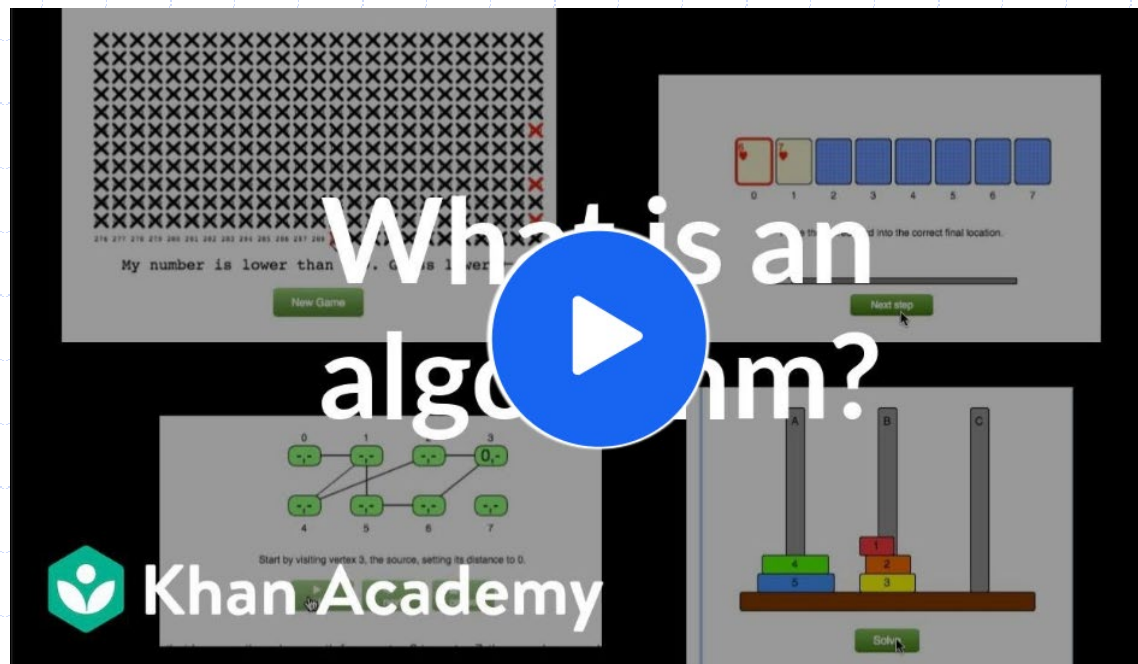
Routing Algorithms

- How does Pixar color a 3D model of a character based on the lighting in a virtual room?

Rendering Algorithms

More examples of algorithms

- ❑ Khan Academy – What is an algorithm and why should you care?



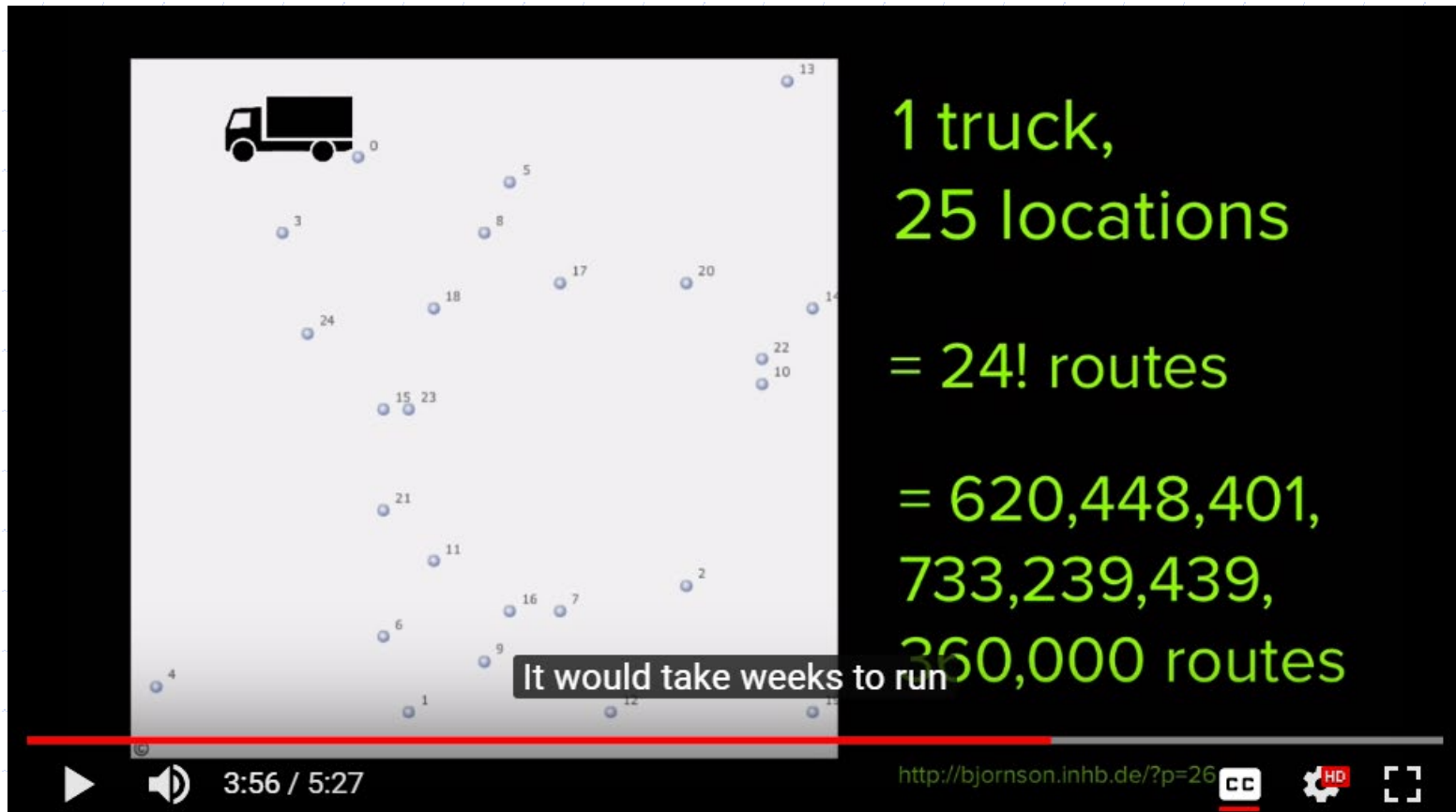
What makes a good algorithm?

- ❑ Correctness
- ❑ Efficiency

- ❑ Correctness vs. Efficiency
 - Best answer?

 - Sometimes, we can live with an algorithm that does not give us the correct answer or best answer for a problem.
 - That is because the only perfect algorithms that we know for the problem takes a really long time to solve.

Correctness vs. Efficiency



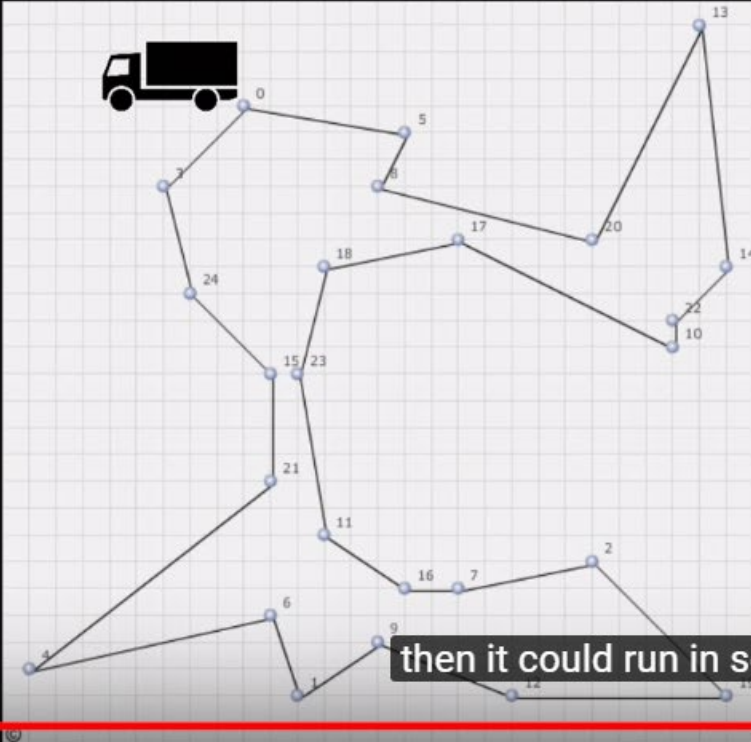
1 truck,
25 locations
= 24! routes
= 620,448,401,
733,239,439,
360,000 routes

It would take weeks to run

3:56 / 5:27

<http://bjornson.inhb.de/?p=26>

Correctness vs. Efficiency



1 truck,
25 locations

...solved using
"Nearest insertion"
algorithm

then it could run in seconds.

4:05 / 5:27

<http://bjornson.inhb.de/?p=26>

Algorithms Analysis

- ❑ How do we measure efficiency of algorithms?
- ❑ Can we simply time how long it takes to run the codes (that implement a particular algorithm)?
- ❑ In fact, we should not as that will depend on many variables e.g. the speed of the computer, the programming language, the compiler that translates the program from the programming language into code that runs directly on the computer, etc.

Algorithms Analysis

- The technique of **Asymptotic Analysis** of algorithms will provide the answers but we will not go into the details in this module/CmU. Briefly, we consider a few things:
 - Size of the input to the algorithm (e.g. how many people are we counting in the room?)
 - How fast a function grows with the input size (e.g. counting people one at a time vs. counting them in pairs, which one will grow faster with input size?)
- We can use the technique of **Asymptotic Analysis** to measure and compare the efficiency of algorithms independent of a particular programming language or hardware.

Asymptotic Analysis

- Asymptotic analysis of an algorithm refers to defining the mathematical boundation of its run-time performance.
- In general, we look at the following scenarios of an algorithm:
 - **Best case** – Minimum time required to run an algorithm
 - **Average case** - Average time required to run an algorithm
 - **Worst case** - Maximum time required to run an algorithm

Asymptotic Notations

- Following are the commonly used **Asymptotic Notations** used to express the running time complexity of an algorithm:
 - **O (Big Oh Notation)**
 - **Ω (Omega Notation)**
 - **θ (Theta Notation)**

Asymptotic Notations

- **O (Big Oh Notation):**
 - The big Oh notation, **$O(n)$** , expresses the upper bound of an algorithm's running time and measures the **worst case time complexity** or the longest amount of time an algorithm can possibly take to complete.

Asymptotic Notations

□ Ω (Omega Notation):

- The omega notation, $\Omega(n)$, expresses the lower bound of an algorithm's running time and measures the **best case time complexity** or the shortest amount of time an algorithm can possibly take to complete.

Asymptotic Notations

□ **Θ (Theta Notation):**

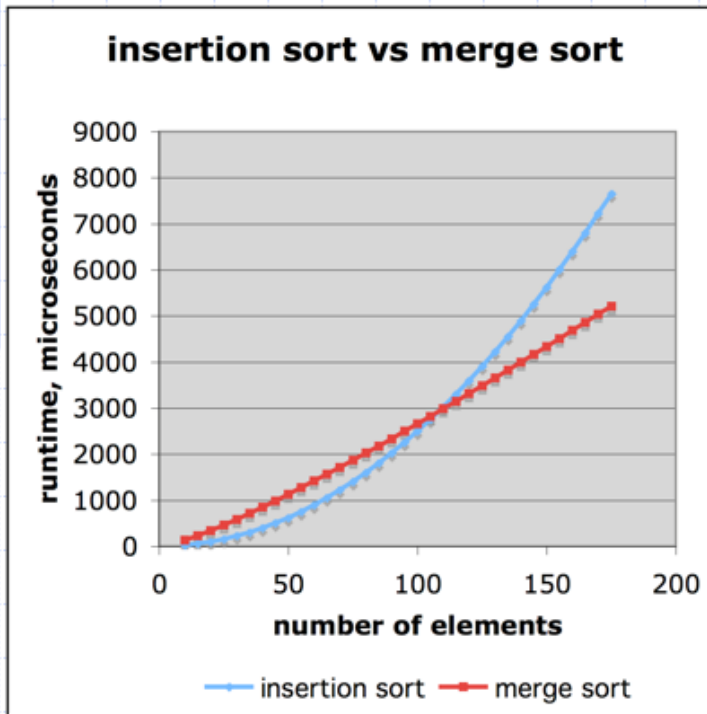
- The theta notation, **$\theta(n)$** , expresses both the lower bound and the upper bound of an algorithm's running time.

Comon Asymptotic Notations

Running Time Complexity	Notation
constant	$O(1)$
logarithmic	$O(\log n)$
linear	$O(n)$
$n \log n$	$O(n \log n)$
quadratic	$O(n^2)$
cubic	$O(n^3)$
polynomial	$n^{O(1)}$
exponential	$2^{O(n)}$

Asymptotic Analysis – A Comparison of Two Algorithms

Comparison of Two Algorithms



insertion sort is
 $n^2 / 4$

merge sort is
 $2 n \lg n$

sort a million items?

insertion sort takes
roughly **70 hours**

while

merge sort takes
roughly **40 seconds**

This is a slow machine, but if
100 x as fast then it's **40 minutes**
versus less than **0.5 seconds**

Asymptotic Analysis – A Comparison of Two Algorithms

- ❑ **An example of using Asymptotic Analysis to compare two algorithms:**
- ❑ In a later topic, we will be looking at various **sorting algorithms** and as you can see from the graph comparing two such algorithms:
 - **Merge Sort** with a running time complexity of **$O(n \log n)$**
 - **Insertion Sort** with a running time complexity of **$O(n^2)$**
- ❑ Merge Sort is more efficient than Insertion Sort as the **input size, n (number of elements to sort)** increases.

Algorithms

- ❑ In this Learning Unit, we will cover various algorithms for searching and sorting e.g., binary search, insertion sort, quick sort etc.

- ❑ What about Data Structures?
 - 2nd half of the semester
 - Stacks, Queues, Linked Lists