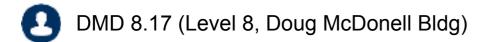


COMP90038 Algorithms and Complexity

Lecture 3: Gro https://eduassistpro.githtlbm/Efficiency (with thanks to Hara edu_assist_pro

Toby Murray







@tobycmurray

Update



- Compulsory Quizzes (first one closes Tuesday Week 3)
- Tutorials start this week
- Background knawledget pateh Expantuterials:
 - Weeks 2 and 3 https://eduassistpro.github.io/
 - Thursday 1-2pm and 2:15-Alice Hoy, Room 101
- Consultation Hours
- Discussion Board

Algorithm Efficiency



Two **algorithms** for computing gcd:

```
Assignment Project Exam Help

https://eduassistpro.github.io/eturn m;

Add WeChat edu_assist_pro
```

Why is one more efficient than the other?

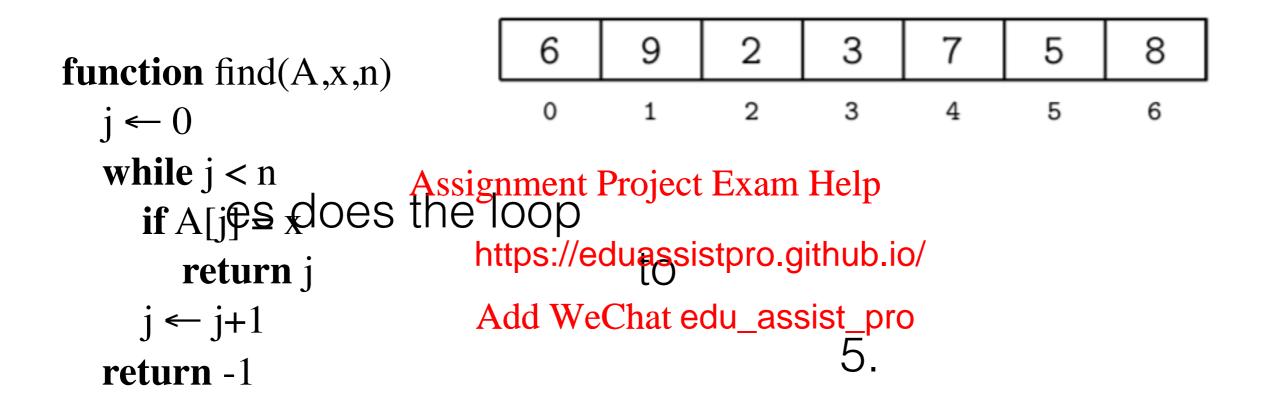
What does "efficient" even mean?

How can we talk about these things precisely?



```
A: Y x: 7 n: 7 j: 2
function find(A,x,n)
  j \leftarrow 0
                             A[i]
  while j < n
                      Assignment Project Exam Help
     if A[j] = x
                        https://eduassistpro.github.io/
       return j
                          Add WeChat edu_assist_pro
    j \leftarrow j+1
  return -1
                            Let's trace the execution of find(Y,7,7)
                                             (returns 4)
```





How many times does the loop run to find 6? 1.

How many times does the loop run to find 99? 7.

(the length of the array)

Assessing Algorithm "Efficiency"



- Resources consumed: time and space
- We want to assess efficiency as a function of input size
 - statement atical vision Project Exam Help
 - Average case
 https://eduassistpro.github.io/
 - Add WeChat edu_assist_pro
- Knowledge about input peculiarities may affect the choice of algorithm
- The right choice of algorithm may also depend on the programming language used for implementation

Running Time Dependencies MELBOURNE

- There are many things that a program's running time depends on:
 - 1. Complexity of the algorithms used
 - 2.Input to the program

Assignment Project Exam Help 3. Underlying machi orv a ory architecture

https://eduassistpro.github.io/4.Language/compiler/operatin

Add WeChat edu_assist_pro

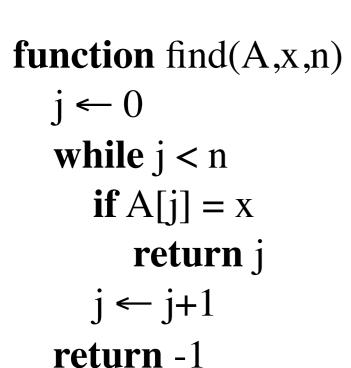
- Since we want to compare algorithms we ignore (3) and (4); just consider units of time
- Use a natural number n to quantify (2)—size of the input
- Express (1) as a function of n



```
3
                                                                5
                                                                       8
                               6
                                      9
function find(A,x,n)
                                             2
                                                   3
                                      1
                                                                5
                               0
                                                                       6
  j \leftarrow 0
  while i < n
                      Assignment Project Exam Help
                                                 measure the size, n,
     if A[j] = x
       return to this albitos: //eduassistpro.github.io/
                           Add WeChat edu_assist_pro
     j \leftarrow j+1
                                  n = the length of the array
  return -1
```

How should we quantify the cost to run this algorithm? roughly, number of times the loop runs (later in this lecture we will be more precise)







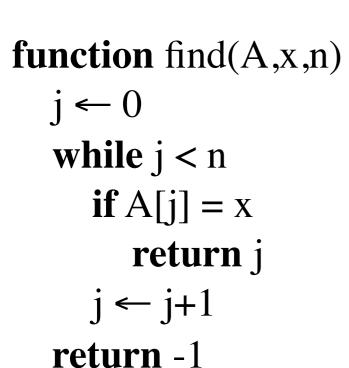
Assignment Project Exam Help

https://eduassistpro.github.id/case input?

Addrwachat edu_assistem't contain the item, x, we are searching for

Worst case time complexity: *n* (since the loop runs *n* times in that case)







Assignment Project Exam Help

https://eduassistpro.ghes.tocase input?

Add We Chatredu_assist here the item, x, we are searching for in the first position

Best case time complexity: 1 (since the loop runs once in that case)

Estimating Time Consumption



- Number of loop iterations is not a good estimate of running time.
- Better is to identify the algorithm's basic operation and how many ti https://eduassistpro.github.io/
- If *c* is the cost of a **basic o** n and *g(n)* is the number of times the operation is performed for input size *n*,

then running time $t(n) \approx c \cdot g(n)$



function find(A,x,n)

$$j \leftarrow 0$$

while $j < n$
if A[j] = x
return j
 $j \leftarrow j+1$

return -1



Assignment Project Exam Help

https://eduassistpro.githcoperation here?

Add WeChat edu_assist_pro the comparison A[j] = x

Rule of thumb: the most expensive operation executed each time in the inner-most loop of the program

Examples: Input Size and Basic Operation



Problem	Size Measure	Basic Operation						
Search in a list Assignment Prøject Exam Help Key comparison of <i>n</i> items								
Multiply two matrices of floats	https://eduassistpro.githu							
Compute an	log n	Float multiplication						
Graph problem	Number of nodes and edges	Visiting a node						

Best, Average and Worst Case



- The running time t(n) may well depend on more than just n
- Worse case: analysis makes the most pessimistic assumptions about the input
- Best case: analysis ignations the input
 about the input
 https://eduassistpro.github.io/
- Average case: analysis alms to edu_assist proceed running time across all possible input of size n
 (Note: not an average of the worst and best cases)
- Amortised analysis takes context of running an algorithm into account, calculates cost spread over many runs. Used for "self-organising" data structures that adapt to their usage

Large Input is what Matters MELBOURNE



Small input does not properly stress an algorithm

Assignment Project Exam Help

https://eduassistpro.github.io/

for small values of and weather edu_assist potois similar

 Only as we let m and n grow large do we witness (big) differences in performance.

Guessing Game Example



 Guess which number I am thinking of, between 1 and n (inclusive). I will tell you if it is higher or lower than each guess.

Assignment Project Exam Help

https://eduassistpro.github.io/

100

Wrong. My number is haan750.

We are **halving** the search space each time.

Basic operation:

(Worse case) complexity: log n

The Tyranny of Growth Rate



n	log ₂ n	n	n log ₂ n	n ²	n³	2 ⁿ	n!
10 ¹	3	10 ¹	3 ·10¹	102	10 ³	10 ³	4 · 106
10 ²	7	. •	ignment Proje	. •	. •	1030	9 · 10157
10 ³	10		Add Wethat			_	_

10³⁰ is 1,000 times the number of nano-seconds since the Big Bang.

At a rate of a trillion (10¹²) operations per second, executing 2¹⁰⁰ operations would take a computer in the order of 10¹⁰ years.

That is more than the estimated age of the Earth

The Tyranny of Growth Rate MELBOURNE

Assignment Project Exam Help

https://eduassistpro.github.io/

Add WeChat edu_assist_pro

Functions Often Met in Algorithm Classification



- 1: Running time independent of input
- **log n:** typical for "divide an conquer" solutions, for example lookup in a balanced search tree
- Linear (n): When Assignment Project Exemplifications on the control of the control
- n log n: Each input once and processing involves other elem for example, sorting.
- **n**², **n**³: Quadratic, cubic. Processing all pairs (triples) of elements.
- 2n: Exponential. Processing all subsets of elements.

Asymptotic Analysis



- We are interested interested in the second interested in the second interested in the second interested in the second in the s
 - https://eduassistpro.github.io/
 - Ignore constan Add WeChat edu_assist_pro
 - Ignore small input sizes

Asymptotics



- f(n) < g(n) iff $\lim_{n \to \infty} \frac{f(n)}{g(n)} = 0$
- That is, g approaches infinity faster than f
- $1 < \log n < n^{\varepsilon}$ Assignment Braject Exam Help where $0 < \varepsilon < 1$ https://eduassistpro.github.io/
- In asymptotic analysis, think big!
 - e.g., $\log n < n^{0.0001}$, even though for $n = 10^{100}$, 100 > 1.023.
 - Try it for $n = 10^{1000000}$

Big-Oh Notation



- O(g(n)) denotes the set of functions that grow no faster than g, asymptotically.
- Formal definition: We write

Assignment Project Exam Help

https://eduassistpro.github.io/

Add WeChat edu_assist_pro

when, for some c and n_0

$$n > n_0 \Rightarrow t(n) < c \cdot g(n)$$

• For example: $1 + 2 + ... + n \in O(n^2)$

Big-Oh: What $t(n) \in O(g(n))$ Means



Assignment Project Exam Help

https://eduassistpro.github.io/

Add WeChat edu_assist_pro

Big-Oh Pitfalls



- Levitin's notation $t(n) \in O(g(n))$ is meaningful, but not standard.
- Other authors use thing.
- As O provides an t is correct to say both $3n \in O(n^2)$ and $3n \in O(n^2)$ and $3n \in O(n^2)$ and $3n \in O(n^2)$ to can see why using '=' is confusing); the latter, $3n \in O(n)$, is of course more precise and useful.
- Note that c and n_0 may be large.

Big-Omega and Big-Theta



- **Big Omega:** $\Omega(g(n))$ denotes the set of functions that grow no slower than g, asymptotically, so Ω is for **lower bounds**_{gnment Project Exam Help}
 - $t(n) \in \Omega(g(n))$ if https://eduassistpro.githgb.ig(n), for some n_0 and C.
- Big Theta: □ is for exact order of growth.
 - $t(n) \in \Theta(g(n))$ iff $t(n) \in O(g(n))$ and $t(n) \in \Omega(g(n))$.

Big-Omega: What $t(n) \in \Omega(g(n))$ Means



Assignment Project Exam Help

https://eduassistpro.github.io/

Add WeChat edu_assist_pro

Big-Theta: What $t(n) \in \Theta(g(n))$ Means



Assignment Project Exam Help

https://eduassistpro.github.io/

Add WeChat edu_assist_pro

Establishing Growth Rate



We can use the definition of O directly.

$$t(n) \in O(g(n))$$
 iff: $n > n_0 \Rightarrow t(n) < c \cdot g(n)$
Assignment Project Exam Help

https://eduassistpro.github.io/

- **Exercise:** use this to show $\frac{1}{4}$ assist_pro $1 + 2 + ... + n \in O(n^2)$
- Also show that: $17n^2 + 85n + 1024 \in O(n^2)$

$$1 + 2 + ... + n \in O(n^2)$$



Find some *c* and n_0 such that, for all $n > n_0$

$$1 + 2 + \dots + n < c \cdot n^2$$

$$1 + 2 + ... + n$$

$$= \frac{n(n+1)}{2}$$

Assignment Project Exam Help

https://eduassistpro.github.io/

Add WeChat edu_assist_pro

$$=\frac{n^2+n}{2}$$

$$< n^2 + n$$
 (for n > 0)

$$< n^2 + n^2 \text{ (for n > 1)}$$

$$= 2n^2$$

$$\sum_{i=1}^{n} i = 1 + 2 + \dots + n = \frac{n(n+1)}{2} \approx \frac{1}{2}n^{2}$$

Choose $n_0 = 1$, c = 2

$$17n^2 + 85n + 1024 \in O(n^2)$$



Find some c and n_0 such that, for all $n > n_0$ $17n^2 + 85n + 1024 < c \cdot n^2$

Guess c = 18 Need to prove:

17 nAssignment Project 4 xam 186/p2

i.e. $85n + \frac{\text{https://eduassistpro.github.io/}}{1.00}$

Add WeChat edu_assist_pro

Guess $n_0 = 1024$ Check if: $85n_0 + 1024 < n_0^2$

 $85 \cdot 1024 + 1024 < 1024 \cdot 1024$

i.e. 86·1024 < 1024·1024 Clearly true.

Choose c = 18, $n_0 = 1024$

$$17n^2 + 85n + 1024 \in O(n^2)$$



Find some c and n_0 such that, for all $n > n_0$ $17n^2 + 85n + 1024 < c \cdot n^2$

Alternative: Let
$$c = 17 + 85 + 1024$$

$$< 17n^2 + 85n^2$$
 https://eduassistpro.github.io/
Add WeChat edu_assist_pro

$$= (17 + 85 + 1024)n^2$$

Choose
$$c = 17 + 85 + 1024$$
, $n_0 = 1$

Of course, this works for *any* polynomial.