My Notes 2024

Department of Computer Science, Copenhagen University (DIKU)
Algorithms and Data Structures

Simon Winther
zlp616@alumni.ku.dk

February 10, 2024

Contents

List of Algorithms

List of Theorems

List of Questions

1 Rules of arithmetic

1.1 Logarithms

$$(\log_b(x))^a = \log_b(x^a) \tag{1.1}$$
$$\log_b(x^a) = a \cdot \log_b(x) \tag{1.2}$$
$$\log_b(x \cdot y) = \log_b(x) + \log_b(y) \tag{1.3}$$
$$\log_b(x/y) = \log_b(x) - \log_b(y) \tag{1.4}$$
$$\log_b(1) = 0 \tag{1.5}$$
$$\log_b(b) = 1 \tag{1.6}$$
$$\log_b(b^x) = x \tag{1.7}$$
$$b^{\log_b(x)} = x \tag{1.8}$$

2 5/2: Introduction, Proofs, Loop Invariants, CLRS 2.1

Chapter Quick Summary Notes (CRLS 2.1)

This box can contain summaries, key points, and important definitions from the reading material assigned (CLRS 2.1). Highlight the main concepts, proofs, and examples discussed in the text.

2.1 Loop Invariants:

Loop invariants are used to prove the correctness of algorithms. They are used to show that the algorithm is working as expected. The loop invariant is a property that holds before and after each iteration of the loop. When you're using a loop invariant, you need to show three things:

1. **Initialization:** It is true prior to the first iteration of the loop.
2. **Maintenance:** If it is true before an iteration of the loop, it remains true before the next iteration.
3. **Termination:** The loop terminates, and when it terminates, the invariant - usually along with the reason that the loop terminated - gives us a useful property that helps show that the algorithm is correct.

When the first two properties are true, the loop invariant is true prior to every iterarion of the loop. When the third property is true, the loop invariant is true after the last iteration of the loop, giving us a useful property that helps show that the algorithm is correct.

A loop-invariant proof is a form of mathematical induction, where to prove

that a property holds, you prove a base case and an inductive step. Here, showing the invariant holds before the first iteration of the loop is like the base case, and showing that the invariant holds from one iteration to the next (from iteration to iteration) is like the inductive step. The third property is perhaps the most important one, because it is where the loop invariant is used to show that the algorithm is correct.

2.2 Proofs:

Test.

2.3 Insertion-Sort and cost analysis

Algorithm 1 Insertion-Sort(A)

Input: $A[1..n]$, an array of numbers.
Output: $A[1..n]$, sorted in non-decreasing order.

		cost	times
1:	**for** `j = 2 to A.length` **do**	c_1	n
2:	`key = A[j]`	c_2	$n-1$
3:	`i = j - 1`	0	$n-1$
4:	**while** `i > 0 and A[i] > key` **do**	c_4	$n-1$
5:	`A[i + 1] = A[i]`	c_5	$\sum_{j=2}^{n} t_j$
6:	`i = i - 1`	c_6	$\sum_{j=2}^{n}(t_j-1)$
7:	`A[i + 1] = key`	c_7	$\sum_{j=2}^{n}(t_j-1)$

where t_j is the number of times the while loop test in line 4 is executed for that value of j. $T(n)$ also called the recurrence relation, is given by:

$$\begin{aligned} T(n) &= c_1 n + c_2(n-1) + c_4(n-1) + c_5\sum_{j=2}^{n} t_j + c_6\sum_{j=2}^{n}(t_j-1) + c_7\sum_{j=2}^{n}(t_j-1) \\ &= (c_1+c_2+c_4)n - (c_2+c_4+c_5+c_6+c_7) + (c_5+c_6+c_7)\sum_{j=2}^{n}(t_j-1) \end{aligned}$$

For the worst case we have $t_j = j$ for $j = 2, 3, \ldots, n$ because each j iteration it has to move the key to the first position and we know that $\sum_{i=1}^{n} = \frac{n(n-1)}{2}$ and so hence we would have $T(n)$ as follows:

$$\begin{aligned} T(n) &= (c_1+c_2+c_4)n - (c_2+c_4+c_5+c_6+c_7) + (c_5+c_6+c_7)\sum_{j=2}^{n}(j-1) \\ &= (c_1+c_2+c_4)n - (c_2+c_4+c_5+c_6+c_7) + (c_5+c_6+c_7)\left(\frac{n(n-1)}{2} - (n-1)\right) \\ &= an^2 + bn + c \end{aligned}$$

And thereby we have shown that the worst-case running time of insertion-sort is $\Theta(n^2)$.

2.4 Exercises:

Question 2.1: Proof (direct proof)

Prove that $x^2 - y^2 = 1$ has no solutions for positive integers x, y (the positive integers are the numbers $1, 2, \ldots$).

Proof for Question 1: We can consider the equation in the form of difference of two squares:

$$x^2 - y^2 = (x - y)(x + y) = 1$$

We can now recall that for positive integers a and b, if $ab = 1$, then both a and b must be 1. Since 1 is the only positive integer multiplied by itself that gives 1. Hence, when we consider the equation $(x - y)(x + y) = 1$, we can see that the only way to get 1 is if $x - y = 1$ and $x + y = 1$. This gives us the system of equations:

$$\begin{cases} x - y = 1 \\ x + y = 1 \end{cases}$$

Adding the two equations gives us $2x + 0y = 2x = 2$, and so $x = 1$. Substituting $x = 1$ into the second equation gives us $1 + y = 1$, and so $y = 0$. But 0 is not a positive integer, and so the equation $x^2 - y^2 = 1$ has no solutions for positive integers x, y. □

Question 2.2: Proof (direct proof)

Prove that $x^2 - y^2 = 10$ has no solutions for positive integers x, y

Proof for Question 2: Again, we can consider the equation in the form of difference of two squares:

$$x^2 - y^2 = (x - y)(x + y) = 10$$

This time we will in fact assume that $\exists x, y \in \mathbb{Z}^+$ such that $x^2 - y^2 = 10$. If we consider the equation $(x - y)(x + y) = 10$, there are four ways to factorize 10 into two numbers, and they are:

$$\begin{cases} x - y = 1, x + y = 10 \\ x - y = 2, x + y = 5 \\ x - y = 5, x + y = 2 \\ x - y = 10, x + y = 1 \end{cases}$$

But because of symmetry, we can see that the first and last equations are the same, and the second and third equations are the same. If we solve the first equations by adding the two equations:

$$x - y + x + y = 1 + 10 \Rightarrow 2x = 11 \Rightarrow x = \frac{11}{2}$$

But x is a positive integer, and so the first equation has no solutions. We can then also solve for the second equations by adding the two equations:

$$x - y + x + y = 2 + 5 \Rightarrow 2x = 7 \Rightarrow x = \frac{7}{2}$$

But x is a rational number, and so the second equation has no solutions. □

Question 2.3: Proof (proof by contradiction)

Prove that if a is a rational number and b is an irrational number, then $a + b$ is an irrational number. (A rational number is one that can be written in the form $\frac{x}{y}$, for some $x, y \in \mathbb{Z}$ and $y \neq 0$).

Proof for Question 3: We can prove this by contradiction. Assume that $a + b$ is a rational number. Then $a + b = \frac{x}{y}$ for some $x, y \in \mathbb{Z}$ and $y \neq 0$. We know that a is a rational number, hence $a = \frac{p}{q}$ for some $p, q \in \mathbb{Z}$ and $q \neq 0$, and so b could be written as:

$$b = \frac{x}{y} - \frac{p}{q} = \frac{xq - py}{qy}$$

But this is a rational number, which is a contradiction because b was assumed to be irrational. Therefore, the initial assumption that $a + b$ is rational must be false, so $a + b$ is irrational. □

Question 2.4: Proof (✠)

Give a formula for the sum of the first n odd numbers.

$$f(n) = \sum_{i=1}^{n} (2i - 1)$$

Question 2.5: Proof (proof by induction)

Show that $n! > 2^n$ for $n \geq 4$.

Proof for Question 5: Let's prove by induction. First we will test our base case, which will be:

$$4! > 2^4$$
$$24 > 16$$

This holds, and now we can assume that $n! > 2^n$ for some $n \geq 4$. We can then show that $(n + 1)! > 2^{n+1}$ as follows:

$$(n + 1)! > 2^{n+1}$$
$$n! \cdot (n + 1) > 2^n \cdot 2^1$$

We know that $n! > 2^n$ by our assumption and $(n+1) \geq 2^1$ hence the inequality holds and thereby the proof is complete and $n! > 2^n$ for $n \geq 4$. □

Question 2.6: Proof (proof by induction)

The Fibonacci numbers are defined as $f_0 = 0, f_1 = 1$, and $f_n = f_{n-1} + f_{n-2}$, for $n \geq 2$. The first Fibonacci numbers are thus $0, 1, 1, 2, 3, 5, 8, 13, 21, \ldots$ Prove that

$$\sum_{i=1}^{n} f_i^2 = f_n f_{n+1}$$

Proof for Question 6: We will prove this by induction. First we will test our base case, which will be:

$$\begin{aligned}
\sum_{i=1}^{2} f_i^2 &= f_2 f_{2+1} \\
f_1^2 + f_2^2 &= f_2 f_3 \\
1^2 + 1^2 &= 1 \cdot 2 \\
2 &= 2
\end{aligned}$$

This holds, and now we can assume that $\sum_{i=1}^{n} f_i^2 = f_n f_{n+1}$ holds for some $n \geq 2$. We can then show that $\sum_{i=1}^{n+1} f_i^2$ holds as well:

$$\begin{aligned}
\sum_{i=1}^{n+1} f_i^2 &= \sum_{i=1}^{n} f_i^2 + f_{n+1}^2 \\
\sum_{i=1}^{n+1} f_i^2 &= f_n f_{n+1} + f_{n+1}^2 \\
\sum_{i=1}^{n+1} f_i^2 &= f_{n+1}(f_n + f_{n+1}) \\
\sum_{i=1}^{n+1} f_i^2 &= f_{n+1}(f_n + f_{n+1}) \\
\sum_{i=1}^{n+1} f_i^2 &= f_{n+1} f_{n+2}
\end{aligned}$$

After proving the base case and assuming the induction hypothesis $\sum_{i=1}^{n} f_i^2 = f_n f_{n+1}$ holds for some $n \geq 2$, we have shown that $\sum_{i=1}^{n+1} f_i^2 = f_{n+1} f_{n+2}$ holds as well. Therefore, the proof is complete. □

Question 2.7: Proof (very good induction proof)

Show that

$$\sum_{i=1}^{n} f_i = f_{n+2} - 1$$

Proof for Question 7: We will prove this by induction. First we will test our base case, which will be:

$$\begin{aligned}\sum_{i=1}^{2} f_i &= f_{2+2} - 1 \\ f_1 + f_2 &= f_4 - 1 \\ 1 + 1 &= 3 - 1 \\ 2 &= 2\end{aligned}$$

This holds, and now we can assume that $\sum_{i=1}^{n} f_i = f_{n+2} - 1$ holds for some $n \geq 2$. We can then show that $\sum_{i=1}^{n+1} f_i$ holds as well:

$$\begin{aligned}\sum_{i=1}^{n+1} f_i &= \sum_{i=1}^{n} f_i + f_{n+1} \\ \sum_{i=1}^{n+1} f_i &= f_{n+2} - 1 + f_{n+1} \\ \sum_{i=1}^{n+1} f_i &= f_{n+2} + f_{n+1} - 1 \\ \sum_{i=1}^{n+1} f_i &= f_{n+3} - 1\end{aligned}$$

After proving the base case and assuming the induction hypothesis $\sum_{i=1}^{n} f_i = f_{n+2} - 1$ holds for some $n \geq 2$, we have shown that $\sum_{i=1}^{n+1} f_i = f_{n+3} - 1$ holds as well. Therefore, the proof is complete. □

Question 2.8: Proof (very nice induction proof too)

Show that

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

Proof for Question 8: We will prove this by induction. First we will test

our base case, which will be:

$$\begin{aligned}\sum_{i=1}^{1}\frac{1}{i(i+1)}&=\frac{1}{1+1}\\ \frac{1}{1(1+1)}&=\frac{1}{2}\\ \frac{1}{2}&=\frac{1}{2}\end{aligned}$$

This holds, and now we can assume that $\sum_{i=1}^{n}\frac{1}{i(i+1)}=\frac{n}{n+1}$ holds for some $n\geq 1$. We can then show that $\sum_{i=1}^{n+1}\frac{1}{i(i+1)}$ holds as well:

$$\begin{aligned}\sum_{i=1}^{n+1}\frac{1}{i(i+1)}&=\sum_{i=1}^{n}\frac{1}{i(i+1)}+\frac{1}{(n+1)(n+1+1)}\\ \sum_{i=1}^{n+1}\frac{1}{i(i+1)}&=\frac{n}{n+1}+\frac{1}{(n+1)(n+2)}\\ \sum_{i=1}^{n+1}\frac{1}{i(i+1)}&=\frac{n(n+1)(n+2)}{(n+1)(n+1)(n+2)}+\frac{1(n+1)}{(n+1)(n+1)(n+2)}\\ \sum_{i=1}^{n+1}\frac{1}{i(i+1)}&=\frac{n(n+1)(n+2)+(n+1)}{(n+1)(n+1)(n+2)}\\ \sum_{i=1}^{n+1}\frac{1}{i(i+1)}&=\frac{n(n+2)+1}{(n+1)(n+2)}\\ \sum_{i=1}^{n+1}\frac{1}{i(i+1)}&=\frac{n^2+2n+1}{(n+1)(n+2)}\\ \sum_{i=1}^{n+1}\frac{1}{i(i+1)}&=\frac{(n+1)^2}{(n+1)(n+2)}\\ \sum_{i=1}^{n+1}\frac{1}{i(i+1)}&=\frac{n+1}{n+2}\end{aligned}$$

After proving the base case and assuming the induction hypothesis $\sum_{i=1}^{n}\frac{1}{i(i+1)}=\frac{n}{n+1}$ holds for some $n\geq 1$, we have shown that $\sum_{i=1}^{n+1}\frac{1}{i(i+1)}=\frac{n+1}{n+2}$ holds as well. Therefore, the proof is complete. □

Question 2.9: Invariants (Opgave 19, A&D (maj 2021), AU)

Given a non-negative integer x and a positive integer y, the following algorithm calculates $\lfloor x/y \rfloor$:

Algorithm 2 Division(x, y)

Input: Integer $x \geq 0$ and $y \geq 1$.
Output: $r = \lfloor x/y \rfloor$

```
1: r ← 0
2: while x ≥ y do
3:     x ← x − y
4:     r ← r + 1
```

The following 5 loop environments are given explain if they are loop invariants or not:

1. $r = \lfloor x/y \rfloor$.
   Answer: No. r is initialized to 0 and the loop invariant states at initialization that $r = \lfloor x/y \rfloor$, which is simply just not true. And for Maintenance it wouldn't hold either because r is incremented by 1 each time the loop runs. It would hold only at termination, but that is not enough to be a loop invariant only to show that the algorithm is correct.

2. $r = \lfloor x_0/y_0 \rfloor$.
   Answer: No, for the same reason as the first loop invariant. And here x_0 and y_0 are static values, which means that in the maintenance step, the values wouldn't change and could not be a loop invariant.

3. $r = \lfloor (x_0 - x)/y \rfloor$.
   Answer: Yes. This is a loop invariant. It holds at initialization, maintenance, and termination. At initialization, $x = x_0$ and so $r = \lfloor (x_0 - x_0)/y \rfloor = \lfloor 0/y \rfloor = 0$. At maintenance,

4. $x + ry = x_0$.
   Answer:

5. $r(x - x_0) = y$.
   Answer:

3 7/2: Time Complexity and Asymptotic Notation, CLRS 2.2, 3

3.1 Time Complexity and Asymptotic Notation

In practice the notation is often used to denote a (unnamed) function, e.g.:

- $g(n) = O(n^2)$ means $g(n) = f(n)$ for some $f(n) \in O(n^2)$
- $g(n) = n^2 + \Omega(n)$ means $g(n) = n^2 + f(n)$ for some $f(n) \in \Omega(n)$

3.1.1 Comparison of Growth Rates for Common Functions

$$1 < \log n < \sqrt{n} < n < n \log n < n^2 < n^3 < \cdots < 2^n < 3^n < \cdots < n^n \tag{3.1}$$

3.1.2 Formal Definitions of Asymptotic Notation

Formal definitions: For a non negative function $g(n) : \mathbb{N} \to \mathbb{R}$ er

$$\begin{aligned}
O(g(n)) &= \{f(n) \mid \exists c > 0, n_0 > 0 : \forall n \geq n_0 \to 0 \leq f(n) \leq cg(n)\} \\
\Omega(g(n)) &= \{f(n) \mid \exists c > 0, n_0 > 0 : \forall n \geq n_0 \to cg(n) \leq f(n)\} \\
\Theta(g(n)) &= \{f(n) \mid \exists c_1, c_2 > 0, n_0 > 0 : \forall n \geq n_0 \to c_1 g(n) \leq f(n) \leq c_2 g(n)\} \\
o(g(n)) &= \{f(n) \mid \forall c > 0 : \exists n_0 > 0 : \forall n \geq n_0 \Rightarrow 0 \leq f(n) < cg(n)\} \\
\omega(g(n)) &= \{f(n) \mid \forall c > 0 : \exists n_0 > 0 : \forall n \geq n_0 \Rightarrow cg(n) < f(n)\}
\end{aligned}$$

Little-o and little-omega: are used to denote strict inequalities, e.g.:

$$\begin{aligned}
f(n) &= o(g(n)) \quad (\text{" } > \text{"}) \\
f(n) &= \Omega(g(n)) \quad (\text{" } \geq \text{"}) \\
f(n) &= \Theta(g(n)) \quad (\text{" } = \text{"}) \\
f(n) &= O(g(n)) \quad (\text{" } \leq \text{"}) \\
f(n) &= o(g(n)) \quad (\text{" } < \text{"})
\end{aligned}$$

3.2 Master Theorem

Theorem 3.2 (Master Theorem)**.** *Let $a \geq 1$ and $b > 1$ be constants, let $f(n)$ be a function, and let $T(n)$ be defined on the non-negative integers by the recurrence*

$$T(n) = aT\left(\frac{n}{b}\right) + f(n)$$

where we interpret n/b to mean either $\lfloor n/b \rfloor$ or $\lceil n/b \rceil$. Then $T(n)$ can be asymptotically estimated as follows:

1. *If $f(n) = O(n^{\log_b a - \epsilon})$ for some $\epsilon > 0$, then $T(n) = \Theta(n^{\log_b a})$.*
2. *If $f(n) = \Theta(n^{\log_b a})$, then $T(n) = \Theta(n^{\log_b a} \log n)$.*
3. *If $f(n) = \Omega(n^{\log_b a + \epsilon})$ for some $\epsilon > 0$, and if $af(n/b) \leq kf(n)$ for some constant $k < 1$ and sufficiently large n, then $T(n) = \Theta(f(n))$.*

3.3 Exercises

Question 3.1: Time Complexity and Asymptotic Notation

Determine the whether the functions below are O, Ω, or Θ of the other functions.

Statement	**Yes**	**No**
$(\log n)^2$ is $O(n^2)$	~~Yes~~	No
$n \cdot \log n$ is $O(n^2)$	~~Yes~~	No
$\sqrt{n}$ is $O((\log n)^3)$	Yes	No
$1 + \log(n^2)$ is $O((\log n)^2)$	Yes	No
$\log n + \log(n!)$ is $O(n^2)$	Yes	No
n^3 is $O(n)$	Yes	No
$\sqrt{n} \cdot \log n$ is $O(n)$	Yes	No
n^3 is $O(\log(n!))$	Yes	No
n^2 is $O(n^{2/3})$	Yes	No
$7 \log n + \log(n!)$ is $O(n \cdot \log n)$	Yes	No
$2 \log n$ is $\Omega(n^{0.01})$	Yes	No
$(\log n)^3 + 3^n$ is $\Omega(2^n)$	Yes	No