Lesson Review 2 — Euclid's algorithm

Isak Oswald
s225375329

April 5, 2025

Question 1

1 Explaining the proof of convergence

Euclid's algorithm is an interest algorithm and was the very first algorithm that I learnt within SIT192, therefore, I am excited to come back and re-visit it with an expanded knowledge. Euclid's algorithm relies on highlighting that the algorithm will eventually reach a remainder of 0, at which point the GCD of both a and b are found. Euclid's algorithm at its core works by replacing the pair of numbers a and b with b and $amodb$ which breaks the problem down from its largest state, into simpler and smaller problems each time and continues until the GCD is obvious (as the problem has been divided into a really small one). At its core, Euclid's algorithm is just repeated subtraction (however we just speed this up with division) of a and b which retains the GCD of the two numbers. You can think about this as numbers on a number line, if we subtract b from a as many times as we can before finding the remainder (which is just $amodb$) we are dividing up the entire number line (a) into smaller chunks (b) and then finding the the GCD. I think that this is important to understand before diving into an actual proof as a high level understanding develops into a more filled out understanding. Have a look at the diagram below.

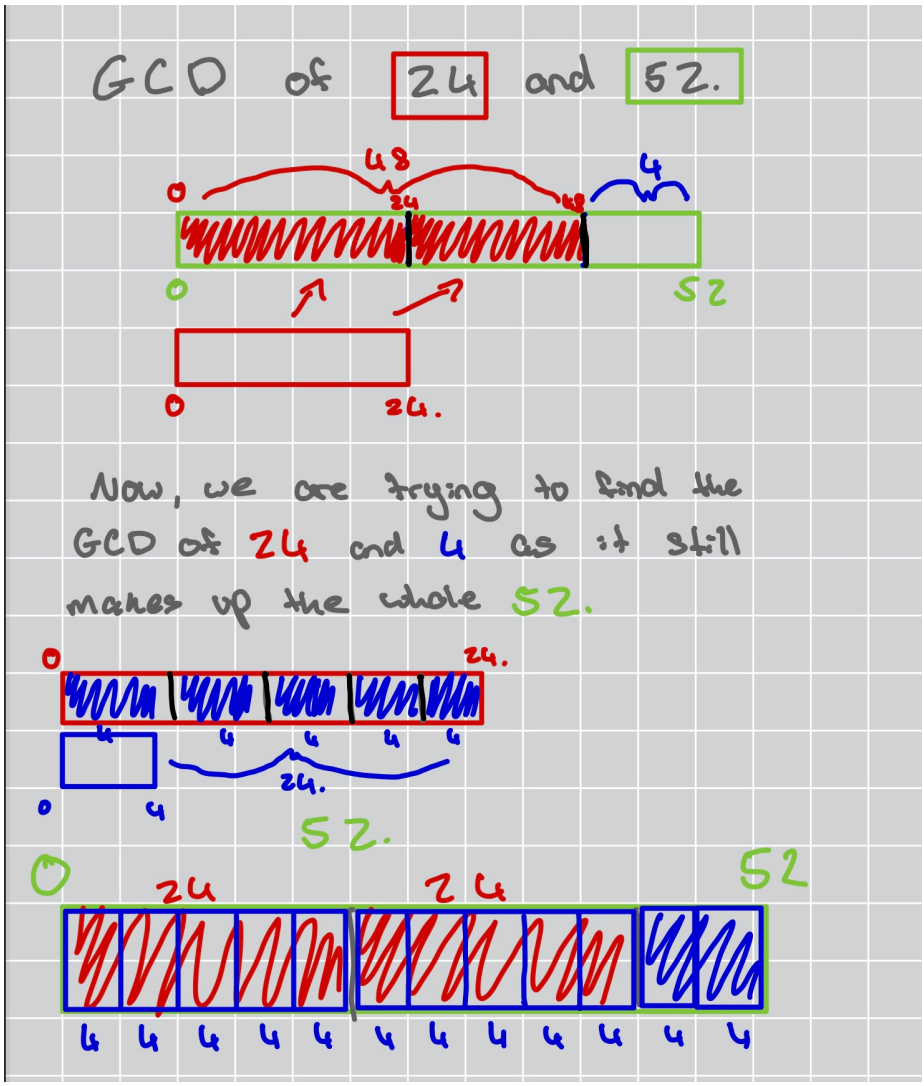

Figure 1: Euclid's Algorithm Assisting Sketch

As you can see we are finding the GCD of 24 and 52 as denoted by the colours. First we take *amodb* which is 4. This means that 24 goes into 52 twice with a remainder, r of 4. Now we can see that two blocks of 24 plus a block of 4 make up the **entire** block of 52. Now, we are trying to find the GCD of 24 and 4 as that GCD will be the GCD of the original a and b (because the smaller blocks make up the larger block of 52). We know that the GCD of 24 and 4 will have a remainder of 0 as 4(6) is 24, this means that 4 makes up both the 24 block (since if it can make up one block it can also make up two)and the remaining 4 block. This means that the GCD of 52 and 24 is indeed 4. As you

can see in this intuitive representation we are just simplifying the problem down into blocks. Now, this is a little informal for me, and I would like to create a more rigorous proof.

1.1 Inductive proof

- Base Case:

The algorithm starts with two integer numbers a and b as we discussed before. After the first step of the algorithm, we are left with the pair $b, a \bmod b$. If $a \bmod b$ is zero, that means the algorithm stops because the remain is zero (as b fits perfectly into a) and therefor the GCD of both a and b is of-course just b. This proves the base case of the algorithm terminating after just one single step of the algorithm (for example when $a = 15$ and $b = 5$). Keep in mind that 5 is also a factor of itself which is why b is the GCD if r is zero.

- Inductive hypothesis:

Assume that in each step of the algorithm starting from the integer pairs (a_1, b_1) after some finite number of steps, it will eventually reach a remainder of 0 and which point we have found our GCD which is b_n. We need to show that the inductive step that the pair $b, a \bmod b$ is a step towards zero. This answers the question "why does it drive towards zero?".

In each step, the size of the second number in the pair decreases (b) because $amodb$ is **always** smaller than b. This is because if we are finding some multiple n such that $a = bn + r$ it will always be smaller then the original b. This proves that the algorithm has a finite number of steps and the algorithms steps are indeed finite. This means that the r in $a = bn_1 + r_1$ will always be less than $a = bn_2 + r_2$ as since the algorithm strictly decreases, $r_0 \geq r_1 \geq r_2 \geq \cdots \geq 0$. If you are familiar with the well ordering principle [1] which states that any non-empty set **must** have a element that is less than all of the following elements and since r is strictly decreasing as discussed before, there will be some number of finite steps where the remained **has** to become 0 no matter how many steps the algorithm takes.

In saying this, this means that the algorithm converges (reaches 0) after a finite amount of steps and will *always* find the GCD of the pair of integers a, b.

Question 2

2 Reflection

2.1 What was the main learning outcome that I took from this activity?

The main thing that I took away from this activity was gaining a deeper understanding of Euclid's algorithm and understanding the 'why' behind why it works in both an intuitive and rigorous way. I worked through the examples on the Cloud Deakin page and was able to gain insight into how the algorithm works on each iteration, and what is happening to both a and b during and after each iteration. I also got to use induction to help prove that the algorithm would terminate after a finite amount of steps which i learned from both my advanced module and the core proofs module.

2.2 How did I approach the tasks?

I approach this task by first completing the questions available on the Cloud Deakin website and if I was confused on the 'why' behind a concept I would either draw the number line method as shown at the start of the activity on my whiteboard to help really visualise what was happening. I also carefully considered the inductive proof, trying my best to explain 'why' in a professional and presentable way.

2.3 Did I work on this activity in a group?

Due to me doing this task over the weekend and not knowing of anyone who is actually up to this task yet (It is only week 5) I was unable to work with other students in the way you might of expected. However, I have had a chat with Julian at the start of the trimester which allowed me to gain some high level insight of why the algorithm works. Me and Julian talked about the algorithm being represented in a circle (like cycles) or through a number line. This basic understanding allowed me to dive deeper into the algorithms behaviour and its proof of convergence when studying the module.

References

[1] Wikipedia contributors. (2025, April 4). *Well-ordering principle.* Wikipedia.