Early Education on Algorithmic Complexity to Instill an Efficient Lifestyle

Lucky Jonathan Chandra - 13517136
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13517136@std.stei.itb.ac.id

***Abstract*—We often quantify the workload for our machines when we design algorithms in order to achieve the most efficient algorithm time-wise and memory-wise. In that case, we should also be able to apply the same principle to human lifestyles. It might be difficult for fully grown adults with already established habits to readjust their daily conducts. However, early education on basic algorithmic complexity should enable young children to build time and effort efficient habits in their perpetual ways to do things, and maintain them throughout their lives.**

***Keywords*—Early Education, Algorithmic Complexity, Building Habits, Efficient Lifestyles.**

I. Introduction

Some people argue that humans are creatures of habits. In many sense, that argument seems logical, as most of the daily routines in the lives of most people usually have reoccurring patterns, not only within the constraint of our occupations, but also in our habitual conducts on the most mundane tasks. We tend to do the same task exactly the same way every single time once we are comfortable with the method. This brings the question of whether or not our established habits are indeed efficient.

The author had witnessed innumerable inefficient actions that people do all the time when it comes to performing the simplest of tasks. Taking some example from the author's personal inaccuracies, the author once found himself walking a certain distance to turn off the light switch near the exit door in a room, walking back to his initial position to retrieve an object, and exiting the room via the only exit door, which was right beside the aforementioned light switch.

Such inefficiencies might seem miniscule when we are unaware of those misconducts ourselves. In reality, the accumulations of our entire life's worth of inefficiency may amount to a large quantity of both time and effort. A five minutes lost each day amounts to approximately 63 days lost after having accumulated for 50 years. Thus, we should aim towards the most efficient ways to do our tasks. Our pathing, the sequential order of our actions, the methods we choose to accomplish our goals, the tools we select to aid us, and many more other factors. Those things do matter.

The study of algorithm complexity starting from an early age is what the author hopes will resolve inefficiencies in human lives. Of course, we will never be able to reach perfection. Nevertheless, we can try to push the limit of what we can do by educating the younger generations to do better than us.

Children at a younger age tend to pick things up faster. That's why early education is usually stressed as one of the most crucial thing for a child growing up. So far in our society, the subjects prioritized for very young children to learn are literacy and basic arithmetic. Once they get past that, they are usually introduced to social education, norms, and ethics. Eventually, they will get introduced to science. This usually happens gradually at the end of their elementary year and as they start to enter junior high school. The author thinks that educating junior high school students in the foundation of Algorithmic Complexity for a meager two-periods-lesson once a week could be very beneficial in the long run.

What can a young teenager accomplish armed with this knowledge? First off, junior high students could arguably at the phase where they experience academic difficulties for the first time. In elementary school, students rarely fail, and if they do, it usually (but not always) has more to do with the environment they are in rather than with the subjects themselves, since most of the subjects taught in the elementary school is inarguably very basic and foundational. So, when studied at this time, this lesson can serve as a catalyst for the students to really start to open a brand new peripheral to their life.

The author had experienced the broadening of peripheral within the author's live two times. It usually happens when people suddenly learn about something they had never thought of before. These kind of experiences feel satisfying, humbling, and fruitful at the same time.

Starting early usually beats starting late, and having this kind of eye opener of a subject as one of the first thing that a junior high student learn, should bear positive results. Starting from quite early in life, the students would be able to organize themselves better.

With better organization, they could also construct in their mind the most optimal course of action when faced with the responsibility of decision making and method choosing, which people will experience a lot. It will also increase in frequency as we get older.

With that reasoning, this writing is going to cover some of the possible concepts that can be brought up as the educational material to be taught. Since the goal of this education is to hone the mindset to do things more efficiently, this paper will focus more on the practical aspects instead of the technical aspects that was covered in the discrete mathematics classes.

II. Theoretical Framework

Algorithmic Complexity is a field of study in computer science. It is concerned about how fast or slow a certain algorithm can perform. The purpose of this study is to determine the efficiency of algorithms from just themselves. The alternative would be to measure the real time it takes to perform the algorithm, which would then depend on the machine used to run the algorithm and adds another variable into count.

Algorithmic Complexity can be categorized into time complexity and space complexity. Time complexity is the concept that deals with the quantification of the time needed for a certain algorithm or program to finish executing, whereas space complexity is the concept that deals with the quantification of the physical memory needed during the same process. For practical use, we are safe to mainly concern ourselves with only time complexity, for the use of optimizing space complexity should be more uncommon. (Although, they do exist. For example, in the management of workspace. Nonetheless, they do not really matter all that much compared to the usage of time in almost every circumstances.)

Time complexity in computer science is often denoted as a numerical function $T(n)$, where n is the input size. Conventionally, the value of T(n) would take into account the characteristic action in a certain algorithm. For instance, in a sorting algorithm, the characteristic action would perhaps be the comparing of two values. The number of times said comparison happen can become determine the value of $T(n)$.

The value of $T(n)$ is no longer bound by external factors such as processing speed, disk speed, brand of compiler, and so forth. In other words, it can also be defined as the number of elementary steps that the program takes in order to complete itself.

Additionally, there is a method to estimate the efficiency of each algorithm asymptotically. This can be particularly useful when we are faced with a complex equation as the result of $T(n)$. For this reason, the Big Oh operator exists, where $T(n) = O(n^2)$ would mean that the algorithm could be represented as one with a quadratic time complexity.

In addition to the Big Oh, there also exist the Big Omega and Big Theta notation that possess a different purpose each.

Usually, besides the characteristic action, other actions (such as variable declaring) can be ignored when calculating the time complexity, since the purpose is to determine the efficiency of the core process. Miscellaneous actions tend to be identical despite the variety within the core process, so they do not need to be taken into account for the time complexity.

It is also common practice to determine the best, worst, and average scenario when calculating time complexity, as sometimes, the time needed depends on the actual data. These values can then be used to evaluate the algorithm's efficiency.

These numbers are reliable, for they apply for any language, machine, or compiler that you use. Since it evaluates algorithms purely from themselves, it is a reliable way to compare algorithms that aims towards the same task, and discern which one is more effective.

III. Algorithmic Complexity Foundations for Junior High School Students

A. Delivery

Usually, in order to explain the concept of algorithmic complexity, it is not uncommon to provide illustrating examples in the form of actual algorithms. However, that would not be optimal for junior high students with no prior education on programming.

It is interesting to consider that there might me a mandatory programming class for young students with the purpose of honing their computational thinking skill early on. However, since that is not the case right now, it would be detrimental to force the students to directly learn even the algorithmic notation just to understand algorithmic complexity.

It would be much easier to deliver the material in the form of word problems. Solving a word problem can harder, yet more engaging compared to straightly technical problem where only the known variables are listed.

In that case, it would not be mandatory for the students to delve into real algorithmic notations just to learn and apply the concept of Algorithmic Complexity. It should be easy enough to understand when taught at an optimal pace with the appropriate difficulty. For example, it might be possible to make a problem like this:

Akhmal has 4 sets of all unique cards. Each set of card contains 3 cards and are placed into a card sleeve. Since Akhmal is a tidy person that doesn't like getting messy by taking out all his cards, to check a card's name, Akhmal has to:

1. *Unseal the sleeve*
2. *Pull out one card halfway out of the sleeve and check its name*
3. *Push it back into the sleeve (if it is not the wanted card)*
4. *Reseal the sleeve (if it is wanted card is not in the sleeve)*

If finding the card is considered done as soon as the card with the correct name is pulled out, and each numbered actions above are assumed to require the same amount of time, which of the following circumstance would allow Akhmal to find a specific card faster:

a. *He remembers the sleeve it's in, but not the card order.*
b. *He remembers the card order, but not the sleeve it's in.*

The answer to above word problem is as follows:

*In a., the best case for Akhmal is if the card is on the first order, so that if he browses the order systematically, he only needs to unseal the sleeve once, and then immediately pulls the wanted card, amounting to **two** actions. Meanwhile, the worst scenario is if the card is on the last order, forcing him to pull and push back the first two cards in the sleeve before finding the correct one, amounting to **six** actions.*

*In b., the best case for Akhmal is if the card is on the first sleeve, which will allow him to unseal the sleeve and immediately pull out the wanted card, resulting in **two** actions, whereas the worst case scenario is to have the card on the last sleeve he checks, forcing him to unseal three sleeves, reseal two sleeves, pulling three cards, and pushing two cards, resulting in*

***ten** actions.*

Although the value of the best case is the same between the two circumstances, circumstance a. is generally better because the worst case scenario in a. takes less time than the worst case scenario in b. by a margin. ***Thus, the answer is a.***

As shown, such word problems could be engaging and challenging, and hopefully interesting to the students. There is not any real prerequisite needed to learn this, as what is used is not more than the standard arithmetic.

The teacher for this subject should act more like a guide rather than a lecturer, letting his/her students to take the initiative, and correct them only when necessary. It would be essential for the lesson to retain as much enthusiasm as possible from the students. Otherwise, it will become stale.

In a lecture based class, the lecturer usually delivers the materials in some sort of one-way communication. This could prove to be a very efficient method when it comes to teaching well-established subjects such as physics, chemistry, *et cetera.* However, rather than trying to feed knowledge, this subject is trying to hone the perceptional skill, so the students themselves must be actively trying to improve to reap the full benefit.

B. Material Coverage

Looking generally at Algorithmic Complexity, it seems unclear at first. Since Algorithmic Complexity has more to do with case-by-case analysis rather than memorized theory, it seems really difficult to arrange a suitable curriculum for it. The memorized theories that do exist, would be far more complex than the average level of junior high students.

What we can do, is to classify several main concepts, and introduce them in turns. Along with the introduction of an increasingly difficult concept each time, increasingly difficult practice problems can be given to further illustrate and test the students' understandings on the concept.

One of the possible concept could be the different levels of algorithm complexity (constant, logarithmic, linear, linear-logarithmic, quadratic, cubic, exponential, factorial). However, this would prove to be a great challenge as most students in their junior grade hasn't learned about lots of things used to explain the differences between the various levels of complexity. Although, it is possible to design the lesson to also act as an introduction to some of the newer number systems previously unknown to the students.

In order to gap the untaught material, the lesson plan needs to be drafted carefully, with taking the knowledge level of the students into consideration. In an actual running school, before implementing such a class, it would be an obvious thing to do to create a survey or another mean of data collecting. Without the proper data and statistics, it is not recommended to draft and decide on what material are supposed to be covered.

However, it is not impossible to do an outline of the core materials. If we reassess what the subject stands for, "Algorithmic Complexity Foundations" that has been described is all about sharpening case-analytical skills in order to enable students to make better decisions and build time efficient habits from early on.

Thus, it is possible to nearly completely rely on the practical sides of things. It is not completely irrational, nor is it completely unconventional. There have been some cases where a lesson plan is built fully on the practical.

The author remembers to have had a class called "Logic" in his junior high years, which consisted of nothing but trying to solve logic puzzles every period, with the main purpose of honing the students deductive reasoning capabilities.

The same idea could stand for this lesson. A lesson plan that consists on nothing but the teacher handing out problems after problems might seem lacking in preparations, but it could also be a solution if it satisfies the goal of the lesson. It cannot be determined just yet, but it cannot also be dismissed as a possibility.

The alternative idea would be to actually attempt to explain various complicated concept throughout this lesson. By underplaying some of the more difficult concepts, this might be possible, but will also be very challenging, a challenge that the author isn't quite sure whether both the teacher and the students would be able to tackle consistently well.

When it concerns junior high school students in their first year, they are mostly only familiars with basic arithmetic. They have barely just started treading the concepts of fractions, decimals, and linear algebra. This limits the availability of which a word problem can be constructed. If it was to be decided that the main format of the lesson would consists of handing out practice after practice with some minimum explanations in between, it would eventually feel repetitive if the limitations of the problem composing leads to the problem itself being lackluster.

C. Computer Science Introductory

For a a lot of people, computer science often seems to be a somewhat obscure. Perhaps it is because it is rarely thought at most schools. Also, it might be because computer science is a relatively new discipline. Some schools have already implemented a programming class one way or another. It is quite beneficial to sharpen the students with the skill of computational thinking.

Having said that, most people are not fortunate enough to actually get to know what is computer science really until later in life. Perhaps, that is why the people that are interested in computer science are not myriads in number. Since people don't really know how computer science really is, it becomes very improbable for them to cultivate interest in it.

On the other hand, the existence of a basic foundational class covering one of the branches of computer science might become a good introduction. Students will get a chance to witness a glimpse of how computer science works, and what are the skills required to prevail in the discipline.

So, in a way, this might also net some potential students that are interested in computer science to find more information and to study even deeper into it. The author would like to think that this particular tidbit is a good quality, since our world has been undergoing digital revolution quite rapidly, and fresh youths could do more good to the fields.

D. Affinity

It is clear that this subject requires a certain level of analytical skill and logic. While it is not impossible for just anyone to prevail at this subject, it is quite impossible to just anyone to enjoy it. Furthermore, the benefit of learning this subject can only be reaped by implementing it daily.

Not everyone likes to think too much about how they do things. Some people prefer to "go with the flow" and could not care less whether what they are doing is efficient or otherwise. The same applies even to young teenagers at junior high school age.

We cannot expect every single one of them to cultivate interest in learning the complex school of computer science, despite this being just the basic foundations. It would not be productive to assume otherwise.

Furthermore, as previously said, there also lies the reason about how enthusiasm is crucial in ensuring the benefits of the lesson could be felt by those who have graduated from it. Making this a mandatory lesson would coerce those who dislike it to force themselves to go through it. This might tempt them to find shortcuts to a passing grade, like memorizing problem types and the typical answers for instance. This would not do, as the goal of the lesson is to hone the students, not to force them to produce a passing grade in any way possible.

Having considered all those reasons, the author feels that if such a class is going to be held in the future, it would be more plausible to make it an elective lesson instead of a mandatory one. Ensuring that it would be taught and implemented well even though the people who actually choose the elective might be a minority.

E. Obstacles

Even on the surface, there seems to be a lot of obstacles blocking the realization of this idea. Some of them are obvious, but relatively easy to solve. Some others might not be that obvious, but relatively harder to solve.

The first thing that comes to mind would be the constraint of the students' mathematical limits. The concept of algorithm optimizations, or in other words, the simplest way to do things, relies heavily on lots of mathematical analysis using mathematical models. As demonstrated, it is possible to undermine the difficulty by omitting some of the harder concepts and prevent the problems in easy to understand word problems.

It would be nice to actually just plan this lesson for high school students. It would allow for more rooms as a high school student would most likely already had a stronger grasp on basic mathematics. At the very least, the concept of polynomials, exponents, and logarithm doesn't sound foreign to them. However, setting this up for high school students would seem to be too late. In order to maximize the impact, it feels like it is more sensible to start as early as possible.

Although, it is also impossible to be dismissed just yet. Without a way of knowing for sure, everything is a possibility until proven otherwise. If we start to consider arranging a lesson on Algorithmic Complexity for high school students, a lot of things would take a completely different turn. It would morph into a relatively easier subject compared to the real one in computer science. The use of algorithmic notations also becomes viable. A lots of new pros and cons will emerge.

Still, the whole purpose of this idea to begin with might have already failed. At the age of 15-17, teenagers often have already established who they are, what defines them, what they like, what they dislike, what affects them, what bothers them, what makes them happy, *etc*. And that would include some well-established habits and daily conduct methods.

In this case, learning the Algorithmic Complexity would become just that, learning a specific branch of computer science. It would be nearly impossible to incorporate the knowledge into the students' lifestyles. That is why the author feels that junior high school students are more suitable as the target of this lesson.

It is also concerning whether the students that do elect for the lesson would find themselves disappointed or regret their decision. As always, everything cannot be for everyone. Some people would just not have the affinity for it, or they might simply lack interest in it. At any case, we cannot hope that even people who seemed to be interested enough to actually elect for the course would keep their interest throughout the semester. Of course, it would be equivalently irrational to assume that all the students will also cultivate and retain interests in computer science after having gone through the lesson. Even so, it is not irrational to assume that at least some number of students would do just that, however small that number is.

Another thing to consider, would be the gap of skills apparent amongst the students themselves. It is often the case that some people would just automatically find themselves very good at what they are learning. Either their affinity for the subject is very high, or some people in general are just talented when it comes to the academics. That is not to say, that students that are not that good in the academics are not talented. That is not the least true in any case. It is unavoidable though, that some people would still get intimidated when their peers stand out so much compared to themselves and seemed to be having a much easier time figuring the subject. No clear solution can be deducted for this problem at the moment, but it is in actuality more of a general problem, since it is not limited to certain subjects. Even existing school subjects that currently still exist are sometimes suffering from this problem. It is true however, that the problem would be more apparent and could feel more frustrating for some subjects rather than the others.

F. Expected Results

After having gone through the whole curriculum, what would we expect of the students? A lesson plan should always have a competency target in which they set a certain level of standard that ensures the students would at least have that level of competency in the specified skill.

It has been previously specified that this lesson is expected to help the students to build better systematical and efficient habits to do their daily tasks, but in what form does it actually appears?

First of all, it is expected that the students would at least achieve some level of procedural understanding. By saying that something is procedural means that it is done with step-by-step actions which can be broken down into a general pattern, that can also apply as the method to different, yet similar problems.

When it comes to programming, procedural programming language is one of the more common ones. A procedural programming language would consist of lines of codes that would execute systematically line per line, which makes it one of the easier to understand programming paradigm.

By learning the basic principle of Algorithmic Complexity, the students are also challenged to formulate a procedural resolution the problems with which they can compare the numerous possible solutions and select the most optimal one from amongst them.

Having learned this concept early might prove to be beneficial should the students attempt to learn programming. Since most of the introductory programming language used widely are those with procedural paradigm, it would be no surprise if the students who had undergone this lesson would find more success at a faster rate when learning programming.

Secondly, it is hoped that formulating the most efficient problem solving method will become like second nature to the students. As with other school subjects, they are usually a bit overcomplicated by purpose, so that once the students are able to consciously solve them and understand them intricately enough, the students would then be able to solve the simpler versions of that matter appearing in their daily lives almost subconsciously.

For example, we were taught algebra in high school, and at times, it might seem a little overcomplicated. It is not rare to hear complains stating that most of the knowledge we earn at school will not be useful in our adult life. I disagree with the motion. For the case of algebra, at the very least, we were taught how equations with variables works. It applies for multiple variables, equations with quadratic or exponential values, and even the more difficult equations containing logarithm or perhaps even trigonometry.

Yes, most people could argue that the problems we do face in our adulthood would not require nearly as advanced knowledge as those given in high school, but one can also argue that it is precisely because of the overcomplicated level of difficulty that we can find it absurdly easy and even do it subconsciously. For instance, after we learn about vector spaces in geometric algebra, calculating the volume of a prism seems like a joke.

That would also hold true for Algorithmic Complexity Foundations. After experiencing the more complex problems throughout the lesson, it is expected that the students could subconsciously bear the fruit of their labor indirectly throughout the small and tedious tasks that they do more effectively every day in their lives.

Lastly, it also sharpens the students' logical abilities in general. In order to prevent dullness, a knife must be constantly sharpened, and so does the brain, in figurative meaning of course. The ability to analyze and subtract a logical reasoning to why things happen can become a very useful capability. It is sought after and quite frankly becomes nearly essential in most discipline relating to science.

In conclusion, there are some benefit that we can envision from the lesson itself. Although, how true it actually is and how effective will it achieve those things in reality still remains to be seen.

IV. Psychological Impact

It is difficult to argue once we enter the realm of human study, but it is also inseparable from almost any topic that concerns human. Prior to committing to a certain course of action, it would be wise to always consider the psychological aspect of things, because not all ideas are possible to be realized, as some of them may face impassable obstacles along the way. Psychology has the potential to be one of the obstacle. We must first determine whether what we are trying to do could bring a positive impact to the mental health of our recipients. Oftentimes, a great idea could get shot down only because of some kind of social barrier that is not even related to the topic of the idea. Which is why, it is crucial to consider the impact of this lesson to the students.

For starter, so far, this lesson has been called "Algorithmic Complexity Foundations", which is unnecessarily intimidating. It sounds hard to learn, and does not really make sense since it contains some harder vocabularies for young teenagers. A friendlier sounding title would need to be thought of. A good example would be something simple to understand like "Basics of Efficient Life" or "Efficiencies in Logics".

Now, other than the main goals of the lesson, there is another thing that can be achieved through this lesson, and that is moralization. Sometimes, demoralizations can occur when people suffer from inferiority complex, or when they feel insecure about themselves. On the contrary, it could be a morale boost when you gain some confidence due to something you obtain one way or another. In this case, the morale booster comes in the form of principal understanding.

People tend to feel more comfortable when they are in their comfort zone, as the term insinuates, and comfort zone, usually means that they understand completely and/or are fully prepared for anything within that zone. Armed with the assurance that you are doing things efficiently, and that you know the reasoning behind why you choose to perform a certain task the way you do, should be a great comfort. For some people, a spark of confidence can do wonders. It can enable them to commit greatness that they never could have done otherwise.

Some people do argue that some things should not be quantified. For example, managing the time that you spend with your family. Some people would consider it immoral for one to allocate his/her time to her family based on pure logic and mathematical reasoning. Some people argue that the matters of the brain should be separated from the matters of the heart. That statement makes it feels like logic is not that natural. Whether logic is natural or not, it will remain an undiscussed controversy in this paper.

Meanwhile, the author thinks that the worst that people could think of this lesson, is just that it is a waste of time. As said before, some people would consider even thinking about some things is totally unnecessary and would opt for their comfortable inefficient conducts as they already befit them.

Ultimately, those who do enjoy this kind of lesson should be able to learn and implement them without any serious repercussion from the society. After confirming that, there should be no more concerns from the psychological aspect. At least, none is currently obvious.

V. CONCLUSION

With everything said, the author concludes that organizing a Foundation of Algorithmic Complexity class for junior high students might be beneficial. However, it is not yet set in stone, as the author had no way to actually prove it. Perhaps, in the future, this uncertainty can be rectified should it actually be done.

VI. ACKNOWLEDGMENT

The author would like to thank **Dr. Ir. Rinaldi Munir, MT.**,as the active lecturer for the “IF2120 Matematika Diskrit” class, for every knowledge and experience that was passed on during classes.. It is mainly thanks to the author’s lecturer that the author is able to write this paper. The author hopes that the value being passed on to the author from the author’s lecturer could be further passed on and be of use to the readers of this paper.

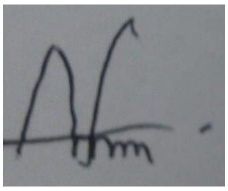
REFERENCES

https://www.cs.cmu.edu/~adamchik/15-121/lectures/Algorithmic%20Complexity/complexity.html ACCESSED : DECEMBER 9TH 2018, 21:17

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 3 Desember 2017



Lucky Jonathan Chandra, 13517136