Nanotechnology Perceptions
ISSN 1660-6795
www.nano-*ntp.com*

Analysis Of Analogies In Secondary Education 'Computer Science' Curriculum

G.V.N.Kishore[1], Dr. M. Vidya Bhargavi[2], Dr. E.Vanilla Kumari[3], Dr. Ch. Shashi Kumar[4] and Dr. Manukonda Pushpa Rajyam[5]

[1]Assistant Professor, Dept. of AI & ML, SASI Institute of Technology and Engineering, Tadepalligudem, West Godavari District, Andhra Pradesh, India.
[2]Associate Professor, Dept. of Mathematics, Stanley College of Engineering and Technology for Women, Abids, Hyderabad, Telangana, India.
[3]Assistant Professor, Dept. of Education, Andhra University, Visakhapatnam, A.P, India.
[4]Assistant Professor, Dept of Mathematics, Vallurupalli Nageswara Rao Vignana Jyothi Institute of Engineering and Technology, Nizampet, Hyderabad, Telangana, India.
[5]Assistant Professor, Institute of Advanced Studies in Education (IASE), Andhra University, Visakhapatnam, Andhra Pradesh, India.

"Analogy" is comparing an unknown concept with a known concept from various angles. In other words, it is a powerful cognitive tool used while teaching new concepts. Although they are effective teaching tools, analogies may not catalyze learning for all the learners. The reason is that no analogy can consist of two concepts which completely correspond to each other. As a result, there are different classifications of analogies. In this study, analogies were accepted in six categories according to their "relationships, presentation format, situation, task, level of richness, and personal". This classification can be utilized actively in Information Technology (IT) classes as well as all other classes. Although there are a lot of studies about Information Technologies in literature, no studies have been found that examine the textbook of Information Technologies in terms of analogies.

Key Words: analogies, concept, information technology, operating system, students.

Introduction

Computer science (CS) is a subject that is hard to learn. Teaching computer science concepts effectively to undergraduate students is also a difficult task. There are two primary reasons why undergraduate students find computer science difficult to comprehend:

1. Most students receive little or no exposure to computer science concepts during their pre-university education. Some students may acquire computer programming skills during high school, but even so, they will not be exposed to concepts such as how the internals of a compiler are organized or how an operating system works.

2. Most computer science concepts are quite abstract. Computing essentially involves coming up with an abstract representation of the problem domain and then programming a solution based on this abstract representation. Students often struggle with abstract thinking as they are

not accustomed to representing the concepts involved in a problem domain in computer memory, a fundamental aspect of computer science.

Through this paper, we demonstrate our dedicated efforts in effectively conveying abstract computer science concepts to undergraduate students by employing analogies. We have used analogy-based teaching in the classroom for the last fourteen years for many advanced CS topics. By employing analogy-based teaching, we establish a mapping between abstract computer science concepts (target concepts) and familiar concrete concepts known to students (familiar concepts). This results in better understanding of the target concept. Analogy-based teaching results in better learning outcomes, as students learn by using their existing knowledge that comes from common daily life examples. It was found that analogy-based teaching makes understanding of a concept easier for students. It significantly improves student learning outcomes. In this paper we present some of the analogies we have developed. We elaborate on two such analogies: one on the topic of process scheduling in a CPU, and another around modular software design. We present the results of our experimental evaluation as well.

Analogy-Based Instruction

Analogies play a significant role in various cognitive processes, such as "problem solving, decision making, argumentation, perception, generalization, creativity, invention, emotion, prediction, explanation, conceptualization and communication". This paper emphasizes the utilization of analogies as an effective means of communicating information related to computer science concepts. We follow structure-mapping theory for our analogies. The theory describes how familiar knowledge about a base domain can be used to understand a less familiar or unfamiliar idea in the target domain. Structure-mapping theory has three components:

1. Structure alignment, i.e., how closely do the base domain and the target domain match each other (a one-to-one correspondence is ideal).
2. Analogical inference, i.e., an inference drawn in the base domain can be transferred to the target domain (e.g., narrow pipes will cause water flow to slow down, analogous to how increased resistance will cause electric current to be reduced).
3. Evaluation, i.e., judging whether the analogy is relevant, plausible, and useful. The inferences drawn in the target domain should be factually correct, i.e., align with reality.
4. They should be useful, i.e., they should provide insight in the target domain.

With the structural alignment theory in mind, we have designed many analogies for advanced computer science subjects in the last fourteen years. We observed that analogical teaching did generate effective learning in students. We developed several analogies in computer science, such as those in the following subjects, to name just a few: (a) parallel processing and pipelining in computer architecture; (b) virtual memory in operating systems; (c) processes in operating systems; (d) asymptotic notations in the analysis and design of algorithms; (e) Chomsky's hierarchy in formal languages and automata theory; (f) lists, stacks, and several topics in data structures; (g) compilers, macros, and the OS in systems programming; (h) the design of the central processing unit (CPU), control unit (CU), and cache in computer architecture.

In SE (software engineering), we have also developed several analogies, the main ones being the following: (a) software maintenance and its analogy with maintaining a physical house; (b) coupling and cohesion in software modules explained through the analogy of a multi-semester degree program in a university; (c) levels of testing for a software system with the analogy of testing of computer chips; (d) the waterfall lifecycle model with a detailed analogy of constructing a house. We describe two analogies in this paper, one of them in detail.

Analogy-Based Teaching in Computer Science

Teaching advanced computer science topics to students is a challenging task. The problem is that students are not familiar with computer science concepts as they have likely not studied them before joining the university. Even if they had studied programming courses earlier, they would not have been exposed to computer science concepts such as formal languages and automata theory, operating systems, computer architecture, compilers, etc. Almost all students learn these concepts for the first time when they study these computer science courses in their university program. A good example of such concepts that are hard to understand is the concept of processes and process scheduling in operating systems. Additionally, several CS topics are abstract. This abstract nature of CS concepts makes it tough for students to grasp them when they face these concepts initially. A good pedagogical technique is to teach students by building upon their prior knowledge. Due to the abstract nature of CS concepts, teaching students by building upon their prior knowledge becomes quite challenging, as their knowledge of advanced CS is minimal.

Thus, analogy-based instruction is particularly helpful for teaching computer science concepts to average students who may not yet have developed the mental ability to abstract complex concepts. Possessing the ability to understand abstract concepts and develop abstract mental models of complex concrete concepts is essential to being a proficient computer scientist. Analogy-based instruction helps beginner and intermediate students of computer science in developing the ability to abstract complex subjects and is important for producing better CS graduates. Our research indicates that learning outcomes for students become better when analogy-based teaching methods are used.

Research indicates that explaining abstract concepts through concrete representations and making a direct mapping between the abstract and the concrete assists students in grasping abstract topics much more easily. If abstract terms are used, then students have a hard time fully understanding the concept. Similarly, teaching a new concept in an exclusively concrete manner curtails students from developing their capacity to generalize and to be able to apply the concept to other similar contexts. To begin with, in computer science, most of the concepts are very abstract. For example, consider the concept of a process in an OS (this concept does not have a tangible or definitive representation). Resorting to analogies is the only way to achieve concretization. Hence, the challenge of teaching CS concepts to computer science undergraduates is solved by resorting to analogies. These analogies use common concepts that students know and to which they can easily relate. The advantage of using analogy has a strong foundation. Employing analogies for teaching CS concepts has two major benefits:

1. Abstract concepts are transformed into tangible or concrete forms, facilitating better comprehension for students. Since in computer science even concrete concepts are abstract, the utilization of analogies that link challenging concepts to familiar and concrete concepts in other areas significantly enhances the attainment of improved and effective learning outcomes. Research indicates that understanding abstract concepts directly is significantly harder.
2. An analogy-based teaching method reinforces the pedagogical principle that a new concept should be taught by building upon existing knowledge that the student possesses related to that concept. Analogizing an abstract concept to a concrete concept taken from daily life allows students to understand new topics much more easily by building upon knowledge that they already possess.

Method

This is a qualitative study aiming to analyze the analogies in the textbook used in the course "Computer Science" at secondary schools in India; and content analysis technique has been employed. In parallel with the purpose of the present study, the document analyzed for the study is textbook "Computer Science" published in 2017 by Ministry of National Education for secondary school level. In this context, "Computer Science" course book has been examined with a critical eye in line with analogy classification.

Results

The examination of distribution of analogies under "According to Relationships" category in Secondary School Level Computer Science course book shows that 23 analogies under structural analogy category, 1 under functional analogy category, and 2 under structural-functional analogy category were used.

When the distribution of the analogies under "According to Presentation Format" category is examined, it is seen that 24 analogies under verbal analogy category, 1 under pictorial category, and 1 under verbal-pictorial category were used. The distribution for "according to situation" category shows that concrete-concrete analogy category has 4, abstract-abstract analogy category has 13, and concrete-abstract analogy category has 9 analogies. For "According to Task" category, the distribution is as 12 analogies under pre-organizing analogy category, and similarly 12 analogies under activating analogy category; and there are 2 analogies under post-organizing analogy category. When the distribution for "According to Level of Richness" is examined, it has been found that there are 25 analogies under basic analogy category, there is 1 analogy under enriched analogy category; in contrast, there has been found any analogies under enlarged analogy category. Lastly, similar to "According to Level of Richness-Enlarged Analogy" there has not been found any analogies under the category named "Personal" throughout Secondary School Level Computer Science course book

Because the subjects and concepts included in the science courses are abstract and theoretical, it is determined that many students have difficulty in perceiving the concepts of science (Anagün, Ağır and Kaynaş, 2010; Ayas and Özmen, 1998; Balkan Kıyıcı, 2008) and therefore, they have developed negative attitudes toward science (Hannover and Kessel, 2004). Today, science education is evaluated with an epistemological approach, focusing on the role of

learners in the personal construction of knowledge (Atasoy, 2004; Hewson, 1992; Ritchie and Russell, 1991). Determining the ideas of the students and correcting them if they are wrong are indispensable for the subsequent learning (Karamustafaoğlu, Özmen and Ayvacı, 2004). The new information will be better understood and remembered when it is meaningful for the student.

Importance of Analogies

Analogies are of great importance in learning, as they can help in reconfiguring existing memory and preparing for new knowledge (Gentner, 1983). Analogy is one of the ways of reasoning; reasoning is defined as the elimination of one another from the relationship between at least two propositions (Çüçen, 1997; Kesercioğlu, Yılmaz, Çavaş and Çavaş, 2004). Analogical thinking plays a key role in the constructivist learning process. In the structuring process of the concepts, analogies have a facilitating effect (Demir, Önen and Şahin, 2011; Duit, 1991; Ekici, Ekici and Aydın, 2007). Newton (2003) categorized the analogies into four: simple, enriched, extended, and metaphor. The simplest analogies in these categories include analogies that emphasize a single similarity dimension, which is not detailed and which is expressed in a similar manner to the target source without explanation. Enriched analogies include a few sources for a target; they are used in simulations that contain more than one similar aspect of the source to identify the target. Expanded metaphors are used in relations between source and new content, as well as in their limitations; in other words, metaphors are used when the daily language of life is the same as the relationship between the target and the source.

When the use of analogy and metaphors in teaching is examined, it is noteworthy that it is the teaching models that make comparisons with similarities between the two systems or elements (Aubusson, Harrison, and Ritchie, 2006; Çelik, 2016; Forceville, 2002), but that carries out this process in different ways. The difference between analogy and metaphor is whether modelling is made explicitly by the nature of comparison (Taber, 2001). The similarity was established during the creation of analogies; similarities, differences, qualifications, and inadequacies should be included between the analogy and target concept (Kesercioğlu, Yılmaz, Çavaş & Çavaş, 2004). In this study, simple, enriched, and expanded analogies were used.

Treagust, Harrison, and Venville (1998) summarized the importance of analogies, which are powerful learning techniques, in science teaching:

- Provides a clear understanding of the concepts from different perspectives.
- Enables learners to gain real-world experience.
- Helps to teach the concepts clearly from a different perspective.
- Summarizes topics in an easily-understandable form.
- Motivates learners by attracting their interests.
- Provides difficult information for learners and teachers to see their mistakes easily.
- Facilitates access to information, concept development, and problem solving.
- Makes unknowns compatible with logic by improving creativity.

When the related items are examined; It can be said that analogies are very important for science teaching in classes and textbooks as it facilitates learning (Guerra-Ramos, 2011; Vosniadou and Skopeliti, 2019). The types of knowledge contained in the sciences are full of abstract concepts that are difficult to understand, as long as they are not linked to our daily experiences (Orgill and Bodner, 2004). There are many different methods and techniques that can be used in the learning environment to teach abstract events, phenomena, concepts, and 757575 Çelik, H, Kırındı, T. & Ayçiçek Kotaman, Y. (2020). In the student-centered learning approach, in order to avoid memorized information, combine the information given to the students with the knowledge they had previously, and try to ensure the active participation of the students in learning, technology-supported education is a useful method in providing the students with rich and self-contained learning activities (Özmen, 2004).

Technology support is an undeniable fact

Technology support is an undeniable fact especially in understanding abstract concepts in science education. In this context, simulation and animation programs are widely used. Simulation is the realization of a subject, system, and event model on a computer. It is widely used in understanding abstract concepts by giving education in laboratories instead of expensive and real environments (Altın, 2009). Animation is the process of manipulating and changing by activating static pictures or images (Akpınar, 2005). Animations can be composed of real images taken with video or they can be displayed as moving images on computer after processing (Altın, 2009). In order to support the teaching of abstract and complex science subjects in science teaching, analogies are included in the Computer Aided Teaching (CAT) material. While preparing the analogies, it is necessary to take into account the persistence of science teaching, which is intended to be realized, as well as better structuring of knowledge for students.

In this context, while creating analogies, it is thought that the use of audiovisual features such as animation, sound effect, color, and moving picture will ease the cognitive burden of knowledge; thus, the science teaching, which is aimed to be realized, will be made more permanent. In the science and technology course dominated by abstract subjects, the use of CAT materials enriched with animation, simulation, video and multimedia, and other supporting techniques in science education is known to increase the interest of students (Akçay, Tüysüz and Feyzioğlu, 2003) and to decrease the spent time for achieving the goal in terms of teachers and students (Kulik, Kulik and Bangert-Drowns, 1985). In recent years, analogies are considered to be one of the most important elements in science-related teaching-learning process (Azizoğlu, Çamurcu and Kırtak Ad, 2014; Dagher, 1995; Duit, 1991; Garde, 1986; Kılıç, 2007; Paul, Lim, Salleh, & Shahrill, 2019; Thiele and Treagust, 1994, Yılmaz, Eryılmaz and Geban, 2002).

In science, teaching concept is very important and students learn the concepts and combine the situations they have learned previously with new situations and simulations in their minds. When we look at the writings of mental processing based on this simulation, we see as analogy, metaphor, mental image, image, or mental model. These concepts serve to the same process in the science and social sciences in the literature (Çelik and Çakır, 2015; Çökelez and Yalçın,

2012; Güveli, İpek, Atasoy and Güveli, 2011; Kavak, 2007; Ören, Ormancı, Babacan, Koparan and Flower, 20; Özdemir, 2012).

Analogy-based learning methods map the concept being learned to a concept well understood by the learner. An analogy is primarily useful when learners do not know the topic being studied. Computer science is an area where the concepts exhibit a high level of abstraction and, hence, are hard for students to comprehend. The use of analogies in instruction can significantly reduce the cognitive load a student faces in learning abstract computer science concepts. The role of analogies in helping students learn computer science topics has not been explored adequately. This paper presents our efforts related to using analogy-based teaching in computer science. Over the last several years, we have developed extensive analogies for many advanced computer science concepts. We have used these analogies extensively in classroom teaching at our institution. We list the analogies that we have developed and used in our classroom teaching and, as illustration, discuss two analogies: one from the field of operating systems and another one in modular software design. We have also conducted experiments to evaluate the impact of using these two analogies on student learning outcomes. Our results confirm our hypothesis that analogy-based instruction techniques are effective and result in improved student learning outcomes.

Discussion and Conclusion

In the present study, which aims at analyzing the analogies in course book officially used in "Computer Science" course at secondary school level in India, the distributionof the analogies in the mentioned course book has been investigated; and it has been seen that when the classification based on "relationship" element of the connection between source-target elements is examined, almost all of the analogies in the course book fall under the sub-category "structural analogy". The most possible reason for this fact can be that Computer Science course content has more tendency for use of structural analogy. When the analogies in "Computer Science" course book for secondary school students are examined according to "According to Presentation Format" category, it is seen that majority of the analogies accumulate under "verbal analogy". In contrast, pictorial and verbal-pictorial categories are considerably in the minority. Although verbal analogies are quite useful, it is a necessity to give place to "pictorial analogies" in course books in an enriching way which especially supports multiple intelligences theory. Considering from the angle of variety of analogies, neglecting thesupport for multiple intelligences is a major deficiency. The examination of analogies under "According to Situation" category in Secondary School Level Computer Science course book shows that majority of analogies fall under "from abstract to abstract" sub-category. This fact is directly related to course content. As in computer science course codes and designs are on digital platforms, it is naturally more probable to try to explain abstract examples and concepts. However, the use of analogies in daily life may differ in this category.

References

1. Alizadeh, M.; Di Eugenio, B.; Harsley, R.; Green, N.A.; Fossati, D.; Omar, A. Study of Analogy in Computer Science Tutorial Dialogs. In Proceedings of the CSEDU Conference, Lisbon, Portugal, 23–25 May 2015; pp. 1–6.

2. Cheo, C. Forty Key CS Concepts Explained in Layman's Terms. 2022.
3. Gadgil, S.; Nokes, T. Analogical scaffolding in collaborative learning. In Proceedings of the Annual Meeting of the Cognitive Science Society, Amsterdam, The Netherlands, 29 July–1 August 2009.
4. Gentner, D. Analogy. Companion Cogn. Sci. 1998, 107–113.
5. Gentner, D.; Colhoun, J. Analogical processes in human thinking and learning. In Towards a Theory of Thinking; Springer: Berlin/Heidelberg, Germany, 2010; pp. 35–48.
6. Gentner, D.; Loewenstein, J.; Thompson, L. Learning and transfer: A general role for analogical encoding. J. Educ. Psychol. 2003, 95, 393.
7. Gollub, J.G.; Bertenthal, M.W.; Labov, J.B.; Curtis, P.C. Learning and Understanding: Improving Advanced Study of Mathematics and Science in U.S. High Schools; National Academy Press: Washington, DC, USA, 2002.
8. Hofstadter, D.R. Analogy as the core of cognition. Analog. Mind Perspect. Cogn. Sci. 2001, 499–538.
9. Lulis, E.; Evens, M.; Michael, J. Implementing analogies in an electronic tutoring system. In Intelligent Tutoring Systems; Springer: Berlin/Heidelberg, Germany, 2004; pp. 751–761.
10. Murray, T.; Schultz, K.; Brown, D.; Clement, J. An analogy-based computer tutor for remediating physics misconceptions. Interact. Learn. Environ. 1990, 1, 79–101.
11. Newby, T.J.; Stepich, D.A. Learning abstract concepts: The use of analogies as a mediational strategy. J. Instr. Dev. 1987, 10, 20–26.
12. Nokes, T.J.; VanLehn, K. Bridging principles and examples through analogy and explanation. In Proceedings of the 8th International Conference for the Learning Sciences, Madrid, Spain, 26–28 April 2008; Volume 3, pp. 100–102.
13. Paul, B. Analogy and Analogical Reasoning. In The Stanford Encyclopaedia of Philosophy, Spring 2019 Edition; Zalta, E.N., Ed.; Stanford University: Stanford, CA, USA, 2019.
14. Saxena, P.; Singh, S.K.; Gupta, G. Analogy-based Instruction for Effective Teaching of Abstract Concepts in Computer Science. In Proceedings of the 7th International Conference on Higher Education Advances (HEAd'21), Valencia, Spain, 22–23 June 2021; pp. 377–385.
15. Schwanenflugel, P.J. Why Are Abstract Concepts Hard to Understand? 1st ed.; Psychology Press: London, UK, 1991.
16. Wikipedia Contributors. Analogy. Wikipedia, The Free Encyclopaedia. 2021. Gentner, D. Structure-Mapping: A Theoretical Framework for Analogy. Cogn. Sci. 1983, 7, 155–170.