Journal of Research in STEM Education

Vol 2, No 1, July 2016, PP 19-38

RESEARCH REPORT

Epistemological, Psychological, Neurosciences, and Cognitive Essence of Computational Thinking

Osman Yasar [a 1]

[a]*State University of New York, USA*

Abstract: *The construct of computational thinking (CT) was popularized a decade ago as an "attitude and skillset" for everyone. However, since it is equated with thinking by computer scientists, the teaching of these skills poses many challenges at K-12 because of their reliance on the use of electronic computers and programming concepts that are often found too abstract and difficult by young students. This article links CT – i.e., thinking generated and facilitated by a computational device – to our typical fundamental cognitive processes by using a model of mind that is aligned with research in cognitive psychology and neuroscience and supported by a decade of empirical data on teaching and learning. Our model indicates that associative and distributive aspects of information storage, retrieval, and processing by a computational mind is the very essence of thinking, particularly deductive and inductive reasoning. We all employ these cognitive processes but not everyone uses them as iteratively, consistently, frequently, and methodologically as scientists. Some scientists have even employed electronic computing tools to boost deductive and inductive uses of their computational minds to expedite the cycle of conceptual change in their work. In this article, we offer a theoretical framework that not only describes the essence of computational thinking but also links it to scientific thinking. We recommend teaching students cognitive habits of conceptual change and reasoning prior to teaching them skills of using electronic devices. Empirical data from a five-year study involving 300 teachers and thousands of students suggests that such an approach helps improve students' critical thinking skills as well as their motivation and readiness to learn electronic CT skills..*

Keywords: Deductive and inductive thinking, cognitive processes, modeling and simulation

Introduction

A decade of discourse since the launch of computational thinking (CT) initiative by the computer science community has resulted in wide acceptance of the following relevant CT skill set for K-12 education (Grover & Pea 2013):

- Abstraction and pattern generalization (including models and simulations)
- Systematic processing of information
- Symbol systems and representations
- Algorithmic notion of flow control
- Structured problem decomposition
- Iterative, recursive, and parallel thinking

[1] Empire Innovation Professor, Department of Computational Science, State University of New York (SUNY)-Brockport 350 New Campus Drive, Brockport, NY 14420, E-mail: oyasar@brockport.edu

Yasar, O. (2016). Epistemological, psychological, neurosciences, and cognitive essence of computational thinking. *Journal of Research in STEM Education*, *2*(1), 19-38

ISSN:2149-8504 (online)

- Conditional logic
- Efficiency and performance constraints
- Debugging and systematic error detection

The idea of adding computational thinking to a child's analytical ability goes back almost four decades (Papert, 1980), yet its recent popularization by Wing (2006) and federal agencies have successfully encouraged many teacher organizations and professional societies to promote its inclusion in education through learning standards (NRC Report, 2012). While the above skills may be appropriate for college freshmen, they all seem to be associated with problem solving and the use of electronic devices with an ultimate goal of preparing tomorrow's programmers. Grover and Pea (2013) attest to that conclusion by arguing that, "Programming is not only a fundamental skill of CS and a key tool for supporting the cognitive tasks involved in CT but a demonstration of computational competencies as well." They also claim that efforts that introduce computing concepts without the use of a computer may be keeping learners from the crucial computational experiences involved in CT's common practice (p. 40). While this is an admission of the fact that obstacles remain in the way of integrating CT practices into grade level curricula, we argue, in contrary to their assessment, that the mere focus on programming is the source of many obstacles in CT education.

Unresolved questions remain in computing education. If we continue to define computational thinking as thinking by a computer scientist during problem solving, then having young students learn problem solving the way that computer scientists do will continue to pose challenges. These challenges are many, including a) prerequisite CS content knowledge needed to engage in the same thinking processes as computer scientists, b) programming skills, and c) access to electronic computing devices. To diminish these challenges, a problem solving and design approach has been recommended to teach computing concepts and principles through CT education (Guzdial, 2008), but this has yet to produce ways to separate CT from programming and the use of electronic devices. In our view, the lack of such separation has precluded us from capturing the true essence of CT, which could have helped narrow down the above list of CT skills to a smaller and more fundamental set of cognitive competencies that can be more easily taught at the K-12 level. Programming electronic devices may be a central tool for computer scientists, but there are non-programming computational tools available to and used by non-CS scientists. Since the goal of the CT initiative is to bring computing to non-CS students, how non-CS scientists use computers may be the key to find ways in which computing is separated from programming. An obvious example is the computational modeling and simulation tools (CMSTs).

While modeling and simulation is associated with one of the most important CT skills, i.e., abstraction, it has not received due attention because of the concern, perhaps, that it would keep learners from more mainstream computing practices as hinted by Grove and Pea. Also, the attention given to abstraction in the CS community is again because of its importance in computer programming (Armoni, 2013). It is true that without abstraction most of the large-scale computer software, including operating systems, compilers, and TCP/IP network software, could not have been written so efficiently. In fact, a good programmer is expected to be so good at abstraction that she should be able to easily oscillate between different levels of abstraction depicted in Fig. 1. Yet, the sad truth is that undergraduate CS students barely move beyond language-specific (level 2) or algorithm-specific (level 3) biases. Reaching level 4 is important to transform appropriate algorithmic and programming skills into different application contexts, yet the teaching of programming itself does not seem to accomplish this (Armoni, 2013).

Dijkstra, an early pioneer in programming, regarded abstraction as the most vital activity of a competent programmer (Armoni, 2013). Other prominent CS educators also see it as one of the most important characteristics of CT (Wing, 2011). So, if modeling and simulation is being recognized to facilitate abstraction skills as noted by Grover & Pea (2013), then an important task to link CT skills to our thought process is to investigate cognitive aspects involved in the practice of modeling and simulation. The good news is that this

task has been recently done and we now want to merge its findings (Yaşar, 2016) with concepts from relevant disciplines to build and examine a theoretical framework that can help us narrow down the CT skillset in a way that can be promoted via modeling and simulation tools readily available at K-12.

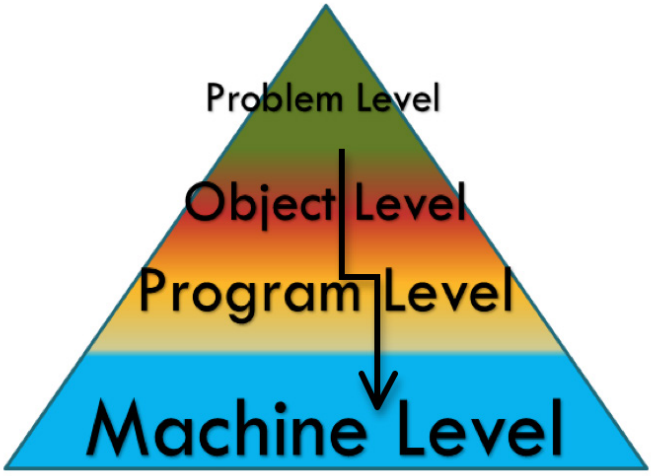

Figure 1. Multi-level abstraction in programming, From low to high: (1) machine-level, (2) program-level, (3) object-level, and (4) problem-level).

In the sections below, we introduce theoretical and empirical considerations from various fields – e.g., computing, neuroscience, education psychology, cognitive sciences and epistemology – as part of the foundation of our framework. These generally revolve around how knowledge is stored, retrieved, processed, and developed. At the core of our framework is the computational theory of mind (Montague, 2006), which basically claims that computational processing of information, regardless of the underlying device (electronic or biological), can facilitate and generate cognition. Assuming modeling and simulation to be a form of device-independent computation, as illustrated in Fig. 2, we will explore in the next section how such an approach might help us link electronic and biological computing. Then, we will present concepts from various fields to establish our framework. Finally, to test the framework we will describe a quasi-experimental design with more than 300 teachers using modeling tools in a diverse set of 15 secondary schools. Detailed findings of this case study have been presented earlier (Yaşar, et al. 2014, 2015, 2016), but their relevance to the essence of computational thinking is a new undertaking here. We will conclude by offering some recommendations and practical considerations for teachers or professional development program designers on how to integrate computational modeling and simulation into K-12 settings. The article will provide a link to a freely available database of curriculum modules and lesson plans teachers and other educators can use to design modeling-based educational materials for their professional development and K-12 classrooms.

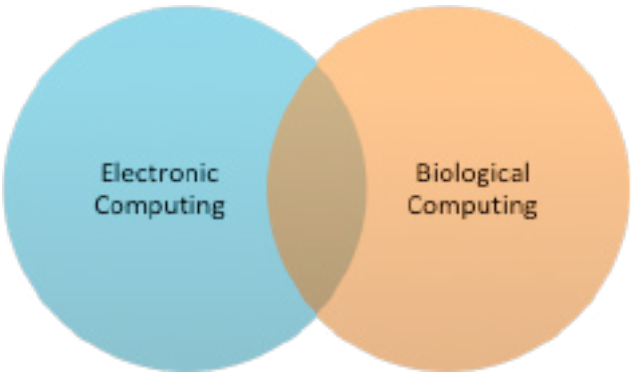

Figure 2. Device-independent elements of computing shown as an overlap of both electronic and biological computing

ISSN:2149-8504 (online)

Device-Independent Computation

Seventy years ago, Alan Turing (1936), widely recognized as the founder of computer science, suggested that if thoughts (i.e., information) can be broken up into simple constructs and algorithmic steps, then machines can add, subtract or rearrange them as our brains do. Electronic machines have since successfully taken many complex and voluminous computations off our brains, and thus further supported the view of brain as a computational device.

Basically, there appears to be two root causes of similarities between electronic and biological computing patterns. One of them, as Turing alluded to, is the invariant behavior of information. That is, information constructs behave either by uniting to make bigger constructs or breaking down to smaller ones. Devices that can track and tally this computable behavior (addition and subtraction) are called computers, regardless of their underlying structure. This duality of basic computation manifests itself in other high level processes as we discuss it later.

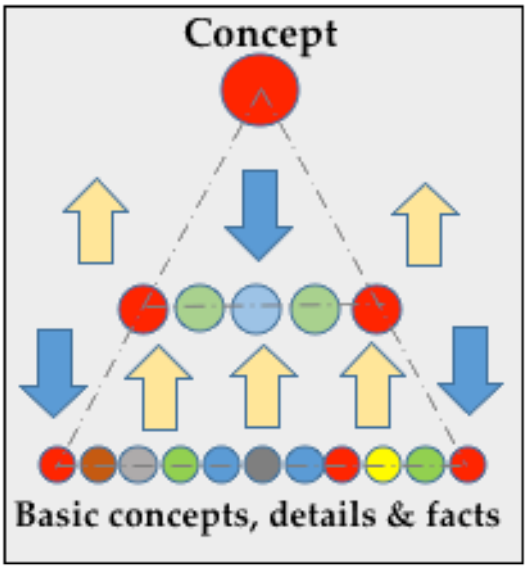

Figure 3. Distributive and associative ways of information storage, retrieval, and processing (Yaşar, 2017).

Another cause may be the control and use of electronic devices by biological computing agents. Our use of an electronic device can certainly reflect the way we use our own biological computing device (i.e., our mind). Their utilization, however, depends on how we use them. So far, we have used electronic devices in various ways, including programming (text-based and visual), office work, communication, visual arts, video games, virtual reality, modeling and simulations. These range from easy tasks (e.g., automation of repetitive and voluminous work) to complex tasks (e.g., solving systems of differential equations for which there is no analytic solution). In almost all these cases, electronic devices basically follow preloaded instructions and therefore offer almost none or little cognitive insight to us, except that virtual reality and computer simulations are known to have generated some insight in scientific research (Oden and Ghattas, 2014).

Modeling has been an important tool for scientific inquiry. As illustrated in Fig. 3, it is an iterative and cyclical process by which a previously constructed model (e.g., a concept or a theory) is analyzed (broken up) first and then synthesized (put together) – after testing, sorting, and updating – to either validate or change the original concept/theory. This cycle is repeated as resources permit until there is confidence in the revised model's validity. In recent years, computers have been very effective in conducting scientific research because they speed up the model building and testing of different scenarios through simulations that provide quick feedback to researchers in order improve the initial model. The role of computational modeling and simulation

tools in scientific and industrial research has been proven beyond doubt because of their accuracy of predicting observed phenomena and improving industrial products (e.g., engines, planes, and cars) and their ability to conduct studies that are impossible experimentally, and ability to conduct studies that are impossible to do experimentally due to size, access and cost (NSF Report, 2006). As a result, modeling and simulation is now regarded as a third pillar of doing science because it facilitates the deductive and inductive cycle of scientific thinking (Oden and Ghattas, 2014; PITAC Report, 2005). Furthermore, modeling and simulation has been found to support deductive and inductive approaches to teaching as briefly reviewed in the next section (Yaşar & Maliekal, 2014; Yaşar 2016). So, judging from its utilization in both scientific research and teaching, one might say that modeling and simulation is a common process through which electronic and biological devices could resonate. The following section will employ this process to explain in simple and understandable terms what we know today about how information gets stored, retrieved and processed by the brain.

Epistemological, Psychological, Neuroscience & Cognitive Aspects of Thinking & Learning

Confidence in our understanding of how the mind works has been hindered by the fact that it involves a delicate, inaccessible, and complicated organ, the brain. Yet, not being able to dig into the most basic level of knowledge, details, and facts did not stop humans from coming up with theories and conclusions. Many fields have their hands in the study of how learning takes place in the mind. Epistemologists study how we know what we know by questioning an observer's subjective view of an objective world. Cognitive, developmental, and educational psychologists conduct empirical research into how people perceive, remember, and think. They form theories of human development and how they can be used in education. Neuroscientists use imaging techniques to understand the brain mechanisms that take part in learning. At the same time, more contemporary sciences of cognition and computing now collectively form theories and models of the mind to study how computation generates cognition. In the sections below, we will review developments in each of these as they relate to our article. We will first introduce what we knew two centuries ago about thinking and learning from an epistemological point of view. Then, we will visit relevant concepts from educational psychology a century ago, followed by views from contemporary sciences of neuroscience and cognitive sciences. In our view, they all reinforce the same general pattern of how our mind stores, retrieves and processes information. This consistency, obviously, is what has encouraged us to propose a framework, which we will introduce later.

Epistemological method of acquiring knowledge: Started a thousand years ago by Alhazen's emphasis on collecting experimental data to test accuracy and reproducibility of proofs, the epistemological method of acquiring knowledge evolved into a more robust form through use of both experimentation and mathematics by Galileo in the 16th century (Morgan, 2007). While natural scientists laid a strong foundation in the 16th century for how to acquire knowledge through empirical observations, philosophers debated for two more centuries whether a scientist's subjective view of the world could be considered as true knowledge. The debate's epistemological framework was about: a) how knowledge is acquired, and b) what the sources of knowledge are (mind, body, or both). This debate actually has not ended yet, because theories on interdependencies of matter and mind, such as how perception shapes up our thoughts through sensory experience and how the matter's existence may have preceded the meaning and essence of life (i.e., existentialism), still continue to mediate between perception and reality, and between philosophy and science using contemporary knowledge in physical, cognitive, and computational sciences (Hawking, 1988; Brown et al., 2014; Montague, 2006).

Historically, there have been two opposing epistemological views, namely rationalism and empiricism. Empiricism claimed that the mind is a blank slate and that it acquires knowledge a posteriori through perception and experiences, and by putting together in a synthetic way related pieces of information. Knowledge acquired this way is not warranted because new experiences may later change its validity. This is none other than the bottom-up (inductive) process shown in Fig. 3. Rationalism, on the other hand, has historically claimed that knowledge is acquired a priori through innate concepts. Innate knowledge is warranted as truth, and decisions and conclusions can be derived from it in an analytic way using deductive reasoning. And, this is none other

than the top-down (deductive) process shown in Fig. 3.

Immanuel Kant (1787) argued against the views of both rationalists and empiricists and created a bridge to lay the foundations of epistemology. He recognized what experience brings to mind as much as what mind itself brings to experience through structural representations. In applying mathematical, logical, and physical representations to study of nature, Kant considered that knowledge developed a posteriori through synthesis could become knowledge a priori later. Furthermore, according to him, a priori cognition of the scientist continues to evolve over the course of science's progress (Rockmore, 2011). The epistemological method Kant established more than two centuries ago has been the method by which scientific knowledge has evolved, although its dual (deductive/inductive) cycle of change has historically been rather on the order of a social timescale, not an individual timescale (Giere, 1993; Kuhn, 1962). Historically, once proven or validated, a hypothesis or an observation was revisited at a very slow pace, sometimes spanning generations, because of both limited resources (time, money, equipment, etc.) and the overwhelming number of other questions begging for an answer or proof. However, the growing scientific knowledge and the number of researchers tackling a problem as well as the increasing capacity of technology have all now shortened the timescale of scientific progress, making us realize that the deductive/inductive process is also happening cognitively at individual timescales. Concepts and theories, which were once considered true and valid, are now quickly being modified or eliminated.

Modeling and testing has been an important tool for scientific research for hundreds of years, although at a slower pace than it is today. In principle, it works exactly as articulated by Kant, and as illustrated as in Fig. 3. Scientists ideally start with a model of reality based on current research, facts, and information. They test the model's predictions against experiment. If results do not match, they, then break down the model deductively into its parts (sub models) to identify what needs to be tweaked. They retest the revised model through what-if scenarios by changing relevant parameters and characteristics of the sub models. By putting together inductively new findings and relationships among sub models, the initial model gets revised. This cycle of modeling, testing, what-if scenarios, synthesis, decision-making, and re-modeling is repeated while resources permit until there is confidence in the revised model's validity. As noted before, computers have recently accelerated this cycle because not only they speed up the model building and testing but also help conduct studies that are impossible experimentally due to size, access and cost.

Educational psychology of learning: Another historical theory that has gone over deaf ears was documented about a century ago by Lev Vygotsky (1930), a Soviet psychologist. Factual details in coursework are often overwhelming, causing frustration and withdrawal for some students. Yet, effortful learning is the key (Brown et al., 2014) and the learner needs to have an interest and necessary background, skills, and confidence to attempt and maintain such effort. Today, we still wonder what to do when some of these ingredients are missing. Vygotsky theorized that the learner can benefit from a little push from his environment (e.g., teachers, peers, parents, instructional pedagogy and technology, etc.) to engage in the process of developmental and effortful learning. Unbeknownst until two decades ago to the American K-12 education system, which puts more emphasis on social development and freedom of choice (Mooney, 2013), his theory suggested that pushing a learner to catch up with his peers was much more important than giving him freedom to choose between effort and withdrawal or between interest and indifference.

Vygotsky's theory basically points out to the benefits of an instructional approach that would employ a general simplistic framework from which instructors can introduce a topic and then move deeper gradually with more content after students gain a level of interest to help them endure the hardships of effortful, constructive, and inductive learning. Such an iterative and stepwise progression toward learning is consistent with the use of deductive and inductive approach of teaching, scaffolding, and the psychology of optimal learning experience, which all emphasize the importance of balancing challenges and abilities to attain an optimal flow of learning (Csikszentmihalyi, 1990).

items falling from their mouths still exist. But, as a result of first relating incoming information deductively to previously stored information and repeated experiences, and then conducting what-if scenarios (i.e., simulations), they eventually conclude inductively that the item has just fallen out of their reach (Mooney, 2013; Brown et al., 2014).

Cognitive science view of information processing: While the distributed structure of neurons and their connections (i.e., hardware) influence cognitive processing (i.e., software), the relationship between software (mind) and hardware (brain) is not a one-to-one relationship. According to computational theory of mind, our mind consists of a hierarchy of many patterns of information processing and, just as the case in electronic computing, these levels may range from basic computations to more complex functions (sequence or structure of instructions) and models (mental representations) of perceived reality and imaginary scenarios (Montague, 2006).

While computational theory of mind has played an important role to separate mind from brain, the effort to model the mind as a rational decision-making computational device has not fully captured all our mental representations, particularly emotions (Goleman, 2006). Artificial intelligence and neural networks may never be able to model the human brain no matter how fast electronic computers become unless we understand what intelligence is and how the human brain makes decisions without exhaustive evaluations of all possible scenarios (Hawkins, 2004). For example, the human mind is known for its energy-efficient operation, consuming as little electricity as a dim light bulb (20 Watts), while computational cognitive modeling and simulation of human brain is expected to need 106 times more electricity – equivalent to a nuclear power plant (Simon, 2016).

One wonders, then, what accounts for the energy efficiency of human brain? Neuropsychologists, as well as evolutionary biologists, point to some structural (hardware) interference by an autopilot limbic system (animal-like brain) to by-pass, simplify, or reduce more elaborate cognitive functions of an evolved neocortex (outer parts of the human brain). It almost appears that we are caught up between two competing brains, as illustrated by the top-down <—>bottom-up cycle in Fig. 3: one that wants to simplify things and one that wants to dig things deeper. Cognitive scientists have developed many similar dual-process theories to study the duality of mind (Sun, 2002). Typically, one of these processes is fast, effortless, automatic, inflexible, nonconscious, and less demanding of working memory, while the other is slow, effortful, controlled, conscious, flexible, and more demanding of working memory (Evans & Frankish, 2009).

Cognitive scientist Read Montague (2006) points to some non-structural (software) tendencies to account for our brain's energy-efficient operation. He suggests that concern for efficiency, as part of our survival, is a major driving factor. While this concern comes at the expense of being slow, noisy, and imprecise, it does assign value, cost, and goals to our thoughts, decisions and action, Montague argues. To assign these attributes, the mind carries out computations, builds models, and conducts hypothetical simulations of different scenarios. While evaluative and hypothetical simulations add additional overhead to decision-making by slowing it down, it still ends up saving it from undertaking more exhaustive computations. According to Montague, the tendency to make trade-offs between simplicity and complexity and between details and generalizations is actually the root driver of our intelligence, and why we have pushed ourselves to be smarter over time.

A model is a simplistic representation of complex and detailed reality, as illustrated in Figure 3. As we move up the pyramid, we inductively generalize the details by decontextualizing (abstracting) it's content. So, by modeling an object or some phenomena we end up with a wrapped package that masks its internal structure (details). An advantage of modeling is that it makes it possible to work with approximate, abstract, or average representations. It is a way of bringing closure to an unending investigation. The human brain uses modeling not only for mental representation of external objects but also for its own internal computations so it can compare their values and costs before making a decision. Many argue that the uniqueness of human intelligence comes from the modeling of thoughts through language and, as we all know, thinking via language constructs appears

to be a major difference between humans and other animals. Jeff Hawkins (2004), a co-inventor of hand-held devices who is now teamed up with neuroscientists to design a mind-like device outside the realm of artificial intelligence or neural networks, claims that the crux of intelligence is the ability to make predictions through mental models. So, as we will see below, modeling and making predictions with them is a core computational process of thinking and learning.

Modeling and Simulation: A Process by Which Everything Seems to Form and Grow

The common pattern in all previous sections appears to be duality (e.g., associative/distributive or inductive/deductive) of information storage, retrieval, and processing by a mind. This is not a new claim as the presence of a dual mind has been discussed for a long time, but capturing its essence is what remains a challenge. Neuropsychologists and evolutionary biologists have come up with some structural (i.e., hardware) reasons to explain the efficiency and quickness with which human brain makes decisions. It involves two competing brains, not the right and the left hemispheres but rather an autopilot limbic system (animal-like brain) that structurally interferes a more evolved neocortex (outer parts of the human brain) to by-pass, simplify, or reduce its elaborate cognitive functions (Sun, 2002). Typically, one of these brains is fast, effortless, automatic, inflexible, nonconscious, and less demanding of working memory, while the other is slow, effortful, controlled, conscious, flexible, and more demanding of working memory.

Cognitive scientist Montague (2006), however, points to some non-structural (i.e., software) tendencies to account for our brain's dual behavior and its energy-efficient operation. He suggests that concern for efficiency, as part of our survival, is a major driving factor. While this concern comes at the expense of being slow, noisy, and imprecise, our brain does assign value, cost, and goals to thoughts, decisions and actions. Montague argues that the mind conducts computations, modeling, and simulations of different scenarios to assign these attributes. While these evaluative simulations add an overhead to decision-making by slowing it down, it still ends up saving it from undertaking more exhaustive computations. So, according to him, the tendency to make trade-offs between simplicity and complexity and between details and generalizations is the root driver of our intelligence and why we have pushed ourselves to be smarter over time. We are still not sure of whether the causes are structural or non-structural, but there appears to be enough evidence from psychology, neuroscience and cognitive sciences about duality of mind regarding its psychological behavior, information storage, retrieval, processing and reasoning, which altogether warrants a synthesizing effort as undertaken here.

We argue that heterogeneity is the essence of duality in computational processing because like other heterogeneous things such as physical matter, information constructs appear to behave in one of only two ways: they either unite to form bigger ones or break down to smaller ones (Yaşar, 2017). So, any computational device, be it electronic or biological, would have to compute by either adding or subtracting information constructs at the basic level. Heterogeneity gives quantifiable information an invariant property (i.e., computability) that constraints how a computational device would process it. So, processing (tallying) of information constructs by any computing device would involve a dichotomy, all the way from addition/subtraction type computation at the basic level to modeling/simulation type computation at higher levels. Both types of these computations are device-independent forms of associative/distributive processing.

But, then, some might wonder what the essence of heterogeneity itself is? It is interesting to note that the essence of heterogeneity is the computable (associative and distributive) behavior that it causes. In layman terms, this is akin to 'what you do is what makes you or destroys you.' So, the degree of heterogeneity, its growth and degradation (breakdown) all depends on its overall dynamics of computable actions. In a way, formation of heterogeneity, whatever the driver is, resembles the act of modeling because both seem to involve adding parts together to form a whole that masks its parts. In general, such action is either driven by external forces or by a collective "trial and error" process controllable by various conditions and rules of engagement— much like a simulation (Yaşar, 2017). In computational research and education, this process is driven by a researcher or a teacher. In cognition, it is driven by a self-aware brain, and in case of physical matter, it is driven by acting forces and fields. So, wherever we turn, we see modeling and simulation as a universal process by which everything

ISSN:2149-8504 (online)

behaves (Yaşar, 2015).

Philosophers and psychologists have been studying the parts-whole dynamics since Plato (Harte, 2002). The ancient saying of "a whole is greater than the sum of its parts" arguably by Aristotle as well as a more modern version of it "a whole is other than the sum of its parts" by the Gestalt psychologist Kurt Koffka (1935) indicate that a union (whole) of parts may have properties not seen in its parts. Part-whole relations continue to occupy researchers' minds such as Findlay & Thagard (2012) who offer many examples from natural and social sciences in their recent paper. For example, a human body is functionally different than its parts; so is an atom or a cell. Another example that is relevant to our subject is that the union of math, computing, and sciences, in the way computational modeling and simulation brings them together, gives rise to a new kind of (deductive/inductive) pedagogy that did not exist in any of these individual subject domains. However, together as a union, they give rise to a computational pedagogy (Yaşar and Maliekal, 2014) that has recently been instrumental to develop of a computational pedagogical content knowledge (CPACK) framework for teacher preparation, as shown in Fig. 4 (Yaşar et.al., 2016). We will later return to the relevance of CPACK in the final section of this article to describe a quasi-experimental study used in the empirical examination of our cognitive framework.

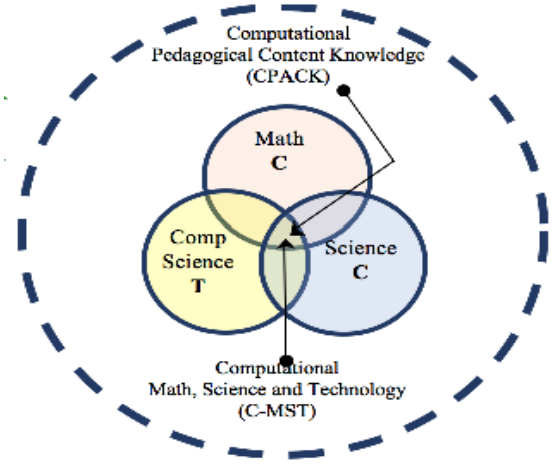

Figure 4. CPACK framework. Computational pedagogy is an inherent outcome of computing, math, science and technology integration

The Essence of Computational Thinking

Our framework is based on epistemological method of Kant (1787), computational theory of Turing (1936), and neuropsychological view of Hebb (1949). All have been around for a long time, but when merged with the latest work on computational pedagogy (Yaşar and Maliekal, 2014; Yaşar, 2016) for a new synthesis, we arrive at an interdisciplinary framework to describe in lay terms what computational thinking is and how cognitive functions are shaped up by computation. The previous sections were all necessary for our discussion to establish our cognitive framework, as they were not a mere repetition of the concepts found in the literature.

Accordingly, we argue that all quantifiable (distinct) things, such as matter and information, behave computationally (Montague, 2006) because they either unite (i.e., addition) or separate (i.e., subtraction) (Yaşar, 2017). We are surrounded by an environment that is flooded by quantifiable things, including matter and information as shown in Fig. 5. Since information is released from the matter's interaction with each other, the computable nature of information may merely reflect quantifiable behavior of physical matter. Or, its quantifiable behavior could be simply due to its own inherent heterogeneous nature. In any case, one can suggest heterogeneity is essential for dynamics to occur and be detected, because in a homogeneous environment all would be the same and not distinguishable.

If we fast-forward from the beginning of our universe to today, which started with a big explosion

of homogeneous energy into heterogeneous units of matter and information, we might say that our brain's natural inclination to process information in an associative/distributive fashion, and to store and retrieve information in a scatter/gather way may all just be a manifestation of heterogeneity-caused duality engrained in the fabric of matter and information. This inclination may just be an evolutionary response, shaped up over many years, to optimize the handling of incoming sensory information whose quantifiable nature only resonates with distributive and associative operations. Accordingly, one can argue that associative processing of information by a computational mind is the essence of inductive reasoning – through which details are put together, focus is placed on general patterns, and priority and importance are assigned to newly acquired information. Inductive reasoning (i.e., abstraction) helps us simplify, categorize, and register key information from sparse, noisy, and ambiguous data for quicker retrieval and processing (Bransford et al. 2000; Brown et al., 2014; Donovan & Bransford, 2005). By the same token, distributive processing of information appears to be the essence of deductive reasoning – through which a general concept is analyzed and broken down in terms of its possible constituencies and their applicability and validity. Deductive reasoning helps us decompose a complex issue by dividing (scattering) the complexity into smaller pieces and then attacking each one separately until a cumulative solution is found (gathered).

We conclude, then, associative (+) and distributive (-) way of processing, storing, and retrieval of information is the very essence of thinking that is generated or facilitated by a computational device. We put this dichotomy at the core of information processing by both electronic and biological computing devices as shown in Fig. 6. And, we expect it to carry itself up to higher level cognitive processes, such as deductive reasoning as a form of distributive processing of information, and inductive reasoning, as a form of associative processing of information. As illustrated before, iterative and cyclical usage of deductive and inductive reasoning is the foundation of conceptual change, a process by which we progress our learning. However, its utilization depends on the underlying device structure and the quality and quantity of the environmental input it receives. Although cognitive researchers have already demonstrated how various forms of information processing could lead to cognitive inferences and generalizations (inductive reasoning) (Tenenbaum et al., 2011; Langley, 2000; Yang, 2009), here we are not concerned about details of computation-to-cognition but rather how duality in fundamental computation could lead to duality in cognitive functions.

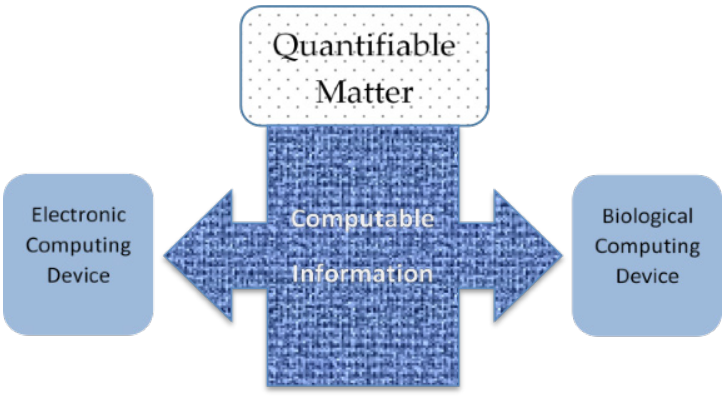

Figure 5. A view that quantifiable nature of information is simply a reflection of heterogeneous behavior of physical matter that emits such information.

The main tenet of our framework is that thinking is generated by a biological computing device and that it could also be equally facilitated, if not generated, by an electronic computing device. So, we equate computational thinking with ordinary thinking generated by a biological computing device and, in that sense, everyone, not just computer scientists, does computational thinking. Perhaps, we should distinguish between biological and electronic computational thinking to indicate that while each form of computing facilitates thinking processes, they also give rise to device-dependent processes. Then, as shown in Fig. 6, electronic CT

ISSN:2149-8504 (online)

would consist of biological CT as well as thinking caused by certain uses of electronic computing devices by a biological agent.

We may all be naturally inclined to employ associative and distributive thinking, however not all of us are equally aware of their importance, nor do we all practice and utilize them fully and equally in the form of deductive and inductive reasoning. Those who use them iteratively and cyclically would be experiencing conceptual change because the concept in question would be a target of decomposition and reconstruction (abstraction). If the concept stays the same, then its validity would be claimed, otherwise a conceptual change will have taken place. This is one of the foundations of scientific thinking (Dunbar & Klahr, 2012; Thagard, 2012). According to Paul Thagard, who specializes in philosophy and cognitive science, scientific thinking processes are no different than those used in everyday living by non-scientists. What he meant is that their essence is the same, and that the difference comes from how these processes are used. As explained in Yaşar et el., (2017), scientific thinking and biological computational thinking use the same cognitive processes as ordinary thinking. The difference is that an ideal scientist is someone who uses her mind's capacity for associative/distributive processing of information in a more iterative, cyclical, consistent, methodological, and habitual way.

Electronic CT skills naturally build upon cognitive processes that both scientists and non-scientists use, as shown in Fig. 6's outer layer. What it adds to them differently would be anything that is tied to the use of an electronic computing device for problem solving. Some of those uses, as mentioned before, would still be common to biological CT because they reflect the thinking of biological computing agents who uses them. And, that is probably why scientists heavily use electronic devices for modeling and simulation, as such use is driven by their own deductive and inductive thinking.

Figure 6: A cognitive framework on the essence of electronic CT skillset in terms of biological CT skills

Of all the characteristics of the electronic CT skill set listed in the literature (Grover & Pea, 2013), the two fundamental ones that have a correspondence with biological CT, as we defined here, are abstraction and decomposition. Others mostly seem to be rather device-dependent skills. Learning scientists and cognitive researchers point to abstraction as an inductive process. An example from our daily lives is that most of us do not care to see how cooks prepare our meals because we do not want to get bogged down with that level of details. Those who do and visit restaurant kitchens soon abandon this behavior, but others who continue to operate at increasingly smaller level of details (the bottom level in Fig. 3) can hardly function in society due to the attention they are paying to details. This myopic view of the world often causes burdens such as delayed action, indecisiveness, or inaction as dramatized by Shakespeare through the Hamlet character. By the same token, decomposition is a deductive process, often used to divide (scatter) complexity of an issue at hand into

smaller pieces and then attack each one separately until a cumulative solution is found (gathered). The famous "divide and conquer" phrase, supposedly by Napoleon, as well as 'many a little makes a mickle' by Benjamin Franklin all point to public awareness of the importance of the decomposition strategy.

If abstraction is considered the packing (modeling) of things, then decomposition is the unpacking and examination of its contents. We all live in a constant cycle of packing and unpacking of information based on our changing need for details and generalizations. This process is akin to the top-down <—>bottom-up cycle in Fig. 3. However, while everyone uses abstraction and decomposition, not all are equally aware of the importance of these two essential biological CT skills, nor are we all practicing and utilizing them fully and equally. They are also essential elements of the electronic CT skillset, as listed earlier, because of their use in programming and problem solving with electronic computers (Armoni, 2013). For example, abstraction is used to distribute the complexity of a code into seemingly independent layers and protocols in such a way to hide the details of how each layer does the requested service. Domain decomposition, on the other hand, is used in parallel computing to distribute the workload among multiple processors. CS educators wish that students would get a chance before college to improve these skills through curricular standards and CT practices.

Results From Use of Our CT Framework in Education

While modeling and simulation capacity was available only to a small group of scientists in national labs a few decades ago, a dramatic increase in access to and power of high performance computing and the drop in its cost in the past 20 years helped spread the use of CMSTs into the manufacturing industry and academic programs such as those described in Yaşar and Landau (2003). It was not until friendly versions of CMSTs were available for K-12 settings that a detailed and thorough empirical research was undertaken to measure their effectiveness on teaching and learning. We ran a 5-year (2003-2008) K-12 outreach program to follow learning theories and national recommendations of offering secondary school students an opportunity to learn science through modeling and simulation. However, what started as an outreach effort by practitioners slowly transformed into a qualitative and quantitative research study to document the impact on teaching and learning. Details of this outreach/research experimentation has been previously published (Yaşar et al., 2014, 2015, 2016). Here, we will briefly mention some of the findings that are relevant to our discussion.

Table 1.

A typical list of user-friendly modeling and computer simulation tools

Interactive Physics (IP): investigate physics concepts. *http://www.design-simulation.com/IP.*
AgentSheets: investigate biology concepts via games & simulations. *http://www.agentsheets.com.*
Geometer's Sketchpad (GSP): model geometrical concepts. *http://www.dynamicgeometry.com.*
Stella: investigate chemistry concepts via modeling of rate of change. *https://www.iseesystems.com*
Project Interactivate: online courseware for exploring STEM concepts. *http://www.shodor.org.*
Excel: constructs hands-on modeling & simulations using rate of change (*new = old + change).*
Scratch: a menu-driven language for creating games and simulations. *http://scratch.mit.edu.*
Python: An object-oriented language with simple and easy to use syntax. *http://www.python.org/.*

A multi-tier teacher-training program supported the use and testing of modeling and simulation tools (Table 1) in both regular classrooms and after school settings. Our hypothesis was that there is a positive relationship between teacher variables (knowledge and ability) and student outcomes (knowledge, ability, and interest). Three independent variables (technology, pedagogy, and training) were considered. Multi-year PD included 80 hours of technology knowledge (TK) training the 1st year, 80 hours of technological content knowledge (TCK) training in the 2nd year, and 40 hours of technological pedagogical content knowledge (TPACK) training in the 3rd year. Teachers received TK training in multiple tools but were offered TCK training

ISSN:2149-8504 (online)

to integrate their choice of tools with their content. While monetary and technology (laptops, smart boards, and software) support were offered, participation was voluntary. Studies of interdisciplinary TPACK (Mishra & Kohler, 2006; Kohler & Mishra, 2008) training on teaching and learning are relatively new but have been well documented (see www.tpack.org). Our focus has been rather on computational pedagogical content knowledge (CPACK; a subset of TPACK) development (Yaşar et al., 2016) and its cognitive framework (Yaşar, 2016). A brief description of the CPACK framework was introduced previously.

More than 300 in-service STEM teachers from 15 schools, including 13 secondary schools (grades 7-12) from an urban (Rochester City, NY) School District (RCSD) and a middle school and a high school from a suburban (Brighton Central, NY) School District (BCSD) took part in activities of this inititative, which included summer professional development for in-service teachers; pre-service courses on computational methods, modeling tools, and pedagogoy; technology and curriculum support for participating schools; project-based afterschool programs; and annual competitions for students. The interdisciplinary aspect of the study and the need for teacher motivation/customization to implement new technologies necessitated a quasi-experimental design with mixed-methods (Creswell, 2012), involving collection and analysis of qualitative data (such as interviews, activity logs, observations) to identify variables as well as to understand and triangulate the quantitative data (such as Likert-scaled surveys, artifacts, report cards, test scores, and standardized exams).

Annually, we had about 50 active teachers in the program who each taught approximately 100 students in a school year. Modeling software tools in Table 1 were made available to all participating teachers, along with supporting technologies such as laptops, LCD projectors, and electronic smartboards. These menu-driven, user-friendly, and non-programming tools allowed students to quickly set up and run a model using an intuitive user interface with no knowledge of equations, scientific laws, and programming or system commands. An after-school project-based challenge program provided students more time and freedom than a regular classroom setting to apply, test, and revise the constructed computational models. More than 80% of teachers surveyed annually indicated that they utilized modeling tools in either classroom instruction or special projects. More than 90% of teachers who used tools provided to them by the project agreed that using modeling tools in their classrooms significantly increased student engagement and made math and science concepts significantly more comprehensible. While science classes utilized technology less due to limited access and lack of science-related modeling examples, in instances where it was utilized, a deeper understanding of science topics was achieved, compared to math topics (83% vs. 76%). Professional evaluators triangulated teacher reports through their own classroom observations.

Student learning data from report cards and NY State exams were found to be consistent with the survey data provided by teachers. For example, Tables 2 through 5 show passing rates (>65/100) in NY State Regents Physics/Math Exams as well as graduation rates in four urban (RCSD) and one suburban (BCSD) high schools with more than 30% of its math and science teacher workforce trained by the initiative. Student responses, except one case with a small sample size, from each school point out to a statistically significant ($0.01 < p < 0.05$) upward trend over the five-year study during which teachers were supported through summer and academic-year workshops, stipends, technology, and mentoring support. District averages are shown in Tables 4 and 5. RCSD passing rate average for NY State Grade 7-8 Math exam also improved: 10%→44%. Improvements over the baseline data were all statistically significant ($p<0.01$). No math teachers from BCSD middle school participated in the program because its baseline passing rate for NY Grade 8 Math was at 89%. Other known factors that may have affected statistics include RCSD's district re-organization into single secondary schools, State's redesign of its exams, and technological reform by these districts. A few control and target comparisons made in early phases of the project, before the control group was lost, consistently show favorable results. For example, a pair of teachers from the same high school taught properties of quadrilaterals in a math class. Class averages for the same unit test were 82.5 (size 24, using modeling tools) versus 49.5 (size 14, using conventional methods). Another study involved State's math exam scores of groups with similar sizes (25 students) in an annual challenge at three levels: Grade 7-8 Math: 64.0 vs. 58.6; Grade 9-10 Math-A: 60.26 vs. 49.54; Grade 11-12 Math-B: 71.9 vs. 55.6.

ISSN:2149-8504 (online)

To circumvent curricular limitations, we offered an afterschool program through which participating teachers and student clubs organized a project-based annual competition. This program was also a way of doing an enriched case study with a qualitative component (e.g., interviews and observations) to explore the meaning of the quantitative trends/findings we learned in the student achievement data. Each year, top three team projects selected from school-based competitions were later submitted to a multi-school competition involving school districts. A rubric with good psychometric properties was developed and tested by computing and teaching experts. Project topics included addressing challenges of environmental issues and misconceptions. They allowed students enough time to progress at their own pace and resolve issues that they wanted to address.

Annually, more than 200 students had a full semester to develop 4-person team projects. Scoring rubric included problem statement, application of the model to a problem of interest, data analysis, teamwork, originality, electronic demonstration, and presentation of the results before a panel. Extra points were given for use of multiple tools, demonstrated understanding of computational, mathematical and scientific content, level of challenge, and knowledge and skills demonstrated beyond team's grade level. The incentives helped push students to go beyond initial job of model construction, playful experimentation, and introductory exposure to STEM concepts.

Table 2.
Passing rate at RCSD high schools

Regents Math-A	Baseline data		5 years later		p value
	Size	Rate	Size	Rate	
School 1	77	5%	427	62%	<0.01
School 2	319	13%	274	61%	<0.01
School 3	441	35%	384	75%	<0.01
School 4	43	21%	262	63%	<0.01

Table 3.
Passing rate at RCSD high schools

Regents Physics	Baseline data		5 years later		p value
	Size	Rate	Size	Rate	
School 1	21	0%	26	22%	<0.05
School 2	240	3%	162	31%	<0.01
School 3	11	0%	6	17%	<0.16
School 4	153	16%	81	26%	<0.05

Table 4.
Passing rate at BCSD high school

Regents Exam	Baseline data		5 years later		p value
	Size	Rate	Size	Rate	
Math-A	51	51%	295	97%	<0.01
Physics	123	52%	132	77%	<0.01
Diploma	259	84%	285	95%	<0.01

ISSN:2149-8504 (online)

Table 5.
Average passing rate at RCSD

Regents Exam	Baseline data		5 years later		p value
	Size	Rate	Size	Rate	
Math-A	880	23%	1347	65%	<0.01
Physics	425	7%	275	27%	<0.01
Diploma	1021	20%	1178	52%	<0.01

A case from this program has been reported by a group of these students in Yaşar et al., (2005, 2006). These papers offer a testimony of how high school students with no prior content knowledge slowly gained a deeper understanding of scientific and computing content. While they initially used tools whose operation they did not understand, they were eventually able to replicate the same simulations using a simple rate-of-change formula (new = old + change) with Excel. This is also consistent with themes extracted from qualitative data involving many team projects (n>300). Modeling examples from these projects are incorporated into lesson plans that are now freely available at http://digitalcommons.brockport.edu/cmst_institute/. They are being utilized at a rate of 80-100 downloads per day by educators around the world.

It appears that students' use of simple hands-on process of numerical integration via Excel helped them realize the virtue of decomposing a problem, as finer decomposition gave them more accurate answers (Yaşar and Maliekal, 2014). Because of limitations of Excel, the need for more accuracy, faster automation, and better control motivated them to seek other tools, including programming languages. A simple loop written in Python basically got them all they needed.

In hands-on modeling, the key is computation of change in the "new = old + change" equation. Since change is caused by laws of nature, it motivates students to learn about scientific laws that drive the change in nature. And, there are only a handful of them because whether they solve for harmonic motion of a pendulum, orbital motion of a planet, or launching of a rocket, change is caused by the same gravitational force. So, by knowing the formula for the force, they can get the change in velocity from acceleration (a=Force/mass) and compute the new velocity. This will give them the change of position (velocity x elapsed time), which they can use to get the new position. If one does this for many instances of time, then they would have a time-dependent profile of position and velocity. Because, solving these problems via a programming language could provide better control over decomposition, accuracy and automation, modeling and simulation is a great way to motivate students to learn knowledge of scientific laws and programming.

Our findings are consistent with a growing body of research that identifies computer simulation as an exemplar of inquiry-guided (inductive) learning through students' active and increasingly independent investigation of questions, problems and issues (Bell et al., 2008;Bell & Smetana, 2008; de Jong & van Joolingen 1998; Rutten et al. 2012; Smetana & Bell 2012; Wieman et al. 2008). Effectiveness of computer simulation in education is also well grounded in contemporary learning theories that recognize the role of abstract thinking and reflection in constructing knowledge and developing ideas and skills (Donovan & Bransford, 2005; Mooney, 2013).

Conclusion

We consider heterogeneity both as the cause and outcome of the dual (associative/distributive) behavior of stuff around us, including matter and information. While the physical matter has been showcasing this behavior through our sensory information for ages, we have only recently theorized that information could be considered in terms of quantifiable constructs. Such a theory by Alan Turing led to invention of electronic computing devices to imitate biological computing devices. Since then, electronic computing devices have grown in complexity and functionality to resemble biological ones. While we do not suggest that they are

the same, they do process information in some common ways at the fundamental level. We argue that one of these common processes is modeling and simulation because it reflects device-independent associative and distributive aspects of information processing.

Our cognitive framework indicates that dual dynamics of information storage, retrieval, and processing by a computational mind is the very essence of thinking (or, computational thinking). While all utilize it in cognitive functioning, not everyone uses it as iteratively, consistently, frequently, and methodologically as scientists. In the past century, scientists have even invented electronic devices to help them automate computations involved in deductive and inductive reasoning of conceptual change process used in the scientific method. We have come to call these computations "modeling and simulation." Recent advances in technology has made real-time computer simulations possible, thereby effectively aiding the scientific progress in the past fifty years.

We infer from empirical data that modeling and simulation carries a constructivist pedagogy whose iterative and cyclical nature mirrors Kant's epistemological method represented in Fig. 3. Basically, modeling provides a general simplistic framework from which instructors can deductively introduce a topic without details, and then move deeper gradually with more content after students gain a level of interest to help them endure the hardships of effortful and constructive learning. This deductive approach takes away the threatening and boring aspect of STEM learning. Simulation, on the other hand, provides a dynamic medium to test the model's predictions, break it into its constitutive parts to run various what-if scenarios, make changes to them if necessary, and put pieces of the puzzle together inductively to come up with a revised model. It provides a dynamic medium for the learner to conduct scientific experiments in a friendly, playful, predictive, eventful, and interactive way to test hypothetical scenarios. This inductive process enables the learner to put pieces of the puzzle to come up with a revised model. Anyone who learns in this fashion would, in fact, be practicing the craft of scientists.

Pedagogical use of modeling and simulation in K-12 classrooms goes well beyond its benefits in math and science education. Students who experience learning in an iterative and cyclical process of deductive and inductive reasoning can transform such thinking to problem solving in computing as well. It can help students develop an understanding of abstraction and decomposition as well as an appreciation for computer programming. A deductive habit in programming practice could help programmers decompose a whole code into its smaller pieces and deal with each one separately until a cumulative solution is found. This is how large codes such as computer and network operating systems are divided up into a hierarchy of layers of varying functionality. Parallel computing is another example of decomposition. At the same time, an inductive habit in programming could have just an opposite benefit by pushing the programmer to see a larger picture and group together seemingly unrelated pieces of programming. Again, this is how complexity of large codes are divided into layers of growing simplicity.

If viewed more carefully from other fields such as philosophy, epistemology, physics, modeling can be quickly seen a more general and pervasive process than its role in electronic and biological computing. It appears to be rather a universal process by which all heterogeneous stuff seems to form and grow. So, as a universal process representing computable behavior of physical matter, it could help put computing at the heart of sciences and convince skeptics that computer science deals with natural phenomena, not artificial phenomena. Accordingly, we argue that it should be lifted beyond its recognition as a tool for scientific research.

While this study was originally motivated by an effort to capture the essence of computational thinking, one of its side benefits has been to link it to scientific thinking. Besides its ramifications for computing education, such a link can help clear up two major myths in science education by not only illustrating that scientific thinking is no different than ordinary thinking but also raising awareness that the scientific method is rather a two-way process, not a one-way linear process as perpetuated to this day by many textbooks and curricular resources. We hope that our perspective will help persuade public and young students that understanding and obtaining the mind of a scientist and an engineer is within their reach. Teaching young minds an awareness of

ISSN:2149-8504 (online)

computation-generated cognition as well as a cognitive habit of conceptual change could help them think like a scientist and be prepared to use electronic computing devices to further such thinking regardless of whether they work as a scientist or not.

Acknowledgement

Support by the National Science Foundation, through grants EHR 0226962, DRL 0410509, DRL 0540824, DRL 0733864, DRL 1614847, and DUE 1136332, is greatly appreciated.

References

Armoni, M. (2013) "On Teaching Abstraction to Computer Science Novices." J. Comp in Math & Science Teaching, 32(3); 265-284.

Bell, L. R., Gess-Newsome, J. and Luft, J. (2008) Technology in the Secondary Science Classroom. National Science Teachers Association (NSTA).

Bransford, J., Brown, A. and Cocking, R. (2000). How People Learn: Brain, Mind, Experience, and School. National Academy Press, Washington, D.C.

Brown, P. C., Roediger, H. L. and McDaniel, M. A. (2014) Make it Stick. The Belknap Press of Harvard.

Creswell, J. W. (2012) Educational Research. 4th Edition. Pearson Education, Inc.

Csikszentmihalyi, M. (1990). Flow: The Psychology of Optimal Experience. New York: Harper Collins.

de Jong, T., & Van Joolingen, W. R. (1998). "Scientific Discovery Learning with Computer Simulations of Conceptual Domains." Review of Educational Research, 68(2); 179-201.

Donovan, S. and Bransford, J. D. (2005). How Students Learn. The National Academies Press, Washington, D.C.

Dunbar, K. N., & Klahr, D. (2012). Scientific Thinking and Reasoning. In K. J. Holyoak and R. G. Morrison (Eds.), The Oxford Handbook of Thinking and Reasoning (pp. 701-718). London: Oxford University Press.

Evans, J. and Frankish, K. (2009). In Two Minds: Dual Processes and Beyond. Oxford University Press: Oxford.

Findlay, S. D. and Thagard, P. (2012). "How parts make up wholes." Frontiers in Physiology, 3, 455. doi: 10.3389/fphys.2012.00455.

Giere, R. N. (1993). Cognitive Models of Science. Minneapolis, MN: University of Minnesota Press.

Goleman, D. (2006). Emotional Intelligence. New York: Bantam Dell.

Grover, S. & Pea, R. (2013). Computational Thinking: A Review of the State of the Field." Educational Researcher, 42(1); 38-43.

Guzdial, Mark. (2008). Paving the way for computational thinking. Communications of the ACM 51(8); 25-27.

Harte, V. (2002). Plato on Parts and Whole: The Metaphysics of Structure. Oxford, NY: Oxford University Press.

Hawking, S. (1988). A Brief History of Time. Random House: New York.

Hawkins, J. (2004). On Intelligence. Times Books: New York.

Hebb, D. (1949). The Organization of Behavior. New York: Wiley & Sons.

Kant, I. (1787). The Critique of Pure Reason. Translated by J. M. D. Meiklejohn. eBook@Adelaide, The University of Adelaide Library, Australia.

Koehler, M. J., and Mishra, P. (2008). "Introducing TPCK," in Handbook of Technological Pedagogical Content Knowledge (TPCK) for Educators. Routledge Press: New York.

Koffka, K. (1935). Principles of Gestalt Psychology. New York: Harcourt, Brace, and World.

Kuhn, T. (1962). The Structure of Scientific Revolutions. Chicago: Univ. of Chicago Press.

Langley, P. (2000). Computational support of scientific discovery. International Journal of Human-Computer Studies, 54, 393-410.

MacDonald, M. (2008) Your Brain: The Missing Manual. O'Reilly Media: Canada.

Mishra, P., Koehler, M. J. (2006). "Technological pedagogical content knowledge: A framework for integrating technology in teacher knowledge." Teachers College Record, 108 (6), 1017-1054.

Montague, R. (2006). How We Make Decisions. Plume Books: New York.

Mooney, C. G. (2013). An Introduction to Dewey, Montessori, Erikson, Piaget, and Vygotsky. St. Paul: Redleaf Press.

Morgan, M. H. (2007) Lost History: The Enduring Legacy of Muslim Scientists, Thinkers, and Artists. Washington D.C.: National Geographic Society.

National Research Council (NRC) Report (2012). A framework for K-12 science education. Washington, DC: National Academies Press.

National Science Foundation (NSF) Report (2006). Simulation-Based Engineering Science: Revolutionizing Engineering Science through Simulation. Washington, D.C.

Oden, T. and Ghattas, O. (2014). Computational Science: The "Third Pillar" of Science. The Academy of Medicine, Engineering & Science of Texas' (TAMEST's) Annual Conference January 16-17, 2014.

Papert, S. (1980). Mindstroms: Children, computers, and powerful ideas. New York: Basic Books.

President's Information Technology Advisory Committee (PITAC) Report (2005). Computational Science: Ensuring America's Competitiveness. Retrieved from http://www.nitrd.gov/pitac/reports/20050609_computational/computational.pdf.

Prince, M. J. and Felder, R. M. 2006. "Inductive Teaching and Learning Methods: Definitions, Comparisons, and Research Bases." J. Engr. Education, 95 (2); 123-138.

Repenning, A. (2012). Programming Goes Back to School. Comm. of the ACM, 55(5), 35-37.

Rockmore, T. (2011). Kant and Phenomenology. Chicago: The University of Chicago Press.

Rutten, N., van Joolingen, R., and van der Veen. (2012). The Learning Effects of Computer Simulations in Science Education. Computer & Education, 58; 136-153.

Simon, H. (2016). Supercomputers and Super-intelligence. 17th SIAM Conference on Parallel Processing for Scientific Computing, Paris, April 12 – 15, 2016.

Smetana, L. K. and Bell, R. L. (2012). Computer Simulations to Support Science Instruction and Learning: A critical review of the literature. Int. J. Science Education, 34 (9); 1337-1370.

Sun, R. (2002). Duality of mind: A bottom-up approach towards cognition. Mahwah, NJ: Lawrence Erlbaum Associates.

Tenenbaum, J. B., Kemp, C., Griffiths, T. L. & Goodman, N. D. (2011). How to Grow a Mind: Statistics, Structure, and Abstraction. Science, 331, 1279-1285.

ISSN:2149-8504 (online)

Thagard, P. (2012). The Cognitive Science of Science. Cambridge, MA: The MIT Press.

Turing, A.M. (1936). On Computable Numbers, with an Application to the Entscheidungs problem. Proceedings of the London Mathematical Society. 2 (1937) 42: 230–265.

Vosniadou, S. (2013). International Handbook of Research on Conceptual Change. 2nd Edition. New York and London: Routledge.

Vygotsky, L. S. (1930). Mind in society: The development of higher psychological processes. Cambridge, MA: Harvard University Press.

Wieman, C. E., Adams, W. K. and Perkins, K. K. (2008). PhET: Simulations That Enhance Learning. Science, 332, 682-83.

Wing, J. M. (2006). Computational Thinking. Comm. of the ACM, 49(3); 33-35.

Wing, J. M. (2011). Research notebook: Computational thinking – What and why? The Link Magazine. March 06, 2011.

Yang, Y. (2009). Target discovery from data mining approaches. Drug Discovery Today, 54 (3-4), 147-154.

Yaşar, O. & Maliekal, J. (2014). Computational Pedagogy. Comp. in Sci. & Eng., 16 (3), 78-88.

Yaşar, O. & Maliekal, J., Veronesi, P. and Little, L. (2014). An Interdisciplinary Approach to Professional Development of Math, Science & Technology Teachers. Comp. in Math & Sci. Teaching, 33 (3), 349-374.

Yaşar, O., Maliekal, J., Veronesi, P., Little, L. and Vattana, S. (2015) Computational Pedagogical Content Knowledge. In L. Liu and D. C. Gibson (Eds), Research Highlights in Technology and Teacher Education (pp. 79-87). ISBN: 978-1-939797-19-3.

Yaşar, O. (2015). A Universal Process: How Mind and Matter Seem to Work, Science Discovery. 3 (6), 76-81. doi: 10.11648/j.sd.20150306.16.

Yaşar, O. (2016). Cognitive Aspects of Computational Modeling & Simulation. J. Computational Science Education, 7 (1), 2-14.

Yaşar, O., Veronesi, P., Maliekal, J., Little, L., Vattana, S., and Yeter, I. (2016). Computational Pedagogy: Fostering A New Method of Teaching. Comp in Edu., 7 (3), 51-72.

Yaşar, O. (2017). Modeling & Simulation: How Everything Seems to Form and Grow. Comp. in Sci. & Eng., 19 (1), 74-78.

Yaşar, O., Maliekal, J., Veronesi, P. and Little, L. (2017). The essence of scientific and engineering thinking and tools to promote it. Proceedings of the American Society for Engineering Education Annual Conference, Columbus, OH, June 25-28.

Yaşar, P., Kashyap, S., & Roxanne, R. (2005). Mathematical and Computational Tools to Observe Kepler's Laws of Motion. MSPNET, http://hub.mspnet.org/index.cfm/14566.

Yaşar, P., Kashyap, S., and Taylor, C. (2006). Limitations of the Accuracy of Numerical Integration & Simulation Technology. MSPNET. http://hub.mspnet.org/index.cfm/14568.