Xu ZW, Tu DD. Three new concepts of future computer science. JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY 26(4): 616–624 July 2011. DOI 10.1007/s11390-011-1161-4

Three New Concepts of Future Computer Science

Zhi-Wei Xu (徐志伟) and Dan-Dan Tu (涂丹丹)
Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China

E-mail: zxu@ict.ac.cn; tudandan@software.ict.ac.cn

Received December 15, 2010; revised May 9, 2011.

Abstract This article presents an observation resulted from the six-year Sino-USA computer science leadership exchanges: the trend towards the emergence of a new computer science that is more universal and fundamental than that in the past. In the 21st century, the field of computer science is experiencing fundamental transformations, from its scope, objects of study, basic metrics, main abstractions, fundamental principles, to its relationship to other sciences and to the human society, while inheriting the basic way of thinking and time-tested body of knowledge accumulated through the past 50 years. We discuss three new concepts related to this trend. They are *computational lens* and *computational thinking* articulated by US scientists, and *ternary computing* for the masses proposed by Chinese scientists. We review the salient features of these concepts, discuss their impact, and summarize future research directions.

Keywords computational lens, computational thinking, ternary computing

1 Introduction

From June 2006 to June 2010, the National Natural Science Foundation of China (NSFC) and the National Science Foundation of the United States (NSF) organized a series of academic exchanges among computer science community leaders of the two countries. The six-year academic exchanges were conducted through workshop presentations, stimulating discussions and debates, and site visits to universities, research institutes, and companies.

The exchanges are fruitful. Computer scientists from the largest developed country and the largest developing country not only discussed research work of computing science and technology, but also exchanged their perspectives on future challenges to computer science research, education, human resources, ecosystem building, and impact to the human society.

Many differences between China and USA emerged from the exchanges. For instance, while the US academic researchers focus more on high-impact transformative research, the R&D work in China currently emphasizes more on relevancies to the nation's economic and social needs. However, the computer science communities of the two countries share many common requirements, challenges and perspectives.

A common observation is that the very role (hence the value proposition) of computer science is changing, in relation to other sciences and to the human society. Computer science should not be viewed only as a human created tools discipline that provides hardware, software and services to other disciplines. Computing is more universal and fundamental. Computation is inherent in Nature and in the human society. Computation exists not only within, but also before and besides the current computer science and the cyberspace.

This insight could significantly impact the future development of computer science in the 21st century, from its scope, objects of study, future research topics, basic measures and metrics, main abstractions, fundamental principles, computing artifacts and systems, to computer science education. The transformations are enrichment, not replacement. We should inherit our field's basic way of thinking and time-tested body of knowledge accumulated through the past 50 years①.

This article discusses three new concepts related to this insight, that arise from the six-year Sino-USA computer science leadership exchanges. They are *computational lens* and *computational thinking* articulated by US scientists, and *ternary computing* for the masses proposed by Chinese scientists. We review the salient features of these three concepts, discuss their impact

---

Survey

The author's work is supported in part by the Frontier Research Project of Chinese Academy of Sciences and the National Basic Research 973 Program of China under Grant Nos. 2011CB302500, 2011CB302800.

① The field of modern computer science can be traced back to Turing's 1936 paper or to the creation of ENIAC in 1946. A more conservative convention is to mark 1962 as the starting year of the computer science discipline. In that year, the first computer science degree program in the United States was formed at Purdue University, and the first computer science professional society was formed within China Electronics Federation, which society in 1985 became an independent China Computer Federation.

and summarize future research directions.

2 Salient Features of Computational Lens

In [1], Richard Karp articulates a new relationship between computer science and other sciences, called *computational lens*. The relationships of computer science and other sciences have grown through four phases, as shown in Table 1. A later phase complements, not replaces, earlier phases. All four relationships exist today.

Table 1. Four Phases of Relationship Between Computer Science and Other Fields of Science and Engineering

Phase	Name	Main Characteristics (computing is used for)
I	Numerical Analysis	Solving equations that model physical phenomena
II	Computational Science	Simulation and visualization of the physical world
III	e-Science	Managing massive experimental data and collaborating via the Net
IV	Computational Lens	Computing as a universal way of thinking

The first phase is *numerical analysis*, where computing is used to solve various systems of equations modeling physical problems. Computer scientists provide computing devices and algorithms for computational speed and accuracy in solving equations.

The second phase is *computational science*, which emphasizes computational simulation and visualization of physical models. Computer scientists provide fast simulation algorithms and help produce simulation results close to the reality.

For instance, we know that proteins fold into three dimensional structures, which determine their biological functions. However, we do not have good systems of equations available relating proteins to their 3D structures. Computer simulation can help in such situations. We utilize first principles, i.e., the particles of a protein still follow basic laws of physics, and use a computer algorithm to mimic the behavior of the particles step by step, to eventually reach a minimized energy configuration, which is deemed to be the 3D structure of the protein. Using straightforward Monte Carlo methods, computer simulation of a protein folding has a time complexity of 200^n, where n is the number of amino acids in the protein. A recent algorithm reduces the computational time complexity to 1.66^n. Furthermore, the produced 3D structures are close to the real ones, with a root mean square deviation of only 0.557 angstrom for some proteins[2].

The third phase is *e-science*, which emphasizes turning massive data into knowledge. Computer science provides hardware and software tools for other sciences, especially efficient data management tools to organize, process and visualize massive, empirically obtained scientific datasets, over distributed networks. This networking nature facilitates scientific collaboration.

All the above three relationships have one commonality: computer science is in the service of other science and engineering fields. The scientific phenomena and processes being studied are perceived as physical, chemical, biological, or social. Computer science is there to offer productivity tools, being they algorithms, hardware or software, to help the investigation of these scientific and engineering phenomena and processes.

The fourth phase, the new relationship of *computational lens*, is qualitatively different and more fundamental: the scientific and engineering phenomena and processes being studied are perceived as *computational*. In many cases, the processes can be perceived as computational because they *are* computational in nature, in that they perform "transformation of information on numeric data"[1]. In other words, the most striking characteristic of computational lens is that it does not merely provide a new set of methods, software or hardware tools to help other sciences. Rather, it is a new metaphor and a new way of thinking.

Evidence abounds supporting this viewpoint of computational lens. Many examples are discussed in [1], ranging from regulation of protein production, metabolism, phase transitions in physical systems, mechanisms of learning, molecular self-assembly, strategic behavior of companies, evolution of Web-based social networks, quantum computing, to interactions between computer science and mathematics.

To summarize, computational lens states the following[1]:

- Many processes in natural, engineered and social systems are computational, in that they perform information transformation.
- Thus, computational lens gives us a new way of thinking in other fields. We can study these natural, engineered and social processes as computational processes, by identifying their computational primitives, discovering their computational models, and finding out their computational requirements or capabilities. These could lead to new scientific insights and engineering methods.
- The traditional computer science can be enriched by studying natural, engineered and social processes as computational processes. In particular, the very concept of computation, the traditional computability theory and algorithm theory may undergo significant changes.

3 Salient Features of Computational Thinking

In [3-4], Jeannette Wing articulates a concept and vision called *computational thinking*. This is a rich concept and far-reaching vision that is still evolving. We can study, understand, and develop computational thinking from the following five angles.

Computational Thinking Is Universal. Computational thinking is not limited to the cyberspace, but a universal way of thinking generally useful in all fields of human endeavor. In this regard, it is similar to mathematical thinking, scientific thinking, and engineering thinking. This echoes Richard Karp's perspective on computational lens. In other words, in addition to the mathematical, scientific, and engineering ways of thinking, the computational way of thinking provides a powerful and general new lens to design complex systems, to solve real and abstract problems, and to understand Nature and the human society. Jeannette Wing envisions that "computational thinking will be instrumental to new discovery and innovation in all fields of endeavour"[4].

Computational Thinking is Fundamental. Computational thinking is not only for experts or information technology professionals. It is such a basic and fundamental capability and way of thinking that every school child should learn, as the kids learn reading, writing, and arithmetic today. Jeannette Wing envisions that "computational thinking will be a fundamental skill used by everyone worldwide by the middle of the 21st Century"[3] and "computational thinking will be an integral part of childhood education"[4].

We can see some signs towards this vision in China. More than 6 million students graduated from colleges in China every year. All of them are expected to have taken a Computing 101 course. Furthermore, "Algorithm" is now a module of Mathematics in Chinese high schools.

Computational Thinking Provides Mental Tools. Teachers and students of computing in China face a challenge: the Chinese society has a persistent view of computing as a tools discipline, providing hardware, software, and service tools to the society. In contrast, the medicine profession is viewed as a value discipline, providing life-long value to the society[5].

Jeannette Wing points out that this is an incomplete perception. It is true that the computing profession provides valuable automation tools to the society. But these "metal tools", manifested as hardware, software, and services, are only part of what we provide. Computing also provides "mental tools", i.e., computational abstractions of the world. These two types of tools work together towards providing life-long value to the society.

Computational Thinking Has Rich Contents. Computational thinking is not merely a philosophical idea, but has rich contents, many of which are yet to be studied and created. In 2004, a committee of the US National Research Council summarized a number of basic features of computer science[6]: computer science studies symbols, abstractions, algorithms, artifacts, exponential growth, limits of computing, and intelligence. A study by the National Natural Science Foundation of China in 2010 characterizes computing from another perspective[4]: the computing field studies all phenomena and relationships surrounding the full lifecycle of information, including production, acquisition, transmission, storage, processing, display, and use of information.

Jeannette Wing offers her perspective that the "essence of computational thinking is abstraction" and "abstractions are the 'mental' tools of computing"[4]. Identifying the right abstractions, defining abstraction layers, and efficiently interpreting abstractions on computers, are all contents of computational thinking.

Computational Thinking Poses Deep Scientific Problems. Computational thinking not only provides metal and mental tools, but also gives a new framework to ask meaningful scientific questions. Jeannette Wing poses six such questions[4]. Her six questions all have to do with another most basic question: "what is computation?" We discuss this question in more details in Subsection 6.2.

4 Salient Features of Ternary Computing

The concept of ternary computing for the masses was originally proposed in 2001, when China started to make its 2001∼2020 mid-term plan of science and technology development. The concept was further developed in the subsequent years in a number of studies by Chinese Academy of Sciences[5,7]. The ternary computing concept has the following salient points.

Computing for the Masses. A most important trend of computing in the 21st century is for computing to reach the masses (i.e., over 80% of the population). Furthermore, computing for the masses requires ordinary citizens to utilize information technology (IT) in a *ternary universe* comprised of the human society, the cyberspace, and the physical world (including Nature and human-engineered things). We are still far from reaching the state of universal use of IT, as most people today access IT only through explicit IT devices such as a PC, a laptop, or a cellular phone.

Utilizing Ternary Resources. In the strict sense, traditional computer science is unary computing. Traditional IT mainly utilizes resources in the cyberspace to solve a problem. We need a new computer science that solves problems by efficiently utilizing resources of the ternary universe. People, machines and things work together computationally in a problem solving process.

Extending the Scope of Computing. Consequently, the scope of the computer science study should not be limited to the cyberspace, but must be extended to *the Net*, which includes all computational aspects of the ternary universe. The human society, the cyberspace and the physical world are all objects of study for computer science. Computation exists not only in the cyberspace, but also in the human society and in the physical world. Computation is a fundamental, inherent activity that Nature and the human society actually do, only that we did not see it clearly before. A most interesting and promising arena is when computing intermingles all the three worlds and a computational process includes cyber-physical-social components.

Enriching Computer Science. The ternary universe perspective enriches traditional computer science by inspiring new cross-disciplinary methodologies and ways of thinking. The problems of enquiry, the basic metrics, the main abstractions, the principles of computer science, as well as other computational sciences, can all be enriched.

As an example, Fig.1 illustrates how to utilize ternary resources to solve two computational problems: 1) ranking households by their electricity consumption behaviors, and 2) upgrading electric devices (appliances) through R&D with field data.

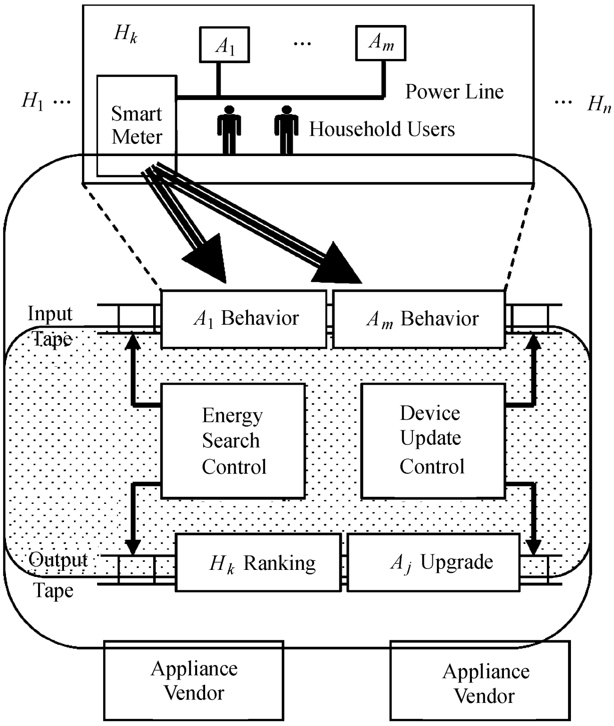

Fig.1. An example of sensing household electric devices to show that ternary (cyber-physical-social) resources are utilized to solve two computational problems.

We want to find out the accurate electricity usage behavior of every household electric device (called appliance) in all the households of a city. In a city like Beijing, we have millions of such households where each household has a few dozens devices ($n > 1$ million and $m =$ tens in Fig.1). Within each household, *things* are the m appliances (lamps, refrigerators, washers, computers, etc.), *people* are the household users, and *machines* are a novel smart meter installed in the power line. Information of how the household uses electricity, i.e., the household users behavior and the behavior of each appliance, exists outside the cyberspace. The smart meter, together with possibly a backend learning computational process, can sense and distinguish the behavior of each of the m appliances, and put the m pieces of behavior data on the Input Tape as input data.

These data are then used by two computational processes (Energy Search and Device Upgrade) to generate on the Output Tape the ranking result of every household H_k and the result of how to upgrade every type of appliance A_j.

An appliance vendor (e.g., a refrigerator manufacturer) can utilize field data on how each of the millions of refrigerators of a particular model behaves in the millions of households, to upgrade and to research and develop better products. This gives the appliance industry the ability to run beta tests and field tests previously not available.

The above is different from a traditional computational model (shown in the shaded box in Fig.1). Such a model (e.g., a narrowly interpreted Turing machine or a RAM model) assumes that the input data is given on the Input Tape when a computational process starts. Then a step-by-step, mechanic algorithm is executed to yield a desired output result on the Output Tape. Everything happens in the cyberspace.

5 Impact of the Three Concepts

The three concepts of computational lens, computational thinking, and ternary computing are articulated recently. But the theses and ideas behind them have been in discussion by the computer science community for quite a few years. These concepts already have four types of impact to the computer science filed.

New Research Agenda. These three concepts have gained support from the scientific communities both in the USA and in China. The US National Science Foundation started in 2008 to fund related projects through its major new initiatives such as Cyber

Enabled Discovery and Innovation (CDI) and Expeditions in Computing. The Chinese Academy of Sciences recently started its 2011～2030 Frontier Research Projects, where the ternary computing concept plays a central role in the future information technology project. The trend towards a ternary universe is also recognized in the strategic study reports on information technology development for the period of 2011～2020, by both the Ministry of Science and Technology of China and the National Natural Science Foundation of China[8].

New Subfields. These three concepts also stimulate the formation of new and often cross-disciplinary subfields and communities. Examples include algorithmic game theory[9] and computational sustainability (e.g., see www.cis.cornell.edu/ics). In China, a subfield called "Internet of physical objects" (物联网) has attracted a lot of attention, and has become a degree program in a few dozen universities.

New Research Topics and Results. These three concepts could start many new research topics. An inspiring and interesting example is Leslie Valiant's computational study of evolution[10].

New Institutions. Examples of new academic institutions include the Institute of Computational Thinking in Carnegie Mellon University, the Institute of Computational Sustainability at Cornell University, and the Frontier Research Institute at Chinese Academy of Science.

6 Future Research Directions

Broadly speaking, there are three directions that need substantial research work: computer science education, fundamental problems of enquiry, and computational models.

6.1 Computer Science Education

Algorithm is now a module of high school mathematics in China. But Jeannette Wing believes that computational thinking is even more basic, as fundamental as reading, writing and arithmetic skills that primary school kids should learn[3]. She poses a challenge: "What are effective ways of learning (teaching) computational thinking by (to) children?"[4]

The current algorithm module in Chinese high schools teaches the algorithm concept as a deterministic process of a finite number of mechanic primitive steps, manifested as a flowchart or a program of programming language statements. It does not involve notions of time or space complexity, nor does it teach students what is or how to design an efficient algorithm. It compensates this lack by offering a lot of examples and exercises drawing from Chinese and Western arithmetic and mathematic sample problems.

What we need is a coherent, fundamental sequence of bodies of knowledge that covers students from the primary school to the graduate school (at least to college). We also need to develop effective learning environments that utilize computing hardware, software and services, as well as resources in the cyber-physical-social ternary universe.

6.2 Fundamental Problems of Enquiry

In the past several years, some computer scientists in China started to ask whether we need to revisit traditional computational models such as the Turing machine model and the von Neumann architecture of computing (cf. [11]). There is definitely such a need (see Subsection 6.3). But a more important issue is to identify fundamental scientific problems of enquiry that need such revisits. If we ask the same type of scientific questions within the framework of say, the decision problem, the Church-Turing hypothesis suggests that the Turing machine model should be enough for computability questions.

In fact, we can see some signs of asking new questions, especially when phrasing a problem crossing the cyberspace boundary and in the framework of the ternary universe. The 2009 DARPA Red Balloon Challenge[12] is such an example. A Chinese example is the real case of sending kids to the Shanghai World Expo. In July 2010, a group of volunteers (people), utilizing the Internet (machines), self organized and worked together by relays of automobiles (things), to transport 10 primary school kids (things) from a remote, poverty stricken village to Shanghai to see the World Expo. This ternary computational process all took 8 days. It begs the question whether similar ternary computation processes exist that benefit 10 thousand kids and finish in 8 hours.

The three concepts of computational lens, computational thinking, and ternary computing all say that computing is universal and fundamental in natural, engineered, societal, and cyber systems. A most basic question is: "What is computation?"

If we cannot answer this question, we can start by asking: What is a computational process? What is *not* computational? What is different of a computational process from a physical, a chemical, or a biological process?

We can appreciate the differential nature of a computational process by focusing on the three attributes of any scientific inquiry: *material*, *method*, and *measurement*. Table 2 compares and contrasts computational processes with chemical processes using this approach.

Table 2. Chemical Process Vs. Computational Process

Attribute	Chemical Process	Computational Process
Material	Substances: Reactants Products	Information: Input data Output data
Method	Chemical Reaction	Algorithmic Execution
Measurement (Metrics)	Macro metrics: Concentration Surface area Temperature Pressure Catalyst Energy Reaction rate	Complexity: Time Space Energy Effort Sensor

The materials involved in a chemical process are two types of *substances*: the initially involved substances called *reactants* and the substances produced by the chemical process called *products*. In contrast, the material used in a computational process is *information*, including *input data* and *output data*.

The method in a chemical process is a chemical *reaction* that transforms substances (with a chemical change). The method in a computational process is the *execution* of an algorithm, comprised of rules explicitly specifying how to combine well defined *primitive* operations (also called *steps*) into an execution to transform information, i.e., to generate output data from input data.

In studying a chemical process, we are concerned with measurements such as listed in Table 2. These are *macro* metrics, in that they are global properties (often emergent properties) of the system performing the chemical process. A computational process is subtly different. It mainly concerns with *complexity* metrics, which, although being themselves global properties of the process, are expressed *in terms of primitive operations* on a well-defined computational model (a traditional one is the Turing machine model). For example, when we say that a computational process of matrix multiplication has a time complexity of $O(n^3)$, we mean it needs to execute $O(n^3)$ number of primitive operations, e.g., integer additions or integer multiplications. That is, the time complexity of the matrix multiplication computational process is connected to, and in fact expressed in terms of, the time cost of primitive operations.

This nature of combining and composing primitives to form process-level Method and Measurement, is a fundamental, differential characteristic of a computational process. This is how we can tell a computational process from other types of processes, with respect to the attributes of Method and Measurement in Table 2.

To summarize, we now have a three-point checklist for what is and what is not a computational process: *A computational process is one that*

1) *involves informational material*;
2) *utilizes a method of algorithmic execution to transform information*; and
3) *concerns with complexity metrics.*

Any process that does not satisfy these three probably essential criteria is not deemed computational, or at least may not be well suited to be studied as a computational process.

This definition of computational process also leaves room and opportunities for enrichment. For instance, while traditional computer science studies time complexity and space complexity, we may need new complexity metrics for ternary computing. Table 2 shows three such metrics: *energy complexity* (how many Joules), *effort complexity* (how much human effort) and *sensor complexity* (how many sensors) are needed in a computational process.

6.3 Computational Models

When traditional computer science was established, we assumed a simple symbiosis system consisting of one human user and one machine. Human provides the goal, algorithm, and input data, while the machine does mechanic execution of algorithmic steps to produce output data. A classic discussion of this symbiosis is [13].

A traditional computational algorithm consists of the following salient points:

- The algorithm is based on an abstract, universal computational model (computer), such as a Turing machine, its equivalent random access machine (RAM).
- The algorithm is an explicitly specified procedure comprised of well-defined primitive operations of the abstract computational model, such as Turing machine steps or RAM instructions.
- The algorithm is given input data with expected output.
- At time T_0, the computer starts to mechanically execute primitive operations.
- At time T_N, the computer ends execution (i.e., stops, or terminates) and produces the output data. It may happen that the computer never terminates.

Traditional algorithms have the following four important assumptions and features:

- *Centralized Procedure.* A traditional algorithm is centralized, in that it is prescribed by one user and executed by one machine. Later generalizations to various parallel computing models do not fundamentally alter this centralization of control feature, when computability problems are concerned.

- *Clear Start and End.* An algorithm has the notion of *termination* when the algorithmic execution ends, thus has a clear start time and a clear stop time. Due to centralization, these two points of time can be determined by a centralized clock. The input data are given before the algorithm starts, and the output data are available upon termination.
- *Mechanic Steps.* Once an algorithmic execution starts, the computer executes well defined primitive steps internally and mechanically, without the need of human intervention or environmental interaction, until the algorithmic execution terminates. Only a few mechanical primitives are enough. For instance, the RAM model can realize any algorithms with assignment, sequence, conditional jump, and arithmetic/logic operations.
- *Exact Results.* Traditional algorithms are expected to always generate exact output results data. In fact, many algorithmic computational problems can be cast as a *decision problem*, where the answer is either 0 or 1.

Traditional computer science is a foundation for the first three relationships in Table 1. They have served sciences well and will continue to advance and prosper.

With the new concepts of computational lens, computational thinking and ternary computing, traditional algorithmic computer science may need to be revisited and transformed, without altering the basic nature of computational process shown in Table 2. All the four assumptions and features of traditional algorithms may need revisits.

Centralization Needs Reexamination. It is not clear whether we can model all computational processes in the cyber-physical-social universe as centralized ones. Even if we can, it is doubtful that it is the best way to proceed with our inquiry.

For instance, a market in economics seems obviously to be a decentralized system[14], where many autonomous agents act on their own behalf, but may interact directly or indirectly with one another. The World Wide Web has been characterized as a decentralized information system[15]. Evolution of species seems to be decentralized. Many games in game theory act like decentralized processes. Even gene expression in a cell may not be best characterized by a centralized process.

Research is needed to make the concepts of centralization and decentralization more precise. Furthermore, we need to investigate computational models and complexity metrics that capture and characterize the algorithmic nature of decentralized processes. A path forward might be to view a decentralized computational process as a network of autonomous agents, each executing its own algorithm but interacting with one another. Thus, a computational process can be studied as an *algorithm network* or *interaction of algorithms*②.

Of course, this raises many new questions: How can we obtain the properties of an algorithm network from its component algorithms? Can these global properties (e.g., time complexity) be composed from those of its component algorithms? Is an algorithm network itself still an algorithm?

Start/End May Be Uncertain. For a decentralized computational process, it may be difficult to state the clear times of start and end. Even for centralized processes, start and end may be uncertain. This makes it hard to keep the traditional algorithmic notion of termination.

For instance, in a nation's economy, a market equilibrium is often a pattern that is difficult to be perceived to have a clear termination (or end) time in the economic process concerned. It may also happen that the market process will reach an equilibrium without any clear, explicit indication, and the process does not end (terminate) there, but continues proceeding.

Other examples of uncertain start/end abound. What is "termination" in the World Wide Web, in Wikipedia, in a social network, or in Nature's evolution? The newly formed field of evolutionary computing studies many systems that do not have a clear notion of termination, in the sense of traditional algorithmic computer science[16].

Mechanic Steps Need Revisiting. Computational processes in the cyber-physical-social universe may involve more than mechanic steps. We may need to consider human in the loop and a process' interaction with its environment.

For instance, many processes on the World Wide Web involve more than mechanic steps, although these processes are computational, with the three characteristics of Table 2. The forming and growth of a social network depend on many individual human users' participation and interaction, not just the computer program running the social network. A biological process in a cell, such as gene expression, is regulated by environmental conditions and involves complex interactions.

Butler Lampson recently argues that "*embodiment*" will be the next big trend in the use of computers. He defines embodiment as computers (information systems) with direct and rich interaction with the physical world[17]. Computational processes in such embodiment consist of more than mechanic steps of traditional algorithms.

Inexact Results Need Consideration. The results

---

②The term *interaction of algorithms* is due to Wei Zhao of US National Science Foundation.

may be inexact in computational processes in the cyber-physical-social universe. Sometimes, the expected output can not be precisely formulated, because we cannot state the decision problem of a computational process in a mechanic way. In other circumstances, the input data may be noisy and incomplete, or the output results may be in a range or trace that may not be subject to exact definition or characterization.

A case in point is information retrieval or Web search, where humans are needed to judge if and which returned results match a user's expectation, and to eliminate false positives.

Another example is Lampson's embodiment, where direct and rich interaction with the physical world requires effective dealing with uncertainty. Lampson even suggests that we should consider probabilistic distribution as a standard data type in our basic computational model, as a new paradigm to cope with uncertainty[16].

Transforming traditional algorithmic computer science may inspire and enable us to ask new scientific questions, to establish bridges between computer science and other fields, and to strive for new scientific insights. Some examples follow.

We may make the following questions scientifically meaningful: What is Wiki computable? What is Web computable? What is cell computable? Can we enrich the polynomial reducibility and the P vs. NP question to answer questions such as "is evolution 'harder' than learning?" Progresses have been made with the last question[10].

We know from chemistry that some substances do not react. We can of course naively view the capability of reaction as computable, and the incapability of reaction as incomputable. But this simple mapping may not be very useful, as we have not utilized the connection to the combination of primitives. A better way of thinking may be to enrich our computability theory, to prove some impossibility results for chemistry, as we did for the Turing machine halting problem. For example, in the future we may be able to prove computationally, that it is impossible, or at least intractable, for harmful substances to escape into environment in certain chemical reactions[18].

The emerging cross-disciplinary field of bioinformatics is interesting, in that it lies between Phase III (e-science) and Phase IV (computational lens) of Table 1. In particular, genome studies involve information Material, such as the basic symbols of A, C, G, T and DNA sequences. Bioinformatics helps by providing effective algorithms and tools for alignment, sequencing, and annotation of genomes. It is still lacking Phase IV characteristics in Method and Measurement in Table 2, as computational execution, complexity metrics and analysis are not quite there yet. We need Turing machine like computational models for genomes, similar to λ-calculus (e.g., some kind of genome calculus) or a formal language theory, to characterize how a certain genome is generated by a formal grammar.

To study the intractability phenomenon, US NSF founded a Center for Computational Intractability (http://intractability.princeton.edu). The central question is: Are intractable problems on current computational models also intractable in Nature?

Umesh Vazirani recently asked the following question[18]: Can the falsifiability (or refutability) of a scientific theory be related to computational complexity? In other words, is it true that a theory is more falsifiable because we can test it computationally with less complexity?

One type of bridge between computer science and another field is the following fact: some fundamental phenomena or mechanisms exist and are observable in both. An example is the phenomenon of phase transition, which exists in physics as well as computer sciences[1]. On the other hand, the mechanism of catalyst in chemical processes corresponds to the enzyme mechanism in biological processes, but we have not yet an explicit concept corresponding to catalyst or enzyme in computational processes, such that its presence will make a process computable or more efficient.

In the algorithmic game theory field, computer scientists have been able to prove results showing that finding a Nash equilibrium is hard (more precisely, is PPAD-complete), even for two-player games[20], by recasting the problem of finding a Nash equilibrium in a traditional, centralized algorithmic setting.

On the other hands, we have many intuitive examples which seem to indicate that decentralized systems are often efficient. For instance, cycles and crises notwithstanding, markets in a nation's economy seem to work fine most of the time. Many processes on the Internet and the Web are fast. Ant colonies function efficiently.

We need to reconcile such intuitions of efficient decentralized systems with negative results such as finding Nash equilibria is computationally hard in a centralized computing model.

7 Conclusions

We have reviewed the three new concepts of computational lens, computational thinking, and ternary computing, and elaborated them with examples. They all state that computational processes are pervasive and fundamental. Computer science is more than a human created tools discipline that provides hardware, software and services to other disciplines. Computation is inherent in Nature and in the human society. Computation exists not only within, but also before and besides the current computer science and the cyberspace.

These three concepts all provide new perspectives on computational systems thinking. Computational lens highlights the algorithmic way of thinking. Computational thinking stresses the essential role of abstractions. Ternary computing emphasizes the utilization of ternary resources of the cyber world, the physical world, and the human society, in problem solving.

These concepts already have four types of impact in both China and the USA, on influencing new national basic research agenda, forming new cross-disciplinary subfields, inspiring new research topics and results, and establishing new academic institutions[21].

A three-point check list for computational process is proposed in this paper: a computational process is characterized by 1) involving informational material, 2) utilizing a method of algorithmic execution to transform information; and 3) concerning with complexity metrics.

Ternary computing, together with computational lens and computational thinking, calls for researching a new science of computing for the human-cyber-physical ternary universe. Four basic assumptions of the traditional algorithmic computer science need revisiting, namely, centralized procedure, clear start and end, mechanic steps, and precise results.

Acknowledgements The author would like to thank Prof. Richard Karp of University of California, Berkeley, for explaining the computational lens concept. The contents of this paper directly benefits from several fruitful discussions with Prof. Karp, when he came to visit China in 2008 and 2009. The author is indebted to Prof. Jeannette Wing and Prof. Wei Zhao of the US National Science Foundation, and Prof. Marc Snir of the University of Illinois, Urbana-Champaign, for sharing their insights on computational thinking.

References

[1] Karp R. Understanding science through the computational lens. *Journal of Computer Science and Technology*, 2011, 26(4): 569-577.

[2] Li S C, Bu D, Xu J, Li M. Fragment-HMM: A new approach to protein structure prediction. *Protein Science*, 2008, 17(11): 1925-1934.

[3] Wing J. Computational thinking. *Communications of the ACM*, 2006, 49(3): 33-35.

[4] Wing J. Computational thinking and thinking about computing. *Philosophical Transactions of the Royal Society*, 2008, 366(1881): 3717-3725.

[5] Xu Z, Li G. Computing for the masses. *Communications of the ACM*. (To appear)

[6] National Research Council, Committee on Fundamentals of Computer Science Reflections: Challenges and Opportunities. Computer Science: Reflections on the Field, Reflections from the Field, National Academies Press, 2004.

[7] Li G (editor). Information Science and Technology in China: A Roadmap to 2050. Science Press Beijing and Springer-Verlag Berlin, 2010.

[8] Information Science Strategic Study Group of the National Natural Science Foundation of China. Strategic Study Report on the Development of Information Science: 2011-2020, August 2010.

[9] Nisan N, Roughgarden T, Tardos E, Vazirani V V (eds.). Algorithmic Game Theory: Cambridge University Press, 2007.

[10] Valiant L. Evolvability. *Journal of the ACM*, 2009, 56(1): 1-21.

[11] Li D, Zhang H. Cloud computing beyond Turing machine. *Communications of the China Computer Federation*, 2009, 5(12): 8-16.

[12] DARPA. DARPA network challenge final standings. http://en.wikipedia.org/wiki/DARPA_Network_Challenge.

[13] Licklider J. Man-computer symbiosis. *IRE Transaction on Human Factors in Electronics*, 1960, HFE-1 (1): 4-11.

[14] Hurwicz L, Reiter S. Designing Economic Mechanisms. Cambridge University Press, 2008.

[15] Berners-Lee T, Hall W, Hendler J, Shadbolt N, Weitzner D J. Creating a science of the Web. *Science*, 2006, 313(5788): 769-771.

[16] Ma P C H, Chan K C C, Yao X, Chiu D K Y. An evolutionary clustering algorithm for gene expression microarray data analysis. *IEEE Transactions on Evolutionary Computation*, 2006, 10(3): 296-314.

[17] Lampson B. The uses of computers: The best is yet to come. In *Nobel Laureates Beijing Forum*, Beijing, China, November, 2008, Speech.

[18] Panel at the Princeton Workshop on the Computational Worldview and the Sciences, December 11, 2006.

[19] Vazirani U. Computational constraints on scientific theories: Insights from quantum computation. In *the Caltech Workshop on the Computational Worldview and the Sciences*, Pasadena, USA, Mar. 15-16, 2007, Speech.

[20] Papadimitriou C H. The complexity of finding Nash equilibria. Algorithmic Game Theory: Cambridge University Press, 2007, 29-51.

[21] Snir M. Computer & information science & engineering — What's all This?. In *the 2nd NSF-NSFC Sino-USA Computer Science Summit*, Washington, DC, USA, July 2008, Keynote Speech.

Zhi-Wei Xu received his Ph.D. degree from the University of Southern California in 1987. He is currently a professor of the Institute of Computing Technology, Chinese Academy of Sciences. His research areas include high-performance computer architecture and network computing science.

Dan-Dan Tu is a Ph.D. candidate at the Institute of Computing Technology, Chinese Academy of Sciences. Her research interests include computational advertising and personalized recommendation.