The fundamental role of computer science concepts in science

Part 1 outlined how computers will play an increasingly important and eventually ubiquitous role in most branches of science, and how they are changing how scientists work. Altogether more radical, however, is the importance of *computer science*. We believe that computer science is poised to become as fundamental to science, and in particular the natural sciences, as mathematics has become to science, and in particular the physical sciences.

Two important pillars underpin this statement: First, computer science concepts and theorems deal with dynamics in a discrete and reactive sense. Calculus, for example, and its more modern derivatives (excuse the pun) is the main way in which mathematics deals with dynamic issues, but it does so in a continuous fashion, with continuous kinds of cause-and-effect; it deals with rates of increase, with feedback loops, with growth and movement, etc. In contrast, computer science deals predominantly with the interactively discrete, which is really what is meant by the term *reactive*, and it is also able to combine this with the continuous. In fact, computer science is the science *dedicated* to the dynamic. In most kinds of complex systems, biology perhaps being the primary example, the discrete is not only more central but is also much harder to deal with. Indeed, biological systems are the most exciting dynamic systems we will ever know; they are predominantly reactive, and they not only behave but also affect, prescribe, cause, program and blueprint other behaviour. In short, the characteristics of computer science are central to the dynamics of biological systems: concurrency, time dependence, cause-effect phenomenon and distributed control.

Second, computer science is also about algorithms and programs, that is, with *generic prescriptions* for creating dynamics. It not only analyses dynamics and writes equations that capture dynamic phenomena, which is what the dynamic parts of mathematics do well (for the continuous case), but computer science *builds* dynamics. And it is this, perhaps more than anything else, that gives computer science some of its most important and special ways of thinking, its tradition and its nature.

Given that many of the most important and fundamental challenges and opportunities for the 21st Century can be characterised by their complexity and dynamics, then computer science clearly – we claim and make a case for here – will be equally fundamental to addressing them. Part 3 of this report outlines some examples of how.

One of the first glimpses of the potential of computer science concepts and tools, augmented with computing, has already been demonstrated in the Human Genome Project, and by the success of structural biology to routinely decipher the three-dimensional structure of proteins. In this and in related sequencing projects, scientists use computers and computerised DNA sequence databases to share, compare, criticise and correct scientific knowledge, thus converging on a consensus sequence quickly and efficiently [15]. These branches of biology succeeded in unleashing the power of computers to their benefit because both have adopted good mathematical abstractions to describe their research such as: the 'DNA-as-string' abstraction (a mathematical string is a finite sequence of symbols) to describe DNA sequences, and the 'protein-as-three-dimensional-labelled-graph' abstraction, to describe the three-dimensional structure of proteins. Armed with good abstractions, these scientists were able to code their knowledge in a mathematical form that is amenable to processing and sharing via computers. We expect that the rest of biology and other scientific disciplines will also be able to make such big strides, with the aid of computers and computer science concepts and tools, by adopting similarly useful abstractions for more complex systems and processes, as explained in the subsection 'Codification of Biology' in the section 'New Conceptual and Technological Tools' below.

The coding of scientific knowledge will not only empower scientists by allowing them to share, compare, criticise and correct scientific knowledge via computers, it will also enable a change in the way science is done. Coded scientific knowledge can be analysed computationally, before any experimentation. It can be checked, computationally, for consistency among coded theories, and for consistency between theories and accumulated data, akin to computer program debugging [16]. When inconsistency among theories is uncovered, it might be resolved by computer-designed 'crucial experiments' [17-19]. Furthermore, computational analysis of theory versus experimental data may suggest additional experiments to be performed, manually or automatically, as described later in the sections 'Integrating Theory, Experiments and Models', and the section 'New Conceptual and Technological Tools, in the subsections 'Artificial Scientists' and 'Prediction Machines'.

We believe that the concepts and tools developed in computer science over the past 70 years will be useful not only at the *'meta level'*, in helping to manage and develop theory, data and experimentation, but most importantly, also at the *'object level'*, in helping to form scientific theories. For example, computer systems and biomolecular systems both start from a small set of elementary components from which, layer by layer, more complex entities are constructed with ever-more sophisticated functions. Computers are networked to perform larger and larger computations; cells form multi-cellular organisms. All existing computers have an essentially similar core design and basic functions, but address a wide range of tasks. Similarly, all cells have a similar core design, yet can survive in radically different environments or fulfil widely differing functions. Hence we believe the abstractions, tools and methods used to specify and study computer systems should illuminate our accumulated knowledge about biomolecular systems [15].

Several fundamental computer science concepts are already on their way to becoming household names in science, and many more will follow. For example, abstraction is a fundamental tool in computer system design: when designing a complex computer system, identifying the right levels of abstraction within the system is perhaps the single most important design decision. Within a computer system, one can easily find a dozen or so such levels, starting from logic gates, logic circuits, functional units, hardware devices, microinstructions, abstract machine

and the machine language, abstractions for memory and communication, high level language, procedures, data types, algorithms, system design, and system specification. Analogously, identifying levels of organisation in biological systems was fundamental to progress in biology: biomolecules (DNA, RNA, proteins), biomolecular machines (polymerases, ribosome, spliceosome) and functional molecular complexes (membranes and pores), signalling pathways, organelles, cells, organs, organisms, and beyond.

As another example, the concepts developed in algebraic concurrency theory, such as concurrency, indeterminism, communication, synchronisation, processes, channels, and messages, may prove essential for the full understanding and codification of complex inter- and intra-cellular biological processes [20]. As a third example, we expect core computer science concepts on interchangeability of program and data, universal computers, interpreters, compilers, meta-interpreters, partial evaluation, and compositional semantics, to prove essential for the full understanding of the role of DNA as program and data, of the universality of cellular design, and of gene regulation and specialisation. As a fourth example, consider the complexity of each biological unit, and organism as a whole, as encoded in its genome. We expect the notion of descriptive complexity, developed by Kolmogorov [21], to play an essential role in understanding and measuring biological complexity at all levels. As a fifth example, modularity and well-defined interfaces are key attributes of good computer design. They ensure that errors in one component may have a limited effect on other components, and therefore can be tracked and corrected. They also ensure that the design can easily be changed and evolved as requirements change. Similarly, we believe that modularity became a fundamental attribute of the evolvable components of biological systems, as non-modular designs were not able to evolve and survive through changes in external conditions. Uncovering the modularity and interfaces of evolvable biological systems is a major challenge of biology, and a computer science perspective on these issues might be of assistance. In general, such advances in science will rely on the development and application of new conceptual and technological tools, discussed in a later chapter.

Ehud Shapiro, David Harel, Christopher Bishop, Stephen Muggleton