Computer Science: The Study of Procedures

Stuart C. Shapiro
Department of Computer Science and Engineering
University at Buffalo, The State University of New York
Buffalo, NY 14260-2000
shapiro@cse.buffalo.edu

June 18, 2001

Introduction

In this paper, I present an argument that Computer Science is a natural science that studies procedures. This requires a discussion of natural sciences, of procedures, of alternative characterizations of Computer Science, and of the nature of computers.

Natural Sciences

Webster's dictionary defines *natural science* as

> *branches of science (as physics, chemistry, biology) that deal with matter, energy, and their interrelations and transformations or with objectively measurable phenomena*
> [Webster, 1993]

Herb Simon, often cited as saying that computer science is a science of the artificial, also says that "natural science is knowledge about natural objects and phenomena" [Simon, 1996, p. 3]. Artificial science, on the other hand, is "knowledge about artificial objects and phenomena" [Simon, 1996, p. 3], whose characteristics include being "synthesized … by human beings… [and being] characterized in terms of function, goals, adaptation" [Simon, 1996, p. 5]. An artifact, he goes on to say, is an interface between an inner environment, "the substance and organization of the artifact itself" [Simon, 1996, p. 6], and an outer environment "in which it performs" [Simon, 1996, p. 6]. The inner and outer environments are the subject of natural sciences, whereas the artificial sciences are concerned with the fit of the inner to the outer environment in satisfying human goals.

Procedures

Webster's definition of *procedure* is:

> *a particular way of doing or of going about the accomplishment of something*
> [Webster, 1993]

In Computer Science, the term *algorithm* is often used, but algorithms are a subclass of procedures. Webster defines *algorithm* as:

> *a step-by-step procedure for solving a problem or accomplishing an end esp. by a computer*
> [Webster, 1993]

However, at least arguably, parallel procedures are not "step-by-step", yet are included in the province of Computer Science. Moreover, Computer Scientists often require algorithms to be both finite and effective, which eliminates those procedures that never naturally terminate, such as operating systems, and procedures that are not guaranteed to produce the correct answer, such as heuristic procedures. Needless to say, both operating systems and heuristic procedures are included in the province of Computer Science.

I am therefore adopting the word "procedure" as the most general term for the way "of going about the accomplishment of something." A computer program is a description of a procedure in a formal language appropriate for directing a computer to carry out the procedure. Similarly, a musical score is a description of a procedure in a language appropriate for a musician to produce a piece of music, and a recipe is a description of a procedure in a language appropriate for a cook to produce a meal.

Procedures are not natural *objects*, but I claim that they are natural *phenomena* that may be, and are, objectively measured, principally in terms of the amount of time they take (for those procedures that terminate), and in terms of the amount of resources they require. In terms of Simon's three-way distinction between inner environment *vs*. interface *vs*. outer environment, a procedure constitutes an inner environment. The interface arises from the choice of a particular procedure in a particular outer environment to accomplish a particular human goal. If procedures are inner environments---natural phenomena that may be objectively measured---then one may have a natural science of procedures. My position is that computer science is that natural science.

Computer Science as a Natural Science

Computer Science is a natural science, since procedures, although not concrete objects, are phenomena that operate in the natural world which constrains them. For example, rather than just being concerned with the difference between procedures that might never terminate *vs*. procedures that are guaranteed to terminate in a finite amount of time, Computer Science is interested in the difference between infeasible procedures, which take an unreasonable amount of time to carry out, *vs*. feasible procedures, which take a reasonable amount of time to carry out.

Like other natural sciences, Computer Science includes both theoreticians and experimentalists. Theoreticians study the boundaries of possible procedures, and general categories of procedures. Some experimentalists consider classes of behaviors, design procedures that are supposed to perform those behaviors, and experiment with variations of those procedures, testing and comparing the different variations in terms of how much of the class of behaviors each can carry out. Other experimentalists study how different variations of procedures affect the amount of time or other resources the variations require. It often takes many experimentalists a long period of time to find the variation of a procedure that only consumes the amount of resources that the theoreticians have proved is the minimal for a class of tasks.

Some Fundamental Principles of Computer Science

Two fundamental principles of computer science are worth mentioning.

One fundamental principle of Computer Science, called the Church-Turing Thesis, implies that a computer can carry out any procedure that any mechanistic device, manufactured or naturally grown, could carry out. This means that various categories of computers, such as Turing Machines, analog computers, serial digital computers and parallel computers can each do what the others can do. Also, it means that if animals and even humans are mechanistic, then computers can do what they can do.

We can analyze any procedure into its basic actions and a set of control structures, which specify how the basic actions are combined. Another fundamental result of computer science is a theorem proved by Corrado Böhm and Giuseppe Jacopini (Böhm and Jacopini , 1966) that any procedure may be constructed using the three control structures:

1. Sequence: Do one action, and then another, and then another, etc.
2. Selection: Do one action or another, based on some condition.
3. Loop: Repeatedly do an action as long as some condition holds.

This characterizes what is needed for a language to be a general-purpose programming language, or, indeed, a general-purpose procedure language.

The Nature of Computers

Although there have been occasional studies of procedures now recognizable as Computer Science as early as Euclid, Computer Science jelled as a field with the invention of the stored-program electronic computer in the mid-1940's. This is because the computer is a general-purpose procedure-following machine. The procedure may be specified in advance and stored in the computer as a "program" to be carried out at any

later time. The importance of the computer's being electronic is the speed at which it can carry out the procedure. A procedure which might require years for a person to perform could be carried out by an electronic computer in seconds. This makes possible experiments with variations of procedures, as mentioned above. Such experiments could not be conducted if each “run” took years to do. The phrase “general-purpose” is not used lightly.

It is often said that a computer is basically a machine that processes numbers, and that everything is ultimately represented in a computer by the two numbers 0 and 1. This is inaccurate. Basically, a computer is a device consisting of a vast number of connected switches. Each switch can be in any one of a set of positions, a switch remains in its setting until changed by a human operator or by the operation of the device, and a change in any switch setting can change the operation of the device. It is a matter of technological convenience and reliability that the number of possible positions of a given switch is usually made to be just two. These two positions may be thought of as 0 and 1, but may just as well be thought of as “off” and “on,” or as “one position” and “the other position.” The power of the computer comes from the fact that the switch settings both determine the operation of the device and can be changed by the operation of the device. (This is the principle of the equivalence of data and program.) The settings of all the switches at the time the device is started *is* the stored program. Groups of switches can be used to represent numbers, but can as easily be used to represent letters of the alphabet, words, *etc*. A working hypothesis of the subfield of Computer Science called Artificial Intelligence is that groups of these switches can be used to represent any thought any human might conceive of.

Computer Science is often characterized as being concerned with information processing. This is misleading since it conjures up the notion of operating on numbers, textual documents, databases, stored images, or other data stored in the computer, and nothing more. In fact, computers can be equipped with a variety of input devices, including those that “sense” electro-magnetic waves (including, but not limited to light), sound, chemicals (including those people interpret as smells), touch, *etc*. Similarly, computers can be equipped with a large variety of output devices including wheels, manipulators, speakers, *etc*. That is, not only can computers operate on stored data, they can also sense and operate on the world and objects in it. Thus computers can perform behaviors in the world including, according to the Church-Turing Thesis, any behavior any other mechanistic device can perform, presumably even anything humans can do. To repeat, computers are general-purpose procedure-following machines, and Computer Science is the general study of procedures.

Like other natural sciences, Computer Science includes both basic and applied science. Basic science is done to understand the world without regard to possible applications. As discussed above, basic Computer Science is done to understand procedures and their properties. However, since computers are general-purpose procedure followers, and can operate both on stored information and on the world, procedures implemented on computers can perform useful functions for people. Applied computer scientists study useful procedures, and produce, as products of this study, computer programs and computer systems that have extensive effects on the operation of our society.

References

Corrado Böhm and Giuseppe Jacopini, Flow diagrams, Turing machines and languages with only two formation rules, *Communications of the ACM 9*, 5 (May 1966) 366-371.

Herbert A. Simon, *The Sciences of the Artificial*, Cambridge: MIT Press, 1996.

Webster's Third New International® Dictionary, Unabridged, Copyright © 1993 Merriam-Webster, Incorporated. Published under license from Merriam-Webster, Incorporated.