A Rebuttal of Dijkstra's Position on Fairness

K. Mani Chandy
Department of Computer Sciences,
University of Texas at Austin
Austin, Texas 78712
{on leave at the California Institute of Technology}

J. Misra
Department of Computer Sciences,
University of Texas at Austin
Austin, Texas 78712

1988

Finite experiments cannot distinguish between fair and unfair implementations. So fairness has no place (Prof. Dijkstra argues) in program design.

Mathematicians and computer scientists often introduce concepts that cannot be verified by experiments, have no analog in the real world, and are patently unimplementable. Some simple examples are: Memory word-lengths are infinite, all hardware operations are guaranteed to terminate (i.e., there are no glitches) and problem sizes can be arbitrarily large. These assumptions are made not because they are "correct" but because they allow programmers to separate concerns: First focus attention on a problem in which the assumptions hold and later study the implications of the assumption. Imagine the difficulty in developing a theory of computation in which the computer store is limited to, say, a million bytes. Not that memory limitations aren't important, but rather that concerns about memory limits have their proper place in the overall design, and experience suggests that a theory of computation should not be based on the memory limits of target computers. The Turing machine is an effective model precisely because it makes the right kind of simplifying assumption (though no physical machine will ever have the infinite store of a Turing machine).

In analyzing the performance of a computer program our goal is to predict the time it takes to execute on a target computer. We have learned over the last four decades that we should not begin design by being concerned with execution times on specific machines, even if short execution time is a key requirement. We have learned to employ abstractions in design rather than be embroiled in concerns about specific machines. Counts of the number of arithmetic operations is one abstraction. Classifying programs by whether their operation counts are exponential or polynomial in the size of the input is another. Fairness is merely one more step toward increasing abstraction. We choose to evaluate these abstractions in terms of the following criterion: Do they make program design simpler or more complex? Whether the abstraction is realizable in the "real-world" is irrelevant.

Certain classes of programs cannot be analyzed in terms of operation counts. Among these are the so called "asynchronous" programs in which many interacting computations may proceed simultaneously at, possibly, differing speeds. These computations may involve communications over geographically distributed computers, accesses to secondary devices or even interactions with human beings—steps that all take finite time but are extremely slow when compared with speeds of operations on a chip; thus all steps are not of comparable time complexity. In many cases of interest the computation never terminates. Even for a qualitative analysis, more simplifying assumptions are needed. One such assumption is fairness; in rough terms, (one simple version of) fairness says that every action in a program is eventually started.

There is an even simpler assumption than fairness that predicts something about the progress of a computation. This is the "minimal progress assumption," introduced by Prof. Dijkstra, which asserts that some action is executed if there is any action that can be executed. The minimal progress assumption can be used to prove absence of system deadlock, and in some cases, eventual program termination. But it cannot be used to prove absence of individual starvation, e.g., a transmitted message is eventually delivered. One major drawback of minimal progress assumption is that no theory of program construction can be solely based upon it: in most cases, the trivial program "skip" implements any specification.

Is fairness real? This metaphysical question, we believe, is in the same league as "do complex numbers exist," a question that profoundly disturbed mathematicians of the 17th and 18th centuries. Clearly no question about complex numbers could be settled by physical experimentation. Empirical evidence of the usefulness of complex numbers later led mathematicians to accept them, and develop elegant theories about them. We no longer question the rationality of using irrational numbers.

We view fairness as a simplifying assumption. The important question to ask is not if fairness is real, but does it help?