technical correspondence

On Nondeterministic Programs

□ A group at the Naval Research Laboratory has been studying the behavior of nondeterministic programs. We have been guided in this study by Dijkstra's *A Discipline of Programming*[1]. We have found a simple way to describe the possible behavior of such programs and we believe that it will prove enlightening to many *Communications* readers. Our characterization has also revealed two errors in the predicates given by Dijkstra. Owners of Dijkstra's book may wish to note these corrections in their copies.

Consider a mechanism S that can transform a system from an initial state to a final state. Let R be a predicate that is either true or false for each system state. Two questions can be asked of each state:

1. Does activation of S with the system in this state lead to termination?
2. Does the final state satisfy R?

Each of these questions admits three possible answers: "yes," "maybe" (in the nondeterministic case) or "no." For deterministic systems, the possible answers to these questions form three ordered pairs:

(yes, yes)
(yes, no)
(no, —) (the second question is meaningless in this case)

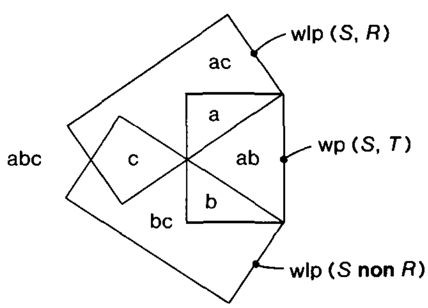
These ordered pairs define a partition of the state space in the deterministic case that corresponds directly with possibilities (a), (b), and (c) listed by Dijkstra at the top of page 21.

The same two questions can be used to partition the state space in the nondeterministic case:

ordered pair		Figure 3.1
(yes, yes)	=	a
(yes, maybe)	=	ab
(yes, no)	=	b
(maybe, yes)	=	ac
(maybe, maybe)	=	abc
(maybe, no)	=	bc
(no, —)	=	c (again, the second question is meaningless)

As shown, this partition corresponds directly to the one given by Dijkstra on pages 21–22 and in Figure 3.1.

Dijkstra's Figure 3.1



In his book, Dijkstra introduces the notation

wp(S, R)

for the "weakest pre-condition" to denote the maximal set of states in which activation of S will lead to termination of the system in a state for which R is true. The predicate that is true for all points in the state space is denoted T; wp(S, T) is thus true for all states that are certain to lead to termination.

For nondeterministic systems, he introduces the "weakest liberal precondition"

wlp(S, R)

to denote the maximal set of states such that activation of S will *not* lead to termination in a state for which R is *false.* That is, for states in wlp(S, R), activation may lead either to termination with R true or to nontermination. The predicate wlp(S, T) is true for all system states, since no state can lead to termination with T not true.

Using this notation, Dijkstra defines formally the seven disjoint regions of the state space listed above. With each predicate, he provides an English description, and his Figure 3.1 illustrates the partition. Table I gives the predicates with the corresponding ordered pairs and regions of that figure. The underlined portions of the predicates for (ac) and (bc) are missing from the text. Note that, since wlp(S, **non** R) does *not* imply **non** wlp(S, R), the final conjuncts are not redundant.

Professor Dijkstra has acknowledged that the expressions in Table I are correct.

Carl E. Landwehr, Code 7593
Naval Research Laboratory
Washington, DC 20375

1. Dijkstra, E. *A Discipline of Programming.* Series in Automatic Computation, Prentice-Hall, Englewood, N.J., © 1976. Figure 3.1 reprinted by permission of Prentice-Hall, Inc.

Table I. Predicates with Corresponding Ordered Pairs and Regions of Figure 3.1

Region	Ordered Pair	Predicate
a	(yes, yes)	wlp(S, R) **and** wp(S, T)
ab	(yes, maybe)	wp(S, T) **and non** wlp(S, R) **and non** wlp(S, **non** R)
b	(yes, no)	wlp(S, **non** R) **and** wp(S, T)
ac	(maybe, yes)	wlp(S, R) **and non** wp(S, T) **and non** wlp (S, **non** R)
abc	(maybe, maybe)	**non** (wlp(S, R) **or** wlp(S, **non** R) **or** wp(S, T)
bc	(maybe, no)	wlp(S, **non** R) **and non** wp(S, T) **and non** wlp(S, R)
c	(no, —)	wlp(S, R) **and** wlp(S, **non** R)