tive compromise between number of required test cases and thoroughness of the test. It is required by important quality standards in aviation, including RTCA/DO-178B, "Software Considerations in Airborne Systems and Equipment Certification," and its European equivalent EUROCAE ED-12B.

Recall the expression `(((a || b) && c) || d) && e`, which required 13 different combinations of condition values for compound condition adequacy. For modified condition/decision adequacy, only 6 combinations are required. Here they have been numbered for easy comparison with the previous table:

	a	b	c	d	e	Decision
(1)	*True*	–	*True*	–	*True*	*True*
(2)	*False*	*True*	*True*	–	*True*	*True*
(3)	*True*	–	*False*	*True*	*True*	*True*
(6)	*True*	–	*True*	–	*False*	*False*
(11)	*True*	–	*False*	*False*	–	*False*
(13)	*False*	*False*	–	*False*	–	*False*

The values underlined in the table independently affect the outcome of the decision. Note that the same test case can cover the values of several basic conditions. For example, test case (1) covers value *True* for the basic conditions a, c and e. Note also that this is not the only possible set of test cases to satisfy the criterion; a different selection of Boolean combinations could be equally effective.

12.5 Path Testing

Decision and condition adequacy criteria force consideration of individual program decisions. Sometimes, though, a fault is revealed only through exercise of some sequence of decisions (i.e., a particular path through the program). It is simple (but impractical, as we will see) to define a coverage criterion based on complete paths rather than individual program decisions

Δ path adequacy criterion

A test suite T for a program P satisfies the path adequacy criterion iff, for each path p of P, there exists at least one test case in T that causes the execution of p.

Δ path coverage

This is equivalent to stating that every path in the control flow graph model of program P is exercised by a test case in T.

The path coverage C_{Path} of T for P is the fraction of paths of program P executed by at least one test case in T.

$$C_{Path} = \frac{\text{number of executed paths}}{\text{number of paths}}$$

Unfortunately, the number of paths in a program with loops is unbounded, so this criterion cannot be satisfied for any but the most trivial programs. For a program with loops, the denominator in the computation of the path coverage becomes infinite, and