NAWCWD TP 8846

Equivalence Class Testing

by
Anthony S. Cantone
Systems Safety Engineering Branch
Systems Engineering Department

APRIL 2020

NAVAL AIR WARFARE CENTER WEAPONS DIVISION
CHINA LAKE, CA 93555-6100

Naval Air Warfare Center Weapons Division

FOREWORD

DO-178B/C guidelines state that an expected activity associated with normal range test cases is that real and integer input variables be exercised using valid equivalence classes and boundary values, and that robustness test case activities include real and integer variables being exercised using equivalence class selection of invalid values. In this report, the meaning of this guideline is explored by a careful development of the concepts of equivalence class along with related concepts such as equivalence relation and the partition of a set.

This report was reviewed for technical accuracy by Glen McCue.

Approved by
K. CHIRKIS, *Head*
Systems Engineering Department
16 April 2020

Under authority of
W. S. DILLON
RDML, U.S. Navy
Commander

Released for publication by
J. L. JOHNSON
Director for Research and Engineering

NAWCWD Technical Publication 8846

Published by .... Technical Communication Office
Collation .... Cover, 7 leaves
First printing .... 5 paper, 7 electronic media

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED (From - To)
16-04-2020	Final	1 September 2019 – 17 March 2020

4. TITLE AND SUBTITLE
Equivalence Class Testing (U)

5a. CONTRACT NUMBER N/A
5b. GRANT NUMBER N/A
5c. PROGRAM ELEMENT NUMBER N/A

6. AUTHOR(S)
Anthony S. Cantone

5d. PROJECT NUMBER N/A
5e. TASK NUMBER N/A
5f. WORK UNIT NUMBER N/A

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Naval Air Warfare Center Weapons Division
1 Administration Circle
China Lake, California 93555-6100

8. PERFORMING ORGANIZATION REPORT NUMBER
NAWCWD TP 8846

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
None

10. SPONSOR/MONITOR'S ACRONYM(S)
N/A

11. SPONSOR/MONITOR'S REPORT NUMBER(S)
N/A

12. DISTRIBUTION/AVAILABILITY STATEMENT
DISTRIBUTION STATEMENT A. Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES
None.

14. ABSTRACT
(U) DO-178B/C guidelines state that an expected activity associated with normal range test cases is that real and integer input variables be exercised using valid equivalence classes and boundary values, and that robustness test case activities include real and integer variables being exercised using equivalence class selection of invalid values. In this report, the meaning of this guideline is explored by a careful development of the concepts of equivalence class along with related concepts such as equivalence relation and the partition of a set. An example of the application of this guideline to white box testing is presented.

15. SUBJECT TERMS
DO-178B, DO-178C, Equivalence Classes, Relations, Software Test

16. SECURITY CLASSIFICATION OF: a. REPORT	b. ABSTRACT	c. THIS PAGE	17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON / 19b. TELEPHONE NUMBER *(include area code)*
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	SAR	12	Anthony Cantone (760) 939-1820

CONTENTS

This page intentionally left blank.

1.0 SCOPE

DO-178B/C states in paragraph 6.4.2.1 (Reference 1) that an activity associated with normal range test cases is that real and integer input variables be exercised using valid equivalence classes and boundary values, and that robustness test case activities include real and integer variables being exercised using equivalence class selection of invalid values. This report explores the definition and construction of these equivalence classes and what it means to choose representatives from equivalence classes for software testing.

2.0 BINARY RELATIONS

The testing process often deals with choosing inputs to functions. These input data sets are sometimes very large. We are told that the input data sets can be divided into subsets, called equivalence classes, from which we can choose “representatives.” What, exactly, is an equivalence class, and what does it mean to choose a representative from an equivalence class?

Equivalence classes are generated by defining binary relations on input data sets. To understand this, we should understand the term “relation, how a relation is defined on a set, and what makes a relation an “equivalence relation.”

One way to represent a **relation** is a **set of ordered pairs**. This is a **binary** relation, a correspondence between the elements of two sets. The two sets could be the same set. Each element of a binary relation is an ordered pair. In the ordered pair (s, t), s is from the first set, and t is from the second set. In this monograph, unless otherwise noted, binary relations defined here will usually be between elements of the same set. An ordered pair (s, t) is said to be in the relation R if the question “Are s and t related according to the relation R?” has an affirmative answer.

Consider non-empty sets S and T, where the elements of S are $\{s_1, s_2, s_3, \ldots\}$ and the elements of T are $\{t_1, t_2, t_3, \ldots\}$. Then a binary relation on the sets S and T (we say on the set SxT), which are all possible ordered pairs with the first element in S and the second element in T, is a subset of ordered pairs from SxT. The set SxT is known as a **Cartesian Product** of sets S and T.

Let us look at a concrete example. Say M is the set of all men, and W is the set of all women. Let us define the relation “likes” on M and W, where we want to determine the truth of “m likes w” for each m in M and each w in W. Say we notice that this relation is true only for (m_1, w_3) and (m_8, w_4), where m_1 and m_8 are two men (i.e., members of M), and w_3 and w_4 are two women (i.e., members of W). We write “likes (MxW)” = $\{(m_1, w_3), (m_8, w_4)\}$. Note that we have to be able to answer the question “Does m_i like w_j?” for each pair (m_i, w_j) in the "Cartesian Product" *M*x*W* in order to determine which ordered pairs in *M*x*W* belong to the relation "likes."

An example of a binary relation on a single set is the following. Suppose *S* is a set of 1,000 line segments in a plane, oriented in a random way with respect to each other (like pick-up sticks). Let us define the binary relation "parallel" on the set *S*. The relation is then a collection of ordered pairs from *S*x*S*. To determine if a pair of lines (s_i, s_j) from *S*x*S* belongs to the relation "parallel," we must be able to answer the question "Is line s_i parallel to line s_j?" in the affirmative.

We may find several lines all parallel to each other. The pairs made of such lines would belong to the relation. There may also be another subset of lines in *S* that are parallel to each other, but no line in that set is parallel to any line in the first set. The pairs made of lines in that set would also belong to the relation and so on. And, finally, there could be a leftover set of lines each of which is parallel to no other lines in *S*.

Note that the original set *S* has been divided into subsets of parallel lines, where no line in any subset is parallel to any line in another subset. There are also the leftover subsets, the subsets of lines parallel to no other line in *S*. This is actually a collection of subsets of one line each. This is the case because since no two distinct lines are parallel in each of the leftover subsets, no pair made of two distinct lines is in the relation. If s_p is an element in one of the leftover subsets of *S*, then the only pair found in the relation "parallel" would be (s_p, s_p). So each line *x* in one of the leftover sets can be considered by itself a subset of *S* associated with the ordered pair (*x*, *x*). We can intuitively confirm two facts:

1. All of these subsets of *S* **COVER** the set *S* (that is, every line in *S* is a member of one of the subsets we previously described), and

2. All of these subsets are **MUTUALLY DISJOINT** (there is no line in *S* that belongs to two or more of the subsets we previously described).

Such an arrangement of subsets of *S* is called a **PARTITION** of *S* (Figure 1). (Notice the distinction: on the one hand, members of the relation, which are ordered pairs from *S*x*S*; on the other hand, members of *S*, on which are induced this collection of subsets called a partition.) Each member of this collection of subsets of *S* is called an **EQUIVALENCE CLASS**, for reasons that will be discussed later.

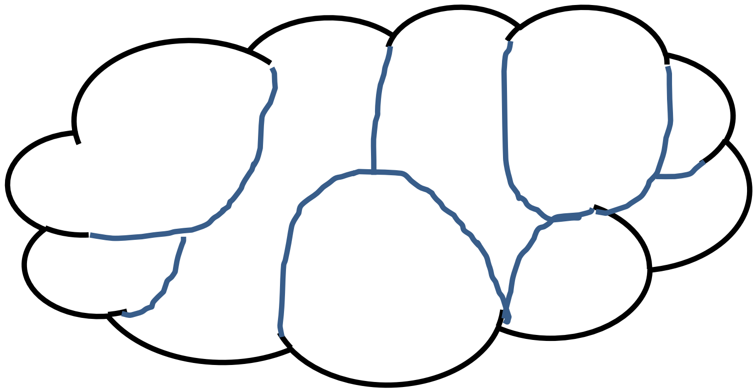

FIGURE 1. Partition of a Set.

Why is this important? Recall that DO-178B/C states in paragraph 6.4.2.1 (Reference 1) that an activity associated with normal range test cases is that real and integer input variables be exercised using valid equivalence classes and boundary values, and that robustness test case activities include real and integer variables being exercised using equivalence class selection of invalid values. “Equivalence Class” is defined in DO-178B/C as follows: “The partition of the input domain of a program such that a test of a representative value of the class is equivalent to a test of other values of the class.” Exactly what does this mean? This monograph will answer this question.

3.0 EXAMPLES OF BINARY RELATIONS

A trivial example: define the following binary relation on the natural numbers N = {1, 2, 3, …}: two natural numbers are related if their difference is a multiple of 2 (i.e., the difference is 0 mod 2). Then we can see that N is divided into two subsets by this relation: a subset that contains the even numbers, and a subset that contains the odd numbers. These two subsets cover N and are disjoint. Therefore they form a partition of N.

What are other examples of binary relations? We are all familiar with the following relations: =, <, >, parallel, perpendicular. The first three relations can be defined on sets of real numbers while the fourth and fifth may be defined on line segments.

4.0 PROPERTIES OF BINARY RELATIONS

Do all binary relations on a set generate a partition on that set? The answer is no. However, there is a way where we can look at some properties of binary relations to determine which relations do generate partitions on sets. This would be useful for the application required in DO-178B/C—we know that if we choose an input value from the set of input values, we will know exactly what the relation on that set of input values is, whether it defines a partition (of subsets) on the set of input values, and whether the value that we choose belongs to more than one subset defined on the set of input values.

Binary relations have many possible properties, for example, reflexive, irreflexive, symmetric, antisymmetric, transitive. However, the properties of interest in defining equivalence classes are the following.

4.1 REFLEXIVE PROPERTY

If S is a non-empty set and R is a binary relation defined on S, then R is said to be **REFLEXIVE** if for *all* elements a in S, aRa. The notation aRa, means the same as (a, a) is in R.) That is, R is reflexive if every element of S is related to itself through R. For example, “=” is reflexive, since for all a in S, a=a. An example of a reflexive relation defined on the natural numbers is the one previously given: two natural numbers a and b are related if $a=b$ mod 2. Since 2 also divides 0, $a=a$ mod 2 must also be true, so R is reflexive.

An example of a binary relation that is not reflexive on the integers is “$<$”, since $a<a$ is not true.

4.2 SYMMETRIC PROPERTY

If S is a non-empty set and R is a binary relation defined on S, then R is said to be **SYMMETRIC** if for all elements a and b in S, aRb implies bRa. Perpendicular and parallel are easily seen to be symmetric relations on the set of line segments in a plane. “=” is a symmetric relation on the set of real numbers.

An example of a binary relation that is not symmetric on the integers is “$<$”, since $a<b$ does not imply that $b<a$.

4.3 TRANSITIVE PROPERTY

If S is a non-empty set and R is a binary relation defined on S, then R is said to be **TRANSITIVE** if for all elements a, b, and c in S, aRb and bRc imply that aRc. “$<$” (less than) is a transitive relation when defined on the integers, as is “$>$” (greater than), “=”, and parallel defined on line segments in the plane.

An example of a binary relation that is not transitive on lines in a plane is "perpendicular." Line *a* perpendicular to line *b*, and line *b* perpendicular to line *c* does not imply line *a* perpendicular to line *c*.

5.0 INDEPENDENCE

It is important to show that the three properties of binary relations presented in Section 4, Properties of Binary Relations, are independent: that is, there are relations that exhibit any two of these properties but do not exhibit the third property.

5.1 REFLEXIVE AND SYMMETRIC BUT NOT TRANSITIVE

Let *S* be the set of real numbers. Define two real numbers *a* and *b* related by *R* if abs($a – b$) ≤ 2.

R is reflexive since abs($a – a$) ≤ 2. *R* is reflexive because 0 is less than or equal to 2.

R is symmetric since abs($a – b$) ≤ 2 ➔abs($b – a$) ≤ 2. *R* is symmetric because both abs($a – b$) and abs($b – a$) have the same values, so both are ≤ 2.

But we can find three real numbers 3, 5, 7 so that abs(3 – 5) ≤ 2 and abs(5 – 7) ≤ 2 does not imply that abs(3 – 7) ≤ 2. Therefore *R* is not transitive.

That is, *R* is a relation that is reflexive and symmetric, but not transitive.

5.2 REFLEXIVE AND TRANSITIVE BUT NOT SYMMETRIC

Let *N* be the set of natural numbers (does not include 0). Let *R* be a binary relation defined on *N* such that if *a* and *b* are natural numbers, they are related by *R* if *a* divides *b*. Since *a* divides *a*, *R* is reflexive.

Given the three natural numbers *a*, *b*, and *c*, if *a* divides *b* and *b* divides *c*, then $a=pb$ and $b=qc$ for some non-zero integers *p* and *q*, and therefore $a=pqc$, i.e., *a* divides *c*. Since *a* divides *b*, and *b* divides *c*, *R* is transitive.

But if *a* and *b* are two different natural numbers such that *aRb* implies *bRa* (that is, *R* is symmetric), or equivalently $a=pb$ and $b=qa$, for some non-zero integers *p* and *q*, then this implies $a=pqa$, and pq=1. This is not possible unless *p* and *q* are each 1, or $a=b$, which is a contradiction of the assumption that *a* and *b* are different. This means if *a* is related to *b* by *R*, then *b* is not related to *a* by *R*, and therefore *R* is not symmetric.

5.3 SYMMETRIC AND TRANSITIVE BUT NOT REFLEXIVE

Given a non-empty set, if one defines a relation *R* on *S* such there are no two members of *S* that are in *R*, then *R* is symmetric (if *aRb*, then *bRa*, but there are no *a* and *b* that satisfy *aRb*, so this is true *vacuously*) and *R* is transitive (similar logic) but to infer that if *aR*b then *bRa*, and then if *aRb* and *bRa*, then *aRa* is nonsense if both "ifs" are false.

That is, for an empty relation, symmetry and transitivity are true but reflexivity is not implied, hence false.

Also, if either the symmetric or transitive property is defined only on some of the elements of *S*, then reflexivity cannot be demonstrated on all the elements of *S*, and so *R* is not reflexive. In general, the following theorem gives the necessary and sufficient condition for a symmetric and transitive binary relation to be reflexive.

Given the set *S*, and the symmetric and transitive binary relation *R* defined on *S*. If, for **every** element *a* in *S* there exists an element *b* in *S* such that *a*R*b*, then by symmetry *b*R*a*, and by transitivity *a*R*a*, and *R* is reflexive.

The key word in the above theorem is "every."

6.0 EQUIVALENCE RELATIONS, EQUIVALENCE CLASSES, AND PARTITIONS

Suppose *R* is a binary relation defined on a non-empty set *S*, and that the relation *R* is reflexive, symmetric, and transitive. Then *R* is known as an **equivalence relation**. An equivalence relation has the interesting property that it induces a **partition** on the set on which it is defined—that is, it divides *S* into subsets, all of which cover *S* and none of which overlap any other (Reference 2). Each of the members of this partition is called an **equivalence class**, as we saw in Section 2. There are many properties of the equivalence classes of a partition, the study of which is beyond the scope of this monograph. Since each element in an equivalence class is, in the sense of the relation *R*, equivalent to any other element of the equivalence class, any element of the equivalence class can be selected as a **representative** of the equivalence class.

In a previous paragraph, we looked at an example involving a set of random line segments in a plane. The binary relation "parallel" was defined on a set of 1,000 randomly oriented line segments in the plane. If we wanted to study the angles that the lines make with each other, we could do this by imposing the parallel relation on this set of line segments. Since "parallel" is easily shown to be reflexive (the line *a* is parallel to itself), symmetric (if line *a* is parallel to line *b*, then line *b* is parallel to line *a*) and transitive (if line *a* is parallel to line *b*, and line *b* is parallel to line *c*, then we can conclude that line *a* is parallel to line *c*), "parallel" is an equivalence relation, and the 1,000 line segments are partitioned. Say there are seven subsets in the partition. Then we can choose a representative line from each subset of the partition (i.e., from each equivalence class) and study the angles that just those seven lines make with each other. Because "parallel" is an equivalence relation, we would know that there are no other lines in *S* that are not parallel to one of the seven lines we chose.

7.0 APPLICATION TO TESTING

Now let us return to the definition of "equivalence class" as defined in DO-178B/C: "The partition of the input domain of a program such that a test of a representative value of the class is equivalent to a test of other values of the class."

By "input domain of a program" this definition must mean the set of all possible values of the input parameters. Moreover, the word "partition" must have the same meaning as in this monograph—a collection of subsets of all the possible values of the input parameters such that the collection covers all the possible values and there are no inputs that belong to more than one of these subsets (i.e., the members of this collection of subsets are mutually disjoint). The definition in DO-178B/C is not quite exact in the sense that each member of the partition should be called an "equivalence class." A more accurate definition would be "One of the members of a partition of the input domain of a program such that a test of the program using any input data element in the member is equivalent to a test of the program using any other input data element in that member." The "member" is of course just a subset of the input domain of the program.

It is known (Reference 2) that given any partition of a set, a suitably defined equivalence relation must exist that is capable of inducing this partition. The DO-178B/C definition of equivalence class implies that there exists a partition of the input data set generated by some equivalence relation. The result of defining this relation on the input domain of a program should permit one to test the program for all the values in any one of the induced equivalence classes by testing a representative value from that equivalence class.

How could this relation be defined? An intuitive guess would be that since the DO-178B/C definition implies there is no difference using one or another input value from an equivalence class, then a good candidate for such a relation (call it *R*) may be as follows: "Two distinct data elements from the input domain are related if each of them drives the program along the same code path."

Is this relation *R* an equivalence relation? If *a* is an input data element that drives the program along a code path *S*, then *aRa,* i.e., *R* is reflexive. If *aRb*, then clearly *bRa,* and *R* is symmetric. If *a, b,* and *c* are distinct input data elements where *aR*b and *bRc*, then *a* drives the program along the same code path as *b*, and *b* drives the program along the same code path as *c*, hence it follows that *a* and *c* drive the program along the same code path, that is, *aRc*. Therefore *R* is transitive, and hence an equivalence relation.

Is the partition induced by *R* on the input domain of a program in fact the partition that is described in the DO-178B/C definition of equivalence class? We can verify this if we understand what this definition means by "…equivalent to a test of other values…". We can interpret this to mean that it would make no difference to the results of the test if any input data element belonging to a particular equivalence class were used instead of any other. We would expect this if any input of a particular equivalence class drives the code execution along the same path as any other input. But this is exactly the definition of the relation that we chose. So we can conclude that the partition of the input domain is exactly the partition generated by the relation *R* previously defined.

8.0 CONCLUSION

This report has explored the meaning of DO-178B/C paragraph 6.4.2.1, the definition and construction of equivalence classes, and what it means to choose representatives from equivalence classes for software testing. It was noted that the definition of equivalence class in DO-178B/C is not quite accurate.

An equivalence class is a member of a partition generated by an equivalence relation. An equivalence relation is a binary relation (i.e., a set of ordered pairs) that exhibits the independent properties "reflexive," "symmetric," and "transitive." If a relation on the input data set can be found that is demonstrated to be an equivalence relation, then a partition of the input data set is generated. Then a test can be formulated where one input (a representative) from each of the partitions is all that is necessary for the test.

A relation that will satisfy this requirement is that two input data elements are related if each of them drives the program along the same code path. It was left to the reader to show that this relation is in fact an equivalence relation, and that therefore a partition (a collection of subsets of the input data set that covers the input data set and is mutually disjoint) is generated from which representative input data elements, one from each of the members of the partition, is all that is necessary to select.

It was also noted for completeness that given a non-empty set, if there is a partition defined on that set, then there exists an equivalence relation defined on that set that will generate that partition. Ideally, if that set is in fact the set of input data to a program, then a candidate for the relation is that two input data elements are related if each of them drives the program along the same code path.

REFERENCES

1. RTCA, Inc. *Software Considerations in Airborne Systems and Equipment Certification*. Washington, D.C., 5 January 2012. (DO-178C.)

2. Pinter, Charles C. *A Book of Abstract Algebra*. 9^{th} Edition. Dover Publications, 2010.

This page intentionally left blank.

INITIAL DISTRIBUTION

1 Defense Technical Information Center, Fort Belvoir, VA

ON-SITE DISTRIBUTION

2 Code D5J4000 (archive copies)
1 Code DC12100 (file copy)
2 Code D510000, Chirkis, K.
4 Code D511000
  Cantone, A. (2)
  Merrill, R. (2)
2 Code D575700, McCue, G.