Chapter 18
T-Way Testing Strategies:
Issues, Challenges, and Practices

Kamal Z. Zamli
Universiti Malaysia Pahang, Malaysia

AbdulRahman A. Alsewari
Universiti Malaysia Pahang, Malaysia

Mohammed I Younis
University of Baghdad, Iraq

ABSTRACT

In line with the advancement of hardware technology and increasing consumer demands for new functionalities and innovations, software applications grew tremendously in term of size over the last decade. This sudden increase in size has a profound impact as far as testing is concerned. Here, more and more unwanted interactions among software systems components, hardware, and operating system are to be expected, rendering increased possibility of faults. To address this issue, many useful interaction-based testing techniques (termed t-way strategies) have been developed in the literature. As an effort to promote awareness and encourage its usage, this chapter surveys the current state-of-the-art and reviews the state-of-practices in the field. In particular, unlike earlier work, this chapter also highlights the different possible adoptions of t-way strategies including uniform interaction, variable strength interaction, and input-output-based relation, that is, to help test engineers make informed decision on the actual use of t-way strategies.

1. INTRODUCTION

As an activity to ensure conformance and quality, software testing is an important phase in the software development lifecycle. Although desirable, exhaustive testing is practically impossible owing to the resource and timing constraints. As a result, many sampling strategies have been developed including that of equivalence partitioning, boundary value, cause and effect graphing as well as decision table mapping. While these traditional static and dynamic sampling strategies are useful for fault detection and prevention, they are not sufficiently effective to tackle bugs due

DOI: 10.4018/978-1-4666-6026-7.ch018

to interaction. Addressing this issue, many t-way strategies (whereby t indicates the interaction strength) have been developed in the literature in the last 15 years. Here, t-way strategies help to search as well as to generate a set of test cases to complete the test suite that covers the required interaction strength at least once from a typically large space of possible test values.

Despite offering significant advantages in terms of test suite reduction (as well as testing costs), t-way testing strategies are not widely adopted in industry (Czerwonka, 2006). Thus, it is important to promote awareness as well as practical use of t-way strategies in order to encourage their applications as well as improve their acceptance within the mainstreams software testing community.

The remaining of this chapter is organized as follows. Section II highlights the fundamentals of t-way testing strategies. Section III surveys the existing t-way strategies. Section IV elaborates on the state-of-practice. Finally, Section V states our conclusion.

2. T-WAY TESTING FUNDAMENTALS

In general, t-way testing can be abstracted to a covering array. Throughout out this chapter, the symbols: p, v, and t are used to refer to number of parameters (or factor), values (or levels) and interaction strength for the covering array respectively. Earlier works suggested two definitions for describing the covering array (Cohen, 2004). The first definition is based on whether or not the numbers of values for each parameter are equal. If the number of values is equal (i.e. uniformly distributed), then the test suite is called Coverage Array (CA). Now, if the number of values in non-uniform, then the test suite is called Mixed Coverage Array (MCA). Unlike earlier work in the literature, this chapter unifies the CA and MCA notations in order to include that of variable strength covering array (VCA), and input output based covering array (IOR).

Normally, the CA takes parameters of N, t, p, and v respectively (i.e. CA(N,t, v^p)). For example, CA (9, 2, 2^4) represents a test suite consisting of 9x4 arrays (i.e. the rows represent the size of test cases (N), and the column represents the parameter (p)). Here, the test suite also covers pairwise interaction for a system with 4 2 valued parameter (see Figure 1). When the CA is the most optimal result, the covering array can be rewritten as CAN (N,t, v^p).

Alternatively, MCA takes parameters of N, t, and Configuration (C). In this case, C captures the parameters and values of each configuration in the following format: v_1^{p1} v_2^{p2}, ….. v_n^{pn} indicating that there are p1 parameters with v1 values, p2 parameters with v2 values, and so on. For example, MCA (8,3,$2^3 1^1$) indicates the test size of 8 which covers 3-way interaction. Here, the configuration takes 4 parameters: 3 2 valued parameter and 1 1 valued parameter (see Figure 2).

In the case of VCA, the parameter consists of N, t, C, and Set (S) (i.e. VCA (N,t,C,S)). Similar to MCA, N,t, and C carry the same meaning. Set S consists of a multi-set of disjoint covering array with strength larger t. For instance (see Figure 3), VCA (12, 2, $3^2 2^2$, (Zamli, et al., 2011)) indicates the test size of 12 for pairwise interaction (with 2 3 valued parameter and 2 2 valued parameter) and 3-way interaction (with 1 3 valued parameter and 2 2 valued parameter).

Building from CA, MCA, and VCA notation, we can express Input Output based relation (IOR) as IOR (N, R,C). Here, N and C take the same meaning given earlier whilst R represents a multi set of parameter relationship definition contributing towards the outputs. For example, for a 4 parameters system with 2 values, and each parameter will be assigned a number 0, 1, 2 and 3 respectively (see Figure 4). Assume two input-output relationships are involved in the outputs (i.e. the first, the second, and the third