A Pairwise Test Suite Generator Based on Melody Search Algorithm

Toh S. Yuen, Ala’a A. Al-Omoush, Foo W. Wen, Goh G. Hau,
Abdul Rahman and A. Alsewari
Department of Software Engineering, Faculty of Computer Systems and
Software Engineering, Universiti Malaysia Pahang, Gambang, Malaysia

Abstract: Melody Search Algorithm (MS) has been adopted to solve many optimization problems such as
combinatorial testing. This paper aims to introduce MS as a pairwise testing strategy called a Pairwise Test suite
generator based Melody Search Algorithm (PTMS). A pairwise testing is an operative approach in the
combinatorial test suite construction. It will minimize the constructed test suite size to save the testing time with
effective defects being detected. The proposed strategy generates the test suite with real-valued variables. A
comparative evaluation revealed that the PTMS is efficient in constructing a minimum test suite with the
existing strategies.

Key words: Pairwise testing, Melody Search (MS) algorithm, test data generation, combinatorial testing,
variables

INTRODUCTION

In this information technology era, there is a huge
influence of high technology and artificial intelligence
when creating new software products. This new method
provides an effective way of bringing high-quality
software products to the end user. Many fields also rely
on this method, especially in the Research and
Development (R and D) area. Evidently, many manual
processes are being performed by certain software
products or artificial intelligence. Basically, every
product created is operated by the combination of
hardware and software to implement each feature
(Perroun et al., 2012). There is a close relationship
between hardware and software; both play important roles
to avoid failure.

Failure of products will occur when a human action
produces some error or bug in the software and this will
lead to the defects which will cause a malfunction when
executed. This problem can cause serious damage to
system function, loss of time especially for a critical
system and higher cost of maintenance. Therefore,
software testing takes first priority in any Software
Development Life Cycle (SDLC) in order to ensure high
quality software and to minimize the occurrence of failure
of the software.

Software testing is defined as the process of
executing a program on finding possible errors and
validating the software or system against its specification
(Myers et al., 1979). It is known from the studies of seven
principles that exhaustive testing is impossible to execute
even the test cases for a real software product (Wang et al.,
2013). A complete testing is impossible because there are
many possible combinations of inputs and pre-condition
test case for software.

Pairwise testing is an effective combinatorial method
of software testing used to minimize the number of test
case that is needed for input parameters to a system and
the interactions between two input parameters values
(McCaffrey, 2010). This strategy generates test cases that
cover all the possible input combinations in order to
include the test data and to reduce the possibilities of
faults due to interaction (Perrouin et al., 2012). There are
many pairwise testing strategies that are available such as
Simulated Annealing (SA) (Cohen et al., 2007), Automatic
Efficient Test Generator (AETG) (Cohen et al., 1997),
Genetic Algorithm (GA) (Flores and Cheon, 2011;
McCaffrey, 2010) Ant Colony Algorithm (ACA)
(Shiba et al., 2004), Bat Pairwise Test Strategy (BPTS)
(Alsaraier et al., 2015), Harmony Search Strategy (PHSS)
(Alsewari and Zamli, 2012), In-Parameter-Order (IPO)
(Lei et al., 2007, 2008) and Interaction Residual Pair Set
(IRPS) (Youns et al., 2008, 2009). Although, the listed
strategies are useful, none of them can give assurance in
producing optimum results for every case study. It
should be noted that some of the existing pairwise
testing strategies are based on optimization algorithms
such as GA, HSS, ACA, BPTS and SA. Therefore,