UAS Self-Adaptive Behavior Testing Automation

To automate the generation and execution of test cases for the UAS application software, a prototype tool named Unmanned-Aircraft-System-Self-Adaptive-Behavior-Testing-Tool (UAS-SBTT) is developed. The UAS-SBTT takes the modeled class diagram, state machine, adaptation rules, UAS variables, and the simulator parameter setting as input. The UAS-SBTT outputs a set of executed test cases and their results. The test cases and their results show the identified faults and whether the SUT executes the required self-adaptive behavior or not during mission execution.

The architecture of the UAS-SBTT is shown in the Tool Architecture figure. The UAS-SBTT consists of two main parts, i.e., Test Case Generation and Test Case Execution. There are a total of ten components, where four components are third-party components and are grey colored, and six components are developed as part of the proposed UAS application software testing approach and are red colored. The artifact that represents the input or outputs of a component is blue. Following, the details of the third-party components are presented briefly, whereas the developed components as part of the UAS-SBTT are discussed in detail.

Test Case Generation

This section presents a description of the steps the developed UAS-SBTT takes for the test case generation.
• Test Case Generator: The purpose of this component is to generate test cases for the SUT using GA. The component takes as input the UAS variables count and their ranges. As an output, the component gives an optimal set of test cases, that maximize the number of faults detected.

• Test Case Generation: The purpose of this component is the transformation of the GA individual into UAS test cases. It takes as input a GA individual and gives as output a test case. This component decodes the values of variables in a GA individual in a form that is understandable by the UAS simulator.

Test Case Execution

This section presents a description of the steps that the UAS-SBTT takes for the test case execution. One of the major components responsible for the test case execution is the Test Driver. A test driver takes as input the generated test cases and executes them.

• Test Case Executor: This is a sub-component of the Test Driver. The purpose of this component is to execute the generated test cases. For this, the component first configures the UAS and then takes as input a test case and executes it on the UAS simulator. As an output, the component gives run-time UAS status. Run-time UAS Behavior Observer: This is a sub-component of the Test Driver. The purpose of this component is to observe the run-time status of the UAS (through the MAVLINK communication protocol[1]) after a UAV self-adaptive behavior scenario is executed in the simulator. As an output, the component gives the UAS run-time status.

• Test Case Evaluator: The purpose of this component is to evaluate the test cases to check if the UAS correctly adapts its behavior or not. This component takes the run-time UAS status as input for evaluation. The test case evaluator makes use of a third-party tool, i.e., OCLSolver [2]. The OCLSolver takes as input the UAS class diagram (UAS-CD (S)), and the adaptation rules (UAS-AR)). It evaluates the UAS-AR on

[1] https://mavlink.io/en/

[2] Shaukat Ali, Muhammad Zohaib Iqbal, Andrea Arcuri, and Lionel Briand. A search-based ocl constraint solver for model-based test data generation. In 2011 11th International Conference on Quality Software, pages 41–50. IEEE, 2011

the UAS-CD (S) instances created using the run-time UAS status and gives as output the constraints evaluation result.

• Test Case Distance Calculator: The purpose of this component is to calculate the fitness value of the test case (GA individual). This component takes as input the constraint evaluation result and computes the fitness.

• Test Case Result File Generator: The purpose of this component is to record the test data from the test case, the constraints evaluated, and the test case evaluation result. The component gives as output a .txt file of evaluation results.