CLAIMS

1. A method for detecting and localizing a fault in a system under test (SUT), the method comprising:

modeling inputs to the SUT as a collection of attribute-value pairs;

assigning a probability value to each attribute-value pair from the collection of attribute-value pairs;

reducing a Cartesian product space comprising all possible combinations of the attribute-value pairs to a reduced test space, at least in part by excluding one or more combinations of attribute-value pairs based on corresponding probability values;

generating an initial set of test vectors that provides complete n-wise coverage of the reduced test space;

generating an initial set of test cases from the initial set of test vectors;

executing the initial set of test cases to obtain a first set of execution results;

determining, based at least in part on the first set of execution results, that one or more test cases failed execution;

generating a set of new test cases from a selected failing test case;

executing the set of new test cases to obtain a second set of execution results; and

detecting and localizing the fault based at least in part on the second set of execution results.

2. The method of claim 1, wherein the Cartesian product space is further reduced based on an architectural restriction of the SUT.

3. The method of claim 2, wherein the architectural restriction comprises at least one of: a requirement that a first attribute cannot take on one or more candidate attribute values if a second attribute has a particular attribute value, a requirement that the first attribute must take on a particular candidate attribute value if the second attribute has the particular attribute value, or a requirement to introduce a third attribute if the second attribute has the particular attribute value.

4. The method of claim 1, wherein an attribute-value pair and a corresponding probability value are represented in a hash.

5. The method of claim 1, wherein a first attribute-value pair that is to be tested is assigned a first probability value that is higher than a second probability value of a second attribute-value pair.

6. The method of claim 1, wherein detecting and localizing the fault based at least in part on the second set of execution results comprises determining a particular combination of attribute values that causes the fault.

7. The method of claim 6, further comprising generating a regression bucket of failing test cases based at least in part on the particular combination of attribute values that causes the fault.

8. A system comprising:

a memory; and

a processor coupled with the memory, the processor configured to perform a method for detecting and localizing a fault in a system under test (SUT), the method comprising:

modeling inputs to the SUT as a collection of attribute-value pairs;

assigning a probability value to each attribute-value pair from the collection of attribute-value pairs;

reducing a Cartesian product space comprising all possible combinations of the attribute-value pairs to a reduced test space, at least in part by excluding one or more combinations of attribute-value pairs based on corresponding probability values;

generating an initial set of test vectors that provides complete n-wise coverage of the reduced test space;

generating an initial set of test cases from the initial set of test vectors;

executing the initial set of test cases to obtain a first set of execution results;

determining, based at least in part on the first set of execution results, that one or more test cases failed execution;

generating a set of new test cases from a selected failing test case;

executing the set of new test cases to obtain a second set of execution results; and

detecting and localizing the fault based at least in part on the second set of execution results.

9. The system of claim 8, wherein the Cartesian product space is further reduced based on an architectural restriction of the SUT.

10. The system of claim 9, wherein the architectural restriction comprises at least one of: a requirement that a first attribute cannot take on one or more candidate attribute values if a second attribute has a particular attribute value, a requirement that the first attribute must take on a particular candidate attribute value if the second attribute has the particular attribute value, or a requirement to introduce a third attribute if the second attribute has the particular attribute value.

11. The system of claim 8, wherein an attribute-value pair and a corresponding probability value are represented in a hash.

12. The system of claim 8, wherein a first attribute-value pair that is to be tested is assigned a first probability value that is higher than a second probability value of a second attribute-value pair.

13. The system of claim 8, wherein detecting and localizing the fault based at least in part on the second set of execution results comprises determining a particular combination of attribute values that causes the fault.

14. The system of claim 13, wherein the method further comprises generating a regression bucket of failing test cases based at least in part on the particular combination of attribute values that causes the fault.

15. A computer program product comprising a non-transitory computer-readable storage media having computer-executable instructions stored thereupon, which when executed by a processor cause the processor to perform a method for detecting and localizing a fault in the SUT, the method comprising:

modeling inputs to the SUT as a collection of attribute-value pairs;

assigning a probability value to each attribute-value pair from the collection of attribute-value pairs;

reducing a Cartesian product space comprising all possible combinations of the attribute-value pairs to a reduced test space, at least in part by excluding one or more combinations of attribute-value pairs based on corresponding probability values;

generating an initial set of test vectors that provides complete n-wise coverage of the reduced test space;

generating an initial set of test cases from the initial set of test vectors;

executing the initial set of test cases to obtain a first set of execution results;

determining, based at least in part on the first set of execution results, that one or more test cases failed execution;

generating a set of new test cases from a selected failing test case;

executing the set of new test cases to obtain a second set of execution results; and

detecting and localizing the fault based at least in part on the second set of execution results.

16. The computer program product of claim 15, wherein the Cartesian product space is further reduced based on an architectural restriction of the SUT.

17. The computer program product of claim 16, wherein the architectural restriction comprises at least one of: a requirement that a first attribute cannot take on one or more candidate attribute values if a second attribute has a particular attribute value, a requirement that the first attribute must take on a particular

candidate attribute value if the second attribute has the particular attribute value, or a requirement to introduce a third attribute if the second attribute has the particular attribute value.

18. The computer program product of claim 15, wherein an attribute-value pair and a corresponding probability value are represented in a hash.

19. The computer program product of claim 15, wherein a first attribute-value pair that is to be tested is assigned a first probability value that is higher than a second probability value of a second attribute-value pair.

20. The computer program product of claim 15, wherein detecting and localizing the fault based at least in part on the second set of execution results comprises determining a particular combination of attribute values that causes the fault.

21. The computer program product of claim 20, wherein the method further comprises generating a regression bucket of failing test cases based at least in part on the particular combination of attribute values that causes the fault.

22. A method for detecting and localizing a fault in a system under test (SUT), the method comprising:

modeling inputs to the SUT as a collection of attribute-value pairs;

assigning a probability value to each attribute-value pair from the collection of attribute-value pairs;

reducing a Cartesian product space comprising all possible combinations of the attribute-value pairs to a reduced test space, at least in part by excluding one or more combinations of attribute-value pairs based on corresponding probability values;

generating an initial set of test vectors that provides complete n-wise coverage of the reduced test space;

generating an initial set of test cases from the initial set of test vectors;

executing the initial set of test cases to obtain a first set of execution results;

determining, based at least in part on the first set of execution results, that one or more test cases failed execution;

generating a set of new test cases from a selected failing test case, wherein generating the set of new test cases comprises generating, in relation to each attribute in the selected failing test case, a respective subset of new test cases at least in part by changing a respective attribute value for the attribute in the selected failing test case to each other candidate attribute value for the attribute that is not present in any of the one or more test cases that failed execution;

executing the set of new test cases to obtain a second set of execution results; and

detecting and localizing the fault based at least in part on the second set of execution results.

23. The method of claim 22, wherein an attribute-value pair and a corresponding probability value are represented in a hash value.

24. A method for detecting and localizing a fault in a system under test (SUT), the method comprising:

modeling inputs to the SUT as a collection of attribute-value pairs;

reducing the test space comprising all possible combinations of the attribute-value pairs to a reduced test space at least in part by excluding one or more combinations of attribute-value pairs based on corresponding probability values;

generating an initial set of test vectors that provides complete n-wise coverage of the reduced test space associated with the collection of attribute-value pairs;

generating an initial set of test cases from the initial set of test vectors;

executing the initial set of test cases to obtain a first set of execution results;

determining, based at least in part on the first set of execution results, that one or more test cases failed execution;

generating a set of new test cases from a selected failing test case, wherein generating the set of new test cases comprises changing a respective attribute value in the selected failing test case to generate each new test case;

executing the set of new test cases to obtain a second set of execution results; and

detecting and localizing the fault based at least in part on the second set of execution results.

25. The method of claim 24, wherein an attribute-value pair and a corresponding probability value are represented in a hash.