Controllable Signature Check Pointing Scheme
For Transient Error Detection

B. Ramamurthy
Department of Computer Science
State University of New York at Buffalo
Buffalo, New York 14260

S. J. Upadhyaya
Department of Electrical and Computer Engineering
State University of New York at Buffalo
Buffalo, New York 14260

Abstract

Almost all the existing signature monitoring schemes use low level language programs to decide the signature check points. We are proposing a new controllable and user transparent signature monitoring scheme that is directly applicable to a High Level Language (HLL) program.

1 Introduction

A program can fail during execution, due to software, hardware or extraneous faults. There are three types of errors caused by the extraneous faults: transient, intermittent and permanent. This paper deals with transient faults. The effect of transient faults during a program execution are often unpredictable and temporary in nature. Due to these characteristics, traditional approaches to error detection are not suitable to detect transient errors. We present a flexible and controllable transient error detection scheme called Controllable Signature Check Pointing Scheme.

Most of the existing schemes are based on the two techniques introduced by Namjoo [3], namely, the Branch Verification (BV) and Path Signature Analysis (PSA). Structural Integrity Checking (SIC) by Lu [1] is of special significance to our work since it is the only work that directly addresses the issues involving HLL constructs. In the new scheme explained below, the control flow complexity factor based on McCabe's [2] cyclomatic complexity of the program control graphs is used to determine the bounds of an interval for check pointing.

2 The New Scheme

The new signature monitoring differs from the existing schemes in three major aspects: (1) the intervals in the program to be monitored are decided at the higher level of abstraction (HLL level), (2) the interval size is not arbitrarily chosen but by applying the control flow and bulk complexity measures, and (3) the interval size could be controlled to suit the application.

The following terms are defined first.

Definition 1: A nested structure is said to be linear if the cyclomatic number of the structure is equal to the deepest nesting level of that structure, assuming the starting level of the structure to be 1.

Definition 2: If the cyclomatic number of a nested structure is greater than the deepest level of nesting in that structure, then the structure is said to be non-linear.

This kind of classification of structures helps in determining with certainty the maximum number of signatures at a check point. The case of the cyclomatic number of a structure being less than the deepest level of nesting is nonexistent.

Definition 3: Interval Control-Flow Complexity Factor (INCCF) is the upper bound of the control flow complexity of a linear structure.

Any non-linear structure could be broken down into linear structures for the application of INCCF. Error latency increases and the coverage decreases as the INCCF value is raised.

2.1 Description of the Scheme

The most important part of the new scheme, that also distinguishes it from the existing schemes, is the process of partitioning the HLL program into intervals. This process is not done in a separate pass through the program but it is integrated into the regular syntax analysis phase of the compiler. The algorithm for signature check pointing is stated in the form of four rules.

Algorithm 1:

1. Consider the first complete structure of the given program: A complete structure is one that starts and ends at level = 1, the outermost level of the program.
2. If it is linear and its control flow complexity $\leq$ INCCF then this structure forms a single interval. There is a signature check point at the end of this interval.

3. If it is linear but its control flow complexity is $>$ INCCF and is equal to say X, then the innermost (X - INCCF) levels are abstracted into a single node. The outer levels now form a structure of complexity equal to INCCF. Add a signature check point at the start of the abstracted node and one at the end of the entire structure. Rules 2 and 3 are applied recursively to the abstracted inner structures.

4. If the structure is nonlinear: Consider each path in the level = Current Level + 1; Apply Rules 2 and 3 to every linear structure at this level. If there is a nonlinear structure apply Rule 4 recursively until there are no more structures left (until the end of the original non-linear structure). Add a signature check point at the end of the non-linear structure.

Example

A program control graph with several complete structures is shown in Fig.1. The location of signature check points for NCCF=3 are shown by asterisks in the graph.

3 Conclusion

In the new scheme, the signature intervals are decided at HLL level. The location of the signature check points and the frequency of occurrence of the check points could be controlled by varying the INCCF. Thus the error coverage and latency, memory overhead and performance degradation could also be tuned to suit the application. Future refinements and extensions include signature check pointing of sequential code and reducing signature embedding overhead.

References

