Summary: Linearizability is the de facto correctness criterion for concurrent data type implementations. Violation of linearizability is witnessed by an error trace in which the outputs of individual operations do not match those of a sequential execution of the same operations. Extensive work has been done in discovering linearizability violations, but little work has been done in trying to provide useful hints to the programmer when a violation is discovered by a tester tool. In this paper, we propose an approach that identifies the root causes of linearizability errors in the form of code blocks whose atomicity is required to restore linearizability. The key insight of this paper is that the problem can be reduced to a simpler algorithmic problem of identifying minimal root causes of conflict serializability violation in an error trace combined with a heuristic for identifying which of these are more likely to be the true root cause of non-linearizability. We propose theoretical results outlining this reduction, and an algorithm to solve the simpler problem. We have implemented our approach and carried out several experiments on realistic concurrent data types demonstrating its efficiency.

For the entire collection see [Zbl 1453.68017].

MSC:
- 68N19 Other programming paradigms (object-oriented, sequential, concurrent, automatic, etc.)
- 68P05 Data structures
- 68Q60 Specification and verification (program logics, model checking, etc.)

Keywords:
- concurrency

Full Text: DOI