ABSTRACT

We build a semantically based assertion language and proof system for partial correctness and deadlock of CSP programs. The assertions are tree-structured and have two types of leaf nodes so that we can distinguish between termination and deadlock. We define syntactic sequential and parallel composition of assertions. The proof system is syntax-directed, and does not require the use of auxiliary variables. The proof rule for parallel composition does not require constraints such as interference-freedom or a cooperation test. It is possible to reason about correctness and deadlock-freedom together, so that we do not require separate proofs of freedom from deadlock and partial correctness.