Implementations of Sized Types for Parallel Complexity of Message-passing Processes

No Author Given

No Institute Given

Abstract. Type systems using sized types have been studied extensively in the context of complexity analysis of functional and parallel programs to formally express, verify and infer complexity bounds on programs. Recent contributions have extended this study to message-passing processes using behavioral types to bound channel synchronizations, providing a sound framework for parallel complexity of pi-calculus processes. We explore the challenges of implementing this work, and present a type checker and a type inference algorithm. Our type checker can verify complexity analyses of some polynomial- and linear time primitive recursive functions, encoded as replicated channel inputs (servers), by using integer programming to bound channel synchronizations. Comparison of parametric complexities is a partial order, so to bound parallel complexities, we introduce the notion of combined complexity: A set of intersecting parametric complexity bounds. Our type inference algorithm first generates a constraint satisfaction problem on sized input/output types that we reduce to a set of polynomial inequality constraints, a solution to which provides a parametric bound on the parallel complexity of a server. We show how our constraint satisfaction problems can be over-approximated and provide a Haskell implementation using the Z3 SMT solver that can provide reasonable bounds on the parallel complexity of some linear time servers.