A Comparison of Software Frameworks for Parallelization of Large-Scale ZI Trader Models

The zero-intelligence trader model is a well-known agent-based representation of supply and demand relations for the study of the emergence of market clearing. While conventional implementations of it employ a variety of agent activation schemes, essentially all are single-threaded. We have parallelized this model using a variety of languages and technologies, including POSIX threads in C, C++11 threads, OpenMP running under C, Clojure, Erlang, Go, Haskell, Java, Python, and Scala. For each, speed-up as a function of the number of threads is studied on a high-performance computing platform. Most of these software systems demonstrate significant performance gains over single-threaded implementations and are capable of taking advantage of a large number of hardware cores. The advantages and disadvantages of each approach are described and compared. Some of these languages are largely functional, some feature persistent data, and some have relatively poor absolute performance. We discuss the utility of them for agent-based models generally, including ease of programming and likely performance improvements on multi-core hardware.