Geophysical Research Abstracts
Vol. 18, EGU2016-7447, 2016
EGU General Assembly 2016


# A Parallel Vector Machine for the PM Programming Language

Tim Bellerby

University of Hull, Geography Environment and Earth Sciences, Geography Environment and Earth Sciences, Hull, United Kingdom (t.j.bellerby@hull.ac.uk)

PM is a new programming language which aims to make the writing of computational geoscience models on parallel hardware accessible to scientists who are not themselves expert parallel programmers. It is based around the concept of communicating operators: language constructs that enable variables local to a single invocation of a parallelised loop to be viewed as if they were arrays spanning the entire loop domain. This mechanism enables different loop invocations (which may or may not be executing on different processors) to exchange information in a manner that extends the successful Communicating Sequential Processes idiom from single messages to collective communication. Communicating operators avoid the additional synchronisation mechanisms, such as atomic variables, required when programming using the Partitioned Global Address Space (PGAS) paradigm. Using a single loop invocation as the fundamental unit of concurrency enables PM to uniformly represent different levels of parallelism from vector operations through shared memory systems to distributed grids.

This paper describes an implementation of PM based on a vectorised virtual machine. On a single processor node, concurrent operations are implemented using masked vector operations. Virtual machine instructions operate on vectors of values and may be unmasked, masked using a Boolean field, or masked using an array of active vector cell locations. Conditional structures (such as if-then-else or while statement implementations) calculate and apply masks to the operations they control. A shift in mask representation from Boolean to location-list occurs when active locations become sufficiently sparse. Parallel loops unfold data structures (or vectors of data structures for nested loops) into vectors of values that may additionally be distributed over multiple computational nodes and then split into micro-threads compatible with the size of the local cache. Inter-node communication is accomplished using standard OpenMP and MPI.

Performance analyses of the PM vector machine, demonstrating its scaling properties with respect to domain size and the number of processor nodes will be presented for a range of hardware configurations.

The PM software and language definition are being made available under unrestrictive MIT and Creative Commons Attribution licenses respectively: www.pm-lang.org.