PGAS parallel languages

Introduction

PGAS or Partitioned Global Address Space is a parallel programming model based on the following concepts:

- multiple execution contexts with separate address spaces, thereby allowing any given execution instance to exploit memory affinity properties of the underlying architecture for good application performance
- access of memory locations on one execution instance by other execution instances

The following programming language extensions implement this concept:

- Coarray Fortran
- Unified parallel C
- Chapel, the Cascade HPCS language under development by Cray
- Titanium, an explicitly parallel dialect of Java
- X10, an experimental HPCS language under development at IBM
- Implicit parallelism is also targeted for Fortress, originally developed under the guidance of Sun Microsystems.

The target is to achieve the following advantages over other parallel paradigms:

- Integration of the type system. This greatly reduces the development and debugging effort for object based parallel codes compared to e.g., the MPI library approach.
- One-sided communication semantics. An efficient implementation can fully exploit the hardware properties (latency and bandwidth) of the underlying system interconnect. Overlap of computation and communication is fully under the programmer's control.
- Optionally, control over memory affinity improves large scale scalability compared e.g., to OpenMP shared data.

This document provides information about PGAS facilities provided on LRZ HPC systems. At present, these are only partial, and not fully optimized implementations which can be used to jumpstart code development and perform parallel algorithm research.

Coarray Fortran

Please consult the Coarray subdocument for the relevant information.

Unified parallel C

Please consult the UPC subdocument for the relevant information.

Chapel

Please consult the Chapel subdocument for the relevant information.