Prof. Jingke Li (lij@pdx.edu); Classes/Labs: TR 1400-1550 @FAB 150, 88-10; Office Hours: TR 1300-1400 @FAB 120-06

tLab 6: Data Parallel Programming

In this lab, you are to practice data parallel programming with Fortran and Chapel.

The `lab6` directory contains a set of Fortran and Chapel programs; most have appeared in last week's lecture. (For simplicity, we use `.f90` suffix for all Fortran programs, even though some actually contain Fortran 95 code.)

Read, compile, and run these programs, to get a feel of these two new languages. The Fortran compiler on the Linux system is called `gfortran`. It can compile any version of program from Fortran 77 to Fortran 2003. If you forget how to compile and run Chapel programs, revisit Lab 1 handout.

1 Array Operations

Consider an array `A` with `N` elements, where `N` is an even number. Here is an example in Fortran:

```
integer:: A(8) = (/ (i,i=1,8) /)   ! A = (1,2,3,4,5,6,7,8)
```

Use array operations to define the following arrays based on array `A`.

1. Arrays `Odd` and `Even`. They hold the odd-indexed and even-indexed elements of `A`, respectively. For the above example, we'll have

   `Odd = (1,3,5,7), Even = (2,4,6,8).`

2. Arrays `Front` and `Back`. They hold the front-half and back-half of the elements of `A`, respectively. For the above example, we'll have

   `Front = (1,2,3,4), Back = (5,6,7,8).`

3. An array `Reverse` that holds `A`'s elements in reverse order. For the above example, we'll have

   `Reverse = (8,7,6,5,4,3,2,1).`

4. An array `Shuffle` that holds the perfect shuffle of `A`'s elements, i.e., alternating elements from `Front` and `Back`. For the above example, we'll have

   `Shuffle = (1,5,2,6,3,7,4,8).`

Implement these arrays in both Fortran 90 and Chapel. Call the programs `arrays1.f90` and `arrays2.chpl`, respectively. Make the array size a parameter `n`. Verify your programs by compiling and running them.

2 Matrix Multiplication

The file `mm.c` contains a matrix multiplication program in C. Convert this program into two versions of Chapel program:

1. A sequential version, `mm1.chpl`.
2. A data parallel version, `mm2.chpl`. Parallelize all the loops in `mm.c`, with `forall`, array operations, and/or reduction operations.

In both programs, represent the parameter `N` by a configurable runtime constant, i.e. `config const`; set its default value to 8. Test your programs with different values of `N`.

Submission

As usual, write a short report summarizing your work with this lab. Submit it with your programs through the "Lab 6" folder on Canvas. The submission deadline is the end of tomorrow (Friday).