Demonstration of Common Runtime System

November 16, 1996

The common runtime system is supposed to create a public domain infrastructure for multiple high-performance languages. We define common code and data descriptors to support data parallelism in a set of High Performance Languages (HPLs). Above this common runtime system, we should provide a specific language interface for each HPL.

Here we have implemented a Fortran language interface for an HPF compiler in NPAC at Syracuse. Our HPF compiler will translate HPF applications into F77 codes using this interface on top of the common runtime systems support.

1 HPF compiler support examples

Four "template" programs illustrate features the HPF compiler must deal with. Each template is parameterized by a handful of constants specifying, for example, data sizes and distribution formats. The supplied program generators dynamically select a set of parameters, within constraints designed to create a program that is

A: legal, and

B: illustrates a particular class of communication requirements.

The Templates

- *Level 1 template.* Deals with the FORALL statement. Illustrates how we deal with one-dimensional array calculations in FORALL statements.
- *Level 2 templates.* Adds a subroutine call. In the subroutine, we deal with the INHERIT property of HPF procedure arguments.
- *Level 3 templates.* Expands the level 2 templates, to deal with calculation of multi-dimension arrays.
- *Level 4 templates.* Deals with the array assignment and where contructs and statements of HPF.

The Demos

For each template a particular set of constraints on the required communications can be selected. Activating the associated link causes a new program to be generated satisfying these constraints. By clicking on further links you have the opportunity to translate the new HPF program to an F77 SPMD program, then to view the translated code. *The translated code makes calls to the Fortran interface of the common run-time.* Finally you can compile the translated code, and run it on the SP2 at NPAC. (You must specify the correct number of processors on which to run the generated HPF code.)

2 Using the common run-time directly from applications

Two of the kernel programs for the *HPF Applications* suite have been translated to C++ and make direct use of a C++ applications-level interface to the common run-time.

The Applications

- *2-D Potts Simulation.* A simulation of the Q-State 2-D Potts model, using the classic Metropolis heatbath algorithm.
- The codes contained here perform a two-dimensional FFT over an example matrix—in this case a photograph of a wolf. The program performs a transform, deletes a specified number of Fourier modes, then reverses the transform.

The Demos

For each application you can view the original HPF source and the C++ "port", containing direct calls to the common run-time. Then you can the compile and run C++ codes. You must specify a sufficient number of processors (4 or more) on which to run the programs. In the FFT case, you must specify the number of modes to delete (maximum 256).