Interfaces of Parallel Programming Language XcalableMP to Parallel Numerical Libraries

Takenori Shimosaka

Programming environment research team
RIKEN AICS

AICS-Café March. 23 2012
Contents

- XcalableMP and Parallel numerical libraries
- A necessity of XcalableMP interface to parallel numerical libraries
- A design of the XcalableMP interface
A Location of XcalableMP in Programming environment

Parallel programming language XcalableMP is directive-based language extensions for C and Fortran90
(A XcalableMP processor of our team is a translator from XcalableMP programming model to
MPI programming model.)

<e.g. Programming environment>

Programming language : C/C++, Fortran, Java, COBOL, Perl, Lisp , XcalableMP, …

Compiler : gcc, gfortran, …

Debugger : gdb, …

Editor : Notepad, vi, emacs

Performance analysis tool

Library: BLAS, LAPACK, ScaLAPACK,…

…..
XcalableMP

PGAS (Parallel programming language for distributed memory systems)

Features:

- Directive-based language extensions for C and Fortran90
- Data distribution and communication are specified by directives.
  
  → A programmer has a Global view on distributed data, then he doesn’t need to handle directly distributed data using local coordinates.
  → Good readability.
  
  Design changes and program modifications are easy.

- It is possible to mix OpenMP and MPI programming into XMP.
  
  → We can easily use the existing programs.
MPI programming model

Global data is distributed by a user!

The user has to calculate by local coordinate, considering global coordinate.

Global coordinate

0 10 20 30

Global array

changed by a user

Local coordinate

node 0

0

node 1

0(10)

node 2

0(20)
If user decides distribution method and timing by XMP directives, global data is distributed by XMP compiler!

Only global coordinate!!

unchanged

Global array

node 0

node 1

node 2
High performance parallel numerical libraries

ScaLAPACK : Scalable Linear Algebra PACKage(http://www.netlib.org/scalapack/)

FFTW : Fast Fourier Transform Library(http://www.fftw.org/)

ParMETIS : Parallel Graph Partitioning and Fill-reducing Matrix Ordering
(http://glaros.dtc.umn.edu/gkhome/metis/parmetis/overview)

MUMPS : a MUltifrontal Massively Parallel sparse direct Solver(http://graal.ens-lyon.fr/MUMPS/)

Lis : a Library of Iterative Solvers for linear systems(http://www.ssisc.org/lis/)

......

......

Most of high performance parallel numerical libraries are implemented by MPI programming model.
A necessity of XcalableMP interface to parallel numerical libraries

Issue: Re-writing every application programs using only XMP is unrealistic.

⇒ In order to develop application programs easily, it is important to make used of existing high performance libraries.

Research objective:

To design a way to make use of MPI parallel libraries in XMP program.
Contents

- XcalableMP and Parallel numerical libraries
- A necessity of XcalableMP interface to parallel numerical libraries
- A design of the XcalableMP interface
Usage of parallel numerical libraries

<Distribution pattern 1>
Usage of parallel numerical libraries

<Distribution pattern 2>

Global data is distributed by a user!
The user has to calculate by local coordinate...
considering global coordinate...
and input descriptor information...

changed by a user

Parallel numerical library

node 0
node 1
node 2
Usage of parallel numerical libraries for XMP

- Global coordinate
- Local coordinate
- Parallel numerical library

Only global coordinate!!

 unchanged

changed by
XMP( interface)

node 0
node 1
node 2
Design of the interface

1. Parallel numerical library routines can be invoked as an XMP procedure through the XMP interface procedure.

2. When the parallel numerical library routine needs information on global distributed arrays, the interface extracts it from the descriptor using some inquiry routines provided by XMP runtime library and passes it to the numerical library routine as arguments.

3. The interface does not affect the behavior of numerical library routines except for restrictions concerning the XMP specification.

XcalableMP Program

```c
#pragma xmp...
...
int main(...){
    ...
    ixmp_xxx(...);
}
```

XcalableMP interface

```c
void ixmp_xxx(...){
    ...
    xmp_array_xxx(...)
    ...
    sub(...);
}
```

Numerical library routine

```c
void sub(...){
    ...
}
```

<Inquiry routines>
```
xmp_array_xxx(xmp_desc_t d, ..)
xmp_align_xxx(xmp_desc_t d, ..)
xmp_dist_xxx(xmp_desc_t d, ..)
```
Sample program for ScaLAPACK (MPI programming model)

Program code

```
bblas_pinfo_(&mypnum,&nprocs);
bblas_get_(&icontxt,&what,&contxt);
bblas_gridinit_(&contxt,order,&nprow,&npcol);
bblas_gridinfo_(&contxt,&nprow,&npcol,&myrow,&mycol);

... ...

for (i=0;i<nrow/nprow;i++){
    for (j=0;j<ncol/npcol;j++){
        a[i][j]=a0[myrow+i*nprow][mycol+j*npcol];  // Calculate by local coordinate!
    }
}

... ...

desca[0]= array information;
....
descb[8]=array information;
pdgesv_(&n,&nrhs,a,&ia,&ja,desca,ipiv,b,
    &ib,&jb,descb,&info);

... ...

bblas_gridexit_(&contxt);
bblas_exit_(&doneflag);
```

Flow of execution

- Initialization of communication process of a numerical library
- Matrix distribution
- Executing solver
- Finalization of communication process of a numerical library
Sample program for ScaLAPACK (XcalableMP programming model)

Program code

```c
#pragma xmp nodes p(nprow,npcol)

#pragma xmp align a[i][j] with t(i,j)
#pragma xmp align ipiv[i][j] with t1(i,j)
#pragma xmp align b[i] with t(*,i)

blacs_pinfo_(&mypnum,&nprocs);
blacs_get(&icontxt,&what,&contxt);
blacs_gridinit(&contxt,order,&nprow,&npcol);
blacs_gridinfo(&contxt,&nprow,&npcol,&myrow,&mycol);

... desca[0]= array information;
... descb[8]= array information;
ixmp_pdgesv(&n,&nrhs,a,&ia,&ja,xmp_desc_of(a),ipiv,b,&ib,&jb,xmp_desc_of(b),&ictxt,&info);

... blacs_gridexit(&contxt);
blacs_exit_(&doneflag);
```

Flow of execution

Definition of data distribution

Initialization of communication process of a numerical library

Matrix distribution (Execute by directives)

Executing solver

Finalization of communication process of a numerical library

Calculate by global coordinate!
Reduce distribution programming cost by directives!

Improve productivity!
Descriptor of XcalableMP

Descriptor : A key to retrieve information of specified global array, template, or node array.

xmp_desc_t : The type which of the descriptor is implementation-dependent.

xmp_desc_of : Built-in operator which is evaluated to a descriptor.
   xmp_desc_of has a operand which is the name of either a global array,
   a template or a node array.
   The resulting descriptor can be used as an input argument of the inquiry procedures.

<Inquiry routines(XcalableMP C)>
xmp_array_xxx(xmp_desc_t d, ..)
xmp_align_xxx(xmp_desc_t d, ..)
xmp_dist_xxx(xmp_desc_t d, ..)
xmp_template_xxx(xmp_desc_t d, ..)
xmp_nodes_xxx(xmp_desc_t d, ..)
## Experiment

Table 1 Experimental environment

<table>
<thead>
<tr>
<th>PC Cluster (node environment)</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td>Intel Xeon X5670(2.93GHz 6core) *2</td>
</tr>
<tr>
<td>Memory</td>
<td>24GB</td>
</tr>
<tr>
<td>Network</td>
<td>1Gbps Ethernet*2 + Infiniband 4x QDR</td>
</tr>
<tr>
<td>OS</td>
<td>Linux 2.6.18</td>
</tr>
<tr>
<td>MPI</td>
<td>Intel MPI Library 4.0 Update2</td>
</tr>
<tr>
<td>C Compiler</td>
<td>gcc4.1.2</td>
</tr>
<tr>
<td>XMP</td>
<td>C language version</td>
</tr>
<tr>
<td>ScaLAPACK</td>
<td>Compile netlib original programs by using “gfortran –O3”</td>
</tr>
<tr>
<td>BLACS</td>
<td>Compile netlib original programs by using “gcc –O3”</td>
</tr>
<tr>
<td>BLAS</td>
<td>Compile netlib original programs by using “gfortran –O3”</td>
</tr>
</tbody>
</table>
## Experiment

### Table 2: Performance results of the sample program

<table>
<thead>
<tr>
<th></th>
<th>MPI program</th>
<th>XMP program</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data distribution</td>
<td>0.0052sec</td>
<td>0.0055sec</td>
</tr>
<tr>
<td>Solver(PDGESV)</td>
<td>1.02sec</td>
<td>1.02sec</td>
</tr>
</tbody>
</table>

The rank of coefficient matrix: 2000  
Data distribution method: Cyclic  
Execution nodes: 1  
Execution processes: 4

⇒ The performance of XMP program is comparable with that of MPI one.
Summary and Future work

- We show that programming productivity is improved by using the XMP programming model and the interface of XMP.
- Currently, we have researched just two parallel numerical libraries. For the other parallel numerical libraries, we need to research the interface of XMP.
Thank you very much for kind attention.