XcalableACC: Extension of XcalableMP using OpenACC for Accelerator Clusters

Motivation

In order to develop an application on accelerator clusters, a programmer selects a combination of MPI and one of the available accelerator programming models, such as OpenACC, OpenCL or CUDA. In particular, OpenACC provides good productivity and high performance. However, MPI programming is often difficult because programmers need to distribute, transfer, and reduce data by using primitive MPI functions.

XcalableACC

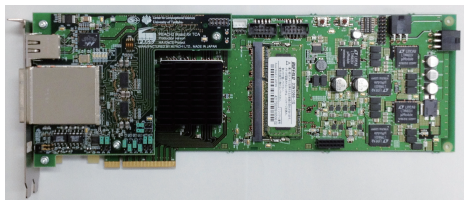
Based on studies of XMP and OpenACC, we believe that a directive-based language is effective for also accelerator clusters. Thus, we propose a new programming model called ***XcalableACC (XACC)***, which is an XMP extension that uses OpenACC. In XACC, programmers can use both XMP and OpenACC directives seamlessly. Therefore, XACC allows programmers to easily develop applications on accelerator clusters. Additionally, XACC also provides directives to transfer data among accelerators efficiently. **"Example 1"** on the right side shows an example of XACC.

Extension of OpenACC for multiple accelerators

In the OpenACC specification 2.0a, it is difficult for a single process to use multiple accelerators. For example, programmers must specify an accelerator number using the **acc_set_device_num()** before each statement, calculate indexes for loop statement on each GPU, and move data among accelerators manually. To reduce the programming cost, we consider an extension of OpenACC to use multiple accelerators. We believe that this extension is also useful for XACC. **"Example 2"** on the right side shows an example of the extension of OpenACC.

Definition and synchronization of a halo region

To develop stencil applications easily, XACC provides shadow and reflect directives. These directives can be used to define a halo region for a distributed array and synchronize it among nodes. In order to transfer data among accelerator memories efficiently, the Omni compiler supports ***Tightly Coupled Accelerators(TCA)*** in addition to the combination of CUDA and MPI, and GPUDirect RDMA over InfiniBand. TCA is a communication architecture based on PCI Express (PCIe) technology. Computation hosts with TCA network are connected with each other using PCIe external cables, which permits direct, low latency data transfers among accelerator memories because they do not need host memory copies, a protocol conversion, and MPI software stack. For more details of TCA, please go to **"Center for Computational Sciences, University of Tsukuba" (#3215 booth).**



Example 1: Loop parallelization

```
double a[N];
#pragma xmp nodes p(*)
#pragma xmp template(0:N-1)
#pragma xmp distribute t(block) onto p
#pragma xmp align a[i] with t(i)

#pragma acc data copy(a)
{
#pragma xmp loop on t(i)
#pragma acc parallel loop
  for(int i=0;i<N;i++){
    a[i] = .. ;
```

Define XMP distributed array

Transfer XMP distributed array to accelerator

OpenACC parallel loop directive parallelizes a loop statement parallelized by XMP loop directive

Example 2: Usage of multiple accelerators

```
#pragma acc device d(*)
float a[N], b[N];
#pragma acc declare device_resident(a,b) ¥
          layout([block]) shadow([1:1]) ¥
                                on_device(d)
..
#pragma acc reflect (b)
#pragma acc parallel loop layout(a[i]) ¥
                                on_device(d)
for (int i=1; i<N-1; i++){
  a[i] = b[i-1] + b[i+1];
  ..
```

Define multiple accelerators

Declare distributed arrays with a halo

Synchronize a halo region

Parallelize a loop statement

Evaluation of the HIMENO benchmark

The HIMENO Benchmark is a stencil application benchmark that evaluates the performance of incompressible fluid analysis code.

```
float p[MIMAX][MJMAX][MKMAX];
#pragma xmp shadow p[1:1][1:1][0]
..
#pragma xmp reflect (p) acc
..
#pragma xmp loop (k,j,i) on t(k,j,i)
#pragma acc parallel loop collapse(3) ..
for(i=1 ; i<MIMAX ; ++i)
  for(j=1 ; j<MJMAX ; ++j){
    for(k=1 ; k<MKMAX ; ++k){
      S0 = p[i+1][j][k] * ..;
```

Define a halo region

Synchronize a halo region

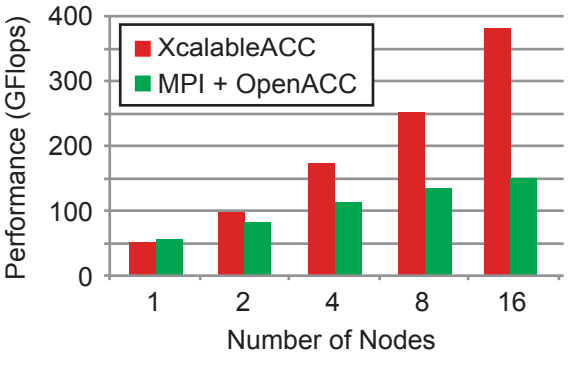
Parallelize a loop statement

Productivity

	Total	Directive		Except for directives
		XcalableMP	OpenACC	
XcalableACC	204	22	9	173
MPI + OpenACC	325	-	15	310

The source lines of the XACC HIMENO Benchmark is less than that of the MPI+OpenACC HIMENO Benchmark. Especially, to develop the XACC HIMENO benchmark, a programmer only adds XMP and OpenACC directives into the sequential Himeno Benchmark.

Performance



The performance of XACC using TCA is up to 2.7 times better than that of MPI + Open ACC using MVAPICH2-GDR 2.0b on HA-PACS/TCA system (NVIDIA K20X, InfiniBand QDR x2 rails).