Center for Computational Sciences, University of Tsukuba
www.ccs.tsukuba.ac.jp

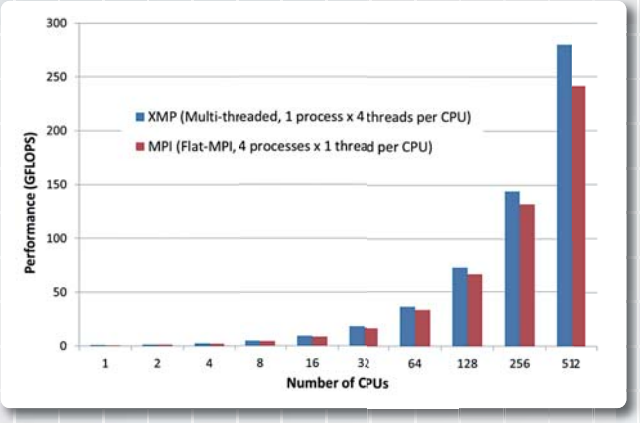
XcalableMP : Directive-Based Language eXtention for Scalable and Performance-Aware Parallel Programming

Overview of XcalableMP

- XcalableMP (XMP) is a PGAS language for distributed memory system.
- XMP extends C99 and Fortran 95 with directives, Co-array syntax, and user APIs.
- XMP supports typical parallelization under global-view programming model.
  - XMP global-view model enables parallelizing the original sequential code using minimal modification with simple directives, like OpenMP.
  - The directives can describe data distribution, work mapping, and inter-node communication for clusters.
  - Many ideas on “global-view” programming are inherited from High Performance Fortran.
- XMP includes Co-arrray Fortran syntax as local-view programming model
  - Co-array syntax in XMP describes one-sided communication between nodes.
- The important design principle of XMP is performance-awareness.
  - All actions of communication and synchronization are taken by directives, different from automatic parallelizing compilers.
  - The user should be aware of what happens by XMP directives in the execution model on the distributed memory architecture.

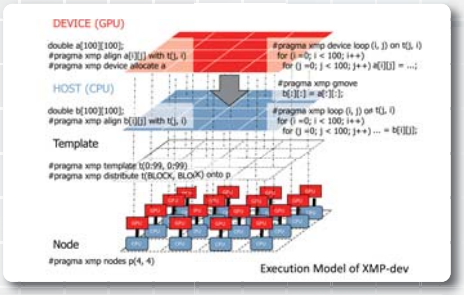
```
#pragma xmp nodes p(2, 2)
#pragma xmp template t(0:(1024*10)-1, 0:(1024*10)-1)
#pragma xmp distribute t(BLOCK, BLOCK) onto p
#pragma xmp align [i][j] with t(j, i) :: u, uu
#pragma xmp shadow u[1][1]
...
 for (k = 0; k < N; k++) {
#pragma xmp reflect (u)
#pragma xmp loop (x, y) on t(x, y) threads
   for (y = 1; y < SIZE-1; y++)
     for (x = 1; x < SIZE-1; x++)
       uu[y][x] = (u[y-1][x] + u[y+1][x] + u[y][x-1] + u[y][x+1]) / 4.0;
#pragma xmp loop (x, y) on t(x, y) threads
   for (y = 1; y < SIZE-1; y++)
     for (x = 1; x < SIZE-1; x++)
       u[y][x] = uu[y][x];
 }
```

code example: Laplace Solver



XcalableMP Acceleration Device Extension (XMP-dev)

- XMP-dev is an extension of XMP for acceleration devices such as GPUs.
- XMP-dev supports clusters equipped with acceleration devices.
- XMP-dev provides directives to describe typical processes of data parallelism for accelerators such as data allocation, transfer and task offloading onto devices.
- Data distribution and inter-node communication for cluster computing can be described in XMP-dev.




```
#pragma xmp align [i] with t(i) :: a, hb, db
#pragma xmp shadow a[*]
#pragma xmp device replicate (a)
#pragma xmp device allocate (db)

int a[N], hb[N], db[N];
void main(void) {
#pragma xmp loop on t(i)
  for (int i = 0; i < N; i++) a[i] = i + 1;
#pragma xmp reflect a
#pragma xmp device replicate_sync in (a)
#pragma xmp device loop on t(i)
  for (int i = 0; i < N; i++) {
    db[i] = 0;
    for (int j = 0; j < N; j++) db[i] += a[j];
  }
#pragma xmp gmove
  hb[:] = db[:];
}
```

code example of XMP-dev

Execution HOST (CPU)

Data Copy HOST to DEVICE

Execution DEVICE (GPU)

Data Copy DEVICE to HOST