XcalableMP

PGAS並列プログラミング言語　XcalableMPの開発進捗・状況

佐藤　三久

Team Leader, Programming Environment Research Team, RIKEN R-CCS
Professor (Cooperative Graduate School Program), University of Tsukuba

XcalableMPについて

- これまで、PCCC XcableMP言語規格部会で仕様を議論し、理研・筑波大で開発を進めてきた。
  - Omni XcalableMP コンパイラ　https://omni-compiler.org/
  - https://github.com/omni-compiler/omni-compiler
  - XcalableMP 1.x 仕様は、データ並列＋Coarrayをベースしたもの
  - MPIのruntimeを使うので、MPIが使えるクラスタならば実行可能

- 富岳で利用可能　（フラグシップ2020の一部として開発・評価）
  - XcalableMP 1.x 仕様
  - Tofu-Dを用いる実装

XcalableMP(XMP) http://www.xcalablemp.org

- **What's XcalableMP (XMP for short)?**
  - A PGAS programming model and language for distributed memory , proposed by **XMP Spec WG**
  - XMP Spec WG is a special interest group to design and draft the specification of XcalableMP language. It is now organized under **PC Cluster Consortium**, Japan. Mainly active in Japan, but open for everybody.

- **Project status (as of June 2019)**
  - XMP Spec **Version 1.4** is available at XMP site. new features: mixed OpenMP and OpenACC , libraries for collective communications.
  - Reference implementation by U. Tsukuba and Riken AICS: **Version 1.3.1 (C and Fortran90)** is available for PC clusters, Cray XT and K computer. Source-to- Source compiler to code with the runtime on top of MPI and GasNet.
- **HPCC class 2 Winner 2013. 2014**

The spec of XcalableMP 1.x is now converged.
We are now moving to XcalableMP 2.0 with global task-based parallel programming and PGAS

- **Language Features**
  - Directive-based language extensions for Fortran and C for PGAS model
  - Global view programming with global-view distributed data structures for data parallelism
    - SPMD execution model as MPI
    - pragmas for data distribution of global array.
    - Work mapping constructs to map works and iteration with affinity to data explicitly.
    - Rich communication and sync directives such as "gmove" and "shadow".
    - Many concepts are inherited from HPF
  - Co-array feature of CAF is adopted as a part of the language spec for local view programming (also defined in C).

Code example

```
int array[YMAX][XMAX];

#pragma xmp nodes p(4)
#pragma xmp template t(YMAX)
#pragma xmp distribute t(block) on p
#pragma xmp align array[i][*] to t(i)

main(){
  int i, j, res;
  res = 0;

#pragma xmp loop on t(i)  reduction(+:res)
  for(i = 0; i < 10; i++)
    for(j = 0; j < 10; j++){
      array[i][j] = func(i, j);
      res += array[i][j];
    }
}
```

data distribution

add to the serial code : incremental parallelization

work sharing and data synchronization

Example of a Global-view XMP Program

- **Collaboration in Scale project (with Tomita's Climate Science Team)**
- **Typical Stencil Code**

```
!$xmp nodes p(npx,npy,npz)

!$xmp template (lx,ly,lz) :: t
!$xmp distribute (block,block,block) onto p :: t

!$xmp align (ix,iy,iz) with t(ix,iy,iz) ::
!$xmp&        sr, se, sm, sp, sn, sl, ...

!$xmp shadow (1,1,1) ::
!$xmp&        sr, se, sm, sp, sn, sl, ...

      ...

!$xmp reflect (sr, sm, sp, se, sn, sl)

!$xmp loop (ix,iy,iz) on t(ix,iy,iz)
      do iz = 1, lz-1
      do iy = 1, ly
      do ix = 1, lx
         wu0 = sm(ix,iy,iz  ) / sr(ix,iy,iz  )
         wu1 = sm(ix,iy,iz+1) / sr(ix,iy,iz+1)
         wv0 = sn(ix,iy,iz  ) / sr(ix,iy,iz  )
         ...
```

declare a node array

declare and distribute a template

align arrays

add shadow area

stencil communication

parallelize loops

Local-view XMP program: Coarray

- **XMP includes the coarray feature imported from Fortran 2008 for the local-view programming.**
  - Basic idea: data declared as *coarray* can be accessed by remote nodes.
  - Coarray in XMP/Fortran is fully compatible with Fortran 2008.

```
real b(8)[*]

if (xmp_node_num() == 1) then
  a(:) = b(:)[2]
```

b is declared as a coarray.

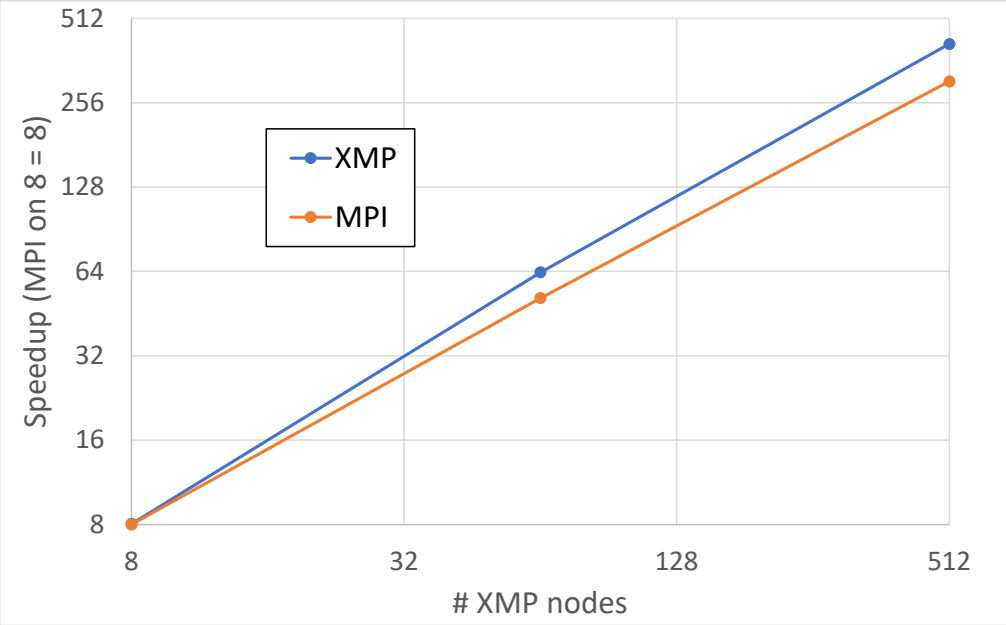
Node 1 *gets* b from node 2.

- **Coarrays can be used in XMP/C.**
  - The subarray notation is also available as an extension.

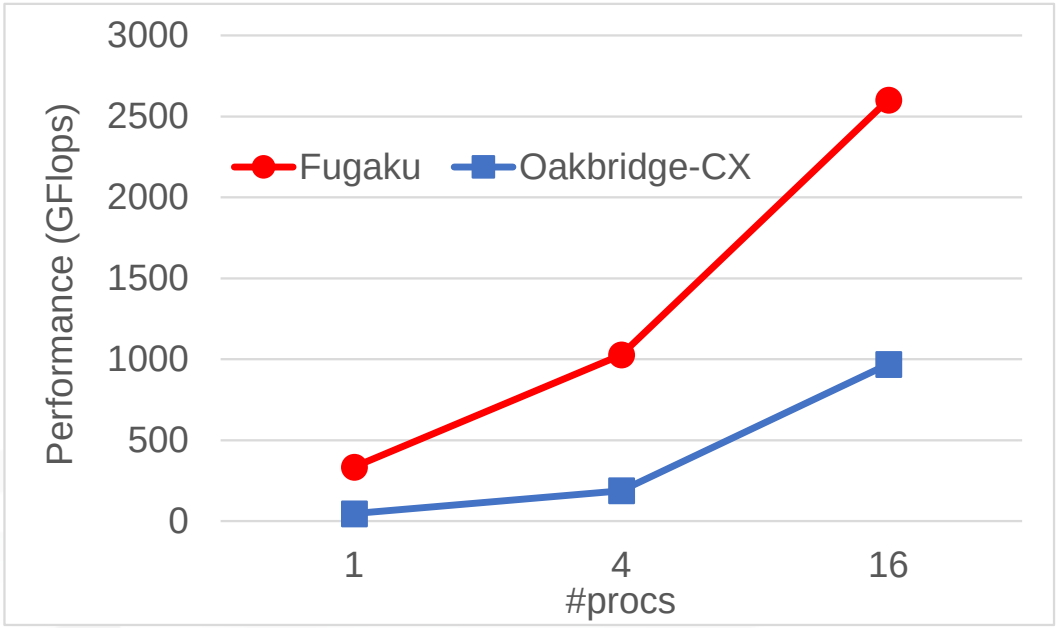
Declaration	`float b[8]:[*];`	
Put	`a[0:3]:[1] = b[3:3];`	puts b to node 1.
Get	`a[0:3] = b[3:3]:[2];`	*gets* b from node 2.
Synchronization	`void xmp_sync_all(int *status)`	

Preliminary Performance of XcalableMP on Fugaku

- XcalableMP was taken as a parallel programming language project for improving the productivity and performance of parallel programing.
- XcalableMP is now available on Fugaku
- Better result than MPI for stencil apps



Speedup of Impact-3D (global view, stencil apps)
Fusion simulation code
Size = 512^3

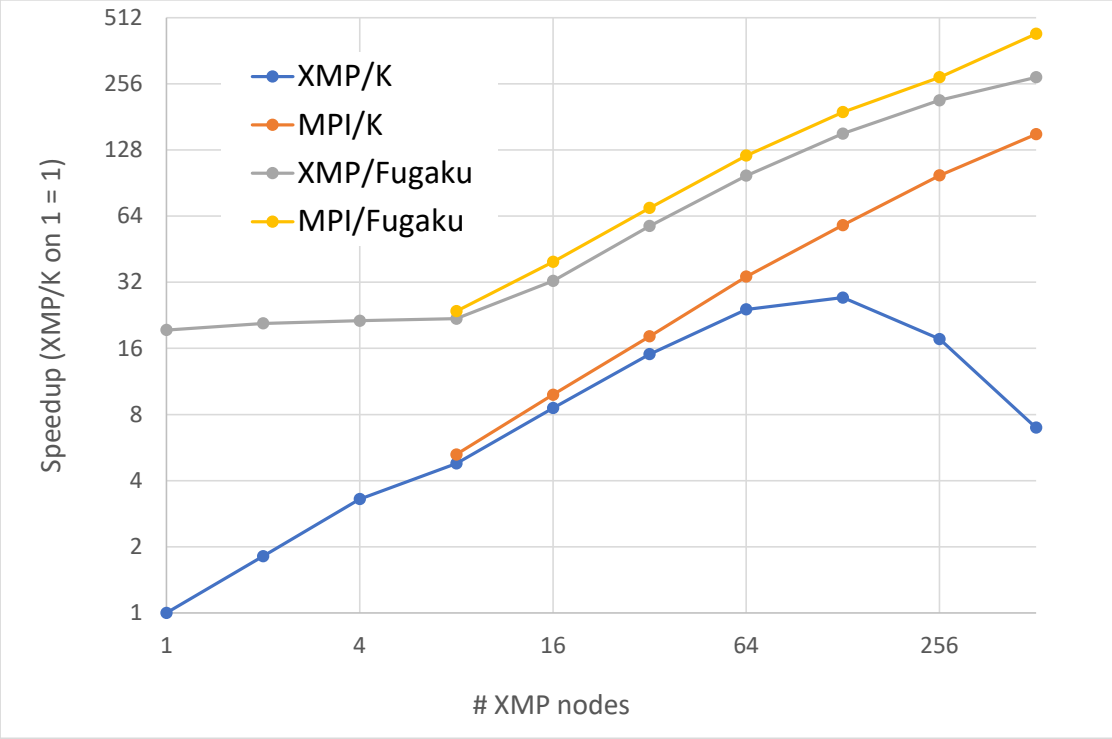


Performance Himeno Benchmark
(12threads/node, 4 proc/node)
Comparing with Cascade Lake
(2.7GHz, 28 core/socket, 2 sockets/node,
28 threads/process)

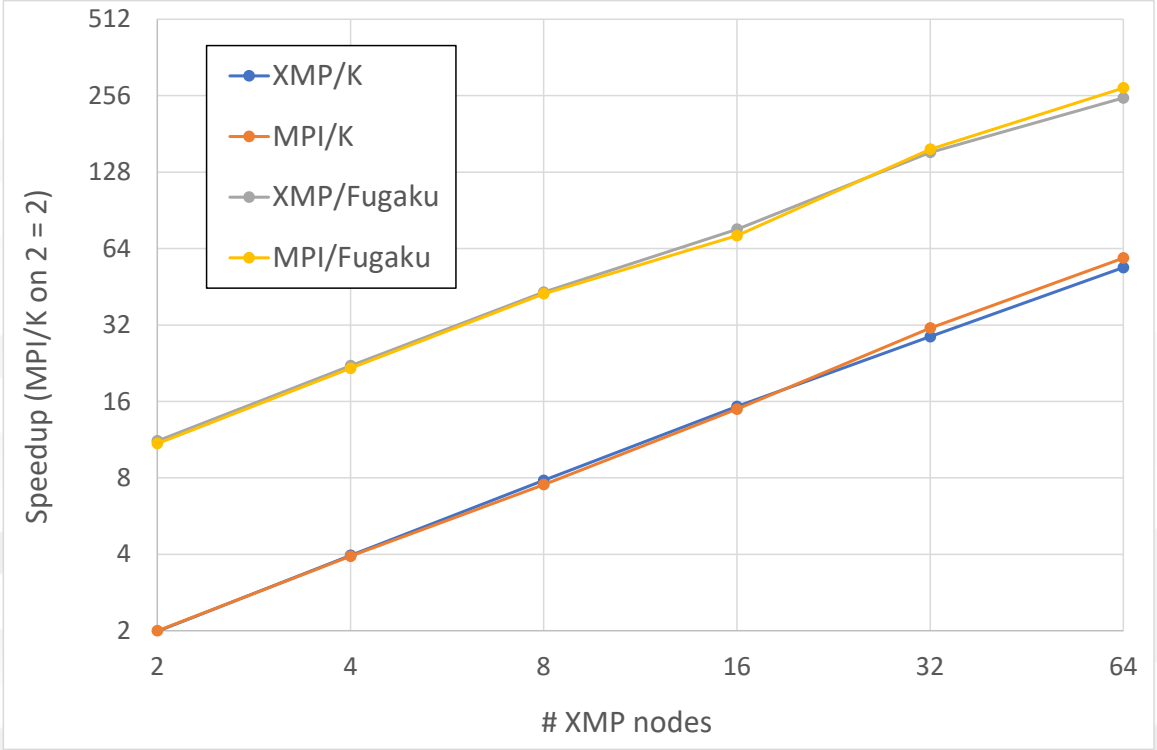
Performance of XcalableMP coarray on Fugaku

- The performance is enhanced by the Fugaku interconnect, Tofu-D, as well as node performance.
- The runtime is implemented using uTofu Libs. (better than MPI)
  - Note that the reason of the performance degradation of the XMP version on the K computer is the overhead of allocation for allocatable coarray used as a buffer for communication. It is improved by removing this overhead by using the uTofu.

QCD (Local view programming, Coarray)



NT-Chem (local view programming, Coarray)

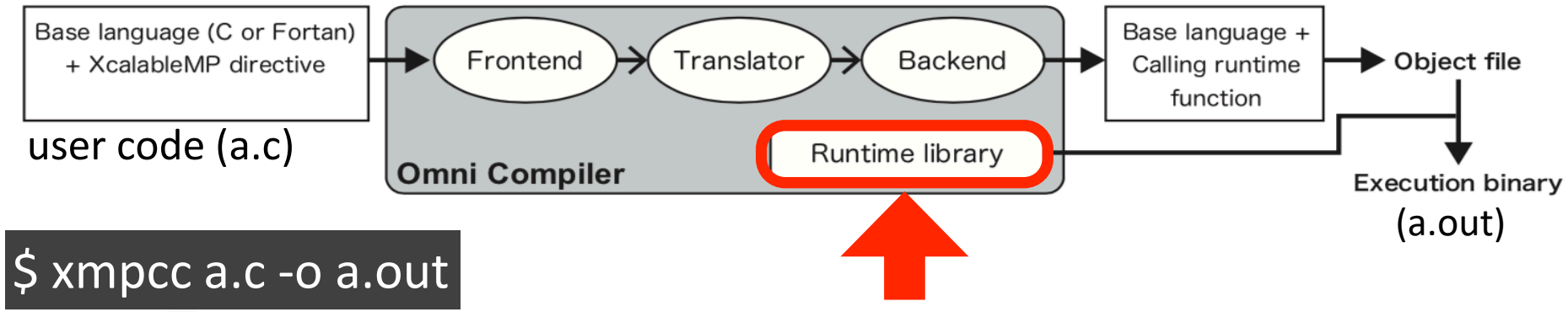


新しい言語は難しい？

- Pythonとのmixed programming: DSL的に使う
  - 並列のカーネルを記述し、それをPythonから呼び出す
- XMP API
  - 言語拡張・コンパイラアプローチでは限界があるの で、XMPとなるべく「同等」なプログラミングができる、ライブラリAPIを定義することを提案したい
  - XMPの基本的なオブジェクトである、ノード集合、テンプレート、分散配列等を、ディスクリプタで表現し、それに対するdirectiveの操作をAPIで表現する。
  - プロトタイプ実装は終了、現在公開を予定 （リポジトリにはある？）

- Pythonによる並列プログラミング
  - 「Scripting XcalableMP = SXMP　分散メモリ並列計算機用スクリプト言語 SXMP の実装と評価」
    新井正樹，津金佳祐，前田宗則，吉川隆英（富士通）
  - 情報処理学会　HPC研究会　http://www.ipsj.or.jp/kenkyukai/event/hpc183.html
  - XMP APIではないが、XMPのruntimeを利用。

Mixed-language programming with Omni XMP compiler




```
$ xmpcc a.c -o a.out
```

Add linkage API functions

- A user code with XMP directives is translated to a parallel code with runtime calls of Omni compiler's runtime library
- The runtime library is implemented in C and MPI
- The translated parallel code is compiled by a native compiler
  - e.g. GNU, Intel, PGI, Cray, and so on
- **Linkage API functions are defined for mixed-language programming**

Serial Python program calls a parallel XMP program

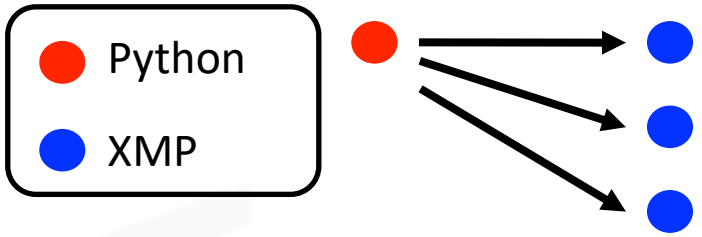
Python program (a.py)

```
import xmp

lib = xmp.Lib("xmp.so")
args = ([1,2,3], [3,4,5])
job = lib.spawn(4, "call_xmp", args, async=True)
// other work
job.wait()
```

XMP program (xmp.c)

```
void call_xmp(long a1[3],
              long a2[3]){
#pragma xmp nodes p[3]
    :
```



- xmp.Lib.spawn() method creates new processes and they work as an XMP program in parallel

```
$ xmpcc -fPIC -shared xmp.c -o xmp.so
$ mpirun -np 1 python a.py
```

Example: Graph Order/degree solver in Python + XMP

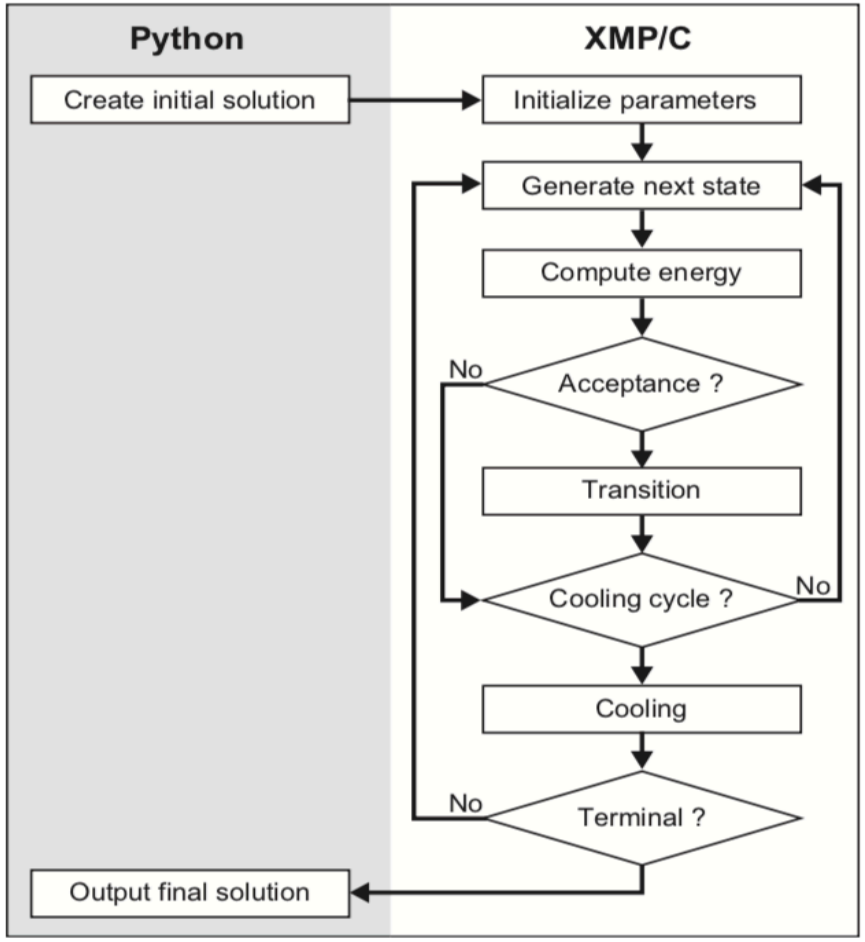
- **Implementation**
  - The existing python program generates an initial solution and saves a final solution
  - XMP program searches an optimal solution (includes calculates diameter and ASPL)
    - Optimization algorithm in XMP uses Simulated Annealing

Python

```
import xmp
 :
lib = xmp.Lib("xmp.so")
args = (vertices, degrees, edge)
lib.call(MPI.COMM_WORLD, "xmp_sa", args)
```

XMP/C

```
void xmp_sa(int vertices, , int degrees,
                int edge[vertices*degrees/2][2]){
  :
#pragma xmp loop on t[i]
 for(int i=0;i<vertices;i++){
    : // Calculate diameter and ASPL
 }
#pragma xmp reduction(max:diameter)
#pragma xmp reduction(+:ASPL)
```



これからの計画

- XMP API
  - 公開・マニュアル
  - C++での利用、wrapper template libraryの設計・開発

- XcalableMP 2.0
  - PGAS上の並列分散タスク並列を中心した仕様
  - プロトタイプ実装があるが、開発はslow-downしている。
  - 現在、富岳でのUCXの実装など、基本的なruntimeの整備をしている状況

関連プロジェクト

- XcalableACC
  - XcalableMP + OpenACC、 GPUつきのクラスタ向けの並列プログラミング言語

- Omni Compiler
  - C & Fortranのsource-to-source コンパイラの開発インフラ
  - OpenMP 4.x のoffload機能の実装
  - OpenMP/OpenACCのOpenCL出力
    - AMD, POCL, FGPA, RISC-Vへの対応