On-the-Fly Synchronization Checking for Interactive Programming in XcalableMP

Tatsuya Abe and Mitsuhisa Sato

P2S2'12
September 10, 2012

Outline

1. What is XcalableMP?
2. Verification of XcalableMP programs
3. Implementation of a Verification Tool
4. Experiment
5. Conclusion, Related Work, and Future Work

Outline

1. What is XcalableMP?
2. Verification of XcalableMP programs
3. Implementation of a Verification Tool
4. Experiment
5. Conclusion, Related Work, and Future Work

XcalableMP

XcalableMP (XMP) is a new programming language.

In XMP, we can write a program to use parallel and distributed computational environments effectively.

XMP = C + directives (~ OpenMP)

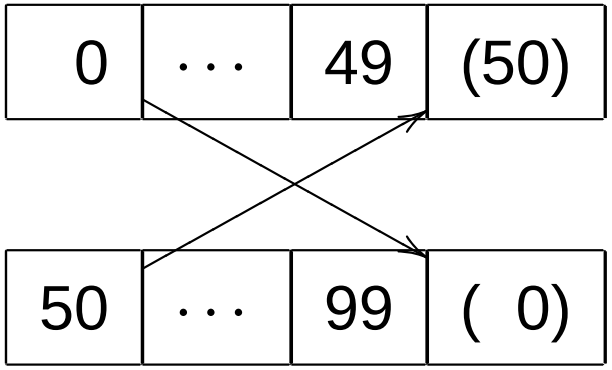
```
#pragma xmp loop on t(i)
for (i=0; i<100; i++) {
 a[i]=a[i]+a[i+1];
}
```

MPI vs. XMP

Message Passing Interface (MPI) is a communication standard for HPC.

```
int a[101];
a[100]=a[0];
for (i=0; i<100; i++) {
 a[i]=a[i]+a[i+1];
}
```

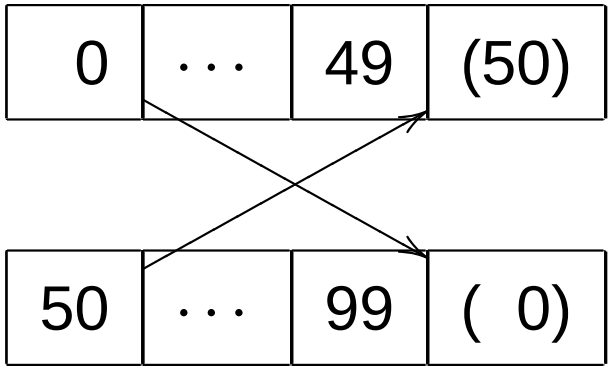
To compute it in parallel, distribute a into two computational nodes and synchronize its boundary.



MPI

To compute it in parallel, distribute a into two computational nodes and synchronize its boundary.

By message passing in MPI:




```
int a[51];
MPI_Comm_rank(MPI_COMM_WORLD, &me);
...
you=(me+1)%2;
MPI_Irecv(&(a[50]), 1, MPI_DOUBLE, you, MPI_ANY_TAG, MPI_COMM_WORLD, &req);
MPI_Send(&(a[0]), 1, MPI_DOUBLE, you, MPI_ANY_TAG, MPI_COMM_WORLD);
MPI_Wait(&req, &stat);
for (i=0; i<50; i++) { a[i]=a[i]+a[i+1]; }
```

XMP

```
#pragma xmp nodes p(2)
#pragma xmp template t(100)
#pragma xmp distribute t(block) onto p
int i, a[100];
#pragma xmp align a[i] with t(i+1)
#pragma xmp shadow a[0:1]
...
#pragma xmp reflect (a) width (/periodic/0:1)
#pragma xmp loop on t(i)
for (i=0; i<100; i++) { a[i]=a[i]+a[i+1]; }
```

Outline

1. What is XcalableMP?
2. Verification of XcalableMP programs
3. Implementation of a Verification Tool
4. Experiment
5. Conclusion, Related Work, and Future Work

Program Verification in XMP

XMP programming ~ directive programming

```
#pragma xmp shadow a[1:3]
#pragma xmp reflect (a) width (1:3)
...
#pragma xmp reflect (a) width (1:2)
```

```
#pragma xmp shadow b[1:2]
#pragma xmp reflect (b) width (1:3)
...(no occurence of b)...
#pragma xmp reflect (b) width (1:2)
```

Redundant bcast

```
#pragma xmp bcast (a) from p(1) on p(2:10)
#pragma xmp bcast (a) from p(2) on p(11:20)
#pragma xmp bcast (a) from p(1) on p(10:11)
```

Redundant lock and unlock

```
#pragma xmp lock (a[1]:[1])
#pragma xmp lock (a[1]:[1])
#pragma xmp unlock (a[2]:[1])
```

Missing Directive

Access distributed arrays without any directive:

```
#pragma xmp distribute t(block) onto p
int i;
int a[100];
#pragma xmp align a[i] with t(i)
...
b=a[0]
```

Static and Light-Weight Checking in XMP

We wish a program verifier would satisfy

- Correctness (no false positive): warning → error
- Completeness (no false negative): error → warning

Types of program verifiers:

- dynamic vs. static
- light-weight vs. heavy-weight

Static and Light-Weight Checking in XMP

We wish a program verifier would satisfy

- Correctness (no false positive): warning → error
- Completeness (no false negative): error → warning

Types of program verifiers:

- dynamic vs. static
- light-weight vs. heavy-weight

The algorithm is in the proceedings.

A work using more heavy-weight methods is on-going.

Outline

1. What is XcalableMP?
2. Verification of XcalableMP programs
3. Implementation of a Verification Tool
4. Experiment
5. Conclusion, Related Work, and Future Work

Implementation of Engine

A stream-processing program tends to be hard to read.

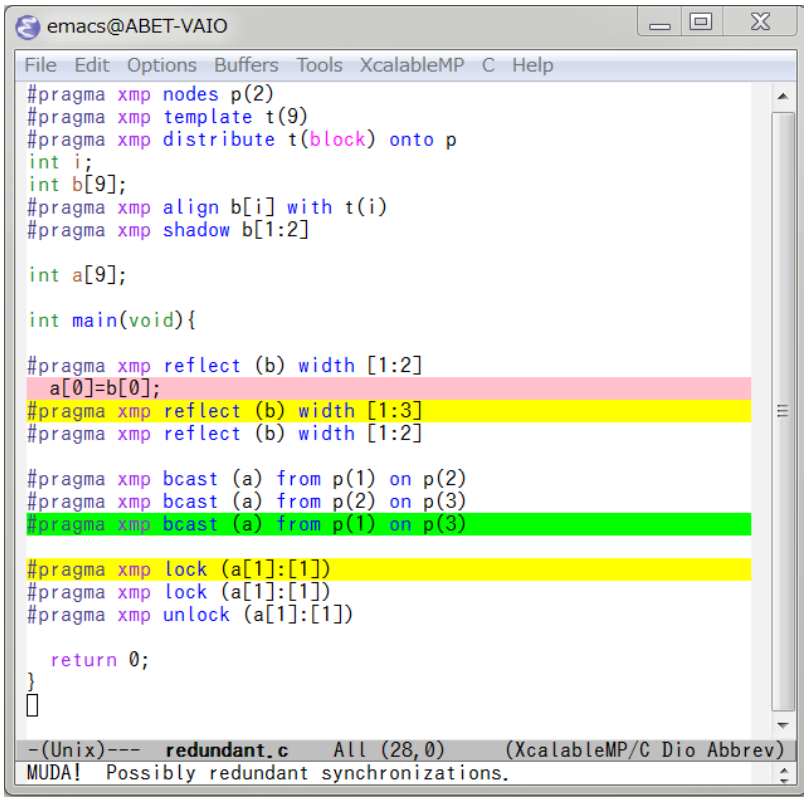
To keep readability of the source code of our tool, use

- a parser combinator library **Parsec**
- **user-defined datatypes** in Haskell, and
- **pattern-matchings** by constructors of the user-defined datatypes.

Implementation of User Interface

The engine takes a source code and returns line numbers. Possible to link any editor.

In this work, link our tool to GNU Emacs.
Every time a buffer is updated, check the source code.

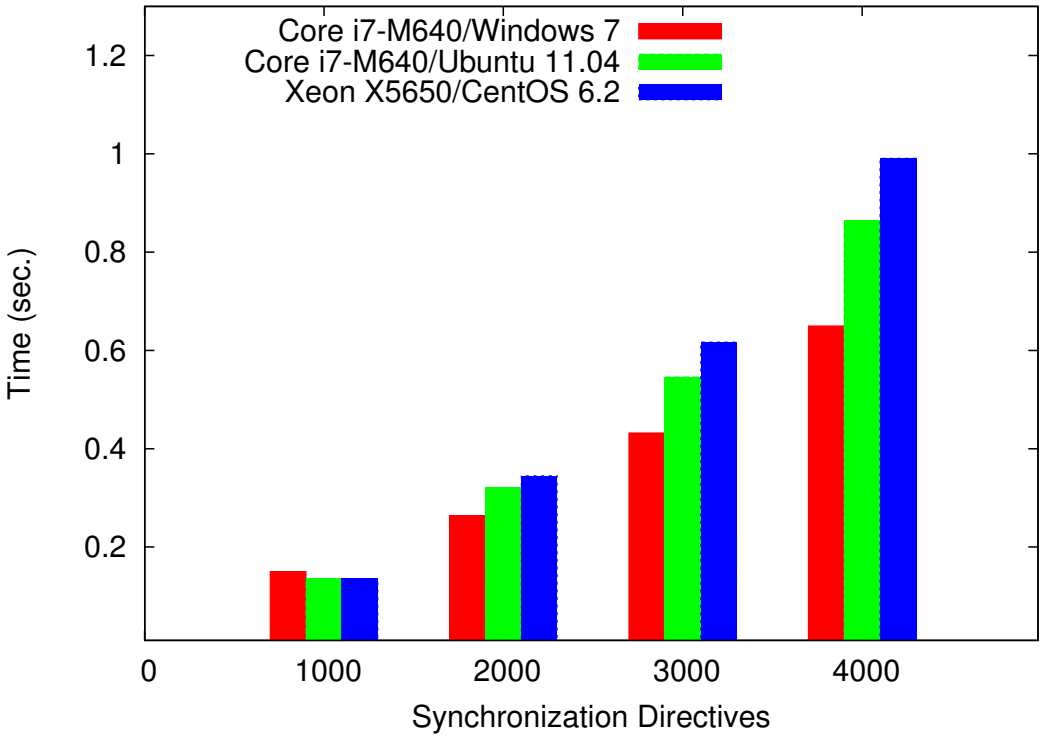


Outline

1. What is XcalableMP?
2. Verification of XcalableMP programs
3. Implementation of a Verification Tool
4. Experiment
5. Conclusion, Related Work, and Future Work

Experiment

Every time find a directive, add it to a table. The worst case is when every directive is suspended to be redundant/missing or not.



The worst case time complexity is $O(n^2)$ where n is not # of synchronizations but # of directives.

Outline

1. What is XcalableMP?
2. Verification of XcalableMP programs
3. Implementation of a Verification Tool
4. Experiment
5. Conclusion, Related Work, and Future Work

Conclusion

We develop a programming tool:

- Checks errors when writing an XcalableMP program,
- Uses XMP's features,
- Linked to GNU Emacs (possibly other IDEs), and
- Runs fast.

Abstract descriptions in XMP are useful to not only **development** of a program but also verification of the program.

Related Work

Verification by using features of languages is standard in imperative, functional, logic, object-oriented, and aspect-oriented programming fields etc.

In PGASs only,

- UPC-SPIN: static, detect race
- UPC-CHECK: run-time check, detect deadlock etc.

In this work, oriented to light-weightness (just like of a spell-checking tool Flyspell or a variable occurence-check Eclipse plugin) and a little complicated error check in XMP.

Future Work

In this work, oriented to light-weight check XMP programs.

- To detect more kinds of errors (race, dead-lock, etc.)
- To detect missing directives by model checking

are left to future work.