


FUJITSU Software

Technical Computing Suite V2.0

A decorative horizontal band with a red-to-dark-red gradient, featuring abstract, glowing white and red lines that swirl and intersect, creating a sense of motion and technology.

Rank Map Automatic Tuning Tools

User's Guide

(PRIMEHPC FX100)

J2UL-1894-01ENZ0(00)
October 2014

Preface

Purpose of This Manual

This guide describes how to use the Rank Map Automatic Tuning Tools (RMATT) that automatically optimizes the locations of MPI ranks. This guide applies to RMATT used on the Linux operating system.

Intended Readers

This guide is intended for those who want to use this tool to optimize large-scale parallel applications. It is assumed that readers of this guide have basic knowledge of Linux commands.

Organization of This Guide

[Chapter 1 Overview of RMATT](#)

Provides an overview of RMATT

[Chapter 2 Communication Pattern Analysis Tool: CMPL](#)

Explains how to use CMPL

[Chapter 3 Bisection Rank Map Tool: BIEM](#)

Explains how to use BIEM

[Chapter 4 Optimization Rank Map Tool: OPTIM](#)

Explains how to use OPTIM

[Chapter 5 Glossary](#)

Defines the terminology used in this guide

[Appendix A Considerations for Using RMATT](#)

Describes the key points to consider when using RMATT

Notation Used in This Manual

Syntax Description Symbols

The typographic conventions used in this guide include symbols that have specific meaning in syntax.

| Symbol name | Symbol | Description |
|-------------------|--------|--|
| Selection symbols | { } | Indicates that only one of the enclosed items can be selected |
| | | Is used as a delimiter in a list of items |
| Optional symbol | [] | Indicates that the enclosed item can be omitted. This symbol has the same meaning as the selection symbol, "{}" |

Export Controls

Exportation/release of this document may require necessary procedures in accordance with the regulations of your resident country and/or US export control laws.

Trademarks

- Linux is a registered trademark or trademark of Linus Torvalds in the U.S. and other countries.
- All other trademarks or registered trademarks appearing in this manual are trademarks or registered trademarks of their respective owners.
- The trademark symbols (TM, (R)) are not necessarily indicated.

About METIS

RMATT uses METIS for optimizing the location of MPI ranks.

METIS 4.0.1 Copyright 1998, Regents of the University of Minnesota

Refer to the METIS manuals and the following documents for information on METIS.

"A Fast and High Quality Multilevel Scheme for Partitioning Irregular Graphs". George Karypis and Vipin Kumar. SIAM Journal on Scientific Computing, Vol. 20, No. 1, pp. 359-392, 1999.

Date of Publication and Version

| Version | Manual code |
|---------------------------|----------------------|
| October 2014, 1st Version | J2UL-1894-01ENZO(00) |

Copyright

Copyright FUJITSU LIMITED 2014

All rights reserved.
The information in this manual is subject to change without notice.

Contents

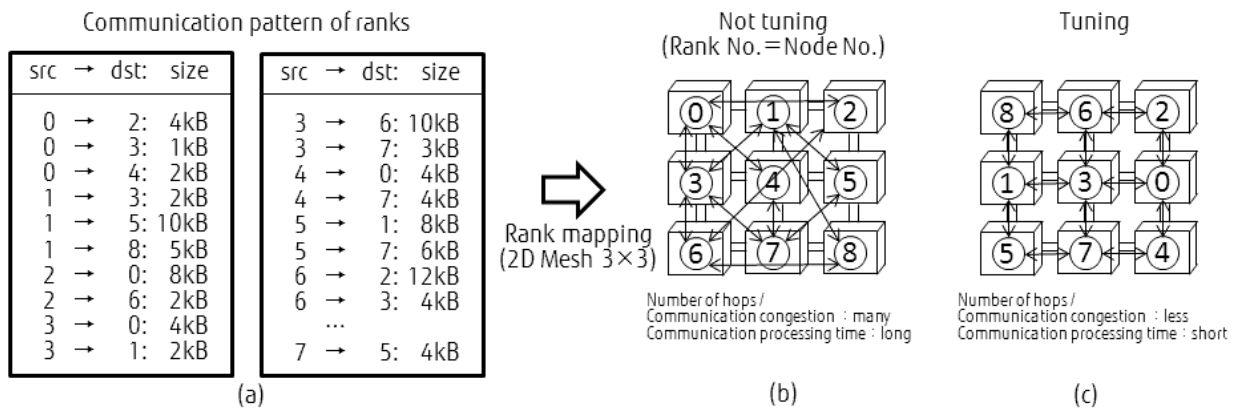
| | |
|--|----|
| Chapter 1 Overview of RMATT..... | 1 |
| 1.1 MPI Rank Location Optimization..... | 1 |
| 1.2 Purpose of RMATT..... | 1 |
| 1.3 RMATT Features..... | 1 |
| 1.4 RMATT Components..... | 2 |
| 1.5 RMATT Commands..... | 2 |
| 1.6 Flow of MPI Rank Location Optimization by RMATT..... | 2 |
| Chapter 2 Communication Pattern Analysis Tool: CMPL..... | 4 |
| 2.1 Overview of CMPL..... | 4 |
| 2.2 CMPL Command..... | 4 |
| 2.3 Output Format..... | 5 |
| 2.4 Environment Variables..... | 5 |
| 2.5 Execution Example..... | 5 |
| Chapter 3 Bisection Rank Map Tool: BIEM..... | 7 |
| 3.1 Overview of BIEM..... | 7 |
| 3.2 BIEM Command..... | 7 |
| 3.3 Output Format..... | 8 |
| 3.4 Environment Variables..... | 8 |
| 3.5 Execution Example..... | 9 |
| Chapter 4 Optimization Rank Map Tool: OPTIM..... | 11 |
| 4.1 Overview of OPTIM..... | 11 |
| 4.2 OPTIM Command..... | 11 |
| 4.3 Output Format..... | 12 |
| 4.4 Environment Variables..... | 12 |
| 4.5 Execution Example..... | 13 |
| Chapter 5 Glossary..... | 16 |
| Appendix A Considerations for Using RMATT..... | 17 |
| A.1 CMPL..... | 17 |
| A.2 BIEM..... | 17 |
| A.3 OPTIM..... | 18 |

Chapter 1 Overview of RMATT

1.1 MPI Rank Location Optimization

If an MPI application is used on a direct network, such as Mesh or Torus, the communication time is determined based on the physical locations of nodes on which MPI ranks are executed. This is because the number of nodes on the communication path (hops) and the number of communication contentions differ according to the locations of MPI ranks. The communication time can be minimized by changing the locations of MPI ranks based on the communication pattern. In this guide, this method is called MPI rank location optimization. An example of rank location optimization is shown below.

Figure 1.1 MPI rank location optimization example



In the above figure, the communication pattern (a) is executed on 2D Mesh 3x3 nodes.

For the communication pattern (a), if MPI ranks are located in a coordinate sequence such as (b), the numbers of hops and communication contentions increase. This increases the communication time.

If the locations of MPI ranks are optimized such as (c), the number of hops and communication contentions decrease, reducing the communication time.

1.2 Purpose of RMATT

In MPI rank location optimization, the locations of MPI ranks is determined and evaluated by trial and error. It is not easy to realize for large-scale MPI applications because this computation is $N!$. The Rank Map Automatic Tuning Tools (RMATT) automatically optimizes MPI rank locations. RMATT reduces the tuning cost for application developers.

1.3 RMATT Features

RMATT automatically optimizes the locations of MPI ranks based on the communication between ranks (communication pattern) and the layout of the network where the MPI application is executed. RMATT creates a file in which the specified optimized MPI rank locations are coded. This file is called the "rank map file". The rank map file can be specified for the rank-map-hostfile parameter of a job script.

The features of RMATT are given below.

Table 1.1 RMATT features

| Feature | Description |
|--------------------------|-------------------------------|
| Scale of MPI application | 32-16,384 ranks. |
| Communication pattern | MPI 1:1 communication |
| Network configuration | 1D-3D Torus. |
| How to locate MPI ranks | Locating one rank in one node |

It is expected that the communication time will be less if an MPI application has the following characteristics:

- The number of communication contentions is large in executing the MPI application without MPI rank location optimization.
- The number of communication hops is large in executing the MPI application without MPI rank optimization.
- The number of nodes on a side of the network configuration is 2^n node.

1.4 RMATT Components

For the automatically rank map tuning, RMATT is composed of the following tools.

- Communication Pattern Analysis Tool CMPL (CoMmunication Pattern anaLyzer)

The communication pattern is analyzed based on the trace data file that the function of the profiler who collects execution information on the application in the time (It is called Tracer at the following) made, and the communication pattern file that RMATT needs to optimize the rank map is made.

As for the trace data file, execution information on the time series of the application that the Tracer collected is recorded. Please refer to "Profiler User's Guide" for Tracer details.

- Rank Map Bisection Processing Tool BISEM (BISEction rankMap)

The OPTIM parameter file and the OPTIM initial rank map file necessary for Rank Topology Optimization tool OPTIM execution are made. The RMATT user executes this tool before executing OPTIM that is the tool that optimize the rank map.

- Rank Topology Optimization Tool OPTIM (OPTImization rankMap)

Optimize the rank map of the target application program, and the optimized rank map file is made. The RMATT user can optimize the rank map of the target application program by specifying the parameter file that BISEM made.

1.5 RMATT Commands

Commands provided by RMATT include:

cmplx

Command for executing CMPL

bisempx

Command for executing BISEM

optimpx

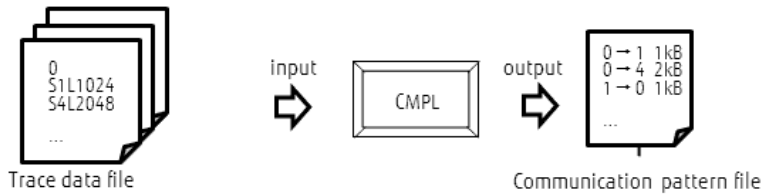
Command for executing OPTIM

1.6 Flow of MPI Rank Location Optimization by RMATT

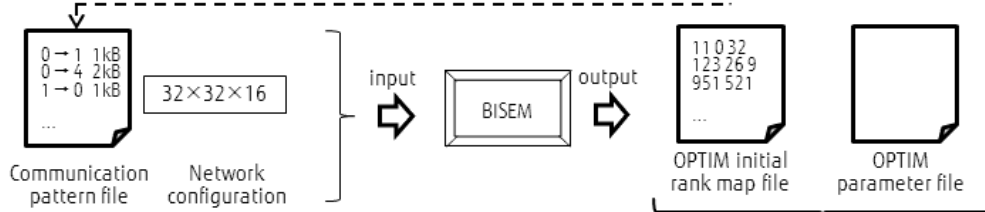
MPI rank location optimization by RMATT is shown below.

Figure 1.2 Flow of MPI rank location optimization by RMATT

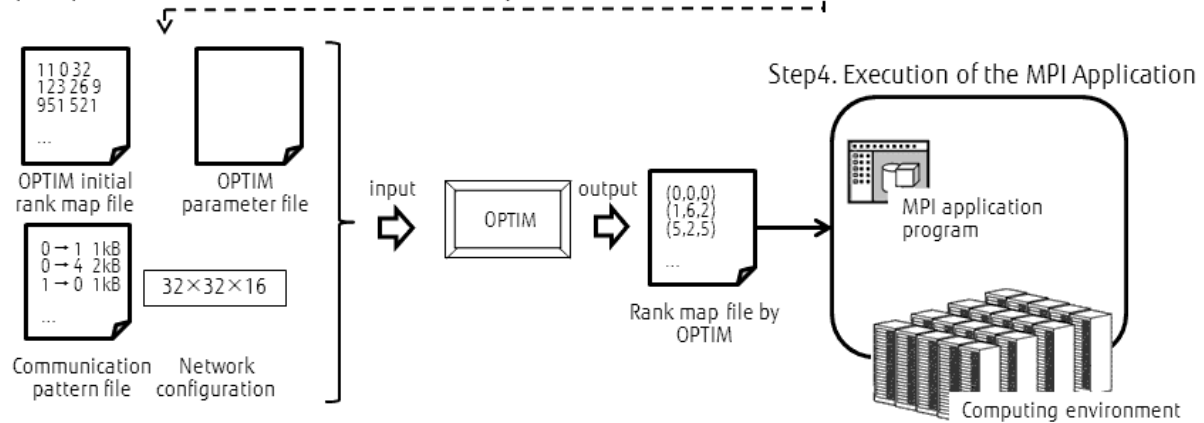
Step1. Analysis of Communication pattern by CMPL



Step2. Bisection MPI Ranks by BISEM



Step3. Optimization of the MPI Rank Locations by OPTIM



1. Analysis of Communication pattern by CMPL

The RMATT user inputs trace data files created by Tracer to CMPL. CMPL analyzes the communication pattern from the trace data files, and creates the communication pattern file.

2. Bisection MPI Ranks by BISEM

The RMATT user inputs the communication pattern file created by CMPL and the network configuration. BISEM performs the initializing process for MPI rank location optimization, and creates the OPTIM parameter file and the OPTIM init rank map file.

3. Optimization of the MPI Rank Locations by OPTIM

The RMATT user inputs the OPTIM parameter file and the OPTIM init rank map file created by BISEM. OPTIM optimizes the locations of MPI ranks, and creates the rank map file.

4. Execution of the MPI Application

The RMATT user executes the MPI application by using the rank map file created by OPTIM.

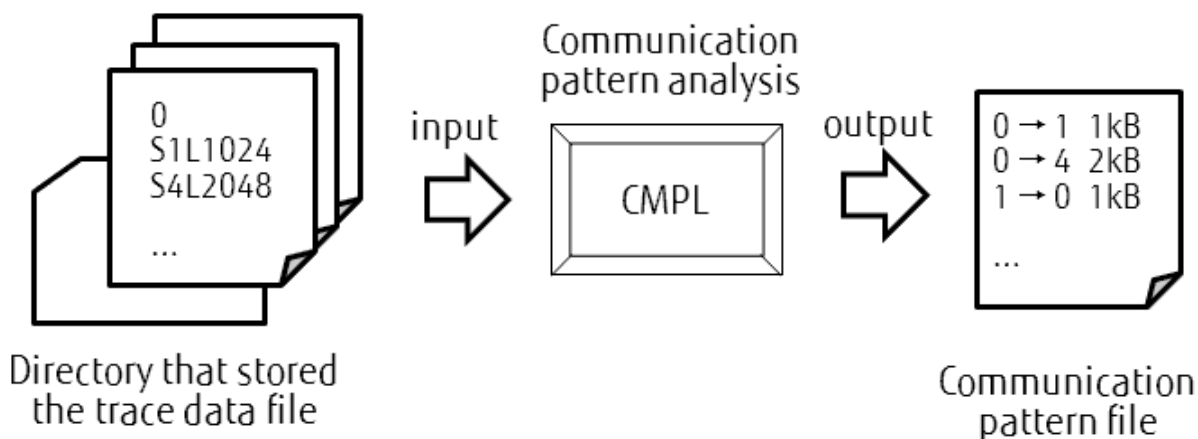
Chapter 2 Communication Pattern Analysis Tool: CMPL

This chapter explains communication pattern analysis tools CMPL.

2.1 Overview of CMPL

CMPL is a tool that analyzes the communication pattern based on the trace data file that Tracer made, and makes the communication pattern file. The following figures show the flow of the communication pattern analysis by CMPL. The RMATT user specifies the directory where the trace data file of the target application program is stored, and executes CMPL. CMPL analyzes the communication pattern, and makes the communication pattern file. When BISEM and OPTIM are executed, this communication pattern file is used.

Figure 2.1 Related Chart of Communication Pattern Analysis by CMPL



Please note the following points when you use CMPL.

- Please let CMPL read the trace data files integrated by the vtunifypx command, etc.
- It doesn't correspond to the application executed by two or more threads in CMPL.
- The trace data file that CMPL reads is only the file non-compressed. To so as not to compress it, please specify no for environment variable VT_COMPRESSION when the application translated by Tracer is executed, or set --nocompress option when the local trace data file is integrated.

The trace data file is a file into which the local trace data file is integrated by the vtunifypx command etc, and execution information on the time series of the target application program is recorded. The local trace data file can be collected by executing the application translated by vtfcpx, vtFCCpx, and the vtfrpx command. Please refer to "Profiler User's Guide" for Tracer of Profiler.

2.2 CMPL Command

Syntax

```
cmplpx -out pattern_file -vt vt_log_dir [-exelog exelog_file] [-help]
```

Parameters

There are two types of parameters:

- Required

These parameters must be specified.

- Optional

It is not necessary to specify a value for these parameters. If a value is not specified for an optional parameter, the default value is used by RMATT.

Parameters of the `cmplx` command are described below.

Table 2.1 `cmplx` command parameters

| Parameter | Type of parameter | Description |
|---|---|---|
| <code>-out <i>pattern_file</i></code> | Required | Specify a name for the communication pattern file. |
| <code>-vt <i>vt_log_dir</i></code> | Required | Specify the name of the directory where trace data files are located. |
| <code>-exelog <i>exelog_file</i></code> | Optional Default value: <code>cmpl<time>.exelog</code> <time> is a character string indicating the CMPL execution time. | Specify a name for the CMPL execution log file. |
| <code>-help</code> | | Displays the Help messages for this command |

2.3 Output Format

When you execute the `cmplx` command, the following files are made.

- Communication pattern file *pattern_file*
- Cmplpx command execution log file *exelog_file*

The communication pattern of the target application program is output to *pattern_file*, and it is possible to use it with RMATT.

The execution log of `cmplx` command is output to *exelog_file*.

2.4 Environment Variables

The following settings are required for using the `cmplx` command.

Table 2.2 Environment variables

| Environment variable | Value |
|----------------------|--|
| PATH | FSDT installation directory /<Version>/bin |
| RMATT | FSDT installation directory /<Version>/etc/rmatt |

"<Version>" is the version of the FSDT package.

2.5 Execution Example

An example of analyzing the communication pattern using trace data files located in the `~/sample/vt` directory is given below. The `~/sample/vt` directory contains 16 trace data files within which the information of 16 MPI processes is coded.

Example of executing the `cmplx` command

```
[rmattuser ~]$ ls -F ~/sample/vt
sample.1.events  sample.2.events  sample.4.events  sample.6.events  sample.8.events  sample.a.events
sample.c.events  sample.e.events
sample.10.events sample.3.events  sample.5.events  sample.7.events  sample.9.events
sample.b.events  sample.d.events  sample.f.events
```

```

[rmattuser ~]$
[rmattuser ~]$ cmplx -out sample.ptn -vt ~/sample/vt -exelog cmpl.exelog
START CMPL@RMATT
"CoMmunication Pattern anaLyzer

"OPTIONS:
Pattern file: sample.ptn
Log file dir: /home/rmattuser/sample/vt
Exe log file: cmpl.exelog

Reading vt log files ...
/home/rmattuser/sample/vt/wrf.1.events (1/16) reading ... done.
/home/rmattuser/sample/vt/wrf.10.events (2/16) reading ... done.
/home/rmattuser/sample/vt/wrf.2.events (3/16) reading ... done.
...
/home/rmattuser/sample/vt/wrf.f.events (16/16) reading ... done.
complete!

Analyzing rank pattern ...
rank0(1/16) analyzing ... done
rank1(2/16) analyzing ... done
rank2(3/16) analyzing ... done
...
rank15(16/16) analyzing ... done
complete!

Writing pattern file: sample.ptn ... complete!

END CMPL with SUCCESS (5sec.)
[rmattuser ~]$
[rmattuser ~]$ ls -F
cmpl.exelog  sample/  sample.ptn

```

In this example, the cmplx command is executed with the following option.

- -out sample.ptn
- -vt ~/sample/vt
- -exelog cmpl.exelog

Sample.ptn is specified for a destination of the communication pattern file with -out option. ~/sample/vt directory is specified for a directory that stored the trace data file with -vt option. Cmpl.exelog is specified for an output file of the execution log of cmplx command with -exelog option.

The made file is as follows.

- sample.ptn
- cmpl.exelog

The communication pattern is analyzed based on the trace data file that exists in ~/sample/vt directory, and cmplx command outputs sample.ptn. The execution log of cmplx command is output to cmpl.exelog.

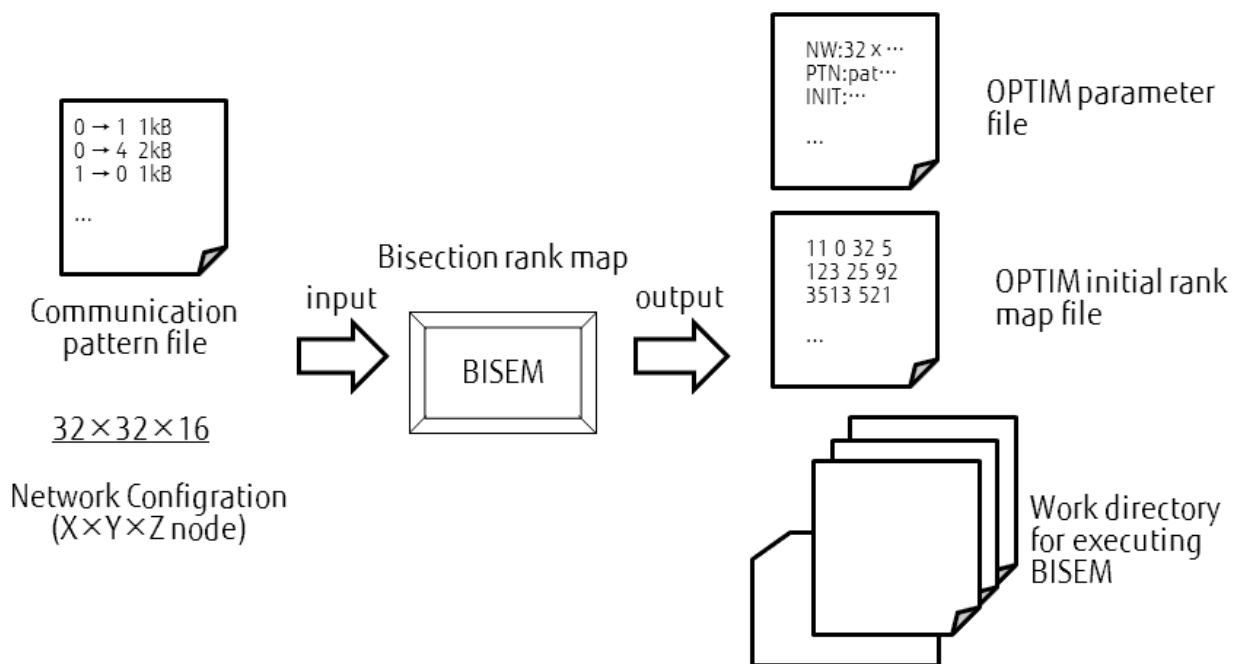
Chapter 3 Bisection Rank Map Tool: BISEM

3.1 Overview of BISEM

BISEM makes the OPTIM parameter file and the OPTIM initial rank map file based on the communication pattern file that CMPL made and network configuration. They are necessary to execute the Optimization rank map tool OPTIM.

The following figure shows the flow of making the OPTIM parameter file and the OPTIM initial rank map file by BISEM. The RMATT user specifies the communication pattern file and network configuration that CMPL made, and executes BISEM. BISEM analyzes the communication pattern and network configuration, and makes the OPTIM parameter file and the OPTIM initial rank map file. When OPTIM is executed, the OPTIM parameter file and the OPTIM initial rank map file are needed.

Figure 3.1 Related Chart of Bisection Rank Map by BISEM



3.2 BISEM Command

Syntax

```
bisempx -out output_file_name -ptn pattern_file -nw XxYxZ [-workdir work_dir] [-exelog exelog_file] [-help]
```

Parameters

There are two types of parameters:

- Required

These parameters must be specified.

- Optional

It is not necessary to specify a value for these parameters. If a value is not specified for an optional parameter, the default value is used by RMATT.

Parameters of the bisempx command are described below.

Table 3.1 bisempx command parameters

| Parameter | Type of parameter | Description |
|------------------------------|--|--|
| -out <i>output_file_name</i> | Required | Specify the file name of the file that the bisempx command outputs. |
| -ptn <i>pattern_file</i> | Required | Specify the file name of the communication pattern file. |
| -nw <i>XxYxZ</i> | Required | Specify the number of nodes of each X,Y,Z axis on the network. |
| -workdir <i>work_dir</i> | Optional default: bisem< <i>time</i> > < <i>time</i> >=Character string that indicates the execution date of bisempx command. | Specify the directory name of the work directory that stores the temporary file of bisempx command. When the specified directory doesn't exist, BISEM makes <i>work_dir</i> . |
| -exelog <i>exelog_file</i> | Optional default: bisem< <i>time</i> >.exelog < <i>time</i> >=Character string that indicates the execution date of bisempx command. | Specify the file name of the file to which the execution log of bisempx command is output. |
| -help | | The help message of the bisempx command is output. |

3.3 Output Format

When you execute the bisempx command, the following files and directories are made.

- OPTIM parameter file *output_file_name.param*
- OPTIM initial rank map file *output_file_name.init*
- Working directory *work_dir* for executing bisempx command
- Bisempx command execution log file *exelog_file*

The OPTIM parameter file is output to *output_file_name.param*. The OPTIM parameter file is necessary for the OPTIM. The parameter for OPTIM, the OPTIM initial rank map file, the communication pattern file of the target application program, and network configuration etc. are specified for the OPTIM parameter file. Please do not edit the output OPTIM parameter file.

The OPTIM initial rank map file is output to *output_file_name.init*. Please do not edit it because it is necessary for OPTIM. The path to the OPTIM initial rank map file is specified in the OPTIM parameter file, so neither move the file nor change the file name, please.

The temporary files that are made of executing bisempx command are stored in the *work_dir* directory. When the *work_dir* directory doesn't exist, the bisempx command makes the *work_dir* directory.

The execution log of bisempx command is output to *exelog_file*.

3.4 Environment Variables

The following settings are required for using the bisempx command.

Table 3.2 Environment variables

| Environment variable | Value |
|----------------------|---|
| PATH | FSDT installation directory /< <i>Version</i> >/bin |
| RMATT | FSDT installation directory /< <i>Version</i> >/etc/rmatt |

"< *Version* >" is the version of the FSDT package.

3.5 Execution Example

The example of executing the bisempx command is as follows. It is assumed that the communication pattern file is sample_512.ptn in ~/sample/ptns and the network configuration is 3D Torus 8x8x8 node.

Example of executing the bisempx command

```
[rmattuser ~]$ bisempx -out sample -ptn ~/sample/ptns/sample_512.ptn -nw 8x8x8 -workdir bisem_work -
exelog bisem.exelog
START BISEM@RMATT
"BISEction rankMap"

OPTIONS:
  Output file name: sample
  Network size: 8x8x8
  Pattern file: /home/rmattuser/sample/ptns/sample_512.ptn
  Work dir: bisem_work
  Exe log file: bisem.exelog

=====
START BISECTION PROCESS
=====
ITRS=1/6 GRPS:1->2
=====
BISECTION GRP 0
  Bisection rank grp0 ... done!
  BISECTION X AXIS
  Bisection network area ... done!
  Bisection all rank grps
    0 ..... 100%
  done!
  Calculating matching score ... done!
    Matching score=2688
BISECTION Y AXIS
  Bisection network area ... done!
...
BISECTION RESULT:
  BIASECTED AXIS: Y (23456)
  RANK GROUP:
    Num of grps: 32
    Num of ranks in grp: 8
  NETWORK AREA:
    Num of areas: 32 (4x4x4)
    Num of nodes in area: 8 (2x2x2)

=====
END BISECTION PROCESS
=====
Generating OPTIM initmap ...
Reading BISEM result ...
  Rank grp file: bisem_work/result.grp
  Network area file: bisem_work/result.area
done!
Checking Bisem result ... OK!
Generating rank map
  0 ..... 100%
complete!
Writing rankmap file: sample.init ...
complete!
complete!
```

```

Generating OPTIM param file ...
sample.param:
# Default OPTIM param file
NW_SIZE: 8x8x8
PTN_FILE: /home/rmattuser/sample/ptns/sample_512.ptn
MAX_ITRS: 1000000
INIT_MAP_FILE: /home/rmattuser/sample.init
NUM_OF_SWAPPED_RANKS: 6
SWAP_RANGE: 3x3x3
NUM_OF_CHILDREN: 1000
INIT_T: 100000
INTERVAL_OF_UPDATE: 2000
FREEZE_POINT: 1.0E-50
ALPHA: 0.95
complete!

END BISEM with SUCCESS (4sec.)
[rmattuser ~]$
[rmattuser ~]$ ls -F
bisem.exelog  bisem_work/  sample/  sample.init  sample.param

```

In this example, the bisempx command is executed with the following option.

- -out sample
- -ptn ~/sample/ptns/sample_512.ptn
- -workdir bisem_work
- -exelog bisem.exelog

The character string "sample" is specified for a file name that BISEM outputs to -out option. Communication pattern file is specified for -ptn option, and network configuration is specified for -nw option. The bisem_work directory is specified to -workdir option for a working directory that stored the temporary files of bisempx command, and bisem.exelog is specified to -exelog option for an output of the execution log of bisempx command.

Made file and directory are as follows.

- sample.param
- sample.init
- bisem.exelog
- bisem_work/(directory)

Based on the execution result of BISEM, sample.param is made as the OPTIM parameter file and sample.init is made as the OPTIM initial rank map file. These files are necessary to execute OPTIM, so do not edit these, please. The path of sample.init is specified in sample.param, so sample.init must not be changed the file name and not move.

The execution log of bisempx command is output to bisem.exelog. The bisem_work directory is a working directory made to store the temporary file of bisempx command.

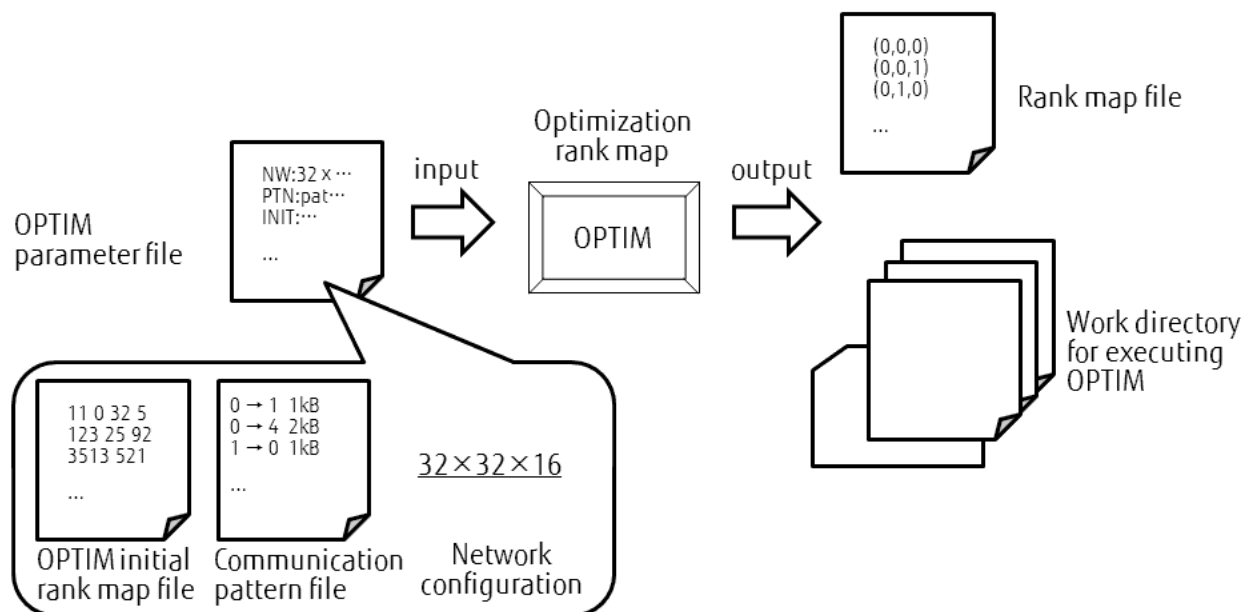
Chapter 4 Optimization Rank Map Tool: OPTIM

4.1 Overview of OPTIM

OPTIM optimizes location of MPI ranks based on the OPTIM parameter file and the OPTIM initial rank map file that BISEM made, and makes the optimized rank map file. The following figure shows the flow of making the rank map file by OPTIM. The RMATT user specifies the OPTIM parameter file that BISEM made and executes OPTIM. The path of the OPTIM initial rank map file, the path of the communication pattern file, and the network configuration are included in the OPTIM parameter file.

OPTIM optimizes location of MPI ranks, and makes the optimized rank map file. The rank map file that OPTIM made can be specified for a value of the rank-map-hostfile parameter in the job script.

Figure 4.1 Related Chart of MPI Rank Location Optimization by OPTIM



4.2 OPTIM Command

Syntax

```
optimpx -out host_file -param param_file [-end min rate] [-workdir work_dir] [-exelog exelog_file][-help]
```

Parameters

There are two types of parameters:

- Required

These parameters must be specified.

- Optional

It is not necessary to specify a value for these parameters. If a value is not specified for an optional parameter, the default value is used by RMATT.

Parameters of the optimpx command are described below.

Table 4.1 optimpv command parameters

| Parameter | Type of parameter | Description |
|----------------------------|--|--|
| -out <i>host_file</i> | Required | Specify the file name of the rank map file that the optimpv command outputs. |
| -param <i>param_file</i> | Required | Specify the OPTIM parameter file. The OPTIM parameter file is made by bisemv command. |
| -end <i>min rate</i> | Optional default: <i>min</i> =10 <i>rate</i> =0.1. The termination condition becomes "The evaluation value will not improve 10% or more in ten minutes". | Specify the termination condition of the optimpv command. The termination condition becomes "The percentage of the evaluation value will not improve <i>rate</i> or more in <i>min</i> minute". |
| -workdir <i>work_dir</i> | Optional default: optim< <i>time</i> > < <i>time</i> >=Character string that indicates the execution date of optimpv command. | Specify the directory name of the work directory that stores the temporary file of optimpv command. When the specified directory doesn't exist, OPTIM makes the directory. |
| -exelog <i>exelog_file</i> | Optional default: optim< <i>time</i> >.exelog < <i>time</i> >=Character string that indicates the execution date of optimpv command. | Specify the file name of the file to which the execution log of optimpv command is output. |
| -help | | The help message of the optimpv command is output. |

4.3 Output Format

When you execute the optimpv command, the following files and directories are made.

- Rank map file *host_file*
- Working directory *work_dir* for executing optimpv command
- Optimpv command execution log file *exelog_file*

The rank map file is output to *host_file* based on the execution result of OPTIM. The rank map file that OPTIM made can be specified for a value of the rank-map-hostfile parameter in the job script. When the optimpv command ends on the way, the rank location optimizing it at that time is output to *host_file* as a rank map file. The temporary files of optimpv command are stored in the *work_dir* directory. When the *work_dir* directory doesn't exist, the optimpv command makes the *work_dir* directory. The execution log of optimpv command is output to *exelog_file*.

4.4 Environment Variables

The following settings are required for using the optimpv command.

Table 4.2 Environment variables

| Environment variable | Value |
|----------------------|---|
| PATH | FSDT installation directory /< <i>Version</i> >/bin |
| RMATT | FSDT installation directory /< <i>Version</i> >/etc/rmatt |

"< *Version* >" is the version of the FSDT package.

4.5 Execution Example

The example of executing the `optimpix` command is as follows. This example uses the OPTIM parameter file `sample.param` and OPTIM initial rank map file `sample.init` made of `bisempix` command in [Example of executing the bisempix command](#).

Example of executing the `optimpix` command

```
[rmattuser ~]$ optimpix -out sample.map -param sample.param -end 1 0.01 -workdir optim_work -exelog
optim.exelog
START OPTIM@RMATT
"OPTImization rankMap"

OPTION:
  MPI host file: sample.map
  Param file: sample.param
  End condition: min=1, rate=0.01
  Work dir: optim_work
  Exe log file: optim.exelog

=====
  START OPTIMIZATION PROCESS
=====
0 1.00 * 3.3461e+08(1.1216e+09) 499418(797160) 16818(11704) 670(1407)
10 0.76 * 2.54376e+08(1.1216e+09) 451822(797160) 16650(11704) 563(1407)
20 0.71 * 2.37729e+08(1.1216e+09) 434606(797160) 16620(11704) 547(1407)
...
180 0.64 0.01 2.14085e+08(1.1216e+09) 414894(797160) 16416(11704) 516(1407)
=====
  END OPTIMIZATION PROCESS
=====

OPTIM RESULT
  RESULT SCORE = 214085304
    Sum traffic = 414894
    Sum hop = 16416
    Max link = 516
  TARGET SCORE = 1121604120
    Sum traffic = 797160
    Sum hop = 11704
    Max link = 1407

Generating rank map file ...
  MPI rankmap file: sample.map
complete!

END OPTIM with SUCCESS (180sec.)
[rmattuser ~]$
[rmattuser ~]$ ls -F
bisem.exelog bisem_work/ optim.exelog optim_work/ sample/ sample.init sample.map sample.param
[rmattuser ~]$
[rmattuser ~]$ cat sample.map
(1, 0, 0)
(1, 2, 1)
(3, 0, 0)
...
(4, 1, 4)
```

In this example, the `optimpix` command is executed with the following option.

- `-out sample.map`
- `-param sample.param`

- -end 10 0.1
- -workdir optim_work
- -exelog optim.exelog

Sample.map is specified to -out option for an output of the rank map file. Sample.param is specified to -param option for OPTIM parameter file. The termination condition of the optimp command is specified to -end option. "-end 1 0.01" means "When the percentage of the evaluation value will not improve 0.01(1%) or more in one minute, end the execution".

Made file and directory are as follows.

- sample.map
- optim.exelog
- Optim_workd/(directory)

Sample.map is a rank map file that is output based on the execution result of OPTIM. It is possible to specify it for a value of the rank-map-hostfile parameter in the job script. Sample.map records the coordinates of the node to which each rank is disposed, so do not edit it, please.

The execution log of optimp command is output to optim.exelog. Optim_work is a work directory that stored the temporary files of optimp command.

As for the optimp command running, the progress report of processing is output. RESULT SCORE and TARGET SCORE of [Example of executing the optimp command](#) indicates the evaluation value of the rank map. The evaluation value is shown by the following expressions, and it is expected that the smaller this value, the shorter the communication processing time.

$$eval (map) = \sum_{i,j \in \text{All ranks}} (hop_{ij} \times size_{ij}) \times \max link$$

Hop_{ij} shows the number of hops between rank i and j , and $size_{ij}$ shows the transfer size between rank i and j . When a total byte number that flows between certain nodes is assumed to be $link$, the maximum of $link$ in all links is assumed to be \max_{link} .

TARGET SCORE is an evaluation value when the rank is arranged on the network in order of X, Y, and Z without doing the rank topology optimization. RESULT SCORE is an evaluation value when the rank is arranged based on the execution result of OPTIM. Sum transffric shows $\sum (hop_{ij} \times size_{ij})$, Sum hop shows $\sum hop_{i,j}$ and Max link shows \max_{link} .

It is the following that pulled out a part of the status of the rank topology optimization processing by OPTIM in [Example of executing the optimp command](#).

Example of the status of MPI rank location optimization by OPTIM

```
0 1.00 * 3.3461e+08(1.1216e+09) 499418(797160) 16818(11704) 670(1407)
10 0.76 * 2.54376e+08(1.1216e+09) 451822(797160) 16650(11704) 563(1407)
20 0.71 * 2.37729e+08(1.1216e+09) 434606(797160) 16620(11704) 547(1407)
...
180 0.64 0.01 2.14085e+08(1.1216e+09) 414894(797160) 16416(11704) 516(1407)
```

Output format and output item of the status are shown as follows.

Figure 4.2 Output format of the status of MPI rank location optimization by OPTIM

```
@time @rate @rate_end @score(@target_score) /
@sumtraffic(target_sumtraffic) @sumhop(@target_sumhop) @maxlink(target_maxlink)
```

Table 4.3 Output item of the status of MPI rank location optimization by OPTIM

| Output item | Meaning |
|--------------------|---|
| @time | Execution time of optimp command (second) |
| @rate | (present evaluation value)/(evaluation value in initial rank map) |
| @rate_end | Improvement rate of evaluation value in time specified with -end option |
| @score | Evaluation value of present rank map |
| @target_score | Value of TARGET SCORE |
| @sumtraffic | Value of $\Sigma(hop \times size)$ in present evaluation value |
| @target_sumtraffic | Value of $\Sigma(hop \times size)$ in TARGET SCORE |
| @sumhop | Value of Σhop in present evaluation value |
| @target_sumhop | Value of Σhop in TARGET SCORE |
| @maxlink | Value of \max_{link} in present evaluation value |
| @target_maxlink | Value of \max_{link} in TARGET SCORE |

In second line of [Example of the status of MPI rank location optimization by OPTIM](#), "10 0.76 * 2.54376e+08(1.1216e+09) 451822(797160) 16650(11704) 563(1407)" is a mianing of "When the ten seconds have passed since it begins to have executed it, the evaluation value is 2.54376e+08, the value of $\Sigma(hop \times size)$ is 451822, the value of Σhop is 16650, the value of \max_{link} is 563, and a present evaluation value is 76% of the evaluation value in the initial rank map". @rate_end is * because it will not reach the one minute specified with -end option.

Chapter 5 Glossary

Rank

Indicates the rank of the MPI application.

Node

Indicates the calculation node from which the rank is executed.

Location of MPI rank

Indicates the rank and node from which that rank is executed in the MPI application.

MPI rank location optimization

The work to decide the MPI rank location so that the communication time is minimized.

Communication pattern

Indicates what communication each rank does in the MPI application.

Concretely, it is the correspondence rank of each rank and the communication size.

Network configuration

Indicates the network topology and the number of nodes of the computing environment that executes the MPI application

Communication pattern file

It is a file where the communication pattern was recorded.

OPTIM parameter file

It is a file where the parameter to execute optimization rank map tool OPTIM was recorded.

OPTIM initial rank map file

It is the file where the initial rank map to execute optimization rank map tool OPTIM was recorded.

Rank map file

It is a file for which the locations of MPI ranks are specified.

Appendix A Considerations for Using RMATT

This chapter explains the considerations for using RMATT.

A.1 CMPL

Trace data file directory

Locate all trace data files for all ranks on a directory specified with the "-vt" parameter. The specified trace data files are not-compressed files. If invalid trace data files are specified, they cannot be guaranteed to process safely.

The cmplx command reads files with the file name extension ".events" as trace data files.

Trace data file

CMPL processes do not read local trace data files, but trace data files unified by the vtunifypx command.

CMPL does not support multi-thread applications.

Only non-compressed trace data files can be read by CMPL. To use non-compressed files, specify "no" for the VT_COMPRESSION environment variable, or specify "-nocompress" as the parameter for the vtunifypx command.

Refer to the "Profiler User's Guide" for information on the Tracer and trace data files.

Communication pattern file

Do not edit the output communication pattern file. If an invalid communication pattern file is used, it cannot be guaranteed to process safely. It is possible to change the name and the path of the communication pattern file.

Output file

If a file created by the cmplx command already exists, the cmplx command overwrites the file.

A.2 BISEM

Communication pattern file

In the -ptn option, please specify the communication pattern file that the cmplx command output. When it violates this, operation is not guaranteed.

The path to the communication pattern file is specified in the OPTIM parameter file that made after the bisempx command is executed. So the communication pattern file must not move file or change its file name after executing the bisempx command. When it violates this, operation is not guaranteed.

Number of ranks and number of nodes

RMATT arranges one rank a node. Please adjust the number of nodes specified by the -nw option to the same value as the number of ranks specified with the communication pattern file. When it violates this, operation is not guaranteed.

It is recommended that the number of nodes of each axis be the power of two. The effect of the tuning rises when it is the power of two.

OPTIM parameter file

Please do not edit the output OPTIM parameter file. When it is edited, operation is not guaranteed.

The change of the file name and the movement of the file are possible.

OPTIM initial rank map file

The path to the OPTIM initial rank map file is specified for the OPTIM parameter file made after the bisempx command is executed. Therefore, the OPTIM initial rank map file must not change its file name or move it. And, please do not edit it. When it violates these, operation is not guaranteed.

Output file and output directory

When the the same name as the file and the directory that bisempx output already exists, they are overwritten.

A.3 OPTIM

OPTIM parameter file

In the -param option, please specify the OPTIM parameter file that bisempx command output. When it violates this, operation is not guaranteed.

In the -ptn option, please specify the communication pattern file that the cmplx command output. When it violates this, operation is not guaranteed.

OPTIM initial rank map file

In the OPTIM parameter file specified with the -param option, the path to the OPTIM initial rank map file is specified when executing the bisempx command. So neither move the OPTIM initial rank map file nor change its file name. When it violates this, operation is not guaranteed.

Termination condition

Please specify $0 < \text{min} < 2^{31}$ and $0.0 < \text{rate} < 1.0$ respectively for the value of min and rate specified by the -end option. When it violates this, operation is not guaranteed.

Output file and output directory

When the the same name as the file and the directory that optimpx output already exists, they are overwritten.