

Intel® MPI Library for Windows* OS

Reference Manual

Copyright © 2003–2011 Intel Corporation

All Rights Reserved

Document Number: 315399-011

Revision: 4.0 Update 3

World Wide Web: <http://www.intel.com>

Contents

1	About this Document	7
1.1	Intended Audience	7
1.2	Using Doc Type Field	7
1.3	Conventions and Symbols.....	8
1.4	Related Information.....	8
2	Command Reference	9
2.1	Compiler Commands.....	9
2.1.1	Compiler Command Options.....	9
2.1.2	Configuration Files.....	11
2.1.3	Profiles	12
2.1.4	Environment Variables	12
2.2	Job Startup Command	13
2.2.1	Global Options	14
2.2.2	Local Options.....	17
2.2.3	Environment Variables	18
2.2.4	Integration with Microsoft* Job Scheduler.....	22
2.2.5	Integration with PBS Pro* Job Scheduler	22
2.3	Simple Multi-Purpose Daemon*	23
2.4	Processor Information Utility.....	24
3	User Authorization.....	28
3.1	Overview	28
3.2	Installation.....	28
3.2.1	Active Directory* Setup.....	28
3.3	Environment Variables	29
4	Tuning Reference	30
4.1	Automatic Tuning Utility	30
4.1.1	Cluster-specific Tuning.....	34
4.1.2	Application-specific Tuning.....	34
4.1.3	Tuning Utility Output	35
4.2	Process Pinning.....	35
4.2.1	Process Identification.....	35
4.2.2	Environment Variables	36
4.2.3	Interoperability with OpenMP*	41
4.3	Fabrics Control.....	50
4.3.1	Communication Fabrics Control	50
4.3.2	Shared Memory Control.....	55
4.3.3	DAPL-capable Network Fabrics Control	62
4.3.4	TCP-capable Network Fabrics Control	71
4.4	Dynamic Process Support	72
4.5	Collective Operation Control.....	73
4.5.1	I_MPI_ADJUST Family.....	73
4.5.2	I_MPI_MSG Family	76
4.6	Compatibility Control	80
4.7	Secure Loading of Dynamic Link Libraries*	81
5	Statistics Gathering Mode.....	83
5.1	Native Statistics Format	83
5.2	IPM Statistics Format	88
5.2.1	Region Control	88
6	ILP64 Support	96
6.1	Using ILP64.....	96
6.2	Known Issues and Limitations.....	96
7	Unified Memory Management	97
8	Graphical Utilities	98

9	Glossary	101
10	Index	102

Disclaimer and Legal Notices

INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER, AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT. UNLESS OTHERWISE AGREED IN WRITING BY INTEL, THE INTEL PRODUCTS ARE NOT DESIGNED NOR INTENDED FOR ANY APPLICATION IN WHICH THE FAILURE OF THE INTEL PRODUCT COULD CREATE A SITUATION WHERE PERSONAL INJURY OR DEATH MAY OCCUR.

Intel may make changes to specifications and product descriptions at any time, without notice. Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined." Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. The information here is subject to change without notice. Do not finalize a design with this information.

The products described in this document may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request. Contact your local Intel sales office or your distributor to obtain the latest specifications and before placing your product order.

Copies of documents which have an order number and are referenced in this document, or other Intel literature, may be obtained by calling 1-800-548-4725, or go to: <http://www.intel.com/design/literature.htm>

Intel processor numbers are not a measure of performance. Processor numbers differentiate features within each processor family, not across different processor families. Go to: http://www.intel.com/products/processor_number/

MPEG-1, MPEG-2, MPEG-4, H.261, H.263, H.264, MP3, DV, VC-1, MJPEG, AC3, AAC, G.711, G.722, G.722.1, G.722.2, AMRWB, Extended AMRWB (AMRWB+), G.167, G.168, G.169, G.723.1, G.726, G.728, G.729, G.729.1, GSM AMR, GSM FR are international standards promoted by ISO, IEC, ITU, ETSI, 3GPP and other organizations. Implementations of these standards, or the standard enabled platforms may require licenses from various entities, including Intel Corporation.

BlueMoon, BunnyPeople, Celeron, Celeron Inside, Centrino, Centrino Inside, Cilk, Core Inside, E-GOLD, i960, Intel, the Intel logo, Intel AppUp, Intel Atom, Intel Atom Inside, Intel Core, Intel Inside, Intel Insider, the Intel Inside logo, Intel NetBurst, Intel NetMerge, Intel NetStructure, Intel SingleDriver, Intel SpeedStep, Intel Sponsors of Tomorrow., the Intel Sponsors of Tomorrow. logo, Intel StrataFlash, Intel vPro, Intel XScale, InTru, the InTru logo, the InTru Inside logo, InTru soundmark, Itanium, Itanium Inside, MCS, MMX, Moblin, Pentium, Pentium Inside, Puma, skool, the skool logo, SMARTi, Sound Mark, The Creators Project, The Journey Inside, Thunderbolt, Ultrabook, vPro Inside, VTune, Xeon, Xeon Inside, X-GOLD, XMM, X-PMU and XPOSYS are trademarks of Intel Corporation in the U.S. and other countries.

* Other names and brands may be claimed as the property of others.

Microsoft, Windows, Visual Studio, Visual C++, and the Windows logo are trademarks, or registered trademarks of Microsoft Corporation in the United States and/or other countries.

Java is a registered trademark of Oracle and/or its affiliates.

Copyright (C) [2003]–[2011], Intel Corporation. All rights reserved.

Optimization Notice

Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice.

Notice revision #20110804

Revision History

Document Number	Revision Number	Description	Revision Date
315399-001	3.1 Beta	Initial release	/07/10/2007
315399-002	3.1	New variables were added. New section "Processor Information Utility" was added. Updated and reviewed for style	/10/02/2007
315399-003	3.1 build 038	Local options were added. Section "GUI Utilities" was updated. Sections "Automatic Tuning Utility", "Integration with Microsoft* job scheduler", "Process Identification", and "Interoperability with OpenMP*" were added	/03/13/2008
315399-004	3.2	Sections "Index" and "Integration with PBS Pro* job scheduler" were added Global options were added Sections "Statistic Gathering Mode", "Automatic Tuning Utility", and "Process Pinning" were updated	/09/04/2008
315399-005	3.2 Update 1	Sections "ILP64 Support", " User authorization" were added	/03/04/2009
315399-006	3.2 Update 2	Section "Processor Information Utility" was updated	/08/04/2009
315399-007	4.0 Beta	Sections "Processor Information Utility", "User Authorization", "Automatic Tuning Utility", "Process Pinning", and "Statistics Gathering Mode" were updated. Sections "Compatibility Control", "Fabrics Control", and "Glossary" were added. Sections "Device Control", "RDMA and RDSSM device control", "Socket Device Control", and "Graphical Utilities" were deleted.	/11/03/2009
315399-008	4.0	Section "Fabrics Control" and "Installation" were updated. Sections "Dynamic Process Support" and "Graphical Utilities" were added.	/02/16/2009
315399-009	4.0 Update 1	Section "Environment Variables", "Process Pinning", " Processor Information Utility ", "Shared Memory Control", "TCP-capable Network Fabrics Control", and "Disclaimer and Legal Notices" were updated.	/09/23/2010
315399-010	4.0 Update 2	Section "Disclaimer and Legal Notices" was updated.	/04/11/2011
315399-011	4.0 Update 3	Sections, "Job Startup Command", "Processor Information Utility", "Automatic Tuning Utility", "Fabrics Control", "Collective Operation Control ", and "Statistics Gathering Mode" were updated. Sections, "IPM Statistics Format" and "Secure Loading of Dynamic Link Libraries" were added.	/08/31/2011

1 About this Document

This *Reference Manual* provides you with a complete command and tuning reference for the Intel® MPI Library.

The Intel® MPI Library is a multi-fabric message passing library that implements the Message Passing Interface, v2 (MPI-2) specification. It provides a standard library across Intel® platforms that enable adoption of MPI -2 functions as their needs dictate.

The Intel® MPI Library enables developers to change or upgrade processors and interconnects as new technology becomes available without changes to the software or to the operating environment.

The library is provided in the following kits:

- *The Intel® MPI Library Runtime Environment* has the tools you need to run programs including, simple multi-purpose daemon (SMPD) and supporting utilities, dynamic (.dll) libraries and documentation.
- *The Intel® MPI Library Development Kit* includes all of the Runtime Environment components plus compilation tools, including compiler commands such as `mpicc`, include files and modules, debug libraries, trace libraries, and test codes.

1.1 Intended Audience

This *Reference Manual* helps an experienced user understand full functionality of the Intel® MPI Library.

1.2 Using Doc Type Field

This *Reference Manual* contains the following sections

Table 1.2-1 Document Organization

Section	Description
Section 1 About this Document	Section 1 introduces this document
Section 2 Command Reference	Section 2 describes options and variables for compiler commands, job startup commands and MPD daemon commands as well
Section 3 User Authorization	Section 3 describes different user authorizations methods
Section 4 Tuning Reference	Section 4 describes environment variables used to influence program behavior and performance at run time
Section 5 Statistics Gathering Mode	Section 5 describes how to obtain statistics of MPI communication operations
Section 6 ILP64 Support	Section 6 describes support provided for the ILP64 programming model
Section 7 Unified Memory Management	Section 7 describes the unified Intel memory management subsystem (<code>i_malloc</code>)
Section 8 Graphical Utilities	Section 8 describes graphical user interface (GUI) utilities distributed with the Intel® MPI Library

Section 9 Index	Section 9 references options and environment variables names
-----------------	--

1.3 Conventions and Symbols

The following conventions are used in this document.

Table 1.3-1 Conventions and Symbols used in this Document

<i>This type style</i>	Document or product names
This type style	Hyperlinks
<code>This type style</code>	Commands, arguments, options, file names
<code>THIS_TYPE_STYLE</code>	Environment variables
<code><this type style></code>	Placeholders for actual values
<code>[items]</code>	Optional items
<code>{ item item }</code>	Selectable items separated by vertical bar(s)
(SDK only)	For Software Development Kit (SDK) users only

1.4 Related Information

The following related documents that might be useful to the user:

[Product Web Site](#)

[Intel® MPI Library Support](#)

[Intel® Cluster Tools Products](#)

[Intel® Software Development Products](#)

2 Command Reference

2.1 Compiler Commands

(SDK only)

The following table lists available MPI compiler commands and the underlying compilers, compiler families, languages, and application binary interfaces (ABIs) that they support.

Table 2.1-1 The Intel® MPI Library Compiler Drivers

Compiler command	Underlying Compiler	Supported Language(s)	Supported ABI (s)
Common Compilers			
<code>mpicc.bat</code>	<code>cl.exe</code>	C	32/64 bit
<code>mpicxx.bat</code>	<code>cl.exe</code>	C++	32/64 bit
<code>mpifc.bat</code>	<code>ifort.exe</code>	Fortran 77/Fortran 95	32/64 bit
Microsoft* Visual C++* Compilers			
<code>mpicl.bat</code>	<code>cl.exe</code>	C/C++	32/64 bit
Intel® Fortran, C++ Compilers Versions 10.0, 10.1, 11.0 and Higher			
<code>mpiicc.bat</code>	<code>icl.exe</code>	C	32/64 bit
<code>mpiicpc.bat</code>	<code>icl.exe</code>	C++	32/64 bit
<code>mpiifort.bat</code>	<code>ifort.exe</code>	Fortran 77/Fortran 95	32/64 bit

NOTE:

- Compiler commands are available only in the Intel® MPI Library Development Kit.
- Compiler commands are in the `<installdir>\<arch>\bin` directory. For the Intel® 64 architecture in 64-bit-enabled compiler commands are in the `<installdir>\em64t\bin` directory and 32-bit compiler commands are in the `<installdir>\ia32\bin` directory.
- Ensure that the corresponding underlying compilers (32-bit or 64-bit, as appropriate) are already in your `PATH`.
- To port existing, MPI-enabled applications to the Intel MPI Library, recompile all sources.
- To display mini-help of a compiler command, execute it without any parameters.

2.1.1 Compiler Command Options

`-mt_mpi`

Use this option to link the thread safe version of the Intel® MPI Library at the following levels: `MPI_THREAD_FUNNELED`, `MPI_THREAD_SERIALIZED`, or `MPI_THREAD_MULTIPLE`. The `MPI_THREAD_FUNNELED` level is provided by default by the thread safe version of the Intel® MPI Library.

NOTE: If you specify either the `-Qopenmp` or the `-Qparallel` options for the Intel® C/C++ Compiler, the thread safe version of the library is used.

NOTE: If you specify one of the following options for the Intel® Fortran Compiler, the thread safe version of the library is used:

- `-Qopenmp`
- `-Qparallel`
- `-threads`
- `-reentrancy`
- `-reentrancy:threaded`

`-profile=<profile_name>`

Use this option to specify an MPI profiling library. The profiling library is selected using one of the following methods:

- Through the configuration file `<profile_name>.conf` located in the `<installdir>\<arch>\etc` directory. See [Profiles](#) for details.
- In the absence of the respective configuration file, by linking the library `lib<profile_name>.lib` located in the same directory as the Intel® MPI Library.

`-t or -trace`

Use the `-t` or `-trace` option to link the resulting executable against the Intel® Trace Collector library.

Use the `-t=log` or `-trace=log` option to link the resulting executable against the logging Intel® MPI Library and the Intel® Trace Collector libraries. The logging libraries trace internal Intel® MPI Library states in addition to the usual MPI function calls.

Include the installation path of the Intel® Trace Collector in the `VT_ROOT` environment variable to use this option.

`-check_mpi`

Use this option to link the resulting executable against the Intel® Trace Collector correctness checking library. Include the installation path of the Intel® Trace Collector into the `VT_ROOT` environment variable to use this option.

`-ilp64`

Use this option to enable ILP64 support. All integer arguments of the Intel MPI Library are treated as 64-bits values in this case.

NOTE: If you specify the `-i8` option for the Intel® Fortran Compiler, you still have to use the ILP64 option for linkage. See [ILP64 Support](#) for details.

`/Zi or /Z7 or /ZI/debug`

Use these options to compile a program in debug mode and link the resulting executable against the debugging version of the Intel® MPI Library. See [Environment Variables, I_MPI_DEBUG](#) for information on how to use additional debugging features with the `/Zi`, `/Z7`, `/ZI` or `debug` builds.

NOTE: The `/ZI` option is only valid for C/C++ compiler.

-O

Use this option to enable optimization.

Setting this option triggers a call to the `libirc` library. Many of those library routines are more highly optimized for Intel microprocessors than for non-Intel microprocessors.

-echo

Use this option to display everything that the command script does.

-show

Use this option to learn how the underlying compiler is invoked. For example, use the following command to see the required compiler flags and options:

```
> mpiicc.bat -show -c test.c
```

Use the following command to see the required linker flags, options, and libraries:

```
> mpiicc.bat -show -o a.exe test.obj
```

This is particularly useful for determining the command line for a complex build procedure that directly uses the underlying compilers.

-show_env

Use this option to see the environment settings in effect when the underlying compiler is invoked.

-[cc, cxx, fc]=<compiler>

Use this option to select the underlying compiler.

For example, use the following command to select the Intel® C++ Compiler:

```
> mpiicc.bat -cc=icl.exe -c test.c
```

For this to work, `icl.exe` should be in your path. Alternatively, you can specify the full path to the compiler.

NOTE: This option works only with the `mpiicc.bat` and the `mpifc.bat` commands.

-V

Use this option to print the compiler driver script version.

2.1.2 Configuration Files

You can create compiler configuration files using the following file naming convention:

```
<installdir>\<arch>\etc\mpi<compiler>-<name>.conf
```

where:

`<compiler>` = {`cc`,`fc`}, depending on the language being compiled

`<name>` = name of underlying compiler

Before compiling or linking to enable changes to the environment on a per compiler command basis, source this file, if it exists.

2.1.3 Profiles

You can select a profile library through the `-profile` option of the Intel® MPI Library compiler drivers. You can also create your own profile as

```
<installdir>\<arch>\etc\<profile_name>.conf
```

The following environment variables can be defined there:

`PROFILE_PRELIB` - libraries (and paths) to include before the Intel® MPI Library

`PROFILE_POSTLIB` - libraries to include after the Intel® MPI Library

`PROFILE_INCPATHS` - C preprocessor arguments for any include files

For instance, create a file `<installdir>\<arch>\etc\myprof.conf` with the following lines:

```
SET PROFILE_PRELIB=<path_to_myprof>\lib\myprof.lib
```

```
SET PROFILE_INCPATHS=-I"<paths_to_myprof>\include"
```

Use the command-line argument `-profile=myprof` for the relevant compile driver to select this new profile.

2.1.4 Environment Variables

I_MPI_{CC, CXX, FC, F77, F90}_PROFILE

Specify a default profiling library.

Syntax

```
I_MPI_{CC, CXX, FC, F77, F90}_PROFILE=<profile_name>
```

Arguments

<code><profile_name></code>	Specify a default profiling library
-----------------------------------	-------------------------------------

Description

Set this environment variable to select a specific MPI profiling library to be used by default. This configuration has the same effect as if `-profile=<profile_name>` were used as an argument to `mpiicc.bat` or another Intel® MPI Library compiler driver.

I_MPI_{CC, CXX, FC, F77, F90}

(MPICH_{CC, CXX, FC, F77, F90})

Set the path/name of the underlying compiler to be used.

Syntax

```
I_MPI_{CC, CXX, FC, F77, F90}=<compiler>
```

Arguments

<code><compiler></code>	Specify the full path/name of compiler to be used
-------------------------------	---

Description

Set this environment variable to select a specific compiler to be used. Specify the full path to the compiler if it is not located in the search path.

NOTE: Some compilers may require additional command line options.

I_MPI_ROOT

Set the Intel® MPI Library installation directory path.

Syntax

```
I_MPI_ROOT=<path>
```

Arguments

<code><path></code>	Specify the installation directory of the Intel® MPI Library
---------------------------	--

Description

Set this environment variable to specify the installation directory of the Intel® MPI Library.

VT_ROOT

Set the Intel® Trace Collector installation directory path.

Syntax

```
VT_ROOT=<path>
```

Arguments

<code><path></code>	Specify the installation directory of the Intel® Trace Collector
---------------------------	--

Description

Set this environment variable to specify the installation directory of the Intel® Trace Collector.

I_MPI_COMPILER_CONFIG_DIR

Set the location of the compiler configuration files.

Syntax

```
I_MPI_COMPILER_CONFIG_DIR=<path>
```

Arguments

<code><path></code>	Specify the location of the compiler configuration files. The default is <code><installdir>\<arch>\etc</code>
---------------------------	---

Description

Set this environment variable to change the default location of the compiler configuration files.

2.2 Job Startup Command

mpiexec

Syntax

```
mpiexec <g-options> <l-options> <executable>
```

or

```
mpiexec <g-options> <l-options> <executable> : \  
<l-options> <executable>
```

or

```
mpiexec -configfile <file>
```

Arguments

<code><g-options></code>	Global options that apply to all MPI processes
<code><l-options></code>	Local options that apply to a single arg-set
<code><executable></code>	<code>.\a.exe</code> or <code>path\name</code> of the executable file
<code><file></code>	File with command-line options

Description

By using the first command-line syntax, you can start all MPI processes of the `<executable>` with the single arg-set. For example, the following command executes `a.out` over the specified `<# of processes>`:

```
> mpiexec -n <# of processes> a.exe
```

By using the second command-line syntax, you can start several MPI programs or the same MPI program with different arg-sets. For example, the following command would run each given executable on a different host:

```
> mpiexec -n 2 -host host1 a.exe : \
      -n 2 -host host2 b.exe
```

In the third command-line syntax, read the command line from specified `<file>`. For a command with a single arg-set, the entire command should be specified on a single line in `<file>`. For a command with multiple arg-sets, each arg-set should be specified on a single, separate line in `<file>`. Global options should always appear at the beginning of the first line in `<file>`.

Simple multi-purpose daemon (SMPD) service must already be running in order for `mpiexec` to succeed.

NOTE: If path to executable is not in the `PATH` on all nodes in the cluster, specify `<executable>` as `<path>\a.exe` rather than `a.exe`.

2.2.1 Global Options

`-machinefile <machine file>`

Use this option to control the process placement through the `<machine file>`. The number of processes to start is controlled by the option `-n` as usual.

A machine file is a list of fully qualified or short host names, one name per line. Blank lines and lines that start with '#' as the first character are ignored.

By repeating a host name you will place additional processes on this host. You can also use the following format to avoid repetition of the same host name: `<host name>:<number of processes>`. For example, the following machine file:

```
host1
host1
host2
host2
host3
```

is equivalent to:

```
host1:2
```

```
host2:2
```

```
host3
```

It is also possible to specify the network interface used for communication for each node: `<host name>:<number of processes> [ifhn=<interface_host_name>]`.

-configfile <filename>

Use this option to specify the file `<filename>` that contains command-line options. Blank lines and lines that start with '#' as the first character are ignored. For example, a configuration file contains the following commands to run the executables `a.exe` and `b.exe` using the `rdssm` device over `host1` and `host2` respectively:

```
-host host1 -env I_MPI_DEBUG 2 -env I_MPI_DEVICE rdssm -n 2 a.exe
-host host2 -env I_MPI_DEBUG 2 -env I_MPI_DEVICE rdssm -n 2 b.exe
```

To launch a MPI application according to the parameters above, use:

```
> mpiexec -configfile <filename>
```

NOTE: This option may only be used alone. It terminates parsing of the `mpiexec` command line.

-g <l-option>

Use this option to apply the named local option `<l-option>` globally. See [Local Options](#) for a list of all local options. During the application startup, the default value is the `-genvuser` option. The options `-genvnone`, `-genvuser`, `-genvall` have the lowest priority, `-genvlist`, `-genvexcl` have higher priority than the previous set. The `-genv` option has the highest priority. Local options have higher priority than the global options.

-l

Use this option to insert the MPI process rank at the beginning of all lines written to standard output.

-tune

Use this option to optimize the Intel® MPI Library performance using the data collected by the `mpitune` utility. If `<configuration_file>` is not mentioned, the best-fit tune options will be selected for the given configurations. Otherwise the given configuration file will be used.

For the Intel® 64 architecture in 64-bit mode, the default location of the configuration files is in the `<installdir>\em64t\etc` directory. For 32-bit mode the files are located in the `<installdir>\ia32\etc` directory. Set the `I_MPI_TUNER_DATA_DIR` environment variable to override the default location.

See [Automatic Tuning Utility](#) for more details.

-p <port> or -port <port>

Use this option to specify the SMPD port `mpiexec` should connect to. This option can be useful if SMPD is using a non-default port number.

-hosts n <host1> <num_proc1> <host2> <num_proc2> ... <hostn> <num_procn>

Use this option to specify a particular hosts list and a number of processes on each of the MPI processes in the current arg-set are to be run. For example, the following command line will run the

executable `a.exe` on the hosts `host1` and `host2`. Two processes will be run on `host1` and one process on `host 2` respectively:

```
> mpiexec -hosts 2 host1 2 host2 1 a.exe
```

-logon

Use this option to prompt for your account name and password.

-delegate

Use this option to enable the domain-based authorization with the delegation ability.

-impersonate

Use this option to enable the limited domain-based authorization. You will not be able to open files on remote machines or access mapped network drives.

-pwdfile <filename>

Use this option to read the account and password from the file specified. Put account on the first line and password on the second one.

-nopopup_debug

Use this option to disable the system popup dialog if the process crashes.

-exitcodes

Use this option to print the process exit codes when each process exits.

-verbose

Use this option to redirect the `smpd` output to `stdout`.

-localroot

Use this option to launch the root process directly from `mpiexec` if the host is local.

-register [-user n]

Use this option to encrypt the user name and password to the registry.

-remove [-user n]

Use this option to delete the encrypted credentials from the registry. If no user index is specified, all entries are removed.

-timeout <seconds>

Use this option to set timeout for the job.

-whoami

Use this option to print the current user name.

-h or -help or --help

Use this option to display the `mpiexec` help message.

2.2.2 Local Options

-n <# of processes> or -np <# of processes>

Use this option to set the number of MPI processes to run with the current arg-set.

-env <ENVVAR> <value>

Use this option to set the `<ENVVAR>` environment variable to specified `<value>` for all MPI processes in the current arg-set.

-envall

Use this option to propagate all environment variables in the current environment.

-envnone

Use this option to suppress propagation of any environment variables to the MPI processes in the current arg-set.

-envlist <list of env var names>

Use this option to pass a list of environment variables with their current values.

-envuser

Use this option to propagate all user environment variables to all MPI processes with the exception of the following environment variables:

```
%ALLUSERSPROFILE%, %APPDATA%, %CommonProgramFiles%, %CommonProgramFiles (x86)%, %COMPUTERNAME%, %HOMEDRIVE%, %HOMEPATH%, %NUMBER_OF_PROCESSORS%, %OS%, %PROCESSOR_ARCHITECTURE%, %PROCESSOR_IDENTIFIER%, %PROCESSOR_LEVEL%, %PROCESSOR_REVISION%, %ProfilePath%, %ProgramFiles%, %ProgramFiles (x86)%, %SystemDrive%, %SystemRoot%, %TEMP%, %TMP%, %USERDNSDOMAIN%, %USERDOMAIN%, %USERNAME%, %USERPROFILE%.
```

This is the default setting.

-envexcl <list of env var names>

Use this option to suppress the propagation of the listed environment variables to the MPI processes in the current arg-set.

-host <nodename>

Use this option to specify a particular `<nodename>` on which the MPI processes in the current arg-set are to be run. For example, the following will run the executable `a.exe` on host `host1` only:

```
> mpiexec -n 2 -host host1 a.exe
```

-path <directory>

Use this option to specify the path to `<executable>` that is to be run in the current arg-set. The separator is ;

-dir <directory> or -wdir <directory>

Use this option to specify the working directory in which *<executable>* is to be run in the current arg-set.

-map <drive:\\host\share>

Use this option to create network mapped drive on nodes before starting *<executable>*. Network drive will be automatically removed after the job completion.

-mapall

Use this option to request creation of all user created network mapped drives on nodes before starting *<executable>*. Network drives will be automatically removed after the job completion.

2.2.3 Environment Variables

I_MPI_DEBUG

Print out debugging information when an MPI program starts running.

Syntax

`I_MPI_DEBUG=<level>[, <flags>]`

Arguments

<code><level></code>	Indicate level of debug information provided
0	Output no debugging information. This is the default value
1	Output verbose error diagnostics
2	Confirm which <code>I_MPI_FABRICS</code> (<code>I_MPI_DEVICE</code>) was used
3	Output effective MPI rank, pid and node mapping table
4	Output process pinning information
5	Output Intel MPI-specific environment variables
> 5	Add extra levels of debug information

<code><flags></code>	Comma-separated list of debug flags
<code>pid</code>	Show process id for each debug message
<code>tid</code>	Show thread id for each debug message for multithreaded library
<code>time</code>	Show time for each debug message
<code>datetime</code>	Show time and date for each debug message
<code>host</code>	Show host name for each debug message
<code>level</code>	Show level for each debug message
<code>scope</code>	Show scope for each debug message
<code>line</code>	Show source line number for each debug message

<code>file</code>	Show source file name for each debug message
<code>nofunc</code>	Do not show routine name
<code>norank</code>	Do not show rank
<code>flock</code>	Synchronize debug output from different process or threads
<code>nobuf</code>	Do not use buffered I/O for debug output

Description

Set this environment variable to control the output of the debugging information.

You can specify the output file name for debug information by setting `I_MPI_DEBUG_OUTPUT` environment variable.

To simplify process identification, add the '+' or '-' sign in front of the numerical value for `I_MPI_DEBUG`. This setting produces debug output lines prefixed with the MPI process rank, a UNIX* OS process id, and a host name as defined at the process launch time. For example, the command:

```
> mpiexec -n <# of processes> -env I_MPI_DEBUG +2 a.exe
```

or

```
> mpiexec -n <# of processes> -env I_MPI_DEBUG +2,pid,host a.exe
```

may produce the following output:

```
[rank#pid@hostname]Debug message
```

NOTE: Compiling with `mpiicc.bat /Zi`, `/Z7` or `/Z7` causes considerable amounts of additional debug information to be printed.

I_MPI_DEBUG_OUTPUT

Set output file name.

Syntax

```
I_MPI_DEBUG_OUTPUT =<arg>
```

Arguments

<code><arg></code>	String value
<code>stdout</code>	Output to stdout - default value
<code>stderr</code>	Output to stderr
<code><file_name></code>	Specify the output file name for debug information

Description

Set this environment variable if you want to split output of debug information from the output produced by an application. If you use format like '%r', '%p' or '%h', rank, pid or host name will be added to the file name accordingly.

I_MPI_PRINT_VERSION

Print library version information.

Syntax

```
I_MPI_PRINT_VERSION=<arg>
```

Arguments

<code><arg></code>	Binary indicator
<code>enable yes on 1</code>	Print library version information.
<code>disable no off 0</code>	No action. This is the default value.

Description

Set this environment variable to enable/disable printing of Intel® MPI library version information when an MPI application starts running.

I_MPI_NETMASK

Choose the network interface for MPI communication over sockets.

Syntax

`I_MPI_NETMASK=<arg>`

Arguments

<code><arg></code>	Define the network interface (string parameter)
<code><interface_mnemonic></code>	Mnemonic of the network interface: <code>ib</code> or <code>eth</code>
<code>ib</code>	Select IPoIB*
<code>eth</code>	Select Ethernet. This is the default value
<code><network_address></code>	Network address. The trailing zero bits imply netmask
<code><network_address/netmask></code>	Network address. The <code><netmask></code> value specifies the netmask length
<code><list of interfaces></code>	A colon separated list of network addresses or interface mnemonics

Description

Set this environment variable to choose the network interface for MPI communication over sockets in the `sock` and `ssm` communication modes. If you specify a list of interfaces, the first available interface on the node will be used for communication.

Examples

- Use the following setting to select the IP over InfiniBand* (IPoIB) fabric:
`I_MPI_NETMASK=ib`
`I_MPI_NETMASK=eth`
- Use the following setting to select particular network for socket communications. This setting implies the `255.255.0.0` netmask:
`I_MPI_NETMASK=192.169.0.0`
- Use the following setting to select a particular network for socket communications with netmask set explicitly:
`I_MPI_NETMASK=192.169.0.0/24`
- Use the following setting to select the specified network interfaces for socket communications:
`I_MPI_NETMASK=192.169.0.5/24:ib0:192.169.0.0`

NOTE: If the library cannot find any suitable interface by the given value of `I_MPI_NETMASK`, the value will be used as a substring to search in the network adapter's description field. And if the substring is found in the description, this network interface will be used for socket

communications. For example, if `I_MPI_NETMASK=myri` and the description field contains something like Myri-10G adapter, this interface will be chosen.

I_MPI_JOB_TIMEOUT

Set the `mpiexec` timeout.

Syntax

`I_MPI_JOB_TIMEOUT=<timeout>`

Arguments

<code><timeout></code>	Define <code>mpiexec</code> timeout period in seconds
<code><n> >= 0</code>	The default timeout value is zero, which means no timeout

Description

Set this environment variable to make `mpiexec` terminate the job in `<timeout>` seconds after its launch. The `<timeout>` value should be greater than zero. Otherwise, the environment variable setting is ignored.

NOTE: Set the `I_MPI_JOB_TIMEOUT` environment variable in the shell environment before executing the `mpiexec` command. Do not use the `-genv` or `-env` options for setting the `<timeout>` value. Those options are used only for passing environment variables to the MPI process environment.

I_MPI_SMPD_VERSION_CHECK

Set this environment variable to enable strong SMPD version check. This configuration is valid only for Windows* OS.

Syntax

`I_MPI_SMPD_VERSION_CHECK=<arg>`

Arguments

<code><arg></code>	Binary indicator
<code>enable, yes, on, 1</code>	Check the version and fail in case of mismatch. This is the default value
<code>disable, no, off, 0</code>	Only print a warning message in case of a version mismatch and continue working

Description

Set this environment variable to control SMPD version check. The Intel® MPI Library terminates application if the versions mismatch is found. To disable SMPD version check set the `I_MPI_SMPD_VERSION_CHECK` environment variable to `disable`.

I_MPI_DAT_LIBRARY

Select a particular DAT library to be used.

Syntax

`I_MPI_DAT_LIBRARY=<library>`

Arguments

<code><library></code>	Specify the exact library to be used instead of the default <code>dat.dll</code>
------------------------------	--

Description

Set this environment variable to select a specific DAT library to be used. Specify the full path to the DAT library if it is not located in the dynamic loader search path.

NOTE: Use this environment variable only if you are going to utilize a DAPL* provider.

I_MPI_TUNER_DATA_DIR

Set an alternate path to the directory with the tuning configuration files.

Syntax

`I_MPI_TUNER_DATA_DIR=<path>`

Arguments

<code><path></code>	Specify the automatic tuning utility output directory. The default value is <code><mpiinstalldir>\em64t\etc</code> or <code><mpiinstalldir>\ia32\etc</code>
---------------------------	---

Description

Set this environment variable to specify an alternate location of the tuning configuration files.

2.2.4 Integration with Microsoft* Job Scheduler

The Intel® MPI Library job startup command `mpiexec` can be called out of Microsoft* Compute Cluster Pack 2003* job scheduler to execute MPI application. In this case the `mpiexec` command automatically inherits the host list, process count and the working directory allocated to the job.

Use the following command to submit MPI job:

```
job submit /numprocessors:4 /stdout:test.out mpiexec -delegate test.exe
```

Make sure `mpiexec` and dynamic libraries are available through `PATH`. The Intel MPI Library environment variables can be registered during the installation process.

2.2.5 Integration with PBS Pro* Job Scheduler

The Intel MPI Library job startup command `mpiexec` can be called out of PBS Pro* job scheduler to execute MPI application. In this case the `mpiexec` command automatically inherits the host list, process count allocated to the job if they were not specified manually by the user. The `mpiexec` reads `%PBS_NODEFILE%` environment variable to count a number of processes and use it as `machinefile`.

Example:

Content of `job` script:

```
REM PBS -l nodes=4:ppn=2
REM PBS -l walltime=1:00:00

cd %PBS_O_WORKDIR%
mpiexec.exe test.exe
```

Use the following command to submit the job:

```
qsub -C "REM PBS" job
```

`mpiexec` will run two processes on each of four nodes for this job.

2.3 Simple Multi-Purpose Daemon*

smpd

Simple multi-purpose daemon.

Syntax

```
smpd.exe [ -h ] [ --help ] [ -port <port> ] [ -d ] \
[ -install | -regserver ] [ -start ] [ -stop ] \
[ -shutdown <hostname> ] [ -status <hostname> ] \
[ -restart <hostname> ] [ -anyport ] [ -hosts ] [ -sethosts ] \
[ -set <option_name> <option_value> ] [ -get <option_name> ] \
[ -tracemon <logfile_name> [ <hostA> <hostB> ... ] ] \
[ -traceoff [ <hostA> <hostB> ... ] ] \
[ -remove | -unregister | -uninstall ] [ -register_spn ] \
[ -remove_spn ] [ -V ] [ -version ]
```

Arguments

-h --help	Display a help message
-p <port> -port <port>	Specify the port that <code>smpd</code> is listening on
-d -debug	Start <code>smpd</code> in debug mode
-install -regserver	Install the <code>smpd</code> service
-start	Start the <code>smpd</code> service
-stop	Stop the <code>smpd</code> service
-shutdown <hostname>	Shutdown <code>smpd</code> on specified <hostname>
-status <hostname>	Get the <code>smpd</code> status on specified <hostname>
-restart <hostname>	Restart <code>smpd</code> on specified <hostname>
-anyport	Use any port for listening
-hosts	Get the <code>smpd</code> ring list
-sethosts	Set the <code>smpd</code> ring from specified hosts. This settings impact all users
-set <option_name> <option_value>	Register the <option_name> key to the HKEY_LOCAL_MACHINE registry key
-get <option_name>	Get the value of the registered <option_name> key from the HKEY_LOCAL_MACHINE registry key
-tracemon <logfile_name> <hostA> <hostB> ...	Restart <code>smpd</code> and store the output into provided <logfile_name>
-traceoff <hostA> <hostB>	Restart <code>smpd</code> without logging the output
-remove -unregserver	Remove <code>smpd</code> service

<code>-uninstall</code>	
<code>-register_spn</code>	Register Service Principle Name (SPN) in the Windows* domain for the cluster node on which this command is executed
<code>-remove_spn</code>	Remove SPN from the Windows* domain for the cluster node on which this command is executed
<code>-V</code>	Get the Intel® MPI Library version info
<code>-version</code>	Display the <code>smpd</code> version information

Description

Simple Multipurpose Daemon* (SMPD) is the Intel® MPI Library process management system for starting parallel jobs. Before running a job, start `smpd` service on each host and connect them into a ring.

Use the `smpd.exe` command to install, uninstall, start or stop SMPD service.

Examples:

1. Use the following command to install SMPD service:

```
> smpd.exe -install
```

NOTE: This command must be run by a user with administrator privileges. After that all users will be able to launch MPI jobs using `mpiexec`.

2. Use the following command to start the SMPD service in debug mode:

```
> smpd.exe -d
```

2.4 Processor Information Utility

cpuinfo

The `cpuinfo` utility provides processor architecture information.

Syntax

```
cpuinfo [[-]<options>]]
```

Arguments

<code><options></code>	Sequence of one-letter options. Each option is responsible for the specific part of printed data
<code>g</code>	General information about single cluster node
<code>i</code>	Logical processors identification

d	Node decomposition table
c	Cache sharing by logical processors
s	Microprocessor signature hexadecimal fields (Intel platform notation)
f	Microprocessor feature flags (Intel platform notation)
A	Replacer of all available options union
gidc	Default sequence
?	Utility usage info

Description

The `cpuinfo` utility prints out the processor architecture information that can be used to define suitable process pinning settings. The output consists of a number of tables. Each table corresponds to one of the single options listed in the arguments table. See the following examples.

- **General information about single node** shows the processor product name, number of packages/sockets on the node, core and threads numbers on the node and within each package, and SMT mode enabling.
- **Logical processor identification** table identifies threads, cores, and packages of each logical processor accordingly.
Processor – logical processor number.
Thread Id – unique processor identifier within a core.
Core Id – unique core identifier within a package.
Package Id – unique package identifier within a node.
- **Node decomposition table** shows the node contents. Each entry contains the information on packages, cores, and logical processors.
Package Id – physical package identifier.
Cores Id – list of core identifiers that belong to this package.
Processors Id – list of processors that belong to this package. This list order directly corresponds to the core list. A group of processors enclosed in brackets belongs to one core.
- **Cache sharing by logical processors** shows information of sizes and processors groups, which share particular cache level.
Size – cache size in bytes.
Processors – a list of processor groups enclosed in the parentheses those share this cache or *no sharing* otherwise.
- **Microprocessor signature** table shows signature values: extended family, extended model, family, model, type, and stepping.
- **Microprocessor feature flags** indicate what features the microprocessor supports. The Intel platform notation is used.

NOTE: The architecture information is available on systems based on the IA-32 and Intel® 64 architectures.

The `cpuinfo` utility is available for both Intel® microprocessors and non-Intel microprocessors, but it may provide only partial information about non-Intel microprocessors.

Examples

`cpuinfo` output for the processor of Intel® microarchitecture code name Sandy Bridge:

```
$ cpuinfo A
```

```
Intel(R) Processor information utility, Version 4.0 Update 3 Build 20110526
Copyright (C) 2005-2011 Intel Corporation. All rights reserved.
```

```
===== Processor composition =====
Processor name      : Genuine Intel(R)
```

Packages (sockets) : 2
 Cores : 16
 Processors (CPUs) : 32
 Cores per package : 8
 Threads per core : 2

==== Processor identification =====

Processor	Thread Id.	Core Id.	Package Id.
0	0	0	0
1	0	1	0
2	0	2	0
3	0	3	0
4	0	4	0
5	0	5	0
6	0	6	0
7	0	7	0
8	0	0	1
9	0	1	1
10	0	2	1
11	0	3	1
12	0	4	1
13	0	5	1
14	0	6	1
15	0	7	1
16	1	0	0
17	1	1	0
18	1	2	0
19	1	3	0
20	1	4	0
21	1	5	0
22	1	6	0
23	1	7	0
24	1	0	1
25	1	1	1
26	1	2	1
27	1	3	1
28	1	4	1
29	1	5	1
30	1	6	1
31	1	7	1

==== Placement on packages =====

Package Id.	Core Id.	Processors
0	0,1,2,3,4,5,6,7	(0,16) (1,17) (2,18) (3,19) (4,20) (5,21) (6,22) (7,23)
1	0,1,2,3,4,5,6,7	(8,24) (9,25) (10,26) (11,27) (12,28) (13,29) (14,30) (15,31)

==== Cache sharing =====

Cache	Size	Processors
L1	32 KB	(0,16) (1,17) (2,18) (3,19) (4,20) (5,21) (6,22) (7,23) (8,24) (9,25) (10,26) (11,27) (12,28) (13,29) (14,30) (15,31)
L2	256 KB	(0,16) (1,17) (2,18) (3,19) (4,20) (5,21) (6,22) (7,23) (8,24) (9,25) (10,26) (11,27) (12,28) (13,29) (14,30) (15,31)
L3	20 MB	(0,1,2,3,4,5,6,7,16,17,18,19,20,21,22,23) (8,9,10,11,12,13,14,15,24,25,26,27,28,29,30,31)

==== Processor Signature =====

xFamily	xModel	Type	Family	Model	Stepping
00	0	2	6	d	3

==== Processor Feature Flags =====

SSE3	PCLMULQ	DTES64	MONITOR	DS-CPL	VMX	SMX	EIST	TM2	SSSE3	CNXT-ID	FMA	CX16	xTPR
1	1	1	1	1	1	1	1	1	1	0	0	1	1

PDCM	PCID	DCA	SSE4.1	SSE4.2	x2APIC	MOVBE	POPCNT	TSC-DEADLINE	AES	XSAVE	OSXSAVE	AVX
1	1	1	1	1	1	0	1	1	1	1	1	1

FPU	VME	DE	PSE	TSC	MSR	PAE	MCE	CX8	APIC	SEP	MTRR	PGE	MCA	CMOV	PAT	PSE-36
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

PSN	CLFSH	DS	ACPI	MMX	FXSR	SSE	SSE2	SS	HTT	TM	PBE
0	1	1	1	1	1	1	1	1	1	1	1

3 User Authorization

3.1 Overview

The Intel® MPI Library supports several authentication methods under Windows* OS:

- The password-based authorization
- The domain-based authorization with the delegation ability
- The limited domain-based authorization

The password-based authorization is the typical method of providing remote computer access using your account name and password.

The domain-based authorization methods use the Security Service Provider Interface (SSPI) provided by Microsoft* in a Windows* environment. The SSPI allows domain to authenticate the user on the remote machine in accordance to the domain policies. You do not need to enter and store your account name and password when using such methods.

NOTE: Both domain-based authorization methods may increase MPI task launch time in comparison with the password-based authorization. This depends on the domain configuration.

NOTE: The limited domain-based authorization restricts your access to the network. You will not be able to open files on remote machines or access mapped network drives.

3.2 Installation

This feature is supported on Windows clusters under the following Microsoft operation systems: Windows* 2000 Server*, Windows 2003 Server*, and Windows 2008 Server*.

Microsoft's Kerberos Distribution Center* must be enabled on your domain controller. This is the default behavior.

Using the domain-based authorization method with the delegation ability requires specific installation of the domain. You can perform this installation in the following ways:

- Use the IMPI installer if you have domain administrator rights
- Follow the actions described in the topic below

3.2.1 Active Directory* Setup

To enable the delegation in the Active Directory* do the following:

1. Log in on the domain controller under the admin account
2. Enable the delegation for cluster nodes:
 - a. Open the **Computers** list in the **Active Directory Users and Computers** administrative utility
 - b. Right click on a desired computer object and select **Properties**
 - c. If the account is located:

- i. in a Windows 2000 functional level domain, check the **Trust computer for delegation** option
 - ii. in a Windows 2003 functional level domain, select the **Delegation** tab and check the **Trust this computer for delegation to any service (Kerberos only)** option
3. Enable the delegation for users:
 - a. Open the **Users** list in the **Active Directory Users and Computers** administrative utility
 - b. Right click on a desired user object and select **Properties**
 - c. Select the **Account** tab and disable the **Account is sensitive and cannot be delegated** option
4. Register Service Principal Name (SPN) for cluster nodes. Use one of the following methods for registering SPN:
 - a. Use the Microsoft*-provided `setspn.exe` utility. For example, execute the following command on the domain controller:


```
setspn.exe -A impi_smpd/<host>:<port>/impi_smpd <host>
```

 where
`<host>` is a cluster node name
`<port>` is a SMPD port. The default value is 8678. Change this number only if your SMPD service uses the non-default port
 - b. Log into each desired node under the admin account and execute the `smpd -register_spn` command

NOTE: In case of any issues with the MPI task start, reboot the machine from which the MPI task is started. Alternatively, execute the `klist purge` command there if the Microsoft*-provided `klist.exe` utility is available.

3.3 Environment Variables

I_MPI_AUTH_METHOD

Select a user authorization method.

Syntax

`I_MPI_AUTH_METHOD=<method>`

Arguments

<code><method></code>	Define an authorization method
<code>password</code>	Use the password-based authorization. This is the default value
<code>delegate</code>	Use the domain-based authorization with delegation ability
<code>impersonate</code>	Use the limited domain-based authorization. You will not be able to open files on remote machines or access mapped network drives

Description

Set this environment variable to select a desired authorization method. If this environment variable is not defined, `mpiexec` uses the password-based authorization method by default.

NOTE: Alternatively, you can change the default behavior by using the `mpiexec -delegate` or `mpiexec -impersonate` options.

4 Tuning Reference

The Intel® MPI Library provides an automatic tuning utility and many environment variables that can be used to influence program behavior and performance at run time.

4.1 Automatic Tuning Utility

mpitune

Use the `mpitune` utility to find optimal settings for the Intel® MPI Library relevant to your cluster configuration or your application.

Syntax

```
mpitune [ -a "<application command line>" ] [ -of <file-name> ] \
[ -t "<test_cmd_line>" ] [ -cm ] [ -d ] [ -D ] \
[ -dl [d1[,d2...[,dN]]] ] [ -fl [f1[,f2...[,fN]]] ] \
[ -hf <hostsfile> ] [ -h ] [ -hr {min:max/min:/:max} ] \
[ -i <count> ] [ -mr {min:max/min:/:max} ] [ -od <outputdir> ] \
[ -odr <outputdir> ] [ -pr {min:max/min:/:max} ] \
[ -sf [file-path] ] [ -ss ] [ -s ] [ -td <dir-path> ] \
[ -tl <minutes> ] [ -mh ] [ -os <opt1,...,optN> ] \
[ -oe <opt1,...,optN> ] [ -V ] [ -vi {percent} ; -vix {X factor} ] \
[ -zb ] [ -t ] [ -so ] [ -ar "reg-expr" ] [ -trf <appoutfile> ] \
[ -m {base|optimized} ] [ -avd {min|max} ] \
[ -co ] [ -sd ] [ -soc ]
```

or

```
mpitune [ --application "<app_cmd_line>" ] [ --output-file <file-name> ] \
[ --test "<test_cmd_line>" ] [ --cluster-mode ] [ --debug ] \
[ --distinct ] [ --device-list [d1[,d2,... [,dN]]] ] \
[ --fabric-list [f1[,f2...[,fN]]] ] \
[ --host-file <hostsfile> ] [ --help ] \
[ --host-range {min:max/min:/:max} ] [ --iterations <count> ] \
[ --message-range {min:max/min:/:max} ] \
[ --output-directory <outputdir> ] \
[ --output-directory-results <outputdir> ] \
[ --ppn-range {min:max/min:/:max} ;  
--perhost-range {min:max/min:/:max} ] \
```

```

[ --session-file [file-path] ] [ --show-session ] [ --silent ] \
[ --temp-directory <dir-path> ] [ --time-limit <minutes> ] \
[ --master-host ] [ --options-set <opt1,...,optN> ] \
[ --options-exclude <opt1,...,optN> ] [ --version ] \
[ --valuable-improvement ; --valuable-improvement-x {X factor} ] \
[ --zero-based ] [ --trace ] [ --scheduler-only ] \
[ --application-regexp "reg-expr" ] \
[ --test-regexp-file <appoutfile> ] [ --model {base|optimized} ] \
[ --application-value-direction {min|max} ] \
[ -co ] [ -sd ] [ -soc ]

```

Arguments

-a \"<app_cmd_line>\" --application \"<app_cmd_line>\"	Switch on the application-specific mode. Quote the full command line as shown including the backslashes
-of <file-name> --output-file <file-name>	Specify the name of the application configuration file to be generated in the application-specific mode. By default, use the file name <code>app.conf</code>
-t \"<test_cmd_line>\" --test \"<test_cmd_line>\"	Replace the default Intel® MPI Benchmarks by the indicated benchmarking program in the cluster-specific mode. Quote the full command line as shown including the backslashes
-cm {exclusive full} -- cluster-mode {exclusive full}	Set the cluster usage mode full – maximum number of tasks will be executed. This is the default mode exclusive – only one task will be executed on the cluster at a time
-d --debug	Print out the debug information
-D --distinct	Tune all options separately from each other. This argument is applicable only for the cluster-specific mode
-dl [d1[,d2...[,dN]]] --device-list [d1[,d2,... [,dN]]]	Select the device(s) you want to tune. Any previously set fabrics are ignored. By default, use all devices listed in the <code><installdir>\<arch>\etc\devices.xml</code> file
-fl [f1[,f2...[,fN]]] --fabric-list [f1[,f2...[,fN]]]	Select the fabric(s) you want to tune. Any previously set devices are ignored. By default, use all fabrics listed in the <code><installdir>\<arch>\etc\fabrics.xml</code> file
-hf <hostsfile> --host-file <hostsfile>	Specify an alternative host file name. By default, use the <code>mpd.hosts</code>
-h --help	Display the help message
-hr {min:max min:/:max} --host-range {min:max min:/:max}	Set the range of hosts used for testing. The default minimum value is 1. The default maximum value is the number of hosts defined by the <code>mpd.hosts</code> or the existing MPD ring. The min: or :max format will use the default values as appropriate
-i <count> --iterations <count>	Define how many times to run each tuning step. Higher iteration counts increase the tuning time, but may also increase the accuracy of the results. The default value is 3

<code>-mr {min:max min:/:max} --message-range {min:max min:/:max}</code>	Set the message size range. The default minimum value is 0. The default maximum value is 4194304 (4mb). By default, the values are given in bytes. They can also be given in the following format: 16kb, 8mb or 2gb. The min: or :max format will use the default values as appropriate
<code>-od <outputdir> --output-directory <outputdir></code>	Specify the directory name for all output files: log-files, session-files, local host-files and report-files. By default, use the current directory. This directory should be accessible from all hosts
<code>-odr <outputdir> --output-directory-results <outputdir></code>	Specify the directory name for the resulting configuration files. By default, use the current directory in the application-specific mode and the <installdir>\<arch>\etc in the cluster-specific mode. If <installdir>\<arch>\etc is unavailable, the current directory is used as the default value in the cluster-specific mode
<code>-pr {min:max min:/:max} --ppn-range {min:max min:/:max} --perhost-range {min:max min:/:max}</code>	Set the maximum number of processes per host. The default minimum value is 1. The default maximum value is the number of cores of the processor. The min: or :max format will use the default values as appropriate
<code>-sf [file-path] --session-file [file-path]</code>	Continue the tuning process starting from the state saved in the file-path session file
<code>-ss --show-session</code>	Show information about the session file and exit. This option works only jointly with the -sf option
<code>-s --silent</code>	Suppress all diagnostics
<code>-td <dir-path> --temp-directory <dir-path></code>	Specify a directory name for the temporary data. The Intel MPI Library uses the mpitunertemp folder in the current directory by default. This directory should be accessible from all hosts
<code>-tl <minutes> --time-limit <minutes></code>	Set mpitune execution time limit in minutes. The default value is 0, which means no limitations
<code>-mh --master-host</code>	Dedicate a single host to run the mpitune
<code>-os <opt1,...,optN> --options-set <opt1,...,optN></code>	Use mpitune to tune the only required options you have set in the option values
<code>-oe <opt1,...,optN> --options-exclude <opt1,...,optN></code>	Exclude the settings of the indicated Intel® MPI Library options from the tuning process
<code>-V --version</code>	Print out the version information
<code>-vi {percent} --valuable-improvement {percent} -vix {X factor} --valuable-improvement-x {X factor}</code>	Control the threshold for performance improvement. The default threshold is 3%
<code>-zb --zero-based</code>	Set zero as the base for all options before tuning. This argument is applicable only for the cluster-specific mode
<code>-t --trace</code>	Print out error information such as error codes and tuner

	traceback
<code>-so --scheduler-only</code>	Create the list of tasks to be executed, display the tasks, and terminate execution
<code>-ar "reg-expr" --application-regexp "reg-expr"</code>	Use <code>reg-expr</code> to determine the performance expectations of the application. This option is applicable only for the application-specific mode. The <code>reg-expr</code> setting should contain only one group of numeric values which will be used by <code>mpitune</code> for analysis. Use backslash for symbols when setting the value of this argument in accordance with the operating system requirements
<code>-trf <appoutfile> --test-regexp-file <appoutfile></code>	Use a test output file to check the correctness of the regular expression. This argument is applicable only for the cluster-specific mode when you use the <code>-ar</code> option
<code>-m {base optimized} --model {base optimized}</code>	Specify the search model: <ul style="list-style-type: none"> Set <code>base</code> to use the old model Set <code>optimized</code> to use the new faster search model. This is the default value
<code>-avd {min max} --application-value-direction {min max}</code>	Specify the direction of the value optimization : <ul style="list-style-type: none"> Set <code>min</code> to specify that lower is better. For example, use this value when optimizing the wall time Set <code>max</code> to specify that higher is better. For example, use this value when optimizing the solver ratio
<code>-co --collectives-only</code>	Tune collective operations only
<code>-sd --save-defaults</code>	Use <code>mpitune</code> to save the default values of the Intel® MPI Library options
<code>-soc --skip-options-check</code>	Specify whether to check the command line options

Deprecated Options

Deprecated Option	New Option
<code>--outdir</code>	<code>-od --output-directory</code>
<code>--verbose</code>	<code>-d --debug</code>
<code>--file</code>	<code>-hf --host-file</code>
<code>--logs</code>	<code>-lf --log-file</code>
<code>--app</code>	<code>-a --application</code>

Description

Use the `mpitune` utility to create a set of Intel® MPI Library configuration files that contain optimal settings for a particular cluster or application. You can reuse these configuration files in the `mpiexec` job launcher by using the `-tune` option. If configuration files from previous `mpitune` sessions exist, `mpitune` creates a copy of the existing files before starting execution.

The MPI tuner utility operates in two modes:

- Cluster-specific, evaluating a given cluster environment using either the Intel® MPI Benchmarks or a user-provided benchmarking program to find the most suitable configuration of the Intel® MPI Library. This mode is used by default.

- Application-specific, evaluating the performance of a given MPI application to find the best configuration for the Intel® MPI Library for the particular application. Application tuning is enabled by the `--application` command line option.

4.1.1 Cluster-specific Tuning

Run this utility once after the Intel® MPI Library installation and after every cluster configuration change (processor or memory upgrade, network reconfiguration, etc.). Do this under the user account that was used for the Intel® MPI Library installation or appropriately set the tuner data directory through the `--output-directory` option and the results directory through the `--output-directory-results` option.

If there are any configuration files in the `<installdir>\<arch>\etc` directory, the recorded Intel® MPI Library configuration settings will be used automatically by `mpiexec` with the `-tune` option.

For example:

- Collect configuration settings for the cluster hosts listed in the `.\mpd.hosts` file by using the Intel® MPI Benchmarks

```
> mpitune
```

- Use the optimal recorded values when running on the cluster

```
> mpiexec -tune -n 32 .\myprog
```

The job launcher will find a proper set of configuration options based on the following execution conditions: communication fabrics, number of hosts and processes, etc. If you have write access permission for `<installdir>\<arch>\etc`, all generated files will be saved in this directory; otherwise the current working directory will be used.

4.1.1.1 Replacing the Default Benchmark

This tuning feature is an extension of the cluster-specific tuning mode in which you specify a benchmarking application that will be used for tuning.

The Intel® MPI Benchmarks executables, which are more optimized for Intel microprocessors than for non-Intel microprocessors, are used by default. This may result in different tuning settings on Intel microprocessors than on non-Intel microprocessors.

For example:

1. Collect the configuration settings for the cluster hosts listed in the `.\mpd.hosts` file by using the desired benchmarking program

```
> mpitune --test \"benchmark -param1 -param2\"
```

2. Use the optimal recorded values for your cluster

```
> mpiexec -tune -n 32 .\myprog
```

4.1.2 Application-specific Tuning

Run the tuning process for any kind of MPI application by specifying its command line to the tuner. Performance is measured as inversed execution time of the given application. To reduce the overall tuning time, use the shortest representative application workload if applicable.

For example:

1. Collect configuration settings for the given application

```
> mpitune --application \"mpiexec -n 32 .\myprog\" -of .\myprog.conf
```

2. Use the optimal recorded values for your application

```
> mpiexec -tune .\myprog.conf -n 32 .\myprog
```

Based on the default tuning rules, the automated tuning utility evaluates a full set of the library configuration parameters to minimize the application execution time. By default, all generated files will be saved in the current working directory.

NOTE: The resulting application configuration file contains the optimal Intel® MPI Library parameters for this particular application only. If you want to tune the Intel® MPI Library for the same application in a different configuration (number of hosts, workload, etc.), you may need to rerun the automated tuning utility by using the desired configuration.

The automated tuning utility will overwrite the existing application configuration files by default. You should use a naming convention for your various application files to create and select the correct file when you need it.

4.1.3 Tuning Utility Output

Upon completion of the tuning process, the Intel® MPI Library tuning utility records the chosen values in the configuration file in the following format:

```
-genv I_MPI_DYNAMIC_CONNECTION 1
-genv I_MPI_ADJUST_REDUCE 1:0-8
```

The Intel MPI Library tuning utility ignores any environment variables that have no effect on the application when the difference between probes is at the noise level (1%). In this case, the utility does not set the environment variable and preserves the default library heuristics.

In the case of the tuning application having significant run-to-run performance variation, the Intel MPI Library tuning utility might select divergent values for the same environment variable under the same conditions. To improve decision accuracy, increase the number of iterations for each test run with the `--iterations` command line option. The default value for the iteration number is 3.

4.2 Process Pinning

Use this feature to pin a particular MPI process to a corresponding CPU and avoid undesired process migration. This feature is available on operating systems that provide the necessary kernel interfaces.

4.2.1 Process Identification

Two schemes are used to identify logical processors in a system:

1. System-defined logical enumeration
2. Topological enumeration based on three-level hierarchical identification through triplets (package/socket, core, thread)

The number of a logical CPU is defined as the corresponding position to this CPU bit in the kernel affinity bit-mask. Use the `cpuinfo` utility, provided with your Intel MPI Library installation, to find out the logical CPU numbers.

The three-level hierarchical identification uses triplets that provide information about processor location and their order. The triplets are hierarchically ordered (package, core, thread).

See the example below for one possible processor numbering where there are two sockets, four cores (two cores per socket), and eight logical processors (two processors per core).

NOTE: Logical and topological enumerations are not the same.

Table 4.2-1 Logical Enumeration

0	4	1	5	2	6	3	7
---	---	---	---	---	---	---	---

Table 4.2-2 Hierarchical Levels

Socket	0	0	0	0	1	1	1	1
Core	0	0	1	1	0	0	1	1
Thread	0	1	0	1	0	1	0	1

Table 4.2-3 Topological Enumeration

0	1	2	3	4	5	6	7
---	---	---	---	---	---	---	---

Use the `cpuinfo` utility to identify the correspondence between the logical and topological enumerations. See [Processor Information Utility](#) for more details.

4.2.2 Environment Variables

`I_MPI_PIN`

Turn on/off process pinning.

Syntax

`I_MPI_PIN=<arg>`

Arguments

<code><arg></code>	Binary indicator
<code>enable yes on 1</code>	Enable process pinning. This is the default value
<code>disable no off 0</code>	Disable processes pinning

Description

Set this environment variable to turn off the process pinning feature of the Intel® MPI Library.

`I_MPI_PIN_PROCESSOR_LIST` (`I_MPI_PIN_PROCS`)

Define a processor subset and the mapping rules for MPI processes within this subset.

Syntax

`I_MPI_PIN_PROCESSOR_LIST=<value>`

The environment variable value has three syntax forms:

1. `<proclist>`
2. `[<procset>] [:[grain=<grain>] [,shift=<shift>] \`
`[,preoffset=<preoffset>] [,postoffset=<postoffset>]`
3. `[<procset>] [:map=<map>]`

Deprecated Syntax

`I_MPI_PIN_PROCS=<proclist>`

NOTE: The `postoffset` keyword has offset alias.

NOTE: The second form of the pinning procedure has three steps:

1. Cyclic shift of the source processor list on `preoffset*grain` value.
2. Round robin shift of the list derived on the first step on `shift*grain` value.
3. Cyclic shift of the list derived on the second step on the `postoffset*grain` value.

The resulting processor list is used for the consecutive mapping of MPI processes (i-th rank is mapped to the i-th list member).

NOTE: The `grain`, `shift`, `preoffset`, and `postoffset` parameters have a unified definition style.

This environment variable is available for both Intel® and non-Intel microprocessors, but it may perform additional optimizations for Intel microprocessors than it performs for non-Intel microprocessors.

Arguments

<code><proclist></code>	A comma-separated list of logical processor numbers and/or ranges of processors. The process with the i-th rank is pinned to the i-th processor in the list. The number should not exceed the amount of processors on a node
<code><l></code>	Processor with logical number <code><l></code>
<code><l>-<m></code>	Range of processors with logical numbers from <code><l></code> to <code><m></code>
<code><k>, <l>-<m></code>	Processors <code><k></code> , as well as <code><l></code> through <code><m></code>

<code><procset></code>	Specify a processor subset based on the topological numeration. The default value is <code>allcores</code>
<code>all</code>	All logical processors. This subset is defined to be the number of CPUs on a node
<code>allcores</code>	All cores (physical CPUs). This subset is defined to be the number of cores on a node. This is the default value If Intel® Hyper-Threading technology is disabled, <code>allcores</code> equals to <code>all</code>
<code>allsockets</code>	All packages/sockets. This subset is defined to be the number of sockets on a node

<code><map></code>	The mapping the pattern used for process placement
<code>bunch</code>	The processes are mapped close as possible on the sockets
<code>scatter</code>	The processes are mapped as remotely as possible so as not to share common resources: FSB, caches, core
<code>spread</code>	The processes are mapped consecutively with the possibility not to share common resources

<code><grain></code>	Specify the pinning granularity cell for a defined <code><procset></code> . The minimal <code><grain></code> is a single element of the <code><procset></code> . The maximal grain is the number of <code><procset></code> elements on a socket. The <code>grain</code> value must be a multiple of the <code><procset></code> value. Otherwise, minimal grain is assumed. The default value is the minimal <code><grain></code>
<code><shift></code>	Specify the round robin shift of the granularity cells for the <code><procset></code> . <code><shift></code> is measured in the defined <code><grain></code> units. The <code><shift></code> value must be a positive integer. Otherwise, no shift is performed. The default value is no shift
<code><preoffset></code>	Specify the cyclic shift of the processor subset <code><procset></code> defined before the round robin shifting on the <code><preoffset></code> value. The value is measured in the defined <code><grain></code> units. The <code><preoffset></code> value must be a non-negative integer. Otherwise, no shift is performed. The default value is no shift
<code><postoffset></code>	Specify the cyclic shift of processor subset <code><procset></code> derived after round robin shifting on the <code><postoffset></code> value. The value is measured in the defined <code><grain></code> units. The <code><postoffset></code> value must be a non-negative integer. Otherwise no shift is performed. The default value is no shift

<code><n></code>	Specify an explicit value of the corresponding parameters previously mentioned. <code><n></code> is non-negative integer
<code>fine</code>	Specify the minimal value of the corresponding parameter
<code>core</code>	Specify the parameter value equal to the amount of the corresponding parameter units contained in one core
<code>cache1</code>	Specify the parameter value equal to the amount of the corresponding parameter units that share an L1 cache
<code>cache2</code>	Specify the parameter value equal to the amount of the corresponding parameter units that share an L2 cache
<code>cache3</code>	Specify the parameter value equal to the amount of the corresponding parameter units that share an L3 cache
<code>cache</code>	The largest value among <code>cache1</code> , <code>cache2</code> , and <code>cache3</code>
<code>socket sock</code>	Specify the parameter value equal to the amount of the corresponding parameter units contained in one physical package/socket
<code>half mid</code>	Specify the parameter value equal to <code>socket/2</code>
<code>third</code>	Specify the parameter value equal to <code>socket/3</code>
<code>quarter</code>	Specify the parameter value equal to <code>socket/4</code>
<code>octavo</code>	Specify the parameter value equal to <code>socket/8</code>

Description

Set the `I_MPI_PIN_PROCESSOR_LIST` environment variable to define the processor placement. In order to avoid conflicts with differing shell syntax, the environment variable value may need to be enclosed in quotes.

NOTE: This environment variable is valid only if `I_MPI_PIN` is enabled.

The `I_MPI_PIN_PROCESSOR_LIST` environment variable has three different syntax variants:

1. Explicit processor list. This comma-separated list is defined in terms of logical processor numbers. The relative node rank of a process is an index to the processor list such that the *i*-th process is pinned to *i*-th list member. This permits the definition of any process placement on the CPUs.

For example, process mapping for `I_MPI_PROCESSOR_LIST=p0,p1,p2,...,pn` is as follows:

Rank on a node	0	1	2	...	n-1	N
Logical CPU	p0	p1	p2	...	pn-1	Pn

2. `grain/shift/offset` mapping. This method provides cyclic shift of a defined `grain` along the processor list with steps equal to `shift * grain` and a single shift on `offset * grain` at the end. This shifting action is repeated `shift` times.

For example: `grain = 2` logical processors, `shift = 3` grains, `offset = 0`.

Legend:

 – MPI process grains

A) – processor grains chosen on the 1st pass

B) - processor grains chosen on the 2nd pass

C) - processor grains chosen on the final 3rd pass

D) Final map table ordered by MPI ranks

A)

0 1			2 3			...	2n-2 2n-1		
0 1	2 3	4 5	6 7	8 9	10 11	...	6n-6 6n-5	6n-4 6n-3	6n-2 6n-1

B)

0 1	2n 2n+1		2 3	2n+2 2n+3		...	2n-2 2n-1	4n-2 4n-1	
0 1	2 3	4 5	6 7	8 9	10 11	...	6n-6 6n-5	6n-4 6n-3	6n-2 6n-1

C)

0 1	2n 2n+1	4n 4n+1	2 3	2n+2 2n+3	4n+2 4n+3	...	2n-2 2n-1	4n-2 4n-1	6n-2 6n-1
0 1	2 3	4 5	6 7	8 9	10 11	...	6n-6 6n-5	6n-4 6n-3	6n-2 6n-1

D)

0	1	2 3	...	2n-2 2n-1	2n 2n+1	2n+2 2n+3	...	4n-2 4n-1	4n 4n+1	4n+2 4n+3	...	6n-2 6n-1
0	1	6 7	...	6n-6 6n-5	2 3	8 9	...	6n-4 6n-3	4 5	10 11	...	6n-2 6n-1

3. Predefined mapping scenario. In this case popular process pinning schemes are defined as keywords selectable at runtime. There are two such scenarios: `bunch` and `scatter`.

In the **bunch** scenario the processes are mapped proportionally to sockets as closely as possible. This makes sense for partial processor loading. In this case the number of processes is less than the number of processors.

In the **scatter** scenario the processes are mapped as remotely as possible so as not to share common resources: FSB, caches, cores.

In the example below there are two sockets, four cores per socket, one logical CPU per core, and two cores per shared cache.

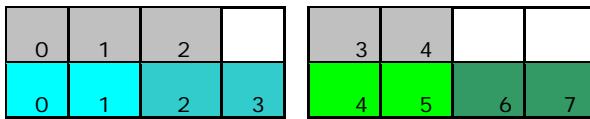
Legend:

gray – MPI processes

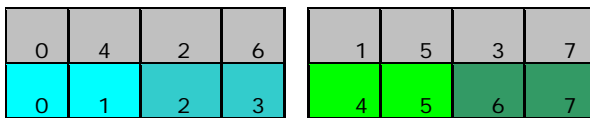
cyan – 1st socket processors

green – 2nd socket processors

Same color defines a processor pair sharing a cache



bunch scenario for 5 processors



scatter scenario for full loading

Examples

- To pin the processes to CPU0 and CPU3 on each node globally, use the following command:


```
> mpiexec.exe -genv I_MPI_PIN_PROCESSOR_LIST 0,3 \
-n <# of processes> <executable>
```
- To pin the processes to different CPUs on each node individually (CPU0 and CPU3 on host1 and CPU0, CPU1 and CPU3 on host2), use the following command:


```
> mpiexec.exe -host host1 -env I_MPI_PIN_PROCESSOR_LIST 0,3 \
-n <# of processes> <executable> : \
-host host2 -env I_MPI_PIN_PROCESSOR_LIST 1,2,3 \
-n <# of processes> <executable>
```
- To print extra debug information about the process pinning, use the following command:


```
> mpiexec.exe -genv I_MPI_DEBUG 4 -m -host host1 \
-env I_MPI_PIN_PROCESSOR_LIST 0,3 -n <# of processes> <executable> :\
-host host2 -env I_MPI_PIN_PROCESSOR_LIST 1,2,3 \
-n <# of processes> <executable>
```

NOTE: If the number of processes is greater than the number of CPUs used for pinning, the process list is wrapped around to the start of the processor list.

I_MPI_PIN_CELL

Set this environment variable to define the pinning resolution granularity. **I_MPI_PIN_CELL** specifies the minimal processor cell allocated when an MPI process is running.

Syntax

I_MPI_PIN_CELL=<cell>

Arguments

<code><cell></code>	Specify the resolution granularity
<code>unit</code>	Basic processor unit (logical CPU)
<code>core</code>	Physical processor core

Description

Set this environment variable to define the processor subset used when a process is running. You can choose from two scenarios:

- all possible CPUs in a system (`unit` value)
- all cores in a system (`core` value)

The environment variable has effect on both pinning kinds:

- one-to-one pinning through the `I_MPI_PIN_PROCESSOR_LIST` environment variable
- one-to-many pinning through the `I_MPI_PIN_DOMAIN` environment variable

The default value rules are:

- If you use `I_MPI_PIN_DOMAIN`, then the cell granularity is `unit`.
- If you use `I_MPI_PIN_PROCESSOR_LIST`, then the following rules apply:
 - When the number of processes is greater than the number of cores, the cell granularity is `unit`.
 - When the number of processes is equal to or less than the number of cores, the cell granularity is `core`.

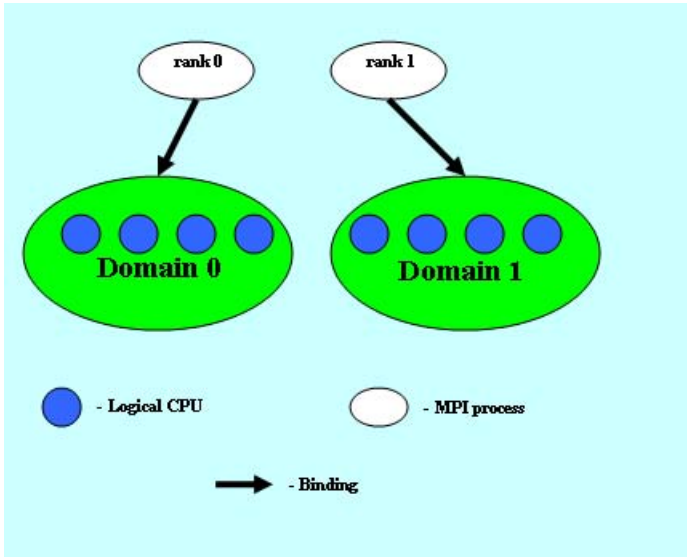
NOTE: The `core` value is not affected by the enabling/disabling of Hyper-threading technology in a system.

4.2.3 Interoperability with OpenMP*

`I_MPI_PIN_DOMAIN`

The Intel MPI Library provides an additional environment variable to control process pinning for hybrid MPI/OpenMP* applications.

This environment variable is used to define a number of non-overlapping subsets (domains) of logical processors on a node, and a set of rules on how MPI processes are bound to these domains by the following formula: **one MPI process per one domain**. See the picture below.



Picture 4.2-1 Domain Example

Each MPI process can create a number of children threads for running within the corresponding domain. The process threads can freely migrate from one logical processor to another within the particular domain. There are no domains defined by default so they should all be created explicitly.

If the `I_MPI_PIN_DOMAIN` environment variable is defined, then the `I_MPI_PIN_PROCESSOR_LIST` environment variable setting is ignored.

If the `I_MPI_PIN_DOMAIN` environment variable is not defined, then MPI processes are pinned according to the current value of the `I_MPI_PIN_PROCESSOR_LIST` environment variable.

The `I_MPI_PIN_DOMAIN` environment variable has the following syntax forms:

1. Domain description through multi-core terms
2. Domain description through domain size and domain member layout
3. Explicit domain description through bit mask

The following tables describe these syntax forms.

Multi-core Shape

`I_MPI_PIN_DOMAIN=<mc-shape>`

<code><mc-shape></code>	Define domains through multi-core terms
<code>core</code>	Each domain consists of the logical processors that share a particular core. The number of domains on a node is equal to the number of cores on the node
<code>socket sock</code>	Each domain consists of the logical processors that share a particular socket. The number of domains on a node is equal to the number of sockets on the node. This is the recommended value
<code>node</code>	All logical processors on a node are arranged into a single domain
<code>cache1</code>	Logical processors that share a particular level 1 cache are arranged into a single domain
<code>cache2</code>	Logical processors that share a particular level 2 cache are arranged into a single domain
<code>cache3</code>	Logical processors that share a particular level 3 cache are arranged into a single domain

cache	The largest domain among <code>cache1</code> , <code>cache2</code> , and <code>cache3</code> is selected
-------	--

Explicit Shape

`I_MPI_PIN_DOMAIN=<size>[:<layout>]`

<code><size></code>	Define a number of logical processors in each domain (domain size)
<code>omp</code>	The domain size is equal to the <code>OMP_NUM_THREADS</code> environment variable value. If the <code>OMP_NUM_THREADS</code> environment variable is not set, each node is treated as a separate domain.
<code>auto</code>	The domain size is defined by the formula <code>size=#cpu/#proc</code> , where <code>#cpu</code> is the number of logical processors on a node, and <code>#proc</code> is the number of the MPI processes started on a node
<code><n></code>	The domain size is defined by a positive decimal number <code><n></code>

<code><layout></code>	Ordering of domain members. The default value is <code>compact</code>
<code>platform</code>	Domain members are ordered according to their BIOS numbering (platform-depended numbering)
<code>compact</code>	Domain members are located as close to each other as possible in terms of common resources (cores, caches, sockets, etc.). This is the default value
<code>scatter</code>	Domain members are located as far away from each other as possible in terms of common resources (cores, caches, sockets, etc.)

Explicit Domain Mask

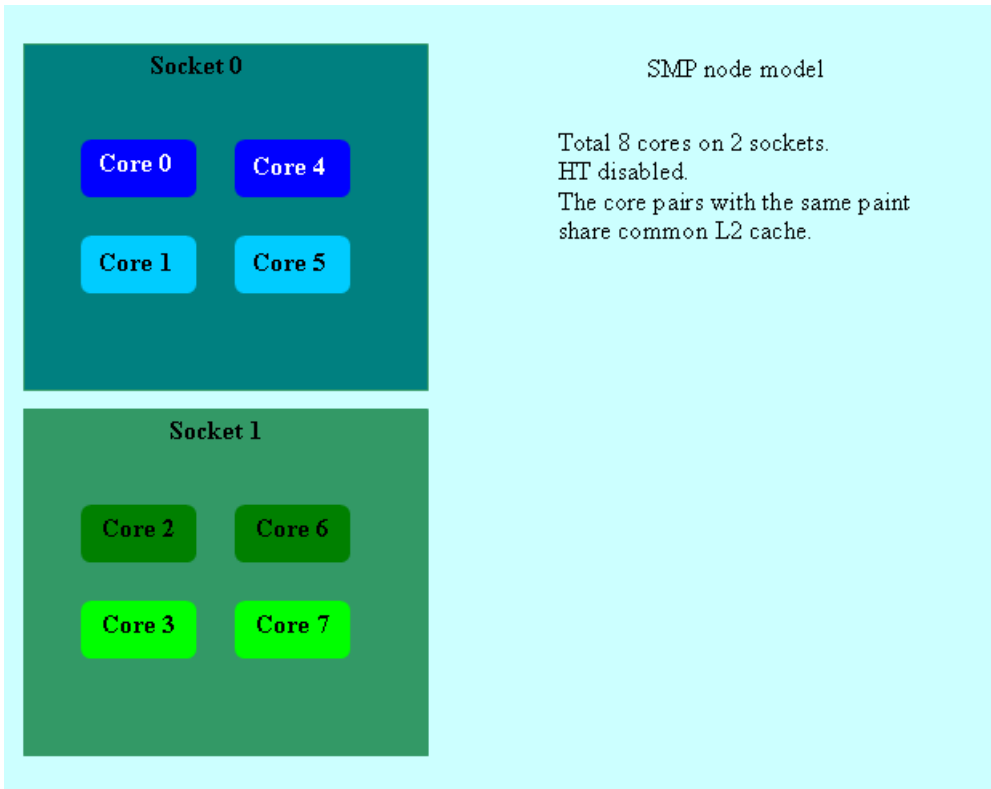
`I_MPI_PIN_DOMAIN=<masklist>`

<code><masklist></code>	Define domains through the comma separated list of hexadecimal numbers (domain masks)
<code>[m₁, ..., m_n]</code>	Each <code>m_i</code> number defines one separate domain. The following rule is used: the <code>ith</code> logical processor is included into the domain if the corresponding <code>m_i</code> value is set to <code>1</code> . All remaining processors are put into a separate domain. BIOS numbering is used

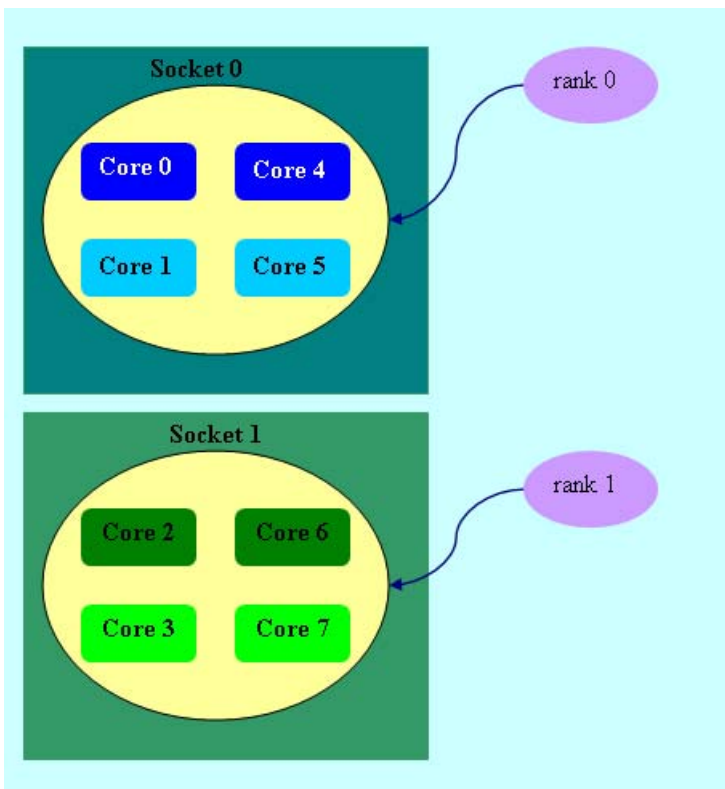
These options are available for both Intel® and non-Intel microprocessors, but they may perform additional optimizations for Intel microprocessors than they perform for non-Intel microprocessors.

NOTE: To pin OpenMP processes/threads inside the domain, the corresponding OpenMP feature (`KMP_AFFINITY` environment variable) should be used.

See the following model of an SMP node in the examples below:

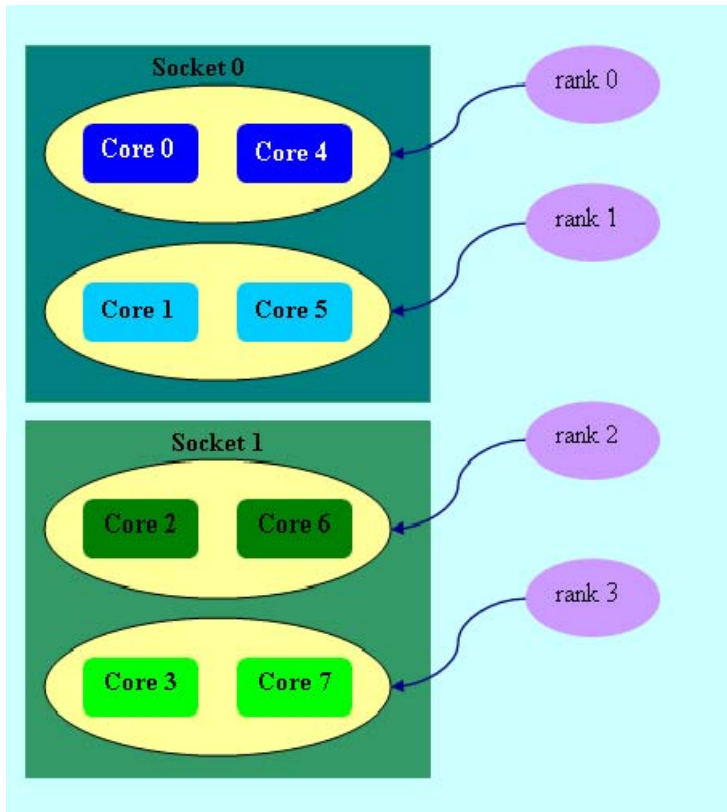


Picture 4.2-2 Node Model



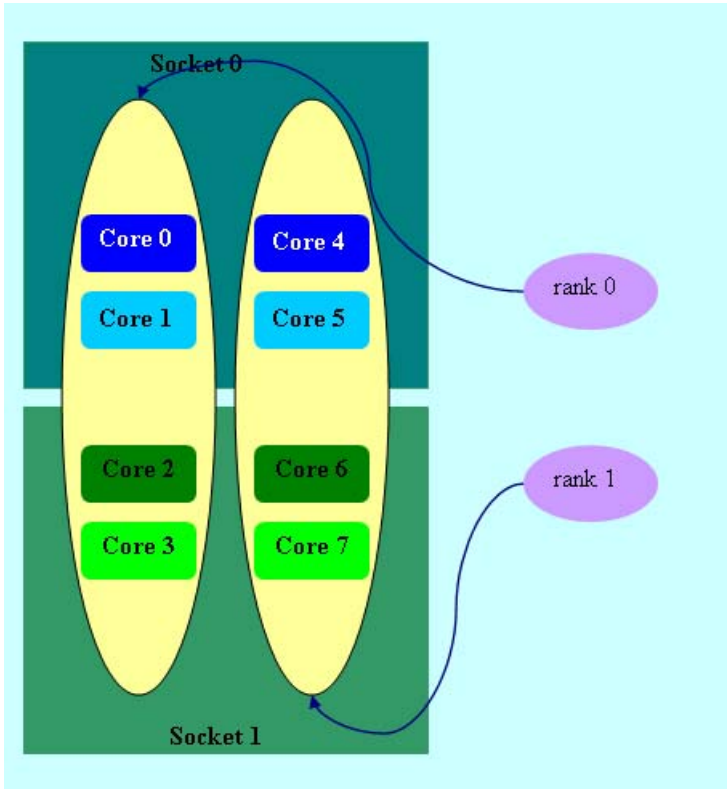
Picture 4.2-3 `mpiexec -n 2 -env I_MPI_PIN_DOMAIN socket .\a.out`

Two domains are defined according to the number of sockets. Process rank 0 can migrate on all cores on the 0-th socket. Process rank 1 can migrate on all cores on the first socket.



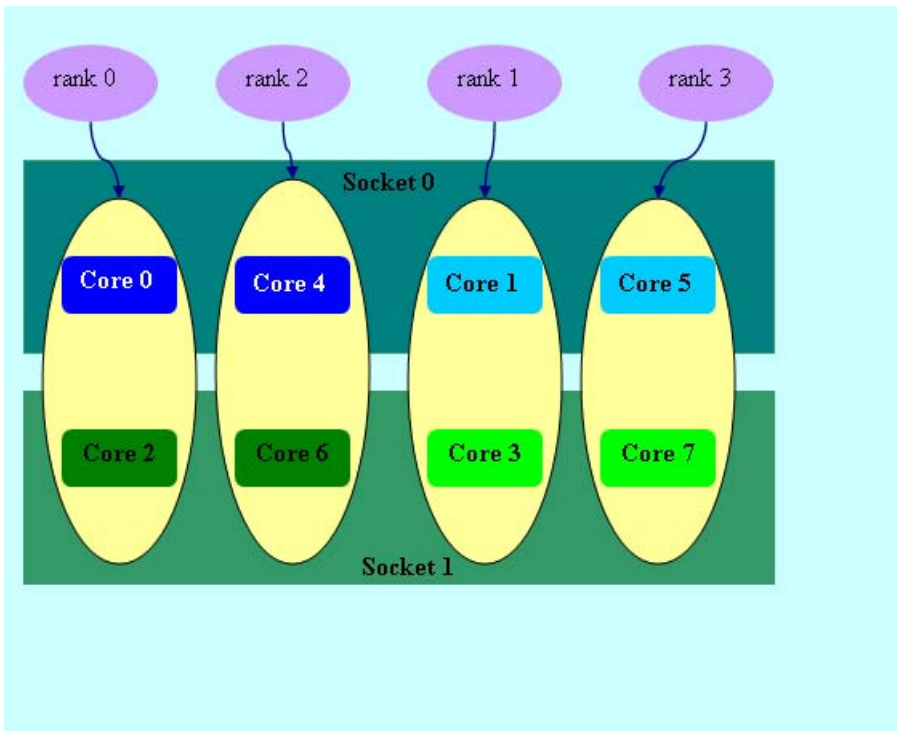
Picture 4.2-4 `mpiexec -n 4 -env I_MPI_PIN_DOMAIN cache2 .\a.out`

Four domains are defined according to the amount of common L2 caches. Process rank 0 runs on cores {0,4} that share an L2 cache. Process rank 1 runs on cores {1,5} that share an L2 cache as well, and so on.



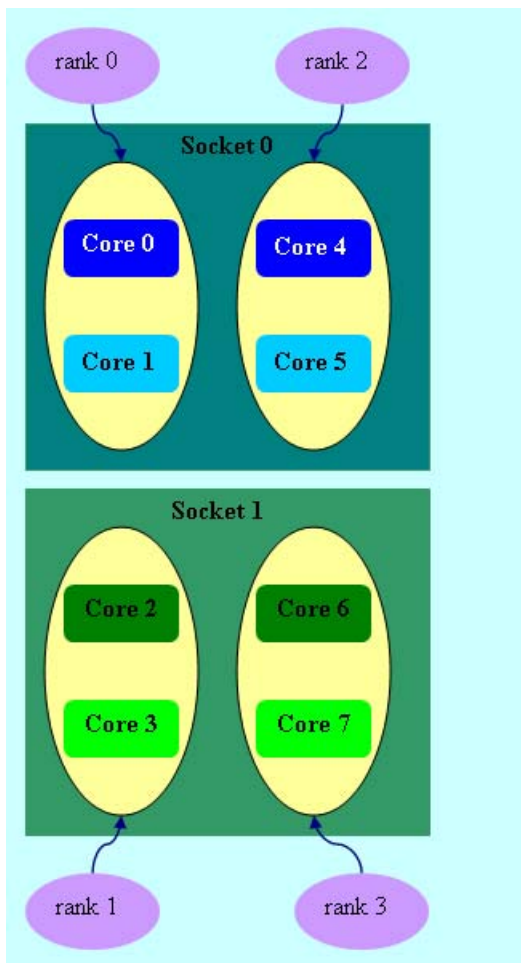
Picture 4.2-5 `mpiexec -n 2 -env I_MPI_PIN_DOMAIN 4:platform .\a.out`

Two domains with size=4 are defined. The first domain contains cores {0,1,2,3}, and the second domain contains cores {4,5,6,7}. Domain members (cores) have consecutive numbering as defined by the `platform` option.



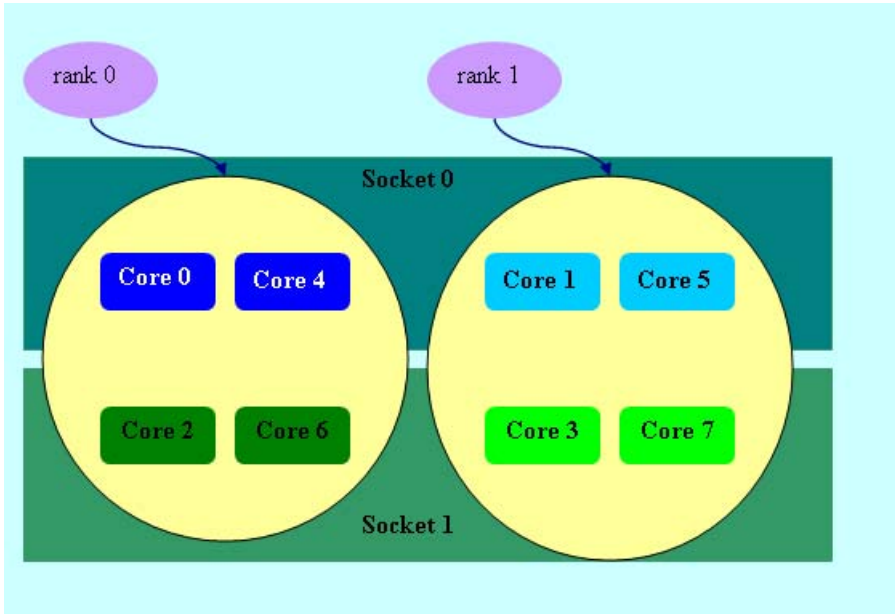
Picture 4.2-6 `mpiexec -n 4 -env I_MPI_PIN_DOMAIN auto:scatter .\a.out`

Domain size=2 (defined by the number of CPUs=8 / number of processes=4), `scatter` layout. Four domains {0,2}, {1,3}, {4,6}, {5,7} are defined. Domain members do not share any common resources.



Picture 4.2-7 `mpiexec -n 4 -env I_MPI_PIN_DOMAIN omp:platform .\a.out`
`setenv OMP_NUM_THREADS=2`

Domain size=2 (defined by `OMP_NUM_THREADS=2`), `platform` layout. Four domains {0,1}, {2,3}, {4,5}, {6,7} are defined. Domain members (cores) have consecutive numbering.



Picture 4.2-8 `mpiexec -n 2 -env I_MPI_PIN_DOMAIN [55,aa] .\a.out`

The first domain is defined by the 0x55 mask. It contains all cores with even numbers {0,2,4,6}. The second domain is defined by the 0xAA mask. It contains all cores with odd numbers {1,3,5,7}.

I_MPI_PIN_ORDER

Set this environment variable to define the mapping order for MPI processes to domains as specified by the `I_MPI_PIN_DOMAIN` environment variable.

Syntax

`I_MPI_PIN_ORDER=<order>`

Arguments

<code><order></code>	Specify the ranking order
<code>range</code>	The domains are ordered according to the processor's BIOS numbering. This is a platform-dependent numbering
<code>scatter</code>	The domains are ordered so that adjacent domains have minimal sharing of common resources
<code>compact</code>	The domains are ordered so that adjacent domains share common resources as much as possible. This is the default value

Description

The optimal setting for this environment variable is application-specific. If adjacent MPI processes prefer to share common resources, such as cores, caches, sockets, FSB, use the `compact` value. Otherwise, use the `scatter` value. Use the `range` value as needed.

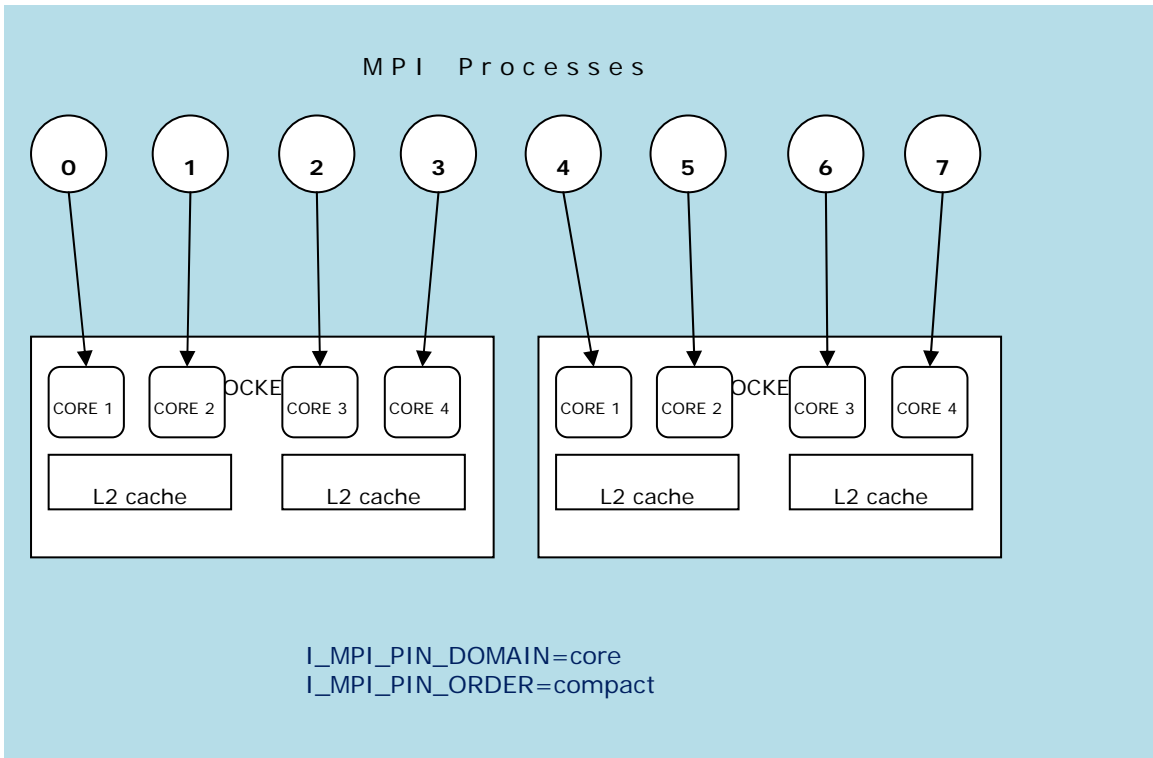
The options `scatter` and `compact` are available for both Intel® and non-Intel microprocessors, but they may perform additional optimizations for Intel microprocessors than they perform for non-Intel microprocessors.

Example

Assume we have:

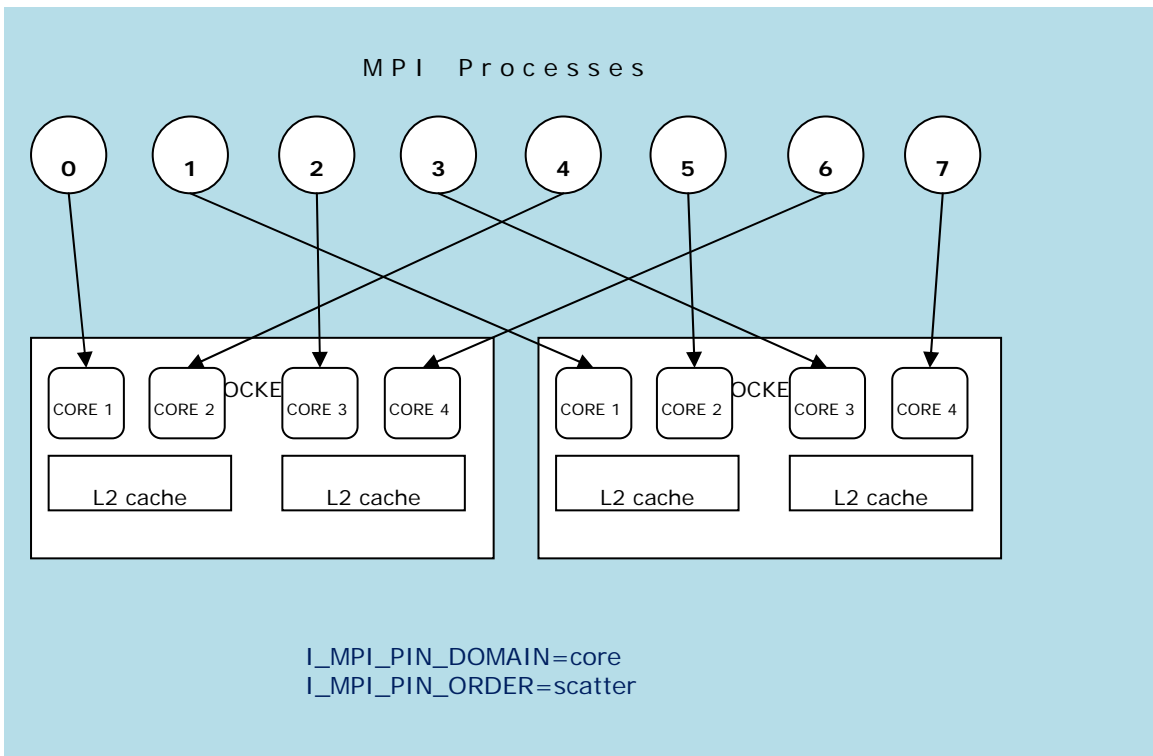
- Two socket node with four cores and a shared L2 cache for corresponding core pairs.
- 8 MPI processes we want to run on the node using

- o The following settings:
`I_MPI_PIN_DOMAIN=core`
`I_MPI_PIN_ORDER=compact`



Picture 4.2-9 Compact Order Example

- o The following settings:
`I_MPI_PIN_DOMAIN=core`
`I_MPI_PIN_ORDER=scatter`



Picture 4.2-10 Scatter Order Example

4.3 Fabrics Control

4.3.1 Communication Fabrics Control

I_MPI_FABRICS

(I_MPI_DEVICE)

Select the particular network fabrics to be used.

Syntax

```
I_MPI_FABRICS=<fabric>/<intra-node fabric>:<inter-nodes fabric>
```

Where *<fabric>* := {shm, dapl, tcp}

<intra-node fabric> := {shm, dapl, tcp}

<inter-nodes fabric> := {dapl, tcp}

Deprecated Syntax

```
I_MPI_DEVICE=<device>[:<provider>]
```

Arguments

<i><fabric></i>	Define a network fabric
shm	Shared-memory
dapl	DAPL-capable network fabrics, such as InfiniBand*, iWarp*, Dolphin*, and XPMEM* (through DAPL*)
tcp	TCP/IP-capable network fabrics, such as Ethernet and InfiniBand* (through IPoIB*)

Correspondence with I_MPI_DEVICE

<i><device></i>	<i><fabric></i>
sock	tcp
shm	shm
ssm	shm:tcp
rdma	dapl
rdssm	shm:dapl
<i><provider></i>	Optional DAPL* provider name (only for the <i>rdma</i> and the <i>rdssm</i> devices) I_MPI_DAPL_PROVIDER= <i><provider></i>

Use the *<provider>* specification only for the {rdma, rdssm} devices.

For example, to select the OFED* InfiniBand* device, use the following command:

```
$ mpiexec -n <# of processes> \
```

```
-env I_MPI_DEVICE rdssm:OpenIB-cma <executable>
```

For these devices, if *<provider>* is not specified, the first DAPL* provider in the `C:\DAT\dat.conf` file is used.

Description

Set this environment variable to select a specific fabric combination. If the pointed fabric(s) is not available, Intel® MPI Library can fall back to other fabric(s). See [I_MPI_FALLBACK](#) for details. If the `I_MPI_FABRICS` environment variable is not defined, Intel® MPI Library selects the most appropriate fabric combination automatically.

The exact combination of fabrics depends on the number of processes started per node.

- If all processes start on one node, the library uses `shm` intra-node communication.
- If the number of started processes is less than or equal to the number of available nodes, the library uses the first available fabric from the fabrics list for inter-nodes communication.
- For other cases, the library uses `shm` for intra-node communication, and the first available fabric from the fabrics list for inter-nodes communication. See [I_MPI_FABRICS_LIST](#) for details.

The `shm` fabric is available for both Intel® and non-Intel microprocessors, but it may perform additional optimizations for Intel microprocessors than it performs for non-Intel microprocessors.

NOTE: The combination of selected fabrics ensures that the job runs, but this combination may not provide the highest possible performance for the given cluster configuration.

For example, to select shared-memory as the chosen fabric, use the following command:

```
$ mpiexec -n <# of processes> -env I_MPI_FABRICS shm <executable>
```

To select shared-memory and DAPL-capable network fabric as the chosen fabric combination, use the following command:

```
$ mpiexec -n <# of processes> -env I_MPI_FABRICS shm:dapl <executable>
```

To enable Intel® MPI Library to select most appropriate fabric combination automatically, use the following command:

```
$ mpiexec -n <# of procs> -machinefile smpd.hosts <executable>
```

Set the level of debug information to 2 or higher to check which fabrics have been initialized. See [I_MPI_DEBUG](#) for details. For example:

```
[0] MPI startup(): shm and dapl data transfer modes
```

or

```
[0] MPI startup(): tcp data transfer mode
```

NOTE: If the `I_MPI_FABRICS` environment variable and the `I_MPI_DEVICE` environment variable are set at the same level (command line, environment, configuration files), the `I_MPI_FABRICS` environment variable has higher priority than the `I_MPI_DEVICE` environment variable.

I_MPI_FABRICS_LIST

Define a fabrics list.

Syntax

```
I_MPI_FABRICS_LIST=<fabrics list>
```

Where *<fabrics list>* := *<fabric>*,...,*<fabric>*

```
<fabric> := {dapl, tcp}
```

Arguments

<code><fabrics list></code>	Specify a fabrics list. The default value is <code>dapl, tcp</code>
-----------------------------------	---

Description

Set this environment variable to define a list of fabrics. The library uses the fabrics list to choose the most appropriate fabrics combination automatically. For information on fabric combination, see [I_MPI_FABRICS](#).

For example, if `I_MPI_FABRICS_LIST=dapl, tcp`, `I_MPI_FABRICS` is not defined and the initialization of DAPL-capable network fabrics fails, the library falls back to TCP-capable network fabric. For information on fallback, see [I_MPI_FALLBACK](#).

I_MPI_FALLBACK

(I_MPI_FALLBACK_DEVICE)

Set this environment variable to enable fallback to the first available fabric.

Syntax

```
I_MPI_FALLBACK=<arg>
```

Deprecated Syntax

```
I_MPI_FALLBACK_DEVICE=<arg>
```

Arguments

<code><arg></code>	Binary indicator
<code>enable yes on 1</code>	Fall back to the first available fabric. This is the default value if <code>I_MPI_FABRICS(I_MPI_DEVICE)</code> environment variable is not set
<code>disable no off 0</code>	Terminate the job if the library cannot initialize the one of the fabrics selected by the <code>I_MPI_FABRICS</code> environment variable. This is the default value if you set <code>I_MPI_FABRICS(I_MPI_DEVICE)</code> environment variable

Description

Set this environment variable to control fallback to the first available fabric.

If `I_MPI_FALLBACK` is set to `enable` and an attempt to initialize a specified fabric fails, the library uses the first available fabric from the list of fabrics. See [I_MPI_FABRICS_LIST](#) for details.

If `I_MPI_FALLBACK` is set to `disable` and an attempt to initialize a specified fabric fails, the library terminates the MPI job.

NOTE: If `I_MPI_FABRICS` is set and `I_MPI_FALLBACK=enable`, the library falls back to fabrics with higher numbers in the fabrics list. For example, if `I_MPI_FABRICS=dapl, I_MPI_FABRICS_LIST=dapl, tcp, I_MPI_FALLBACK=enable` and the initialization of DAPL-capable network fabrics fails, the library falls back to TCP-capable network fabric.

I_MPI_EAGER_THRESHOLD

Change the eager/rendezvous message size threshold for all devices.

Syntax

```
I_MPI_EAGER_THRESHOLD=<nbytes>
```

Arguments

<code><nbytes></code>	Set the eager/rendezvous message size threshold
<code>> 0</code>	The default <code><nbytes></code> value is equal to 262144 bytes

Description

Set this environment variable to control the protocol used for point-to-point communication:

- Messages shorter than or equal in size to `<nbytes>` are sent using the eager protocol.
- Messages larger than `<nbytes>` are sent using the rendezvous protocol. The rendezvous protocol uses memory more efficiently.

I_MPI_INTRANODE_EAGER_THRESHOLD

Change the eager/rendezvous message size threshold for intra-node communication mode.

Syntax

`I_MPI_INTRANODE_EAGER_THRESHOLD=<nbytes>`

Arguments

<code><nbytes></code>	Set the eager/rendezvous message size threshold for intra-node communication
<code>> 0</code>	The default <code><nbytes></code> value is equal to 262144 bytes for all fabrics except <code>shm</code> . For <code>shm</code> , cutover point is equal to the value of <code>I_MPI_SHM_CELL_SIZE</code> environment variable

Description

Set this environment variable to change the protocol used for communication within the node:

- Messages shorter than or equal in size to `<nbytes>` are sent using the eager protocol.
- Messages larger than `<nbytes>` are sent using the rendezvous protocol. The rendezvous protocol uses the memory more efficiently.

If `I_MPI_INTRANODE_EAGER_THRESHOLD` is not set, the value of `I_MPI_EAGER_THRESHOLD` is used.

I_MPI_INTRANODE_DIRECT_COPY

Turn on/off the intranode direct copy communication mode.

Syntax

`I_MPI_INTRANODE_DIRECT_COPY=<arg>`

Arguments

<code><arg></code>	Binary indicator
<code>enable yes on 1</code>	Turn on the direct copy communication mode. This is the default value
<code>disable no off 0</code>	Turn off the direct copy communication mode

Description

Set this environment variable to specify the communication mode within the node. If the direct copy communication mode is enabled, data transfer algorithms are selected according to the following scheme:

- Messages shorter than or equal to the threshold value of the `I_MPI_INTRANODE_EAGER_THRESHOLD` environment variable are transferred using the shared memory.
- Messages larger than the threshold value of the `I_MPI_INTRANODE_EAGER_THRESHOLD` environment variable are transferred through the direct process memory access.

I_MPI_SPIN_COUNT

Control the spin count value.

Syntax

`I_MPI_SPIN_COUNT=<scout>`

Arguments

<code><scout></code>	Define the loop spin count when polling fabric(s)
<code>> 0</code>	The default <code><scout></code> value is equal to <code>1</code> when more than one process runs per processor/core. Otherwise the value equals <code>250</code>

Description

Set the spin count limit. The loop for polling the fabric(s) spins `<scout>` times before freeing the processes if no incoming messages are received for processing. Smaller values for `<scout>` cause the Intel® MPI Library to release the processor more frequently.

Use the `I_MPI_SPIN_COUNT` environment variable for tuning application performance. The best value for `<scout>` can be chosen on an experimental basis. It depends on the particular computational environment and application.

I_MPI_SCALABLE_OPTIMIZATION

Turn on/off scalable optimization of the network fabric communication.

Syntax

`I_MPI_SCALABLE_OPTIMIZATION=<arg>`

Deprecated Syntax

`I_MPI SOCK_SCALABLE_OPTIMIZATION=<arg>`

Arguments

<code><arg></code>	Binary indicator
<code>enable yes on 1</code>	Turn on scalable optimization of the network fabric communication. This is the default for 16 or more processes
<code>disable no off 0</code>	Turn off scalable optimization of the network fabric communication. This is the default for less than 16 processes

Description

Set this environment variable to enable scalable optimization of the network fabric communication. In most cases, using optimization decreases latency and increases bandwidth for a large number of processes.

I_MPI_WAIT_MODE

Turn on/off wait mode.

Syntax`I_MPI_WAIT_MODE=<arg>`**Arguments**

<code><arg></code>	Binary indicator
<code>enable yes on 1</code>	Turn on the wait mode
<code>disable no off 0</code>	Turn off the wait mode. This is the default

Description

Set this environment variable to control the wait mode. If this mode is enabled, the processes wait for receiving messages without polling the fabric(s). This mode can save CPU time for other tasks.

I_MPI_DYNAMIC_CONNECTION**(I_MPI_USE_DYNAMIC_CONNECTIONS)**

Turn on/off the dynamic connection establishment.

Syntax`I_MPI_DYNAMIC_CONNECTION=<arg>`**Deprecated Syntax**`I_MPI_USE_DYNAMIC_CONNECTIONS=<arg>`**Arguments**

<code><arg></code>	Binary indicator
<code>enable yes on 1</code>	Turn on the dynamic connection establishment. This is the default for 64 or more processes
<code>disable no off 0</code>	Turn off the dynamic connection establishment. This is the default for less than 64 processes

Description

Set this environment variable to control dynamic connection establishment.

- If this mode is enabled, all connections are established at the time of the first communication between each pair of processes.
- If this mode is disabled, all connections are established upfront.

The default value depends on a number of processes in the MPI job. The dynamic connection establishment is off if a total number of processes is less than 64.

4.3.2 Shared Memory Control**I_MPI_SHM_CACHE_BYPASS****(I_MPI_CACHE_BYPASS)**

Control the message transfer algorithm for the shared memory.

Syntax

`I_MPI_SHM_CACHE_BYPASS=<arg>`

Deprecated Syntax

`I_MPI_CACHE_BYPASS=<arg>`

Arguments

<code><arg></code>	Binary indicator
<code>enable yes on 1</code>	Enable message transfer bypass cache. This is the default value
<code>disable no off 0</code>	Disable message transfer bypass cache

Description

Set this environment variable to enable/disable message transfer bypass cache for the shared memory. When enabled, the MPI sends the messages greater than or equal in size to the value specified by the `I_MPI_SHM_CACHE_BYPASS_THRESHOLD` environment variable through the bypass cache. This feature is enabled by default.

I_MPI_SHM_CACHE_BYPASS_THRESHOLDS

(I_MPI_CACHE_BYPASS_THRESHOLDS)

Set the messages copying algorithm threshold.

Syntax

`I_MPI_SHM_CACHE_BYPASS_THRESHOLDS=<nb_send>, <nb_recv> [, <nb_send_pk>, <nb_recv_pk>]`

Deprecated Syntax

`I_MPI_CACHE_BYPASS_THRESHOLDS=<nb_send>, <nb_recv> [, <nb_send_pk>, <nb_recv_pk>]`

Arguments

<code><nb_send></code>	Set the threshold for sent messages in the following situations: Processes are pinned on cores that are not located in the same physical processor package Processes are not pinned
<code>>= 0</code>	<ul style="list-style-type: none"> For machines optimized with Intel® Streaming SIMD Extensions 4.2 (Intel® SSE4.2) and Intel® AES New Instructions (Intel® AES-NI), the default <code><nb_send></code> value is <code>-1</code>. This value disables the copying bypass cache For other architectures, the default <code><nb_send></code> value is <code>16,384</code> bytes
<code><nb_recv></code>	Set the threshold for received messages in the following situations: Processes are pinned on cores that are not located in the same physical processor package Processes are not pinned
<code>>= 0</code>	<ul style="list-style-type: none"> For machines optimized with Intel® SSE4.2, the default <code><nb_send></code> value is <code>-1</code>. This value disables the copying bypass cache For machines optimized with Intel® AES-NI, the default <code><nb_send></code> value is <code>MAX(1Mb, L3/NP)</code>, where <code>L3</code> indicates the size of Level 3 cache and <code>NP</code> indicates the number of processes

	<p>on the node</p> <ul style="list-style-type: none"> For other architectures, the default <code><nb_recv_pk></code> value is <code>2,097,152</code> bytes
<code><nb_send_pk></code>	Set the threshold for sent messages when processes are pinned on cores located in the same physical processor package
<code>>= 0</code>	The default <code><nb_send_pk></code> value is <code>-1</code> (copying bypass cache is disabled)
<code><nb_recv_pk></code>	Set the threshold for received messages when processes are pinned on cores located in the same physical processor package
<code>>= 0</code>	<ul style="list-style-type: none"> For machines optimized with Intel® SSE4.2, the default <code><nb_send></code> value is <code>-1</code>. This value disables the copying bypass cache For machines optimized with Intel® AES-NI, the default <code><nb_send></code> value is <code>MAX(1Mb, L3/NP)</code>, where <code>L3</code> indicates the size of Level 3 cache and <code>NP</code> indicates the number of processes on the node For other architectures, the default <code><nb_recv_pk></code> value is <code>2,097,152</code> bytes

Description

Set this environment variable to control the thresholds for the message copying algorithm. MPI copies messages greater than or equal in size to the defined threshold values so that the messages bypass the cache. The value of `-1` disables cache bypass. This environment variable is valid only when `I_MPI_SHM_CACHE_BYPASS` is enabled.

This environment variable is available for both Intel and non-Intel microprocessors, but it may perform additional optimizations for Intel microprocessors than it performs for non-Intel microprocessors.

I_MPI_SHM_FBOX

Control the usage of the shared memory fast-boxes.

Syntax

`I_MPI_SHM_FBOX=<arg>`

Arguments

<code><arg></code>	Binary indicator
<code>enable yes on 1</code>	Turn on the usage of fast-boxes. This is the default value
<code>disable no off 0</code>	Turn off the usage of fast-boxes

Description

Set this environment variable to control the usage of fast-boxes. Each pair of MPI processes on the same computing node has two shared memory fast-boxes, for sending and receiving eager messages.

Turn off the usage of fast-boxes to avoid the overhead of message synchronization when the application uses mass transfer of short non-blocking messages.

I_MPI_SHM_FBOX_SIZE

Set the size of the shared memory fast-box.

Syntax

`I_MPI_SHM_FBOX_SIZE=<nbytes>`

Arguments

<code><nbytes></code>	Size of shared memory fastbox in bytes
<code>> 0</code>	The default <code><nbytes></code> value is equal to <code>65,472</code> bytes

Description

Set this environment variable to define the size of shared memory fast-boxes. The value must be multiple of 64.

I_MPI_SHM_CELL_NUM

Change the number of cells in the shared memory receiving queue.

Syntax

`I_MPI_SHM_CELL_NUM=<num>`

Arguments

<code><num></code>	The number of shared memory cells
<code>> 0</code>	The default value is <code>128</code>

Description

Set this environment variable to define the number of cells in the shared memory receive queue. Each MPI process has own shared memory receive queue, where other processes put eager messages. The queue is used when shared memory fast-boxes are blocked by another MPI request.

I_MPI_SHM_CELL_SIZE

Change the size of shared memory cell.

Syntax

`I_MPI_SHM_CELL_SIZE=<nbytes>`

Arguments

<code><nbytes></code>	Size of shared memory cell in bytes
<code>> 0</code>	The default <code><nbytes></code> value is equal to <code>65,472</code> bytes

Description

Set this environment variable to define the size of shared memory cells. The value must be a multiple of 64.

If a value is set, `I_MPI_INTRANODE_EAGER_THRESHOLD` is also changed and becomes equal to the given value.

I_MPI_SHM_LMT

Control the usage of large message transfer (LMT) mechanism for the shared memory.

Syntax

`I_MPI_SHM_LMT=<arg>`

Deprecated Syntax

`I_MPI_INTRANODE_DIRECT_COPY=<arg>`

Arguments

<code><arg></code>	Binary indicator
<code>shm</code>	Turn on the usage of shared memory copy LMT mechanism. This is the default value on Linux OS*
<code>direct</code>	Turn on the usage of direct copy LMT mechanism. This is the default value on Windows OS*
<code>disable no off 0</code>	Turn off the usage of LMT mechanism

Description

Set this environment variable to control the usage of the large message transfer (LMT) mechanism. To transfer rendezvous messages, you can use the LMT mechanism by employing either of the following implementations:

- Use intermediate shared memory queues to send messages.
- Use direct copy mechanism that transfers messages without intermediate buffer.

NOTE: Two arguments of the `I_MPI_SHM_LMT` environment variable are related to the `I_MPI_INTRANODE_DIRECT_COPY` environment variable:

- `I_MPI_SHM_LMT=direct` is equal to the deprecated setting `I_MPI_INTRANODE_DIRECT_COPY=enable`.
- `I_MPI_SHM_LMT=shm` is equal to the deprecated setting `I_MPI_INTRANODE_DIRECT_COPY=disable`.

`I_MPI_SHM_LMT_BUFFER_NUM`

`(I_MPI_SHM_NUM_BUFFERS)`

Change the number of shared memory buffers for the large message transfer (LMT) mechanism.

Syntax

`I_MPI_SHM_LMT_BUFFER_NUM=<num>`

Deprecated Syntax

`I_MPI_SHM_NUM_BUFFERS=<num>`

Arguments

<code><num></code>	The number of shared memory buffers for each process pair
<code>> 0</code>	The default value is 8

Description

Set this environment variable to define the number of shared memory buffers between each process pair.

`I_MPI_SHM_LMT_BUFFER_SIZE`

`(I_MPI_SHM_BUFFER_SIZE)`

Change the size of shared memory buffers for the LMT mechanism.

Syntax

```
I_MPI_SHM_LMT_BUFFER_SIZE=<nbytes>
```

Deprecated Syntax

```
I_MPI_SHM_BUFFER_SIZE=<nbytes>
```

Arguments

<code><nbytes></code>	The size of shared memory buffers in bytes
<code>> 0</code>	The default <code><nbytes></code> value is equal to <code>32,768</code> bytes

Description

Set this environment variable to define the size of shared memory buffers for each pair of processes.

I_MPI_SSHM

Control the usage of the scalable shared memory mechanism.

Syntax

```
I_MPI_SSHM =<arg>
```

Arguments

<code><arg></code>	Binary indicator
<code>enable yes on 1</code>	Turn on the usage of this mechanism
<code>disable no off 0</code>	Turn off the usage of this mechanism. This is the default value

Description

Set this environment variable to control the usage of an alternative shared memory mechanism. This mechanism replaces the shared memory fast-boxes, receive queues and LMT mechanism.

If a value is set, the `I_MPI_INTRANODE_EAGER_THRESHOLD` environment variable is changed and becomes equal to `262,144` bytes.

I_MPI_SSHM_BUFFER_NUM

Change the number of shared memory buffers for the alternative shared memory mechanism.

Syntax

```
I_MPI_SSHM_BUFFER_NUM=<num>
```

Arguments

<code><num></code>	The number of shared memory buffers for each process pair
<code>> 0</code>	The default value is <code>4</code>

Description

Set this environment variable to define the number of shared memory buffers between each process pair.

I_MPI_SSHM_BUFFER_SIZE

Change the size of shared memory buffers for the alternative shared memory mechanism.

Syntax

```
I_MPI_SSHM_BUFFER_SIZE=<nbytes>
```

Arguments

<code><nbytes></code>	The size of shared memory buffers in bytes
<code>> 0</code>	The default <code><nbytes></code> value is <code>65,472</code> bytes

Description

Set this environment variable to define the size of shared memory buffers for each pair of processes.

I_MPI_SSHM_DYNAMIC_CONNECTION

Control the dynamic connection establishment for the alternative shared memory mechanism.

Syntax

```
I_MPI_SSHM_DYNAMIC_CONNECTION=<arg>
```

Arguments

<code><arg></code>	Binary indicator
<code>enable yes on 1</code>	Turn on the dynamic connection establishment
<code>disable no off 0</code>	Turn off the dynamic connection establishment. This is the default value

Description

Set this environment variable to control the dynamic connection establishment.

- If this mode is enabled, all connections are established at the time of the first communication between each pair of processes.
- If this mode is disabled, all connections are established upfront.

I_MPI_SHM_BYPASS**(I_MPI_INTRANODE_SHMEM_BYPASS, I_MPI_USE_DAPL_INTRANODE)**

Turn on/off the intra-node communication mode through network fabric along with `shm`.

Syntax

```
I_MPI_SHM_BYPASS=<arg>
```

Deprecated Syntaxes

```
I_MPI_INTRANODE_SHMEM_BYPASS=<arg>
```

```
I_MPI_USE_DAPL_INTRANODE=<arg>
```

Arguments

<code><arg></code>	Binary indicator
<code>enable yes on 1</code>	Turn on the intra-node communication through network fabric
<code>disable no off 0</code>	Turn off the intra-node communication through network fabric. This is the default

Description

Set this environment variable to specify the communication mode within the node. If the intra-node communication mode through network fabric is enabled, data transfer algorithms are selected according to the following scheme:

- Messages shorter than or equal in size to the threshold value of the `I_MPI_INTRANODE_EAGER_THRESHOLD` environment variable are transferred using shared memory.
- Messages larger than the threshold value of the `I_MPI_INTRANODE_EAGER_THRESHOLD` environment variable are transferred through the network fabric layer.

NOTE: This environment variable is applicable only when shared memory and a network fabric are turned on either by default or by setting the `I_MPI_FABRICS` environment variable to `shm:<fabric>` or an equivalent `I_MPI_DEVICE` setting. This mode is available only for `dapl` and `tcp` fabrics.

I_MPI_SHM_SPIN_COUNT

Control the spin count value for the shared memory fabric.

Syntax

`I_MPI_SHM_SPIN_COUNT=<scout>`

Arguments

<code><scout></code>	Define the spin count of the loop when polling the <code>shm</code> fabric
<code>> 0</code>	The default <code><scout></code> value is equal to <code>100</code> spins

Description

Set the spin count limit of the shared memory fabric to increase the frequency of polling. This configuration allows polling of the `shm` fabric `<scout>` times before the control is passed to the overall network fabric polling mechanism. See [I_MPI_SPIN_COUNT](#) for details on higher level fabrics polling.

To tune application performance, use the `I_MPI_SHM_SPIN_COUNT` environment variable. You can choose the best value for `<scout>` on an experimental basis. It depends largely on the application and the particular computation environment. An increase in the `<scout>` value will benefit multi-core platforms when the application uses topological algorithms for message passing.

4.3.3 DAPL-capable Network Fabrics Control

I_MPI_DAPL_PROVIDER

Define the DAPL provider to load.

Syntax

`I_MPI_DAPL_PROVIDER=<name>`

Arguments

<code><name></code>	Define the name of DAPL provider to load
---------------------------	--

Description

This environment variable is applicable only when shared memory and a network fabric are turned on either by default or by setting the `I_MPI_FABRICS` environment variable to `shm:<fabric>` or an equivalent `I_MPI_DEVICE` setting. This mode is available only for `dapl` and `tcp` fabrics.

I_MPI_DAT_LIBRARY

Select the DAT library to be used for DAPL provider.

Syntax

`I_MPI_DAT_LIBRARY=<library>`

Arguments

<code><library></code>	Specify the DAT library for DAPL provider to be used. Default values are <code>dat.dll</code> for DAPL* 1.2 providers and <code>dat2.dll</code> for DAPL* 2.0 providers
------------------------------	---

Description

Set this environment variable to select a specific DAT library to be used for DAPL provider. If the library is not located in the dynamic loader search path, specify the full path to the DAT library. This environment variable affects only DAPL capable fabrics.

I_MPI_DAPL_TRANSLATION_CACHE

(I_MPI_RDMA_TRANSLATION_CACHE)

Turn on/off the memory registration cache in the DAPL path.

Syntax

`I_MPI_DAPL_TRANSLATION_CACHE=<arg>`

Deprecated Syntax

`I_MPI_RDMA_TRANSLATION_CACHE=<arg>`

Arguments

<code><arg></code>	Binary indicator
<code>enable yes on 1</code>	Turn on the memory registration cache. This is the default
<code>disable no off 0</code>	Turn off the memory registration cache

Description

Set this environment variable to turn on/off the memory registration cache in the DAPL path.

The cache substantially increases performance, but may lead to correctness issues in certain rare situations. See product *Release Notes* for further details.

I_MPI_DAPL_TRANSLATION_CACHE_AVL_TREE

Enable/disable the AVL tree* based implementation of the RDMA translation cache in the DAPL path.

Syntax

`I_MPI_DAPL_TRANSLATION_CACHE_AVL_TREE=<arg>`

Arguments

<code><arg></code>	Binary indicator
<code>enable yes on 1</code>	Turn on the AVL tree based RDMA translation cache
<code>disable no off 0</code>	Turn off the AVL tree based RDMA translation cache. This is the default value

Description

Set this environment variable to enable the AVL tree based implementation of RDMA translation cache in the DAPL path. In some cases, the AVL tree based RDMA translation cache is faster than the default implementation.

I_MPI_DAPL_DIRECT_COPY_THRESHOLD**(I_MPI_RDMA_EAGER_THRESHOLD, RDMA_IBA_EAGER_THRESHOLD)**

Change the threshold of the DAPL direct-copy protocol.

Syntax

`I_MPI_DAPL_DIRECT_COPY_THRESHOLD=<nbytes>`

Deprecated Syntaxes

`I_MPI_RDMA_EAGER_THRESHOLD=<nbytes>`

`RDMA_IBA_EAGER_THRESHOLD=<nbytes>`

Arguments

<code><nbytes></code>	Define the DAPL direct-copy protocol threshold
<code>> 0</code>	The default <code><nbytes></code> value is equal to <code>16,456</code> bytes

Description

Set this environment variable to control the DAPL direct-copy protocol threshold. Data transfer algorithms for the DAPL-capable network fabrics are selected based on the following scheme:

- Messages shorter than or equal to `<nbytes>` are sent through the internal pre-registered buffers. This approach is faster for short messages.
- Messages larger than `<nbytes>` are sent using the direct copy protocol. It does not use any buffering but involves registration of memory on sender and receiver sides. This approach is faster for large messages.

This environment variable is available for both Intel® and non-Intel microprocessors, but it may perform additional optimizations for Intel microprocessors than it performs for non-Intel microprocessors.

I_MPI_DAPL_EAGER_MESSAGE_AGGREGATION

Control the use of concatenation for adjourned MPI send requests. Adjourned MPI send requests are those that cannot be sent immediately.

Syntax

`I_MPI_DAPL_EAGER_MESSAGE_AGGREGATION=<arg>`

Arguments

<code><arg></code>	Binary indicator
<code>enable yes on 1</code>	Enable the concatenation for adjourned MPI send requests
<code>disable no off 0</code>	Disable the concatenation for adjourned MPI send requests. This is the default value

Set this environment variable to control the use of concatenation for adjourned MPI send requests intended for the same MPI rank. In some cases, this mode can improve the performance of applications, especially when it calls `MPI_Isend()` is used with short message sizes and the same destination rank, such as:

```

    for( i = 0; i< NMSG; i++)
    {
        ret = MPI_Isend( sbuf[i], MSG_SIZE, datatype, dest , tag, comm,
&req_send[i]);
    }

```

I_MPI_DAPL_DYNAMIC_CONNECTION_MODE

(I_MPI_DYNAMIC_CONNECTION_MODE, I_MPI_DYNAMIC_CONNECTIONS_MODE)

Choose the algorithm for establishing the DAPL* connections.

Syntax

I_MPI_DAPL_DYNAMIC_CONNECTION_MODE=<arg>

Deprecated Syntax

I_MPI_DYNAMIC_CONNECTION_MODE=<arg>

I_MPI_DYNAMIC_CONNECTIONS_MODE=<arg>

Arguments

<arg>	Mode selector
reject	Deny one of the two simultaneous connection requests. This is the default
disconnect	Deny one of the two simultaneous connection requests after both connections have been established

Description

Set this environment variable to choose the algorithm for handling dynamically established connections for DAPL-capable fabrics according to the following scheme:

- In the **reject** mode, if two processes initiate the connection simultaneously, one of the requests is rejected.
- In the **disconnect** mode, both connections are established, but then one is disconnected. The **disconnect** mode is provided to avoid a bug in certain DAPL* providers.

I_MPI_DAPL_SCALABLE_PROGRESS

(I_MPI_RDMA_SCALABLE_PROGRESS)

Turn on/off scalable algorithm for DAPL read progress.

Syntax

I_MPI_DAPL_SCALABLE_PROGRESS=<arg>

Deprecated Syntax

I_MPI_RDMA_SCALABLE_PROGRESS=<arg>

Arguments

<arg>	Binary indicator
enable yes on 1	Turn on scalable algorithm. When the number of processes is larger than 128, this is the default value

<code>disable no off 0</code>	Turn off scalable algorithm. When the number of processes is less than or equal to <code>128</code> , this is the default value
-------------------------------------	---

Description

Set this environment variable to enable scalable algorithm for the DAPL read progress. In some cases, this provides advantages for systems with many processes.

I_MPI_DAPL_BUFFER_NUM

(I_MPI_RDMA_BUFFER_NUM, NUM_RDMA_BUFFER)

Change the number of internal pre-registered buffers for each process pair in the DAPL path.

Syntax

`I_MPI_DAPL_BUFFER_NUM=<nbuf>`

Deprecated Syntaxes

`I_MPI_RDMA_BUFFER_NUM=<nbuf>`

`NUM_RDMA_BUFFER=<nbuf>`

Arguments

<code><nbuf></code>	Define the number of buffers for each pair in a process group
<code>> 0</code>	The default value is <code>16</code>

Description

Set this environment variable to change the number of the internal pre-registered buffers for each process pair in the DAPL path.

NOTE: The more pre-registered buffers are available, the more memory is used for every established connection.

I_MPI_DAPL_BUFFER_SIZE

(I_MPI_RDMA_BUFFER_SIZE, I_MPI_RDMA_VBUF_TOTAL_SIZE)

Change the size of internal pre-registered buffers for each process pair in the DAPL path.

Syntax

`I_MPI_DAPL_BUFFER_SIZE=<nbytes>`

Deprecated Syntaxes

`I_MPI_RDMA_BUFFER_SIZE=<nbytes>`

`I_MPI_RDMA_VBUF_TOTAL_SIZE=<nbytes>`

Arguments

<code><nbytes></code>	Define the size of pre-registered buffers
<code>> 0</code>	The default <code><nbytes></code> value is equal to <code>16,640</code> bytes

Description

Set this environment variable to define the size of the internal pre-registered buffer for each process pair in the DAPL path. The actual size is calculated by adjusting the `<nbytes>` to align the buffer to an optimal value.

I_MPI_DAPL_RNDV_BUFFER_ALIGNMENT

(I_MPI_RDMA_RNDV_BUFFER_ALIGNMENT, I_MPI_RDMA_RNDV_BUF_ALIGN)

Define the alignment of the sending buffer for the DAPL direct-copy transfers.

Syntax

I_MPI_DAPL_RNDV_BUFFER_ALIGNMENT=<arg>

Deprecated Syntaxes

I_MPI_RDMA_RNDV_BUFFER_ALIGNMENT=<arg>

I_MPI_RDMA_RNDV_BUF_ALIGN=<arg>

Arguments

<arg>	Define the alignment for the sending buffer
> 0 and a power of 2	The default value is 128

Set this environment variable to define the alignment of the sending buffer for DAPL direct-copy transfers. When a buffer specified in a DAPL operation is aligned to an optimal value, data transfer bandwidth may be increased.

I_MPI_DAPL_RDMA_RNDV_WRITE

(I_MPI_RDMA_RNDV_WRITE, I_MPI_USE_RENDEZVOUS_RDMA_WRITE)

Turn on/off the RDMA Write-based rendezvous direct-copy protocol in the DAPL path.

Syntax

I_MPI_DAPL_RDMA_RNDV_WRITE=<arg>

Deprecated Syntaxes

I_MPI_RDMA_RNDV_WRITE=<arg>

I_MPI_USE_RENDEZVOUS_RDMA_WRITE=<arg>

Arguments

<arg>	Binary indicator
enable yes on 1	Turn on the RDMA Write rendezvous direct-copy protocol
disable no off 0	Turn off the RDMA Write rendezvous direct-copy protocol

Description

Set this environment variable to select the RDMA Write-based rendezvous direct-copy protocol in the DAPL path. Certain DAPL* providers have a slow RDMA Read implementation on certain platforms. Switching on the rendezvous direct-copy protocol based on the RDMA Write operation can increase performance in these cases. The default value depends on the DAPL provider attributes.

I_MPI_DAPL_CHECK_MAX_RDMA_SIZE

(I_MPI_RDMA_CHECK_MAX_RDMA_SIZE)

Check the value of the DAPL attribute `max_rdma_size`.

Syntax

`I_MPI_DAPL_CHECK_MAX_RDMA_SIZE=<arg>`

Deprecated Syntax

`I_MPI_RDMA_CHECK_MAX_RDMA_SIZE=<arg>`

Arguments

<code><arg></code>	Binary indicator
<code>enable yes on 1</code>	Check the value of the DAPL* attribute <code>max_rdma_size</code>
<code>disable no off 0</code>	Do not check the value of the DAPL* attribute <code>max_rdma_size</code> . This is the default value

Description

Set this environment variable to control message fragmentation according to the following scheme:

- If this mode is enabled, the Intel® MPI Library fragmentizes the messages bigger than the value of the DAPL attribute `max_rdma_size`
- If this mode is disabled, the Intel® MPI Library does not take into account the value of the DAPL attribute `max_rdma_size` for message fragmentation

I_MPI_DAPL_MAX_MSG_SIZE

(I_MPI_RDMA_MAX_MSG_SIZE)

Control message fragmentation threshold.

Syntax

`I_MPI_DAPL_MAX_MSG_SIZE=<nbytes>`

Deprecated Syntax

`I_MPI_RDMA_MAX_MSG_SIZE=<nbytes>`

Arguments

<code><nbytes></code>	Define the maximum message size that can be sent through DAPL without fragmentation
<code>> 0</code>	If the <code>I_MPI_DAPL_CHECK_MAX_RDMA_SIZE</code> environment variable is enabled, the default <code><nbytes></code> value is equal to the <code>max_rdma_size</code> DAPL* attribute value. Otherwise the default value is <code>MAX_INT</code>

Description

Set this environment variable to control message fragmentation size according to the following scheme:

- If the `I_MPI_DAPL_CHECK_MAX_RDMA_SIZE` environment variable is set to `disable`, the Intel® MPI Library fragmentizes the messages whose sizes are greater than `<nbytes>`.
- If the `I_MPI_DAPL_CHECK_MAX_RDMA_SIZE` environment variable is set to `enable`, the Intel® MPI Library fragmentizes the messages whose sizes are greater than the minimum of `<nbytes>` and the `max_rdma_size` DAPL* attribute value.

I_MPI_DAPL_CONN_EVD_SIZE

(I_MPI_RDMA_CONN_EVD_SIZE, I_MPI_CONN_EVD_QLEN)

Define the event queue size of the DAPL event dispatcher for connections.

Syntax

```
I_MPI_DAPL_CONN_EVD_SIZE=<size>
```

Deprecated Syntaxes

```
I_MPI_RDMA_CONN_EVD_SIZE=<size>
```

```
I_MPI_CONN_EVD_QLEN=<size>
```

Arguments

<size>	Define the length of the event queue
> 0	The default value is $2 * \text{number of processes} + 32$ in the MPI job

Description

Set this environment variable to define the event queue size of the DAPL event dispatcher that handles connection related events. If this environment variable is set, the minimum value between <size> and the value obtained from the provider is used as the size of the event queue. The provider is required to supply a queue size that is at least equal to the calculated value, but it can also provide a larger queue size.

I_MPI_DAPL_SR_THRESHOLD

Change the threshold of switching send/recv to rdma path for DAPL wait mode.

Syntax

```
I_MPI_DAPL_SR_THRESHOLD<arg>
```

Arguments

<nbytes>	Define the message size threshold of switching send/recv to rdma
>= 0	The default <nbytes> value is 256 bytes

Description

Set this environment variable to control the protocol used for point-to-point communication in DAPL wait mode:

- Messages shorter than or equal in size to <nbytes> are sent using DAPL send/recv data transfer operations.
- Messages greater in size than <nbytes> are sent using DAPL RDMA WRITE or RDMA WRITE immediate data transfer operations.

I_MPI_DAPL_SR_BUF_NUM

Change the number of internal pre-registered buffers for each process pair used in DAPL wait mode for send/recv path.

Syntax

```
I_MPI_DAPL_SR_BUF_NUM=<nbuf>
```

Arguments

<nbuf>	Define the number of send/recv buffers for each pair in a process
--------	---

	group
> 0	The default value is 32

Description

Set this environment variable to change the number of the internal send/recv pre-registered buffers for each process pair.

I_MPI_DAPL_RDMA_WRITE_IMM**(I_MPI_RDMA_WRITE_IMM)**

Enable/disable RDMA Write with immediate data InfiniBand* (IB) extension in DAPL wait mode.

Syntax

`I_MPI_DAPL_RDMA_WRITE_IMM=<arg>`

Deprecated syntax

`I_MPI_RDMA_WRITE_IMM=<arg>`

Arguments

<code><arg></code>	Binary indicator
<code>enable yes on 1</code>	Turn on RDMA Write with immediate data IB extension
<code>disable no off 0</code>	Turn off RDMA Write with immediate data IB extension

Description

Set this environment variable to utilize RDMA Write with immediate data IB extension. The algorithm is enabled if this environment variable is set and a certain DAPL provider attribute indicates that RDMA Write with immediate data IB extension is supported.

I_MPI_DAPL_DESIRED_STATIC_CONNECTIONS_NUM

Define the number of processes that establish DAPL static connections at the same time.

Syntax

`I_MPI_DAPL_DESIRED_STATIC_CONNECTIONS_NUM=<num_processes>`

Arguments

<code><num_processes></code>	Define the number of processes that establish DAPL static connections at the same time
> 0	The default <code><num_processes></code> value is equal to 256

Description

Set this environment variable to control the algorithm of DAPL static connections establishment.

If the number of processes in the MPI job is less than or equal to `<num_processes>`, all MPI processes establish the static connections simultaneously. Otherwise, the processes are distributed into several groups. The number of processes in each group is calculated to be close to `<num_processes>`. Then static connections are established, in several iterations, including the intergroup connection setup.

4.3.4 TCP-capable Network Fabrics Control

`I_MPI_TCP_NETMASK`

(`I_MPI_NETMASK`)

Choose the network interface for MPI communication over TCP-capable network fabrics.

Syntax

`I_MPI_TCP_NETMASK=<arg>`

Arguments

<code><arg></code>	Define the network interface (string parameter)
<code><interface_mnemonic></code>	Mnemonic of the network interface: <code>ib</code> or <code>eth</code>
<code>ib</code>	Select IPoIB*
<code>eth</code>	Select Ethernet. This is the default value
<code><interface_name></code>	Name of the network interface Usually the UNIX* driver name followed by the unit number
<code><network_address>></code>	Network address. The trailing zero bits imply netmask
<code><network_address/netmask></code>	Network address. The <code><netmask></code> value specifies the netmask length
<code><list of interfaces></code>	A colon separated list of network addresses and interface names

Description

Set this environment variable to choose the network interface for MPI communication over TCP-capable network fabrics. If you specify a list of interfaces, the first available interface on the node will be used for communication.

Examples

- Use the following setting to select the IP over InfiniBand (IPoIB) fabric:
`I_MPI_TCP_NETMASK=ib`
- Use the following setting to select the specified network interface for socket communications:
`I_MPI_TCP_NETMASK=ib0`
- Use the following setting to select the specified network for socket communications. This setting implies the `255.255.0.0` netmask:
`I_MPI_TCP_NETMASK=192.169.0.0`
- Use the following setting to select the specified network for socket communications with netmask set explicitly:
`I_MPI_TCP_NETMASK=192.169.0.0/24`
- Use the following setting to select the specified network interfaces for socket communications:
`I_MPI_TCP_NETMASK=192.169.0.5/24:ib0:192.169.0.0`

`I_MPI_TCP_BUFFER_SIZE`

Change the size of the TCP socket buffers.

Syntax

`I_MPI_TCP_BUFFER_SIZE=<nbytes>`

Arguments

<code><nbytes></code>	Define the size of the TCP socket buffers
<code>> 0</code>	The default <code><nbytes></code> value is equal to 128 Kb.

Description

Set this environment variable to define the size of the TCP socket buffers.

Use the `I_MPI_TCP_BUFFER_SIZE` environment variable for tuning your application's performance for a given number of processes.

NOTE: TCP socket buffers of a large size can require more memory for an application with large number of processes. Alternatively, TCP socket buffers of a small size can considerably decrease the bandwidth of each socket connection especially for 10 Gigabit Ethernet and IPoIB (see [I_MPI_TCP_NETMASK](#) for details).

4.4 Dynamic Process Support

The Intel® MPI Library provides support for the MPI-2 process model what allows creation and cooperative termination of processes after an MPI application has started. It provides

- a mechanism to establish communication between the newly created processes and the existing MPI application
- a process attachment mechanism to establish communication between two existing MPI applications even when one of them does not spawn the other

The set of hosts indicated within `<machine_file>` (see [mpixec](#) for details) is used for placement of spawned processes. The spawned processes are placed onto different hosts in round-robin or per-host fashion. The first spawned process is placed after the last process of the parent group. A specific network fabric combination is selected using the usual fabrics selection algorithm (see [I_MPI_FABRICS](#) and [I_MPI_FABRICS_LIST](#) for details).

For example, to run a dynamic application, use the following command:

```
$ mpiexec -n 1 -machinefile smpd.hosts -gwdir <path_to_executable>
-genv I_MPI_FABRICS shm:tcp <spawn_app>
```

In this example, the `spawn_app` spawns 4 dynamic processes. If the `smpd.hosts` contains the following information:

```
host1
host2
host3
host4
```

the original spawning process is placed on `host1`, while the dynamic processes is distributed as follows: 1 – on `host2`, 2 – on `host3`, 3 – on `host4`, and 4 – again on `host1`.

If the `smpd.hosts` contains the following information:

```
host1:2
host2:2
```

the ordinary process is placed on `host1`, while the dynamic processes is distributed as follows: 1 – on `host1`, 2 and 3 – on `host2`, and 4 – on `host1`.

To run a client-server application, use the following command on the server host:

```
$ mpiexec -n 1 -genv I_MPI_FABRICS shm:tcp <server_app> > <port_name>
```

and the following command on the intended client hosts:

```
$ mpiexec -n 1 -genv I_MPI_FABRICS shm:tcp <client_app> < <port_name>
```

To run a simple MPI_COMM_JOIN based application, use the following commands on the intended host:

```
$ mpiexec -n 1 -genv I_MPI_FABRICS shm:tcp <join_server_app> < <port_number>
```

```
$ mpiexec -n 1 -genv I_MPI_FABRICS shm:tcp <join_client_app> < <port_number>
```

4.5 Collective Operation Control

Each collective operation in the Intel® MPI Library supports a number of communication algorithms. In addition to highly optimized default settings, the library provides two ways to control the algorithm selection explicitly: the novel `I_MPI_ADJUST` environment variable family and the deprecated `I_MPI_MSG` environment variable family. They are described in the following sections.

These environment variables are available for both Intel® and non-Intel microprocessors, but they may perform additional optimizations for Intel microprocessors than they perform for non-Intel microprocessors.

4.5.1 I_MPI_ADJUST Family

I_MPI_ADJUST_<opname>

Control collective operation algorithm selection.

Syntax

```
I_MPI_ADJUST_<opname>=<algid>[:<conditions>] [;<algid>:<conditions>[...]]
```

Arguments

<algid>	Algorithm identifier
>= 0	The default value of zero selects optimized default settings

<conditions>	A comma separated list of conditions. An empty list selects all message sizes and process combinations
<l>	Messages of size <l>
<l>-<m>	Messages of size from <l> to <m>, inclusive
<l>@<p>	Messages of size <l> and number of processes <p>
<l>-<m>@<p>-<q>	Messages of size from <l> to <m> and number of processes from <p> to <q>, inclusive

Description

Set this environment variable to select the desired algorithm(s) for the collective operation *<opname>* under particular conditions. Each collective operation has its own environment variable and algorithms. See below.

Table 4.5-1 Environment Variables, Collective Operations, and Algorithms

Environment Variable	Collective Operation	Algorithms
<code>I_MPI_ADJUST_ALLGATHER</code>	<code>MPI_Allgather</code>	<ol style="list-style-type: none"> 1. Recursive doubling algorithm 2. Bruck's algorithm 3. Ring algorithm 4. Topology aware Gather + Bcast algorithm
<code>I_MPI_ADJUST_ALLGATHERV</code>	<code>MPI_Allgatherv</code>	<ol style="list-style-type: none"> 1. Recursive doubling algorithm 2. Bruck's algorithm 3. Ring algorithm 4. Topology aware Gather + Bcast algorithm
<code>I_MPI_ADJUST_ALLREDUCE</code>	<code>MPI_Allreduce</code>	<ol style="list-style-type: none"> 1. Recursive doubling algorithm 2. Rabenseifner's algorithm 3. Reduce + Bcast algorithm 4. Topology aware Reduce + Bcast algorithm 5. Binomial gather + scatter algorithm 6. Topology aware binomial gather + scatter algorithm 7. Shumilin's ring algorithm 8. Ring algorithm
<code>I_MPI_ADJUST_ALLTOALL</code>	<code>MPI_Alltoall</code>	<ol style="list-style-type: none"> 1. Bruck's algorithm 2. Isend/Irecv + waitall algorithm 3. Pair wise exchange algorithm 4. Plum's algorithm
<code>I_MPI_ADJUST_ALLTOALLV</code>	<code>MPI_Alltoallv</code>	<ol style="list-style-type: none"> 1. Isend/Irecv + waitall algorithm 2. Plum's algorithm
<code>I_MPI_ADJUST_ALLTOALLW</code>	<code>MPI_Alltoallw</code>	<ol style="list-style-type: none"> 1. Isend/Irecv + waitall algorithm
<code>I_MPI_ADJUST_BARRIER</code>	<code>MPI_Barrier</code>	<ol style="list-style-type: none"> 1. Dissemination algorithm 2. Recursive doubling algorithm 3. Topology aware dissemination algorithm 4. Topology aware recursive doubling algorithm 5. Binomial gather + scatter algorithm 6. Topology aware binomial gather + scatter algorithm

I_MPI_ADJUST_BCAST	MPI_Bcast	<ol style="list-style-type: none"> 1. Binomial algorithm 2. Recursive doubling algorithm 3. Ring algorithm 4. Topology aware binomial algorithm 5. Topology aware recursive doubling algorithm 6. Topology aware ring algorithm 7. Shumilin's bcast algorithm
I_MPI_ADJUST_EXSCAN	MPI_Exscan	<ol style="list-style-type: none"> 1. Partial results gathering algorithm 2. Partial results gathering regarding algorithm layout of processes
I_MPI_ADJUST_GATHER	MPI_Gather	<ol style="list-style-type: none"> 1. Binomial algorithm 2. Topology aware binomial algorithm 3. Shumilin's algorithm
I_MPI_ADJUST_GATHERV	MPI_Gatherv	<ol style="list-style-type: none"> 1. Linear algorithm 2. Topology aware linear algorithm
I_MPI_ADJUST_REDUCE_SCATTER	MPI_Reduce_scatter	<ol style="list-style-type: none"> 1. Recursive having algorithm 2. Pair wise exchange algorithm 3. Recursive doubling algorithm 4. Reduce + Scatterv algorithm 5. Topology aware Reduce + Scatterv algorithm
I_MPI_ADJUST_REDUCE	MPI_Reduce	<ol style="list-style-type: none"> 1. Shumilin's algorithm 2. Binomial algorithm 3. Topology aware Shumilin's algorithm 4. Topology aware binomial algorithm 5. Rabenseifner's algorithm 6. Topology aware Rabenseifner's algorithm
I_MPI_ADJUST_SCAN	MPI_Scan	<ol style="list-style-type: none"> 1. Partial results gathering algorithm 2. Topology aware partial results gathering algorithm
I_MPI_ADJUST_SCATTER	MPI_Scatter	<ol style="list-style-type: none"> 1. Binomial algorithm 2. Topology aware binomial algorithm 3. Shumilin's algorithm
I_MPI_ADJUST_SCATTERV	MPI_Scatterv	<ol style="list-style-type: none"> 1. Linear algorithm 2. Topology aware linear algorithm

The message size calculation rules for the collective operations are described in table below. Here, **n/a** means that the corresponding interval `<l>-<m>` should be omitted.

Table 4.5-2 Message Collective Functions

Collective Function	Message Size Formula
<code>MPI_Allgather</code>	<code>recv_count*recv_type_size</code>
<code>MPI_Allgatherv</code>	<code>total_recv_count*recv_type_size</code>
<code>MPI_Allreduce</code>	<code>count*type_size</code>
<code>MPI_Alltoall</code>	<code>send_count*send_type_size</code>
<code>MPI_Alltoallv</code>	n/a
<code>MPI_Alltoallw</code>	n/a
<code>MPI_Barrier</code>	n/a
<code>MPI_Bcast</code>	<code>count*type_size</code>
<code>MPI_Exscan</code>	<code>count*type_size</code>
<code>MPI_Gather</code>	<code>recv_count*recv_type_size</code> if <code>MPI_IN_PLACE</code> is used, otherwise <code>send_count*send_type_size</code>
<code>MPI_Gatherv</code>	n/a
<code>MPI_Reduce_scatter</code>	<code>total_recv_count*type_size</code>
<code>MPI_Reduce</code>	<code>count*type_size</code>
<code>MPI_Scan</code>	<code>count*type_size</code>
<code>MPI_Scatter</code>	<code>send_count*send_type_size</code> if <code>MPI_IN_PLACE</code> is used, otherwise <code>recv_count*recv_type_size</code>
<code>MPI_Scatterv</code>	n/a

Examples:

- Use the following settings to select second algorithm for `MPI_Reduce` operation:
`I_MPI_ADJUST_REDUCE=2`
- Use the following settings to define algorithms for `MPI_scatter` operation:
`I_MPI_ADJUST_REDUCE_SCATTER=4:0-100,5001-10000;1:101-3200,2:3201-5000;3`

In this case algorithm four will be used for the message sizes from 0 up to 100 bytes and from 5001 to 10000 bytes, algorithm one will be used for the message sizes from 101 up to 3200 bytes, algorithm two will be used for the message sizes from 3201 up to 5000 bytes, and algorithm three will be used for all other messages.

4.5.2 I_MPI_MSG Family

These environment variables are deprecated and supported mostly for backward compatibility. Use the `I_MPI_ADJUST` environment variable family whenever possible.

I_MPI_FAST_COLLECTIVES

Control default library behavior during selection of appropriate collective algorithm for specific execution situation.

Syntax

`I_MPI_FAST_COLLECTIVES=<arg>`

Arguments

<code><arg></code>	Binary indicator
<code>enable yes on 1</code>	Fast collective algorithms are used. This is the default value
<code>disable no off 0</code>	Slower and safer collective algorithms are used

Description

The Intel® MPI Library uses advanced collective algorithms designed for better application performance by default. The implementation makes the following assumptions:

- It is safe to utilize the flexibility of the MPI standard regarding the order of execution of the collective operations to take advantage of the process layout and other opportunities.
- There is enough memory available for allocating additional internal buffers.

Set the `I_MPI_FAST_COLLECTIVES` environment variable to `disable` if you need to obtain results that do not depend on the physical process layout or other factors.

NOTE: Some optimizations controlled by this environment variable are of an experimental nature. In case of failure, turn off the collective optimizations and repeat the run.

I_MPI_BCAST_NUM_PROCS

Control `MPI_Bcast` algorithm thresholds.

Syntax

`I_MPI_BCAST_NUM_PROCS=<nproc>`

Arguments

<code><nproc></code>	Define the number of processes threshold for choosing the <code>MPI_Bcast</code> algorithm
<code>> 0</code>	The default value is 8

I_MPI_BCAST_MSG

Control `MPI_Bcast` algorithm thresholds.

Syntax

`I_MPI_BCAST_MSG=<nbytes1, nbytes2>`

Arguments

<code><nbytes1, nbytes2></code>	Define the message size threshold range (in bytes) for choosing the <code>MPI_Bcast</code> algorithm
<code>> 0</code> <code>nbytes2 >= nbytes1</code>	The default pair of values is 12288,524288

Description

Set these environment variables to control the selection of the three possible `MPI_Bcast` algorithms according to the following scheme (See Table 4.5-1 for algorithm descriptions):

1. The first algorithm is selected if the message size is less than `<nbytes1>`, or the number of processes in the operation is less than `<nproc>`.
2. The second algorithm is selected if the message size is greater than or equal to `<nbytes1>` and less than `<nbytes2>`, and the number of processes in the operation is a power of two.
3. If none of the above conditions is satisfied, the third algorithm is selected.

I_MPI_ALLTOALL_NUM_PROCS

Control `MPI_Alltoall` algorithm thresholds.

Syntax

`I_MPI_ALLTOALL_NUM_PROCS=<nproc>`

Arguments

<code><nproc></code>	Define the number of processes threshold for choosing the <code>MPI_Alltoall</code> algorithm
<code>> 0</code>	The default value is 8

I_MPI_ALLTOALL_MSG

Control `MPI_Alltoall` algorithm thresholds.

Syntax

`I_MPI_ALLTOALL_MSG=<nbytes1,nbytes2>`

Arguments

<code><nbytes1,nbytes2></code>	Defines the message size threshold range (in bytes) for choosing the <code>MPI_Alltoall</code> algorithm
<code>> 0</code> <code>nbytes2 >= nbytes1</code>	The default pair of values is 256,32768

Description

Set these environment variables to control the selection of the three possible `MPI_Alltoall` algorithms according to the following scheme (See Table 4.5-1 for algorithm descriptions):

1. The first algorithm is selected if the message size is greater than or equal to `<nbytes1>`, and the number of processes in the operation is not less than `<nproc>`.
2. The second algorithm is selected if the message size is greater than `<nbytes1>` and less than or equal to `<nbytes2>`, or if the message size is less than `<nbytes2>` and the number of processes in the operation is less than `<nproc>`.
3. If none of the above conditions is satisfied, the third algorithm is selected.

I_MPI_ALLGATHER_MSG

Control `MPI_Allgather` algorithm thresholds.

Syntax

`I_MPI_ALLGATHER_MSG=<nbytes1,nbytes2>`

Arguments

<code><nbytes1,nbytes2></code>	Define the message size threshold range (in bytes) for choosing
--------------------------------------	---

	the <code>MPI_Allgather</code> algorithm
<code>> 0</code> <code>nbytes2 >= nbytes1</code>	The default pair of values is 81920,524288

Description

Set this environment variable to control the selection of the three possible `MPI_Allgather` algorithms according to the following scheme (See Table 4.5-1 for algorithm descriptions):

1. The first algorithm is selected if the message size is less than `<nbytes2>` and the number of processes in the operation is a power of two.
2. The second algorithm is selected if the message size is less than `<nbytes1>` and number of processes in the operation is not a power of two.
3. If none of the above conditions is satisfied, the third algorithm is selected.

I_MPI_ALLREDUCE_MSG

Control `MPI_Allreduce` algorithm thresholds.

Syntax

`I_MPI_ALLREDUCE_MSG=<nbytes>`

Arguments

<code><nbytes></code>	Define the message size threshold (in bytes) for choosing the <code>MPI_Allreduce</code> algorithm
<code>> 0</code>	The default value is 2048

Description

Set this environment variable to control the selection of the two possible `MPI_Allreduce` algorithms according to the following scheme (See Table 4.5-1 for algorithm descriptions):

1. The first algorithm is selected if the message size is less than or equal `<nbytes>`, or the reduction operation is user-defined, or the count argument is less than the nearest power of two less than or equal to the number of processes.
2. If the above condition is not satisfied, the second algorithm is selected.

I_MPI_REDS CAT_MSG

Control the `MPI_Reduce_scatter` algorithm thresholds.

Syntax

`I_MPI_REDS CAT_MSG=<nbytes1,nbytes2>`

Arguments

<code><nbytes></code>	Define the message size threshold range (in bytes) for choosing the <code>MPI_Reduce_scatter</code> algorithm
<code>> 0</code>	The default value is 512,524288

Description

Set this environment variable to control the selection of the three possible `MPI_Reduce_scatter` algorithms according to the following scheme (See Table 4.5-1 for algorithm descriptions):

1. The first algorithm is selected if the reduction operation is commutative and the message size is less than `<nbytes2>`.

2. The second algorithm is selected if the reduction operation is commutative and the message size is greater than or equal to *<nbytes2>*, or if the reduction operation is not commutative and the message size is greater than or equal to *<nbytes1>*.
3. If none of the above conditions is satisfied, the third algorithm is selected.

I_MPI_SCATTER_MSG

Control *MPI_Scatter* algorithm thresholds.

Syntax

I_MPI_SCATTER_MSG=*<nbytes>*

Arguments

<i><nbytes></i>	Define the buffer size threshold range (in bytes) for choosing the <i>MPI_Scatter</i> algorithm
<i>> 0</i>	The default value is 2048

Description

Set this environment variable to control the selection of the two possible *MPI_Scatter* algorithms according to the following scheme (See Table 4.5-1 for algorithm descriptions):

1. The first algorithm is selected on the intercommunicators if the message size is greater than *<nbytes>*.
2. If the above condition is not satisfied, the second algorithm is selected.

I_MPI_GATHER_MSG

Control *MPI_Gather* algorithm thresholds.

Syntax

I_MPI_GATHER_MSG=*<nbytes>*

Arguments

<i><nbytes></i>	Define the buffer size threshold range (in bytes) for choosing the <i>MPI_Gather</i> algorithm
<i>> 0</i>	The default value is 2048

Description

Set this environment variable to control the selection of the two possible *MPI_Gather* algorithms according to the following scheme (See Table 4.5-1 for algorithm descriptions):

1. The first algorithm is selected on the intercommunicators if the message size is greater than *<nbytes>*.
2. If the above condition is not satisfied, the second algorithm is selected.

4.6 Compatibility Control

The Intel® MPI Library 4.0 implements the MPI-2.1 standard. The following MPI routines are changed:

- *MPI_Cart_create*
- *MPI_Cart_map*
- *MPI_Cart_sub*

- `MPI_Graph_create`

If your application depends on the strict pre-MPI-2.1 behavior, set the environment variable `I_MPI_COMPATIBILITY` to 3.

I_MPI_COMPATIBILITY

Select the runtime compatibility mode.

Syntax

`I_MPI_COMPATIBILITY=<value>`

Arguments

<code><value></code>	Define compatibility mode
3	Enable the Intel® MPI Library 3.x compatibility mode
4	Disable the Intel® MPI Library 3.x compatibility mode and enable the Intel® MPI Library 4.0 compatible mode. This is the default value

Description

Set this environment variable to choose the Intel® MPI runtime compatible mode.

4.7 Secure Loading of Dynamic Link Libraries*

The Intel® MPI Library provides enhanced security options for the loading of Dynamic Link Libraries*. You can enable the enhanced security mode for the dynamic library loading, as well as define a set of directories in which the library will attempt to locate an external DLL*.

The security options are placed in the `HKEY_LOCAL_MACHINE\Software\Intel\MPI` protected Windows* registry key. The location prevents the options from being changed with non-administrative privileges.

SecureDynamicLibraryLoading

Select the secure DLL loading mode.

Syntax

`SecureDynamicLibraryLoading=<value>`

Arguments

<code><value></code>	Binary indicator
<code>enable yes on 1</code>	Enable the secure DLL loading mode
<code>disable no off 0</code>	Disable the secure DLL loading mode. This is the default value

Description

Use `HKEY_LOCAL_MACHINE\Software\Intel\MPI` registry key to define the `SecureDynamicLibraryLoading` registry entry. Set this entry to enable the secure DLL loading mode.

I_MPI_DAT_LIBRARY

Select a particular DAT library to be used in the DLL enhanced security mode.

Syntax

`I_MPI_DAT_LIBRARY=<library>`

Arguments

<code><library></code>	Specify the name of the library to be loaded
------------------------------	--

Description

In the secure DLL loading mode, the library changes the default-defined set of directories to locate DLLs. Therefore, the current working directory and the directories that are listed in the `PATH` environment variable may be ignored. To select a specific external DAT library to be loaded, define the `I_MPI_DAT_LIBRARY` entry of the `HKEY_LOCAL_MACHINE\Software\Intel\MPI` registry key. Specify the full path to the DAT library.

NOTE: The `I_MPI_DAT_LIBRARY` environment variable has no effect in the secure DLL loading mode. See [I_MPI_DAT_LIBRARY](#) for more details.

SecurePath

Specify a set of directories to locate an external DLL.

Syntax

`SecurePath=<path> [;<path> [...]]`

Arguments

<code><path></code>	Specify the path to a directory
---------------------------	---------------------------------

Description

Use `HKEY_LOCAL_MACHINE\Software\Intel\MPI` registry key to define the `SecurePath` registry entry. Set this entry to specify a set of directories to locate an external DLL in the secure DLL loading mode. Use a safe set of directories instead of some publicly writable directories to avoid insecure library loading.

NOTE: Use this option when the library is unable to load a DLL in the secure DLL loading mode. The option has no effect if the secure DLL loading mode is turned off.

5 Statistics Gathering Mode

5.1 Native Statistics Format

The Intel® MPI Library has a built-in statistics gathering facility that collects essential performance data without disturbing the application execution. The collected information is output onto a text file. This section describes the environment variables used to control the built-in statistics gathering facility, and provides example output files.

I_MPI_STATS

Control statistics collection.

Syntax

```
I_MPI_STATS=[n-] m
```

Arguments

<code>n, m</code>	Possible stats levels of the output information
<code>1</code>	Output the amount of data sent by each process
<code>2</code>	Output the number of calls and amount of transferred data
<code>3</code>	Output statistics combined according to the actual arguments
<code>4</code>	Output statistics defined by a buckets list
<code>10</code>	Output collective operation statistics for all communication contexts

Description

Set this environment variable to control the amount of the statistics information collected and output onto the log file. No statistics are output by default.

NOTE: `n, m` represent the positive integer numbers to define the range of output information. The statistics from level `n` to level `m` inclusive are output. Omitted `n` value assumes to be `1`.

I_MPI_STATS_SCOPE

Select the subsystem(s) to collect statistics for.

Syntax

```
I_MPI_STATS_SCOPE=<subsystem>[:<ops>] [;<subsystem>[:<ops>] [...]]
```

Arguments

<code><subsystem></code>	Define the target subsystem(s)
<code>all</code>	Collect statistics data for all operations. This is the default value
<code>coll</code>	Collect statistics data for all collective operations
<code>p2p</code>	Collect statistics data for all point-to-point operations

<code><ops></code>	Define the target operations as a comma separated list
<code>Allgather</code>	<code>MPI_Allgather</code>
<code>Allgatherv</code>	<code>MPI_Allgatherv</code>
<code>Allreduce</code>	<code>MPI_Allreduce</code>
<code>Alltoall</code>	<code>MPI_Alltoall</code>
<code>Alltoallv</code>	<code>MPI_Alltoallv</code>
<code>Alltoallw</code>	<code>MPI_Alltoallw</code>
<code>Barrier</code>	<code>MPI_Barrier</code>
<code>Bcast</code>	<code>MPI_Bcast</code>
<code>Exscan</code>	<code>MPI_Exscan</code>
<code>Gather</code>	<code>MPI_Gather</code>
<code>Gatherv</code>	<code>MPI_Gatherv</code>
<code>Reduce_scatter</code>	<code>MPI_Reduce_scatter</code>
<code>Reduce</code>	<code>MPI_Reduce</code>
<code>Scan</code>	<code>MPI_Scan</code>
<code>Scatter</code>	<code>MPI_Scatter</code>
<code>Scatterv</code>	<code>MPI_Scatterv</code>
<code>Send</code>	Standard transfers (<code>MPI_Send</code> , <code>MPI_Isend</code> , <code>MPI_Send_init</code>)
<code>Bsend</code>	Buffered transfers (<code>MPI_Bsend</code> , <code>MPI_Ibsend</code> , <code>MPI_Bsend_init</code>)
<code>Csend</code>	Point-to-point operations inside the collectives. This internal operation serves all collectives
<code>Rsend</code>	Ready transfers (<code>MPI_Rsend</code> , <code>MPI_Irsend</code> , <code>MPI_Rsend_init</code>)
<code>Ssend</code>	Synchronous transfers (<code>MPI_Ssend</code> , <code>MPI_Issend</code> , <code>MPI_Ssend_init</code>)

Description

Set this environment variable to select the target subsystem to collect statistics for. All collective and point-to-point operations, including the point-to-point operations performed inside the collectives are covered by default.

Examples

- The default settings are equivalent to:
`I_MPI_STATS_SCOPE=coll;p2p`
- Use the following settings to collect statistics for the `MPI_Bcast`, `MPI_Reduce`, and all point-to-point operations:
`I_MPI_STATS_SCOPE=p2p;coll:bcast,reduce`

Use the following settings to collect statistics for the point-to-point operations inside the collectives:
`I_MPI_STATS_SCOPE=p2p:csend`

I_MPI_STATS_BUCKETS

Identify a list of ranges for message sizes and communicator sizes that will be used for collecting statistics.

Syntax

```
I_MPI_STATS_BUCKETS=<msg> [@<proc>] [, <msg> [@<proc>]]...
```

Arguments

<msg>	Specify range of message sizes in bytes
<l>	Single value of message size
<l>-<m>	Range from <l> to <m>

<proc>	Specify range of processes (ranks) for collective operations
<p>	Single value of communicator size
<p>-<q>	Range from <p> to <q>

Description

Set the `I_MPI_STATS_BUCKETS` environment variable to define a set of ranges for message sizes and communicator sizes.

Level 4 of the statistics provides profile information for these ranges.

If `I_MPI_STATS_BUCKETS` environment variable is not used, then level 4 statistics is not gathered.

If a range is omitted then the maximum possible range is assumed.

Examples

To specify short messages (from 0 to 1000 bytes) and long messages (from 50000 to 100000 bytes), use the following setting:

```
-env I_MPI_STATS_BUCKETS 0-1000,50000-100000
```

To specify messages that have 16 bytes in size and circulate within four process communicators, use the following setting:

```
-env I_MPI_STATS_BUCKETS "16@4"
```

NOTE: When the @ symbol is present, the environment variable value must be enclosed in quotes.

I_MPI_STATS_FILE

Define the statistics output file name.

Syntax

```
I_MPI_STATS_FILE=<name>
```

Arguments

<name>	Define the statistics output file name
--------	--

Description

Set this environment variable to define the statistics output file. The stats.txt file is created in the current directory by default.

The statistics data is blocked and ordered according to the process ranks in the `MPI_COMM_WORLD` communicator. The timing data is presented in microseconds. For example, with the following settings in effect

```
I_MPI_STATS=4
I_MPI_STATS_SCOPE=p2p;coll:allreduce
```

the statistics output for a simple program that performs only one `MPI_Allreduce` operation may look as follows:

```
Intel(R) MPI Library Version 4.0
_____ MPI Communication Statistics _____

Stats level: 4
P2P scope:< FULL >
Collectives scope:< Allreduce >

~~~~ Process 0 of 2 on node svlmpihead01 lifetime = 414.13

Data Transfers
Src      Dst      Amount(MB)  Transfers
-----
000 --> 000      0.000000e+00  0
000 --> 001      7.629395e-06  2
=====
Totals                7.629395e-06  2

Communication Activity
Operation      Volume(MB)  Calls
-----
P2P
Csend          7.629395e-06  2
Send           0.000000e+00  0
Bsend          0.000000e+00  0
Rsend          0.000000e+00  0
Ssend          0.000000e+00  0
Collectives
Allreduce      7.629395e-06   2
=====

Communication Activity by actual args
P2P
Operation      Dst      Message size  Calls
```

```

-----
Csend
1      1      4      2
Collectives
Operation      Context      Algo      Comm size      Message size      Calls      Cost(%)
-----
Allreduce
1              0              1      2              4              2      44.96
=====

```

~~~~ Process 1 of 2 on node svlmpihead01 lifetime = 306.13

```

Data Transfers
Src      Dst      Amount(MB)      Transfers
-----
001 --> 000      7.629395e-06      2
001 --> 001      0.000000e+00      0
=====
Totals              7.629395e-06      2

```

```

Communication Activity
Operation      Volume(MB)      Calls
-----
P2P
Csend          7.629395e-06      2
Send           0.000000e+00      0
Bsend          0.000000e+00      0
Rsend          0.000000e+00      0
Ssend          0.000000e+00      0
Collectives
Allreduce      7.629395e-06      2
=====

```

```

Communication Activity by actual args
P2P
Operation      Dst      Message size      Calls
-----
Csend
1      0      4      2
Collectives
Operation      Context      Comm size      Message size      Calls      Cost(%)
-----
Allreduce

```

```

1      0      2      4      2      37.93
=====
_____ End of stats.txt file _____

```

In the example above all times are measured in microseconds. The message sizes are counted in bytes. **MB** means megabyte equal to 2<sup>20</sup> or 1 048 576 bytes. The process life time is calculated as a stretch of time between `MPI_Init` and `MPI_Finalize`. The **Cost** field represents a particular collective operation execution time as a percentage of the process life time.

## 5.2 IPM Statistics Format

The Intel® MPI Library supports integrated performance monitoring (IPM) summary format as part of the built-in statistics gathering mechanism described above. You do not need to modify the source code or re-link your application to collect this information.

The `I_MPI_STATS_BUCKETS` environment variable is not applicable to the IPM format. The `I_MPI_STATS_ACCURACY` environment variable is available to control extra functionality.

The Intel® MPI Library also supports an optional `ipm` region feature. This feature requires the source code modification. The `MPI_Pcontrol` function can be used.

### 5.2.1 Region Control

Region is a named part of the source code marked by the start/end points through the standard `MPI_Pcontrol` function calls. The `MPI_Pcontrol` function isn't used for the following special permanent regions:

- Main region contains statistics information about all MPI calls from `MPI_Init` to `MPI_Finalize`. Main region gets the "\*" name in output.
- Complementary region contains statistics information not included into any named region. The region gets the "ipm\_noregion" name in output.

If named regions are not used, the main regions and the complementary regions are identical and the complementary region is ignored.

Each region contains its own independent statistics information about MPI functions called inside the region.

The Intel® MPI Library supports the following types of regions:

- Discontiguous (several open and close).
- Intersected.
- Covering a subset of MPI processes (part of the `MPI_COMM_WORLD` environment variable).

A region is opened by the `MPI_Pcontrol(1, name)` call and closed by the `MPI_Pcontrol(-1, name)` call where `name` is a zero terminated string with the region name.

All open regions are closed automatically inside the `MPI_Finalize` environment variable.

## I\_MPI\_STATS

Control the statistics data output format.

### Syntax

`I_MPI_STATS=<level>`

### Argument

|                            |                                     |
|----------------------------|-------------------------------------|
| <code>&lt;level&gt;</code> | Level of statistics data            |
| <code>ipm</code>           | Summary data throughout all regions |

|                        |                    |
|------------------------|--------------------|
| <code>ipm:terse</code> | Basic summary data |
|------------------------|--------------------|

### Description

Set this environment variable to `ipm` to get the statistics output that contains region summary. Set this environment variable to `ipm:terse` argument to get the brief statistics output.

## I\_MPI\_STATS\_FILE

Define the output file name.

### Syntax

`I_MPI_STATS_FILE=<name>`

### Argument

|                           |                                         |
|---------------------------|-----------------------------------------|
| <code>&lt;name&gt;</code> | File name for statistics data gathering |
|---------------------------|-----------------------------------------|

### Description

Set this environment variable to change the statistics output file name from the default name of `stats.ipm`.

## I\_MPI\_STATS\_SCOPE

Define a semicolon separated list of subsets of MPI functions for statistics gathering.

### Syntax

`I_MPI_STATS_SCOPE=<subset> [ ; <subset> [ ; ... ] ]`

### Argument

| <code>&lt;subset&gt;</code> | Target subset                                                       |
|-----------------------------|---------------------------------------------------------------------|
| <code>all2all</code>        | Collect statistics data for all to all kind of collective functions |
| <code>all2one</code>        | Collect statistics data for all to one kind of collective functions |
| <code>attr</code>           | Collect statistics data for attribute control functions             |
| <code>comm</code>           | Collect statistics data for communicator control functions          |
| <code>err</code>            | Collect statistics data for error handling functions                |
| <code>group</code>          | Collect statistics data for group support functions                 |
| <code>init</code>           | Collect statistics data for initialize/finalize functions           |
| <code>io</code>             | Collect statistics data for input/output support function           |
| <code>one2all</code>        | Collect statistics data for one to all kind of collective functions |
| <code>recv</code>           | Collect statistics data for receive functions                       |
| <code>req</code>            | Collect statistics data for request support functions               |
| <code>rma</code>            | Collect statistics data for one sided communication functions       |
| <code>scan</code>           | Collect statistics data for scan collective functions               |
| <code>send</code>           | Collect statistics data for send functions                          |
| <code>sendrecv</code>       | Collect statistics data for send/receive functions                  |
| <code>serv</code>           | Collect statistics data for additional service functions            |
| <code>spawn</code>          | Collect statistics data for dynamic process functions               |
| <code>status</code>         | Collect statistics data for status control function                 |

|                   |                                                         |
|-------------------|---------------------------------------------------------|
| <code>sync</code> | Collect statistics data for barrier synchronization     |
| <code>time</code> | Collect statistics data for timing support functions    |
| <code>topo</code> | Collect statistics data for topology support functions  |
| <code>type</code> | Collect statistics data for data type support functions |

### Description

Use this environment variable to define a subset or subsets of MPI functions for statistics gathering specified by the following table. A union of all subsets is used by default.

Table 5.2-1 Stats Subsets of MPI Functions

|                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <p><b>all2all</b></p> <p><i>MPI_Allgather</i><br/> <i>MPI_Allgatherv</i><br/> <i>MPI_Allreduce</i><br/> <i>MPI_Alltoall</i><br/> <i>MPI_Alltoallv</i><br/> <i>MPI_Alltoallw</i><br/> <i>MPI_Reduce_scatter</i></p> <p><b>all2one</b></p> <p><i>MPI_Gather</i><br/> <i>MPI_Gatherv</i><br/> <i>MPI_Reduce</i></p> <p><b>attr</b></p> <p><i>MPI_Comm_create_keyval</i><br/> <i>MPI_Comm_delete_attr</i><br/> <i>MPI_Comm_free_keyval</i><br/> <i>MPI_Comm_get_attr</i><br/> <i>MPI_Comm_set_attr</i><br/> <i>MPI_Comm_get_name</i><br/> <i>MPI_Comm_set_name</i><br/> <i>MPI_Type_create_keyval</i><br/> <i>MPI_Type_delete_attr</i><br/> <i>MPI_Type_free_keyval</i><br/> <i>MPI_Type_get_attr</i><br/> <i>MPI_Type_get_name</i><br/> <i>MPI_Type_set_attr</i><br/> <i>MPI_Type_set_name</i><br/> <i>MPI_Win_create_keyval</i><br/> <i>MPI_Win_delete_attr</i><br/> <i>MPI_Win_free_keyval</i><br/> <i>MPI_Win_get_attr</i><br/> <i>MPI_Win_get_name</i><br/> <i>MPI_Win_set_attr</i><br/> <i>MPI_Win_set_name</i><br/> <i>MPI_Get_processor_name</i></p> <p><b>comm</b></p> <p><i>MPI_Comm_compare</i><br/> <i>MPI_Comm_create</i><br/> <i>MPI_Comm_dup</i><br/> <i>MPI_Comm_free</i><br/> <i>MPI_Comm_get_name</i><br/> <i>MPI_Comm_group</i><br/> <i>MPI_Comm_rank</i><br/> <i>MPI_Comm_remote_group</i><br/> <i>MPI_Comm_remote_size</i><br/> <i>MPI_Comm_set_name</i><br/> <i>MPI_Comm_size</i><br/> <i>MPI_Comm_split</i><br/> <i>MPI_Comm_test_inter</i><br/> <i>MPI_Intercomm_create</i><br/> <i>MPI_Intercomm_merge</i></p> <p><b>err</b></p> <p><i>MPI_Add_error_class</i><br/> <i>MPI_Add_error_code</i><br/> <i>MPI_Add_error_string</i><br/> <i>MPI_Comm_call_errhandler</i><br/> <i>MPI_Comm_create_errhandler</i><br/> <i>MPI_Comm_get_errhandler</i><br/> <i>MPI_Comm_set_errhandler</i><br/> <i>MPI_Errhandler_free</i><br/> <i>MPI_Error_class</i><br/> <i>MPI_Error_string</i><br/> <i>MPI_File_call_errhandler</i><br/> <i>MPI_File_create_errhandler</i></p> | <p><i>MPI_File_get_errhandler</i><br/> <i>MPI_File_set_errhandler</i><br/> <i>MPI_Win_call_errhandler</i><br/> <i>MPI_Win_create_errhandler</i><br/> <i>MPI_Win_get_errhandler</i><br/> <i>MPI_Win_set_errhandler</i></p> <p><b>group</b></p> <p><i>MPI_Group_compare</i><br/> <i>MPI_Group_difference</i><br/> <i>MPI_Group_excl</i><br/> <i>MPI_Group_free</i><br/> <i>MPI_Group_incl</i><br/> <i>MPI_Group_intersection</i><br/> <i>MPI_Group_range_excl</i><br/> <i>MPI_Group_range_incl</i><br/> <i>MPI_Group_rank</i><br/> <i>MPI_Group_size</i><br/> <i>MPI_Group_translate_ranks</i><br/> <i>MPI_Group_union</i></p> <p><b>init</b></p> <p><i>MPI_Init</i><br/> <i>MPI_Init_thread</i><br/> <i>MPI_Finalize</i></p> <p><b>io</b></p> <p><i>MPI_File_close</i><br/> <i>MPI_File_delete</i><br/> <i>MPI_File_get_amode</i><br/> <i>MPI_File_get_atomicity</i><br/> <i>MPI_File_get_byte_offset</i><br/> <i>MPI_File_get_group</i><br/> <i>MPI_File_get_info</i><br/> <i>MPI_File_get_position</i><br/> <i>MPI_File_get_position_shared</i><br/> <i>MPI_File_get_size</i><br/> <i>MPI_File_get_type_extent</i><br/> <i>MPI_File_get_view</i><br/> <i>MPI_File_iread_at</i><br/> <i>MPI_File_iread</i><br/> <i>MPI_File_iread_shared</i><br/> <i>MPI_File_iwrite_at</i><br/> <i>MPI_File_iwrite</i><br/> <i>MPI_File_iwrite_shared</i><br/> <i>MPI_File_open</i><br/> <i>MPI_File_preallocate</i><br/> <i>MPI_File_read_all_begin</i><br/> <i>MPI_File_read_all_end</i><br/> <i>MPI_File_read_all</i><br/> <i>MPI_File_read_at_all_begin</i><br/> <i>MPI_File_read_at_all_end</i><br/> <i>MPI_File_read_at_all</i><br/> <i>MPI_File_read_at</i><br/> <i>MPI_File_read</i><br/> <i>MPI_File_read_ordered_begin</i><br/> <i>MPI_File_read_ordered_end</i><br/> <i>MPI_File_read_ordered</i><br/> <i>MPI_File_read_shared</i><br/> <i>MPI_File_seek</i><br/> <i>MPI_File_seek_shared</i><br/> <i>MPI_File_set_atomicity</i><br/> <i>MPI_File_set_info</i><br/> <i>MPI_File_set_size</i><br/> <i>MPI_File_set_view</i><br/> <i>MPI_File_sync</i><br/> <i>MPI_File_write_all_begin</i><br/> <i>MPI_File_write_all_end</i><br/> <i>MPI_File_write_all</i><br/> <i>MPI_File_write_at_all_begin</i><br/> <i>MPI_File_write_at_all_end</i></p> |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

|                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <p> <i>MPI_File_write_at_all</i><br/> <i>MPI_File_write_at</i><br/> <i>MPI_File_write</i><br/> <i>MPI_File_write_ordered_begin</i><br/> <i>MPI_File_write_ordered_end</i><br/> <i>MPI_File_write_ordered</i><br/> <i>MPI_File_write_shared</i><br/> <i>MPI_Register_datarep</i> </p> <p><b>one2all</b></p> <p> <i>MPI_Bcast</i><br/> <i>MPI_Scatter</i><br/> <i>MPI_Scatterv</i> </p> <p><b>recv</b></p> <p> <i>MPI_Recv</i><br/> <i>MPI_Irecv</i><br/> <i>MPI_Recv_init</i><br/> <i>MPI_Probe</i><br/> <i>MPI_Iprobe</i> </p> <p><b>req</b></p> <p> <i>MPI_Start</i><br/> <i>MPI_Startall</i><br/> <i>MPI_Wait</i><br/> <i>MPI_Waitall</i><br/> <i>MPI_Waitany</i><br/> <i>MPI_Waitsome</i><br/> <i>MPI_Test</i><br/> <i>MPI_Testall</i><br/> <i>MPI_Testany</i><br/> <i>MPI_Testsome</i><br/> <i>MPI_Cancel</i><br/> <i>MPI_Grequest_start</i><br/> <i>MPI_Grequest_complete</i><br/> <i>MPI_Request_get_status</i><br/> <i>MPI_Request_free</i> </p> <p><b>rma</b></p> <p> <i>MPI_Accumulate</i><br/> <i>MPI_Get</i><br/> <i>MPI_Put</i><br/> <i>MPI_Win_complete</i><br/> <i>MPI_Win_create</i><br/> <i>MPI_Win_fence</i><br/> <i>MPI_Win_free</i><br/> <i>MPI_Win_get_group</i><br/> <i>MPI_Win_lock</i><br/> <i>MPI_Win_post</i><br/> <i>MPI_Win_start</i><br/> <i>MPI_Win_test</i><br/> <i>MPI_Win_unlock</i><br/> <i>MPI_Win_wait</i> </p> <p><b>scan</b></p> <p> <i>MPI_Exscan</i><br/> <i>MPI_Scan</i> </p> <p><b>send</b></p> <p> <i>MPI_Send</i><br/> <i>MPI_Bsend</i><br/> <i>MPI_Rsend</i><br/> <i>MPI_Ssend</i><br/> <i>MPI_Isend</i> </p> | <p> <i>MPI_Ibsend</i><br/> <i>MPI_Irsend</i><br/> <i>MPI_Issend</i><br/> <i>MPI_Send_init</i><br/> <i>MPI_Bsend_init</i><br/> <i>MPI_Rsend_init</i><br/> <i>MPI_Ssend_init</i> </p> <p><b>sendrecv</b></p> <p> <i>MPI_Sendrecv</i><br/> <i>MPI_Sendrecv_replace</i> </p> <p><b>serv</b></p> <p> <i>MPI_Alloc_mem</i><br/> <i>MPI_Free_mem</i><br/> <i>MPI_Buffer_attach</i><br/> <i>MPI_Buffer_detach</i><br/> <i>MPI_Op_create</i><br/> <i>MPI_Op_free</i> </p> <p><b>spawn</b></p> <p> <i>MPI_Close_port</i><br/> <i>MPI_Comm_accept</i><br/> <i>MPI_Comm_connect</i><br/> <i>MPI_Comm_disconnect</i><br/> <i>MPI_Comm_get_parent</i><br/> <i>MPI_Comm_join</i><br/> <i>MPI_Comm_spawn</i><br/> <i>MPI_Comm_spawn_multiple</i><br/> <i>MPI_Lookup_name</i><br/> <i>MPI_Open_port</i><br/> <i>MPI_Publish_name</i><br/> <i>MPI_Unpublish_name</i> </p> <p><b>status</b></p> <p> <i>MPI_Get_count</i><br/> <i>MPI_Status_set_elements</i><br/> <i>MPI_Status_set_cancelled</i><br/> <i>MPI_Test_cancelled</i> </p> <p><b>sync</b></p> <p> <i>MPI_Barrier</i> </p> <p><b>time</b></p> <p> <i>MPI_Wtick</i><br/> <i>MPI_Wtime</i> </p> <p><b>topo</b></p> <p> <i>MPI_Cart_coords</i><br/> <i>MPI_Cart_create</i><br/> <i>MPI_Cart_get</i><br/> <i>MPI_Cart_map</i><br/> <i>MPI_Cart_rank</i><br/> <i>MPI_Cart_shift</i><br/> <i>MPI_Cart_sub</i><br/> <i>MPI_Cartdim_get</i><br/> <i>MPI_Dims_create</i><br/> <i>MPI_Graph_create</i><br/> <i>MPI_Graph_get</i><br/> <i>MPI_Graph_map</i><br/> <i>MPI_Graph_neighbors</i><br/> <i>MPI_Graphdims_get</i><br/> <i>MPI_Graph_neighbors_count</i><br/> <i>MPI_Topo_test</i> </p> |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

|                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |  |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--|
| <b>type</b><br><i>MPI_Get_address</i><br><i>MPI_Get_elements</i><br><i>MPI_Pack</i><br><i>MPI_Pack_external</i><br><i>MPI_Pack_external_size</i><br><i>MPI_Pack_size</i><br><i>MPI_Type_commit</i><br><i>MPI_Type_contiguous</i><br><i>MPI_Type_create_darray</i><br><i>MPI_Type_create_hindexed</i><br><i>MPI_Type_create_hvector</i><br><i>MPI_Type_create_indexed_block</i><br><i>MPI_Type_create_resized</i><br><i>MPI_Type_create_struct</i><br><i>MPI_Type_create_subarray</i><br><i>MPI_Type_dup</i><br><i>MPI_Type_free</i><br><i>MPI_Type_get_contents</i><br><i>MPI_Type_get_envelope</i><br><i>MPI_Type_get_extent</i><br><i>MPI_Type_get_true_extent</i><br><i>MPI_Type_indexed</i><br><i>MPI_Type_size</i><br><i>MPI_Type_vector</i><br><i>MPI_Unpack_external</i><br><i>MPI_Unpack</i> |  |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--|

## I\_MPI\_STATS\_ACCURACY

Use the `I_MPI_STATS_ACCURACY` environment variable to decrease statistics output.

### Syntax

`I_MPI_STATS_ACCURACY=<percentage>`

### Argument

|                                 |                       |
|---------------------------------|-----------------------|
| <code>&lt;percentage&gt;</code> | Float threshold value |
|---------------------------------|-----------------------|

### Description

Set this environment variable to collect data only on those MPI functions that take a larger portion of the elapsed time as a percentage of the total time spent inside all MPI calls.

### Example

The following example represents the simple application code and IPM summary statistics format:

```
int main (int argc, char *argv[])
{
    int i, rank, size, nsend, nrecv;

    MPI_Init (&argc, &argv);

    MPI_Comm_rank (MPI_COMM_WORLD, &rank);
    nsend = rank;

    MPI_Wtime();

    for (i = 0; i < 200; i++)
    {
        MPI_Barrier(MPI_COMM_WORLD);
    }

    /* open "reduce" region for all processes */
    MPI_Pcontrol(1, "reduce");
    for (i = 0; i < 1000; i++)
        MPI_Reduce(&nsend, &nrecv, 1, MPI_INT, MPI_MAX, 0, MPI_COMM_WORLD);

    /* close "reduce" region */
}
```

```

MPI_Pcontrol(-1, "reduce");

if (rank == 0)
{
    /* "send" region for 0-th process only */
    MPI_Pcontrol(1, "send");
    MPI_Send(&nsend, 1, MPI_INT, 1, 1, MPI_COMM_WORLD);
    MPI_Pcontrol(-1, "send");
}
if (rank == 1)
{
    MPI_Recv(&nrecv, 1, MPI_INT, 0, 1, MPI_COMM_WORLD, MPI_STATUS_IGNORE);
}

/* reopen "reduce" region */
MPI_Pcontrol(1, "reduce");
for (i = 0; i < 1000; i++)
    MPI_Reduce(&nsend, &nrecv, 1, MPI_INT, MPI_MAX, 0, MPI_COMM_WORLD);

MPI_Wtime();
MPI_Finalize ();

return 0;
}

```

Command:

```
mpirun -n 4 -env I_MPI_STATS ipm:terse ./a.out
```

Statistics output:

```

#####
#
# command : unknown (completed)
# host    : SVLMPICL704/Windows          mpi_tasks : 4 on 1 nodes
# start   : 06/17/11/14:10:40           wallclock : 0.037681 sec
# stop    : 06/17/11/14:10:40           %comm    : 99.17
# gbytes  : 0.00000e+000 total          gflop/sec : NA
#
#####

```

Command:

```
mpirun -n 4 -env I_MPI_STATS ipm ./a.out
```

Stats output:

```

#####
#
# command : unknown (completed)
# host    : SVLMPICL704/Windows          mpi_tasks : 4 on 1 nodes
# start   : 06/17/11/14:10:40           wallclock : 0.037681 sec
# stop    : 06/17/11/14:10:40           %comm    : 99.17
# gbytes  : 0.00000e+000 total          gflop/sec : NA
#
#####
# region : * [ntasks] = 4
#
#
#          [total]          <avg>          min          max
# entries          4          1          1          1
# wallclock        0.118763      0.0296908    0.0207312    0.0376814
# user             0.0156001      0.00390002   0          0.0156001
# system           0          0          0          0
# mpi              0.117782      0.0294454    0.0204467    0.0374543
# %comm            99.1735      98.6278      99.3973
# gflop/sec        NA          NA          NA          NA
# gbytes           0          0          0          0
#
#
#          [time]          [calls]          <%mpi>          <%wall>
# MPI_Init         0.0944392          4          80.18          79.52
# MPI_Reduce       0.0183164          8000         15.55          15.42
# MPI_Recv         0.00327056          1          2.78          2.75
# MPI_Barrier      0.00174499          800          1.48          1.47
# MPI_Send         4.23448e-006          1          0.00          0.00

```

```

# MPI_Finalize          3.07963e-006  4          0.00          0.00
# MPI_Wtime            1.53982e-006  8          0.00          0.00
# MPI_Comm_rank       1.5398e-006  4          0.00          0.00
# MPI_TOTAL           0.117782      8822      100.00       99.17
#####
# region : reduce [ntasks] = 4
#
# [total] <avg> min max
# entries 8 2 2
# wallclock 0.0190786 0.00476966 0.00273201 0.00665929
# user 0 0 0
# system 0 0 0
# mpi 0.0183199 0.00457997 0.00255377 0.00643987
# %comm 96.0231 93.4761 97.0543
# gflop/sec NA NA NA NA
# gbytes 0 0 0
#
# [time] [calls] <%mpi> <%wall>
# MPI_Reduce 0.0183164 8000 99.98 96.00
# MPI_Finalize 3.07963e-006 4 0.02 0.02
# MPI_Wtime 3.84956e-007 4 0.00 0.00
# MPI_TOTAL 0.0183199 8008 100.00 96.02
#####
# region : send [ntasks] = 4
#
# [total] <avg> min max
# entries 1 0 0 1
# wallclock 1.22389e-005 3.05971e-006 1e-006 9.23885e-006
# user 0 0 0
# system 0 0 0
# mpi 4.23448e-006 1.05862e-006 0 4.23448e-006
# %comm 34.5986 0 45.8334
# gflop/sec NA NA NA NA
# gbytes 0 0 0
#
# [time] [calls] <%mpi> <%wall>
# MPI_Send 4.23448e-006 1 100.00 34.60
#####
# region : ipm_noregion [ntasks] = 4
#
# [total] <avg> min max
# entries 13 3 3 4
# wallclock 0.0996611 0.0249153 0.0140604 0.0349467
# user 0.0156001 0.00390002 0 0.0156001
# system 0 0 0
# mpi 0.0994574 0.0248644 0.0140026 0.0349006
# %comm 99.7957 99.5893 99.8678
# gflop/sec NA NA NA NA
# gbytes 0 0 0
#
# [time] [calls] <%mpi> <%wall>
# MPI_Init 0.0944392 4 94.95 94.76
# MPI_Recv 0.00327056 1 3.29 3.28
# MPI_Barrier 0.00174499 800 1.75 1.75
# MPI_Comm_rank 1.5398e-006 4 0.00 0.00
# MPI_Wtime 1.15486e-006 4 0.00 0.00
# MPI_TOTAL 0.0994574 813 100.00 99.80

```

# 6 ILP64 Support

---

The term *ILP64* means that integer, long, and pointer data entities all occupy 8 bytes. This differs from the more conventional LP64 model in which only long and pointer data entities occupy 8 bytes while integer entities stay at 4 byte size. More information on the historical background and the programming model philosophy can be found for example in [http://www.unix.org/version2/whatsnew/lp64\\_wp.html](http://www.unix.org/version2/whatsnew/lp64_wp.html)

## 6.1 Using ILP64

Below are several options to enable the ILP64 interface

- Use the Fortran\* compiler driver option `-i8` for separate compilation and the `-ilp64` option for separate linkage. For example,
 

```
> mpiifort -i8 -c test.f
> mpiifort -ilp64 -o test test.o
```
- Use the `mpiexec -ilp64` option to preload the ILP64 interface. For example,
 

```
> mpiexec -ilp64 -n 2 .\myprog
```

## 6.2 Known Issues and Limitations

- Data type counts and other arguments with values larger than  $2^{31}-1$  are not supported.
- Special MPI types `MPI_FLOAT_INT`, `MPI_DOUBLE_INT`, `MPI_LONG_INT`, `MPI_SHORT_INT`, `MPI_2INT`, `MPI_LONG_DOUBLE_INT`, `MPI_2INTEGER` are not changed and still use a 4-byte integer field.
- Predefined communicator attributes `MPI_APPNUM`, `MPI_HOST`, `MPI_IO`, `MPI_LASTUSED`, `MPI_TAG_UB`, `MPI_UNIVERSE_SIZE`, and `MPI_WTIME_IS_GLOBAL` are returned by the functions `MPI_GET_ATTR` and `MPI_COMM_GET_ATTR` as 4-byte integers. The same holds for the predefined attributes that may be attached to the window and file objects.
- Do not use the `-i8` option to compile MPI callback functions, such as error handling functions, user-defined reduction operations, etc.
- You have to use a special ITC library if you desire to use the Intel Trace Collector with the Intel MPI ILP64 executable files. If necessary, the Intel MPI `mpiifort` compiler driver will select the correct ITC library automatically.
- Use the `mpif.h` file instead of the MPI module in Fortran90 applications. The Fortran module supports 32-bit `INTEGER` size only.
- There is currently no support for C and C++ applications.

## 7 Unified Memory Management

---

Intel® MPI Library provides a way to replace the memory management subsystem by a user defined package. The following function pointers may optionally be set by the user:

- `i_malloc`
- `i_calloc`
- `i_realloc`
- `i_free`

These pointers also affect the C++ `new` and `delete` operators.

The respective standard C library functions are used by default.

To use the unified memory management subsystem, link your application against the `libimalloc.dll`.

The following contrived source code snippet illustrates the usage of the unified memory subsystem:

```
#include <i_malloc.h>
#include <my_malloc.h>

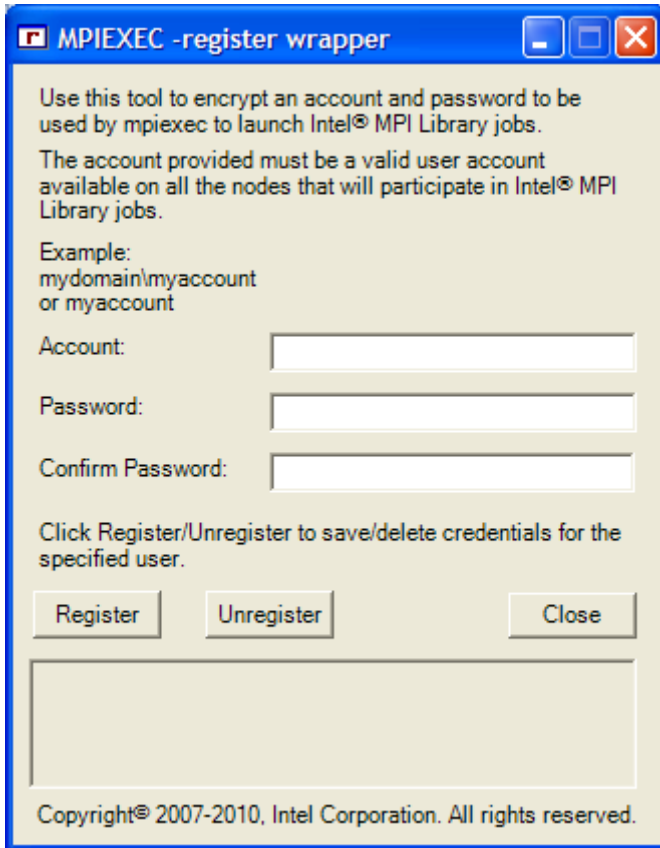
int main( int argc, int argv )
{
    // override normal pointers
    i_malloc = my_malloc;
    i_calloc = my_calloc;
    i_realloc = my_realloc;
    i_free = my_free;

#ifdef _WIN32
    // also override pointers used by DLLs
    i_malloc_dll = my_malloc;
    i_calloc_dll = my_calloc;
    i_realloc_dll = my_realloc;
    i_free_dll = my_free;
#endif

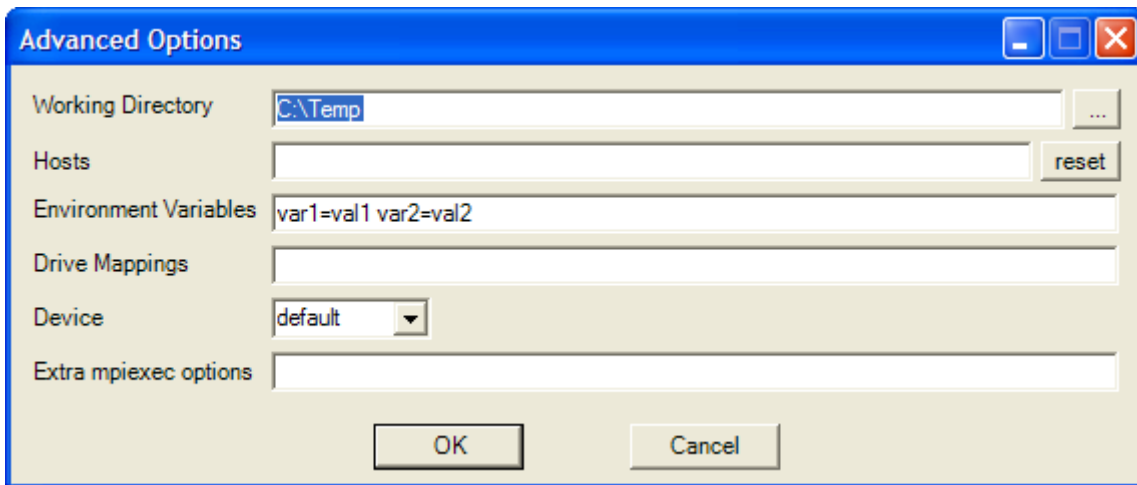
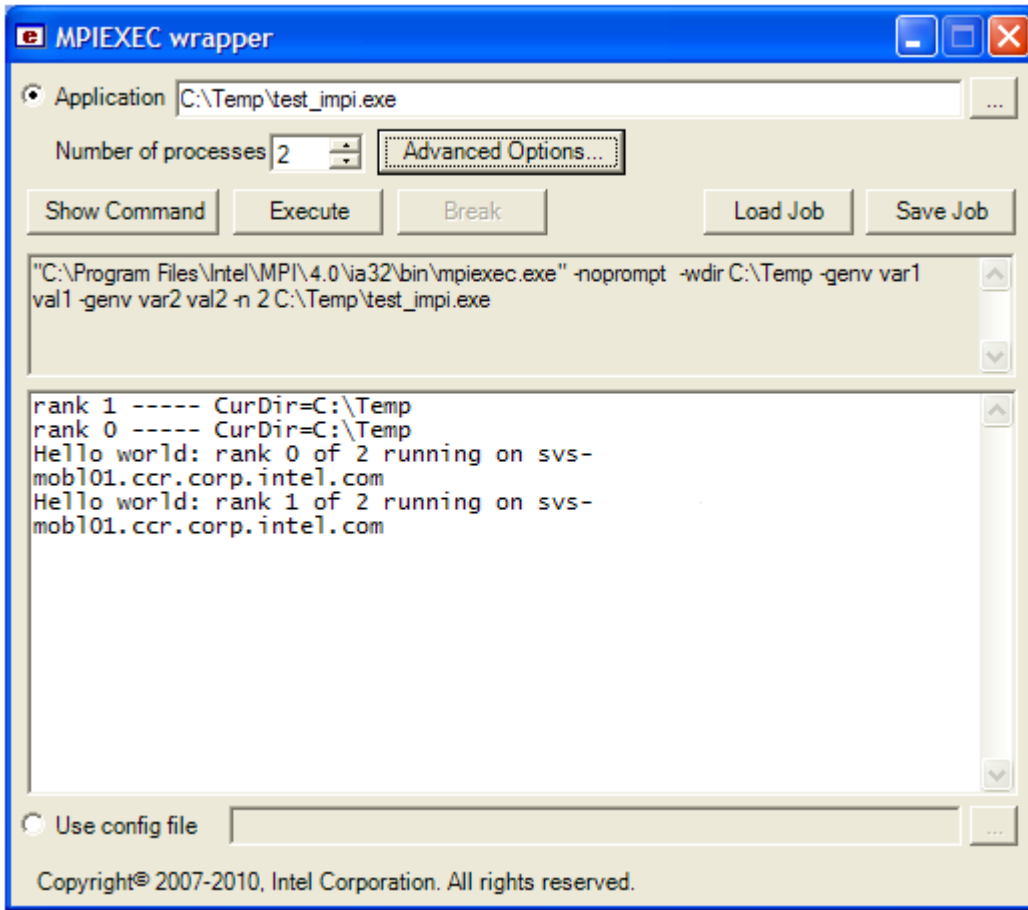
    // now start using Intel(R) libraries
}
```

## 8 Graphical Utilities

The Intel® MPI Library provides three graphical utilities: `wmpiregister`, `wmpiexec`, and `wmpiconfig`. These utilities simplify using the Intel® MPI Library under Windows\* OS.



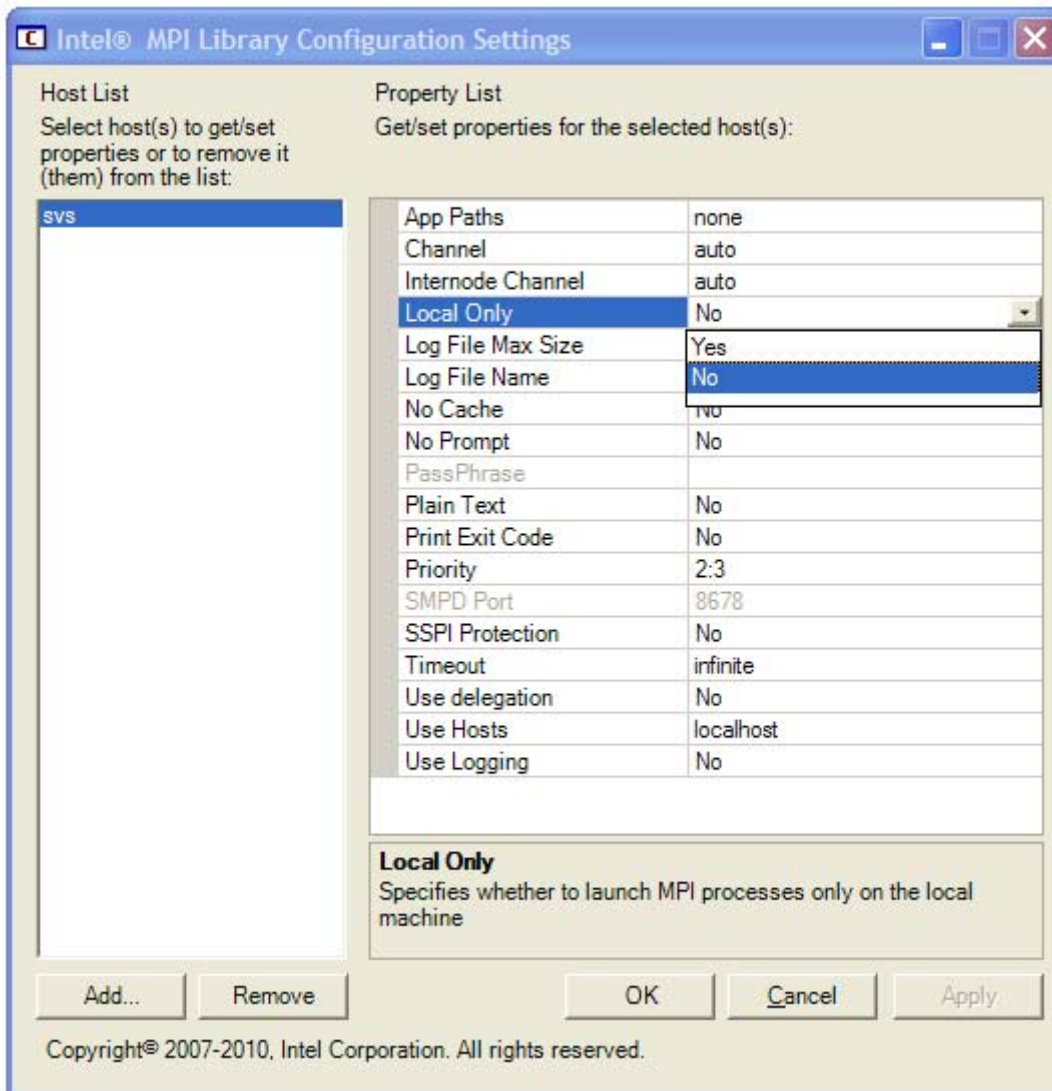
Use the `wmpiregister` utility to encrypt and store your account name and a password. The specified account and the password are used for all subsequent MPI jobs you start. The first time you use this utility, you need to enter an account name and a password during the first `wmpiexec` invocation.



Use the `wmpiexec` utility as a graphical interface to the `mpiexec` command. This utility allows you to:

1. Describe the job by specifying the following:
  - an application to run
  - a number of instances
  - host names
  - a communication device to be used
  - a working directory for the MPI processes
  - environment variables to be set for the MPI processes
  - drive mappings to be used
  - extra MPI options for `wmpiexec`

2. Save the job description using the **Save Job** button (optional).
3. Load the job description using the **Load Job** button (optional).
4. View the actual `mpiexec` command line using the **Show Command** button.
5. Launch the job using the **Execute** button.
6. Break the job execution using the **Break** button.



Use the `wmpiconfig` utility to view/change the Intel® MPI Library settings for different hosts. This affects every job run on that host. The work with the `wmpiconfig` utility can be split into three steps:

1. Select the host(s) for which you want to change the Intel® MPI Library settings, and add them to the host list using the **Add** button.
2. Select host(s) in the host list to view the properties. If more than one host name is selected, intersection of the properties is displayed.
3. Change properties for the selected host(s) and press the **Apply** button in confirmation.

## 9 Glossary

---

|                                   |                                                                                                                                                                                                                                    |
|-----------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>hyper-threading technology</b> | A feature within the IA-32 family of processors, where each processor core provides the functionality of more than one logical processor.                                                                                          |
| <b>logical processor</b>          | The basic modularity of processor hardware resource that allows a software executive (OS) to dispatch task or execute a thread context. Each logical processor can execute only one thread context at a time.                      |
| <b>multi-core processor</b>       | A physical processor that contains more than one processor core.                                                                                                                                                                   |
| <b>multi-processor platform</b>   | A computer system made of two or more physical packages.                                                                                                                                                                           |
| <b>processor core</b>             | The circuitry that provides dedicated functionalities to decode, execute instructions, and transfer data between certain sub-systems in a physical package. A processor core may contain one or more logical processors.           |
| <b>physical package</b>           | The physical package of a microprocessor capable of executing one or more threads of software at the same time. Each physical package plugs into a physical socket. Each physical package may contain one or more processor cores. |
| <b>processor topology</b>         | Hierarchical relationships of “shared vs. dedicated” hardware resources within a computing platform using physical package capable of one or more forms of hardware multi-threading.                                               |

# 10 Index

- /Z7, 10
- /Zi, 10
- /ZI/debug, 10
- {cc, cxx, fc}=<compiler>, 11
- Active Directory\*, 28
- application, 34
- check\_mpi, 10
- configfile <filename>, 15
- cpuinfo, 24, 25, 35
- delegate, 16
- dir <directory>, 18
- echo, 11
- env <ENVVAR> <value>, 17
- envall, 17
- envexcl <list of env var names>, 17
- envlist <list of env var names>, 17
- envnone, 17
- envuser, 17
- exitcodes, 16
- g<l-option>, 15
- h, 17, 23, 31
- help, 17, 23, 30, 31
- help, 17
- host <nodename>, 17
- hosts, 15, 23
- I\_MPI\_{CC, CXX, FC, F77, F90}, 12
- I\_MPI\_{CC, CXX, FC, F77, F90}\_PROFILE, 12
- I\_MPI\_ADJUST, 73, 76
- I\_MPI\_ADJUST\_<opname>, 73
- I\_MPI\_ALLGATHER\_MSG, 78
- I\_MPI\_ALLREDUCE\_MSG, 79
- I\_MPI\_ALLTOALL\_MSG, 78
- I\_MPI\_ALLTOALL\_NUM\_PROCS, 78
- I\_MPI\_AUTH\_METHOD, 29
- I\_MPI\_BCAST\_MSG, 77
- I\_MPI\_BCAST\_NUM\_PROCS, 77
- I\_MPI\_CACHE\_BYPASS, 55
- I\_MPI\_CACHE\_BYPASS\_THRESHOLDS, 56
- I\_MPI\_COMPATIBILITY, 81
- I\_MPI\_COMPILER\_CONFIG\_DIR, 13
- I\_MPI\_CONN\_EVD\_QLEN, 69
- I\_MPI\_DAPL\_BUFFER\_NUM, 66
- I\_MPI\_DAPL\_BUFFER\_SIZE, 66
- I\_MPI\_DAPL\_CHECK\_MAX\_RDMA\_SIZE, 68
- I\_MPI\_DAPL\_CONN\_EVD\_SIZE, 69
- I\_MPI\_DAPL\_DESIRED\_STATIC\_CONNECTIONS\_NUM, 70
- I\_MPI\_DAPL\_DIRECT\_COPY\_THRESHOLD, 64
- I\_MPI\_DAPL\_EAGER\_MESSAGE\_AGGREGATION, 64
- I\_MPI\_DAPL\_MAX\_MSG\_SIZE, 68
- I\_MPI\_DAPL\_PROVIDER, 50, 62
- I\_MPI\_DAPL\_RDMA\_RNDV\_WRITE, 67
- I\_MPI\_DAPL\_RNDV\_BUFFER\_ALIGNMENT, 67
- I\_MPI\_DAPL\_SCALABLE\_PROGRESS, 65
- I\_MPI\_DAPL\_SR\_BUF\_NUM, 69
- I\_MPI\_DAPL\_SR\_THRESHOLD, 69
- I\_MPI\_DAPL\_TRANSLATION\_CACHE, 63
- I\_MPI\_DAPL\_TRANSLATION\_CACHE\_AVL\_TREE, 63
- I\_MPI\_DAT\_LIBRARY, 21, 63, 81
- I\_MPI\_DEBUG, 10, 18, 19
- I\_MPI\_DEBUG\_OUTPUT, 19
- I\_MPI\_DEVICE, 18, 51, 62
- I\_MPI\_DYNAMIC\_CONNECTION, 55
- I\_MPI\_DYNAMIC\_CONNECTION\_MODE, 65
- I\_MPI\_EAGER\_THRESHOLD, 52, 53
- I\_MPI\_FABRICS, 18, 50, 51, 52
- I\_MPI\_FABRICS\_LIST, 51, 52
- I\_MPI\_FALLBACK, 51, 52
- I\_MPI\_FALLBACK\_DEVICE, 52
- I\_MPI\_FAST\_COLLECTIVES, 77
- I\_MPI\_GATHER\_MSG, 80
- I\_MPI\_INTRANODE\_DIRECT\_COPY, 53, 59
- I\_MPI\_INTRANODE\_EAGER\_THRESHOLD, 53, 54, 58, 60, 62
- I\_MPI\_INTRANODE\_SHMEM\_BYPASS, 61
- I\_MPI\_JOB\_TIMEOUT, 21
- I\_MPI\_NETMASK, 20, 71
- I\_MPI\_PIN, 36
- I\_MPI\_PIN\_CELL, 40
- I\_MPI\_PIN\_DOMAIN, 41, 42, 43
- I\_MPI\_PIN\_ORDER, 48, 49
- I\_MPI\_PIN\_PROCESSOR\_LIST, 36, 38, 39, 41
- I\_MPI\_PIN\_PROCS, 36
- I\_MPI\_PRINT\_VERSION, 19
- I\_MPI\_RDMA\_BUFFER\_NUM, 66
- I\_MPI\_RDMA\_BUFFER\_SIZE, 66
- I\_MPI\_RDMA\_CHECK\_MAX\_RDMA\_SIZE, 68
- I\_MPI\_RDMA\_CONN\_EVD\_SIZE, 69
- I\_MPI\_RDMA\_EAGER\_THRESHOLD, 64
- I\_MPI\_RDMA\_MAX\_MSG\_SIZE, 68
- I\_MPI\_RDMA\_RNDV\_BUF\_ALIGN, 67
- I\_MPI\_RDMA\_RNDV\_BUFFER\_ALIGNMENT, 67

I\_MPI\_RDMA\_RNDV\_WRITE, 67  
 I\_MPI\_RDMA\_SCALABLE\_PROGRESS, 65  
 I\_MPI\_RDMA\_TRANSLATION\_CACHE, 63  
 I\_MPI\_RDMA\_VBUF\_TOTAL\_SIZE, 66  
 I\_MPI\_RDMA\_WRITE\_IMM, 70  
 I\_MPI\_REDS CAT\_MSG, 79  
 I\_MPI\_ROOT, 13  
 I\_MPI\_SCALABLE\_OPTIMIZATION, 54  
 I\_MPI\_SCATTER\_MSG, 80  
 I\_MPI\_SHM\_BUFFER\_SIZE, 59, 60  
 I\_MPI\_SHM\_BYPASS, 61  
 I\_MPI\_SHM\_CACHE\_BYPASS, 55  
 I\_MPI\_SHM\_CACHE\_BYPASS\_THRESHOLDS, 56  
 I\_MPI\_SHM\_CELL\_NUM, 58  
 I\_MPI\_SHM\_CELL\_SIZE, 53, 58  
 I\_MPI\_SHM\_FBOX\_SIZE, 57  
 I\_MPI\_SHM\_LMT, 58  
 I\_MPI\_SHM\_LMT\_BUFFER\_NUM, 59  
 I\_MPI\_SHM\_LMT\_BUFFER\_SIZE, 59  
 I\_MPI\_SHM\_NUM\_BUFFERS, 59  
 I\_MPI\_SHM\_SPIN\_COUNT, 62  
 I\_MPI\_SMPD\_VERSION\_CHECK, 21  
 I\_MPI SOCK\_SCALABLE\_OPTIMIZATION, 54  
 I\_MPI\_SPIN\_COUNT, 54  
 I\_MPI\_SSHM, 60  
 I\_MPI\_SSHM\_BUFFER\_NUM, 60  
 I\_MPI\_SSHM\_BUFFER\_SIZE, 60  
 I\_MPI\_SSHM\_DYNAMIC\_CONNECTION, 61  
 I\_MPI\_STATS, 83, 88  
 I\_MPI\_STATS\_ACCURACY, 88, 93  
 I\_MPI\_STATS\_BUCKETS, 85, 88  
 I\_MPI\_STATS\_FILE, 85, 89  
 I\_MPI\_STATS\_SCOPE, 83, 89  
 I\_MPI\_TCP\_BUFFER\_SIZE, 71  
 I\_MPI\_TCP\_NETMASK, 71  
 I\_MPI\_TUNER\_DATA\_DIR, 15, 22  
 I\_MPI\_USE\_DAPL\_INTRANODE, 61  
 I\_MPI\_USE\_DYNAMIC\_CONNECTIONS, 55  
 I\_MPI\_USE\_RENDEZVOUS\_RDMA\_WRITE, 67  
 I\_MPI\_WAIT\_MODE, 54  
 -ilp64, 10  
 -impersonate, 16  
 IPM, 88, 93  
 --iterations, 35  
 -l, 15  
 large message transfer (LMT), 58, 59  
 LMT, 59, 60  
 -localroot, 16  
 -logon, 16  
 LP64, 96  
 -machinefile <machine file>, 14  
 -map <drive:\host\share>, 18  
 -mapall, 18  
 MAX\_INT, 68  
 max\_rdma\_size, 68  
 mpd, 31  
 MPI\_Allgather, 76, 78  
 MPI\_Allgatherv, 76  
 MPI\_Allreduce, 76, 79, 86  
 MPI\_Alltoall, 76, 78  
 MPI\_Alltoallv, 76  
 MPI\_Alltoallw, 76  
 MPI\_Barrier, 76  
 MPI\_Bcast, 76, 77, 84  
 MPI\_COMM\_WORLD, 86  
 MPI\_Exscan, 76  
 MPI\_Gather, 76, 80  
 MPI\_Gatherv, 76  
 MPI\_Init, 88  
 MPI\_Pcontrol, 88  
 MPI\_Reduce, 76, 84  
 MPI\_Reduce\_scatter, 76, 79  
 MPI\_Scan, 76  
 MPI\_scatter, 76  
 MPI\_Scatter, 76, 80  
 MPI\_Scatterv, 76  
 MPICH\_{CC, CXX, FC, F77, F90}, 12  
 mpiexec, 13, 14, 15, 21, 22, 24, 29, 33, 34, 50, 98, 99, 100  
 mpitune, 15, 30, 33  
 -mt\_mpi, 9  
 -n, 14  
 -n <# of processes>, 17  
 -nopopup\_debug, 16  
 -np <# of processes>, 17  
 NUM\_RDMA\_BUFFER, 66  
 -O, 10  
 -p <port>, 15  
 -path <directory>, 17  
 PI\_Finalize, 88  
 -port <port>, 15  
 -profile, 12  
 -profile=<profile\_name>, 10, 12  
 -pwdfile <filename>, 16  
 RDMA\_IBA\_EAGER\_THRESHOLD, 64  
 -register [-user n], 16  
 -remove [-user n], 16  
 SecureDynamicLibraryLoading, 81  
 SecurePath, 82  
 Security Service Provider Interface (SSPI), 28  
 Service Principal Name (SPN), 29  
 Service Principle Name (SPN), 24  
 -show, 11  
 -show\_env, 11  
 smpd, 23

SMPD, 14, 21, 24

-t, 10

-timeout <*seconds*>, 16

-trace, 10

-tune, 15, 33, 34

-v, 11

-verbose, 16

VT\_ROOT, 10, 13

-wdir <*directory*>, 18

-whoami, 16