1. はじめに

本書は、Intelコンパイラ を東京工業大学学術国際情報センターの TSUBAME3 で利用する方法について説明しています。
また、TSUBAME3を利用するにあたっては、TSUBAME利用の手引きもご覧下さい。
サーバの利用環境や注意事項などが詳細に記述されていますので、よく読んでください。

1.1. 利用できるバージョン

TSUBAME3で利用可能な最新バージョンについてはTSUBAME計算サービスWebサイトの アプリケーション ページをご確認下さい。
研究に支障がない限り、バグ修正の入っている最新版をご利用下さい。

1.1.1. バージョンの切り替え

本システムでは、moduleコマンドを使用することでコンパイラやアプリケーション利用環境の切り替えを行うことができます。特にバージョンの指定がない場合は、バージョン 17.0.4.196 がロードされるようになっています。

例: Ver 18.0.1.163

```
module load intel/18.0.1.163
```

例: Ver 19.0.0.117

```
module load intel/19.0.0.117
```

1.2. マニュアル

マニュアルはTSUBAMEの次のディレクトリ下にあります。

バージョン	ディレクトリ
18.0.1.163	/apps/t3/sles12sp2/isv/intel/documentation_2018/en /apps/t3/sles12sp2/isv/intel/documentation_2018/ja
19.0.0.117	/apps/t3/sles12sp2/isv/intel/documentation_2019/en

オンラインドキュメントを用意しています。(東工大学内からのみ閲覧可能)

Parallel Studio XE 2017: Getting Started with the InteR C++ Compiler 17.0 for Linux*
Parallel Studio XE 2017: Getting Started with the IntelR Fortran Compiler 17.0 for Linux*
Getting Started with IntelR VTune Amplifier XE 2017 for Linux* OS
Getting Started with IntelR Trace Analyzer and Collector for Linux* OS
Getting Started with IntelR Inspector
Getting Started with IntelR Math Kernel Library 2017 for Linux*
Getting Started with IntelR MPI Library for Linux* OS

2. 利用方法

2.1. Intelコンパイラ

Intelコンパイラの各コマンドは以下のとおりです。

コマンド名とコマンド形式

コマンド	言語	コマンド形式
ifort	Fortran 77/90/95	`$ ifort [オプション] source_file`
icc	C	`$ icc [オプション] source_file`
icpc	C++	`$ icpc [オプション] source_file`

利用する際は、moduleコマンドでintelを読み込んでください。--helpオプションを指定して頂くとコンパイラオプションの一覧が表示されます。

2.1.1. コンパイルの主なオプション

コンパイルの最適化オプションを以下に示します。

コンパイラの主なオプション

オプション	説明
-O0	すべての最適化を無効にします。
-O1	最適化を有効にします。コードサイズを大きくするだけで高速化に影響を与えるような一部の最適化を無効にします。
-O2	最適化を有効にします。一般的に推奨される最適化レベルです。ベクトル化は O2 以上のレベルで有効になります。Oオプションを指定しない場合、デフォルトでこちらが指定されます。

オプション	説明
-O3	O2 よりも積極的に最適化を行い、融合、アンロールとジャムのブロック、IF 文の折りたたみなど、より強力なループ変換を有効にします。
-xCORE-AVX2	Intel プロセッサー向けのIntel アドバンスト・ベクトル・エクステンション 2 (Intel AVX2)、Intel AVX、SSE4.2、SSE4.1、SSSE3、SSE3、SSE2、SSE 命令を生成します。Intel AVX2 命令セット対応のIntel プロセッサー向けに最適化します。
-xSSE4.2	Intel プロセッサー向けのIntel SSE4 高効率および高速な文字列処理命令、Intel SSE4 ベクトル化コンパイラ命令およびメディア・アクセラレーター命令、およびIntel SSSE3、SSE3、SSE2、SSE 命令を生成します。Intel SSE4.2 命令セット対応のIntel プロセッサー向けに最適化します。
-xSSSE3	Intel プロセッサー向けのIntel SSSE3、SSE3、SSE2、SSE 命令を生成します。Intel SSSE3 命令セット対応のIntel プロセッサー向けに最適化します。xオプションを指定しない場合、デフォルトでこちらが指定されます。
-xHOST	ホストプロセッサで利用可能な最上位の命令セットを生成します。Intel(R) Xeon(R) CPU E5-2680 v4(計算ノードのCPU)では-xCORE-AVX2となります。
-qopt-report=n	最適化レポートを生成します。デフォルトでは、レポートは.optrpt 拡張子を持つファイルに出力されます。nには、0 (レポートなし) から5 (最も詳しい) の詳細レベルを指定します。デフォルトは 2 です。
-fp-model precise	浮動小数点演算のセマンティクスを制御します。浮動小数点データの精度に影響する最適化を無効にし、中間結果をソースで定義された精度まで丸めます。
-g	-gオプションはオブジェクト・ファイルのサイズを大きくするシンボリック・デバッグ情報をオブジェクト・ファイルに生成するようにコンパイラに指示します。
-traceback	このオプションは、ランタイム時に致命的なエラーが発生したとき、ソースファイルのトレースバック情報を表示できるように、オブジェクト・ファイル内に補足情報を生成するようにコンパイラに指示します。 致命的なエラーが発生すると、コールスタックの 16 進アドレス (プログラム・カウンター・トレース) とともに、ソースファイル、ルーチン名、および行番号の相関情報が表示されます。 マップファイルとエラーが発生したときに表示されるスタックの 16 進アドレスを使用することで、エラーの原因を特定できます。 このオプションを指定すると、実行プログラムのサイズが増えます。

2.1.1.1. コンパイルの推奨最適化オプション

コンパイルの推奨最適化オプションを以下に示します。本システムに搭載しているIntel Xeon E5-2680 v4は、Intel AVX2命令セットに対応していますので、-xCORE-AVX2オプションを指定することができます。
-xCORE-AVX2を指定すると、コンパイラがソースコードを解析し、最適なAVX2、AVX、SSE命令を生成します。
推奨最適化オプションは積極的な最適化を行い、かつ安全なオプションです。
最適化のために計算の順序を変更する可能性があり、結果に誤差が生じる場合があります。

推奨最適化オプション

オプション	説明
-O3	O2 最適化を行い、融合、アンロールとジャムのブロック、IF 文の折りたたみなど、より強力なループ変換を有効にします。
-xCORE-AVX2	Intel プロセッサー向けのIntel アドバンスト・ベクトル・エクステンション 2 (Intel AVX2)、Intel AVX、SSE4.2、SSE4.1、SSSE3、SSE3、SSE2、SSE 命令を生成します。Intel AVX2 命令セット対応のIntel プロセッサー向けに最適化します。

上記のオプションを使用することにより、プログラムの性能が悪化した場合、最適化のレベルを-O2に下げるかベクトル化のオプションを変更してください。また、結果が一致していない場合、浮動小数点のオプションも試してみてください。

2.1.2. Intel 64アーキテクチャーのメモリモデル指定

次のいずれかのメモリモデルを使用して実行バイナリを作成します。

メモリモデル

メモリモデル	説明
small (-mcmodel=small)	コードとデータのすべてのアクセスが、命令ポインター (IP) 相対アドレス指定で行われるように、コードとデータはアドレス空間の最初の 2GB までに制限されます。 -mcmodelオプションを指定しない場合、デフォルトでこちらが指定されます。
medium (-mcmodel=medium)	コードはアドレス空間の最初の 2GB までに制限されますが、データは制限されません。コードは IP 相対アドレス指定でアクセスできますが、データのアクセスは絶対アドレス指定を使用する必要があります。
large (-mcmodel=large)	コードもデータも制限されません。コードもデータもアクセスは絶対アドレス指定を使用します。

IP 相対アドレス指定は 32 ビットのみ必要ですが、絶対アドレス指定は 64 ビット必要です。これは、コードサイズとパフォーマンスに影響します。(IP 相対アドレス指定の方が多少速くアクセスできます。)
プログラム内の共通ブロック、グローバルデータ、静的データの合計が2GBを越えるとき、リンク時に次のエラーメッセージが出力されます。

```
<some lib.a library>(some .o): In Function <function>:
  : relocation truncated to fit: R_X86_64_PC32 <some symbol>
……
  : relocation truncated to fit: R_X86_64_PC32 <some symbol>
```

この場合は、-mcmodel=mediumと-shared-intelを指定してコンパイル/リンクして下さい。
medium メモリモデルまたは large メモリモデルを指定した場合、Intelのランタイム・ライブラリの適切なダイナミック・バージョンが使用されるように、-shared-intel コンパイラ・オプションも指定する必要があります。

2.2. 並列化

2.2.1. スレッド並列(OpenMPと自動並列化)

OpenMP、自動並列化によるスレッド並列によりプログラムの高速化ができます。
OpenMP、自動並列化を使用する場合のコマンド形式を以下に示します。

コマンド形式(OpenMP/自動並列化)

	言語	コマンド形式
OpenMP	Fortran 77/90/95	`$ ifort -qopenmp [オプション] source_file`
	C	`$ icc -qopenmp [オプション] source_file`
	C++	`$ icpc -qopenmp [オプション] source_file`
自動並列化	Fortran 77/90/95	`$ ifort -parallel [オプション] source_file`
	C	`$ icc -parallel [オプション] source_file`
	C++	`$ icpc -parallel [オプション] source_file`

'qopt-report-phase=openmp" プションを使用することでOpenMP最適化フェーズのレポートを作成することができます。
'qopt-report-phase=par" プションを使用することで自動並列化フェーズのレポートを作成することができます。

2.2.2. プロセス並列(MPI)

Fortran/C/C++ プログラムに MPIライブラリをリンクし、プロセス並列プログラムを作成/実行することができます。
MPIを使用する場合のコマンド形式を以下に示します。
利用する際は、moduleコマンドで各MPIを読み込んでください。

コマンド形式(MPI)

MPIライブラリ	言語	コマンド形式
Intel MPI	Fortran 77/90/95	`$ mpiifort [オプション] source_file`
	C	`$ mpiicc [オプション] source_file`
	C++	`$ mpiicpc [オプション] source_file`

MPIライブラリ	言語	コマンド形式
Open MPI	Fortran 77/90/95	`$ mpifort [オプション] source_file`
	C	`$ mpicc [オプション] source_file`
	C++	`$ mpicxx [オプション] source_file`
SGI MPT	Fortran 77/90/95	`$ mpif90 [オプション] source_file`
	C	`$ mpicc [オプション] source_file`
	C++	`$ mpicxx [オプション] source_file`

2.3. 数値計算ライブラリ

2.3.1. Intel MKL

Intelマス・カーネル・ライブラリ (Intel MKL) は、工学、科学、金融系アプリケーション向けに高度に最適化され、広範囲にスレッド化された数学関数を含むライブラリです。Intel MKL は、線形代数、高速フーリエ変換 (FFT)、ベクトルマス、直接法および反復法スパースソルバー、乱数ジェネレーター、その他を含む包括的なサポートを提供します。

- BLAS
- BLACS
- LAPACK
- ScaLAPACK
- PBLAS
- スパースソルバー
- 拡張固有値ソルバー
- ベクトル数学関数 (VML)
- 統計関数 (VSL)
- コンベンショナル DFT とクラスタ DFT
-

偏微分方程式のサポート

- 非線形最適化問題ソルバー
- データ・フィッティング関数
- FFTWへのラッパー・インタフェース

2.3.2. Intel MKLのリンク方法

Intel MKLのリンク方法は用途に応じてオプションを選んでください。コンパイラ・オプション -mklを利用してリンクすることができます。
FFTWへのラッパー・インタフェースを利用するとき、コンパイル時にFFTWのヘッダーファイルが必要です。コンパイル時に、`-I $MKLROOT/include/fftw` を指定して下さい。
スレッド並列化版のIntel MKL(Intel コンパイラ & SGI MPT)を以下に示します。

-mklオプションを使用したリンク方法(Intel MKL逐次版)

オプション	説明
`-mkl=sequential`	BLAS, LAPACK, DFT, FFTW(ラッパー・インタフェース)
`-mkl=sequential -lmkl_scalapack_lp64 -lmkl_blacs_sgimpt_lp64 -lmpi`	ScaLAPACK, BLACS(SGI MPT)

-mklオプションを使用したリンク方法(Intel MKLスレッド並列化版)

オプション	説明
`-mkl=parallel`	BLAS, LAPACK, DFT, FFTW(ラッパー・インタフェース)
`-mkl=parallel -lmkl_scalapack_lp64 -lmkl_blacs_sgimpt_lp64 -lmpi`	ScaLAPACK, BLACS(SGI MPT)

`-mkl` オプションを使わずに必要なライブラリを全て明示的にリンクする場合は、以下のように指定して下さい。

Intel MKLのリンク方法(Intel MKL逐次版)

オプション	説明
`-lmkl_intel_lp64 -lmkl_sequential -lmkl_core`	BLAS, LAPACK, DFT, FFTW(ラッパー・インタフェース)
`-lmkl_scalapack_lp64 -lmkl_blacs_sgimpt_lp64\ -lmkl_intel_lp64 -lmkl_sequential -lmkl_core -lmpi`	ScaLAPACK, BLACS(SGI MPT)

Intel MKLのリンク方法(Intel MKLスレッド並列化版)

オプション	説明
`-lmkl_intel_lp64 -lmkl_intel_thread -lmkl_core -liomp5 -lpthread`	BLAS, LAPACK, DFT, FFTW(ラッパー・インタフェース)
`-lmkl_scalapack_lp64 -lmkl_blacs_sgimpt_lp64 -lmkl_intel_lp64\ -lmkl_intel_thread -lmkl_core -liomp5 -lpthread -lmpi`	ScaLAPACK, BLACS(SGI MPT)

【補足】
- 以下のページでは適切なリンク行を自動的に生成します。
Intel Math kernel Library Link Line Advisor
MKLの使用方法（ダイナミックリンクまたはスタティックリンク、逐次版またはスレッド並列化版など）を指定しますと適切なリンク行を表示します。

2.3.3. 64ビット整数型に対応したライブラリのリンク

64ビット整数型に対応したライブラリ（ILP64ライブラリ）は以下の目的で提供されています。

- 大規模なデータ配列 (要素数 2^31-1 以上) をサポートする
- `-i8` コンパイラ・オプションを使用して Fortran コードをコンパイルできるようにする

上記で、`-lmkl_intel_lp64` が指定されていますが、ILP64ライブラリをリンクするときには、`-lmkl_intel_ilp64` を指定して下さい。
`-mkl` オプションではILP64ライブラリをリンクすることはできません。

ILP64用のコンパイル
- Fortran : `ifort -i8 …`
- C/C++ : `icc -DMKL_ILP64 …`

【補足】
- `-i8` または `-DMKL_ILP64` オプションを使用してコンパイルしたプログラムと LP64 ライブラリをリンクすると、予測できない結果や誤出力が発生する場合があります。

2.3.4. 実行スレッド数

ジョブスクリプト内の `OMP_NUM_THREADS` で設定したスレッド数で実行されます。
ジョブスクリプトについては、SMP並列を参照してください。
MKLのスレッド数のみを変更したいときはMKLの環境変数 `MKL_NUM_THREADS` を設定してください。

2.3.5. ライセンス使用状況の確認

Intelコンパイラのライセンス利用状況を以下のコマンドで確認できます。

```
$ lmutil lmstat -S INTEL -c 27011@lice0:27011@remote:27011@t3ldap1
```

改訂履歴

改定日付	内容
2019/07/02	mkdocs版作成
2017/09/15	初版作成