インテル® ソフトウェア開発ツールで始める

コードの現代化と最適化

最適化済みのライブラリー群の活用

2019 年 4 月
iSUS 編集長
すがわら　きよふみ

内容

インテル® パフォーマンス・ライブラリー

- インテル® Distribution for Python*
- インテル® マス・カーネル・ライブラリー (インテル® MKL)
- インテル® インテグレーテッド・パフォーマンス・プリミティブ (インテル® IPP)
- インテル® スレッディング・ビルディング・ブロック (インテル® TBB)
- インテル® Data Analytics Acceleration Library (インテル® DAAL)

© 2019 iSUS *その他の社名、製品名などは、一般に各社の表示、商標または登録商標です。

インテル® Parallel Studio XE のコンポーネント

包括的なソフトウェア開発ツールスイート

Composer Edition

ビルド
コンパイラーとライブラリー

インテル® C/C++、Fortran コンパイラー

インテル® MKL[1]

インテル® DAAL[2]

インテル® TBB[3]
C++ スレッド・ライブラリー

インテル® IPP[4]
画像、信号、データ処理

インテル® Distribution for Python*
ハイパフォーマンスな Python*

Professional Edition

解析
解析ツール

インテル® VTune™ Amplifier
パフォーマンス・プロファイラー

インテル® Inspector
メモリー/スレッドのデバッガー

インテル® Advisor
ベクトル化の最適化、スレッドのプロトタイプ生成、フローグラフ解析

Cluster Edition

スケール
クラスターツール

インテル® MPI ライブラリー
メッセージ・パッシング・インターフェイス・ライブラリー

インテル® Trace Analyzer & Collector
MPI チューニングと解析

インテル® Cluster Checker
クラスター診断エキスパート・システム

オペレーティング・システム: Windows*、Linux*、macOS*[5]

インテル® アーキテクチャー・ベースのプラットフォーム

© 2019 iSUS *その他の社名、製品名などは、一般に各社の表示、商標または登録商標です。

[1] インテル® マス・カーネル・ライブラリー
[2] インテル® データ・アナリティクス・アクセラレーション・ライブラリー
[3] インテル® スレッディング・ビルディング・ブロック
[4] インテル® インテグレーテッド・パフォーマンス・プリミティブ
[5] インテル® Parallel Studio XE Composer Edition でのみ利用可能

インテル® Distribution for Python* で Python* を高速化

科学計算、データ解析、マシンラーニング、ディープラーニング向けのハイパフォーマンスな Python*

高速なパフォーマンス	優れた生産性	エコシステムの互換性
パフォーマンス・ライブラリー、並列化、マルチスレッド化、言語拡張	**事前ビルドの高速化されたパッケージ**	**Python* 2.7/3.6、conda、pip をサポート**
▪ インテル® MKL とインテル® DAAL により NumPy*/SciPy*/scikit-learn を高速化 ▪ scikit-learn、pyDAAL、TensorFlow*、Caffe* によるデータ解析、マシンラーニング、ディープラーニング ▪ Numba* と Cython* によるスケーリング ▪ 最適化された mpi4py、Dask & PySpark で動作 ▪ 最新のインテル® アーキテクチャー向けに最適化	▪ 数値計算、マシンラーニング/ディープラーニング、HPC、データ解析向けの事前ビルドの最適化されたパッケージ ▪ 既存の Python* から簡単に移行可能 - コード変更不要 ▪ Jupyter* Notebook、Matplotlib を含む ▪ 商用開発を含むすべてのユーザーに無料で提供	▪ Python* 2.7/3.6 をサポート、最適化を Anaconda* ディストリビューションに統合 ▪ ディストリビューションおよび最適化されたパッケージは conda、pip、apt get、yum、DockerHub から利用可能、さまざまなパフォーマンスの最適化を Anaconda* ディストリビューションに統合 ▪ 最適化はメインの Python* トランクに反映される ▪ インテル® Parallel Studio XE を通してプライオリティー・サポートが受けられる

オペレーティング・システム: Windows*、Linux*、macOS*[1]

インテル® アーキテクチャー・ベースのプラットフォーム

詳細: isus.jp/python-distribution/

[1]インテル® Parallel Studio XE Composer Edition でのみ利用可能

© 2019 iSUS *その他の社名、製品名などは、一般に各社の表示、商標または登録商標です。

インテル® Distribution for Python* で Python* を高速化

ネイティブコードに近いパフォーマンスを実現

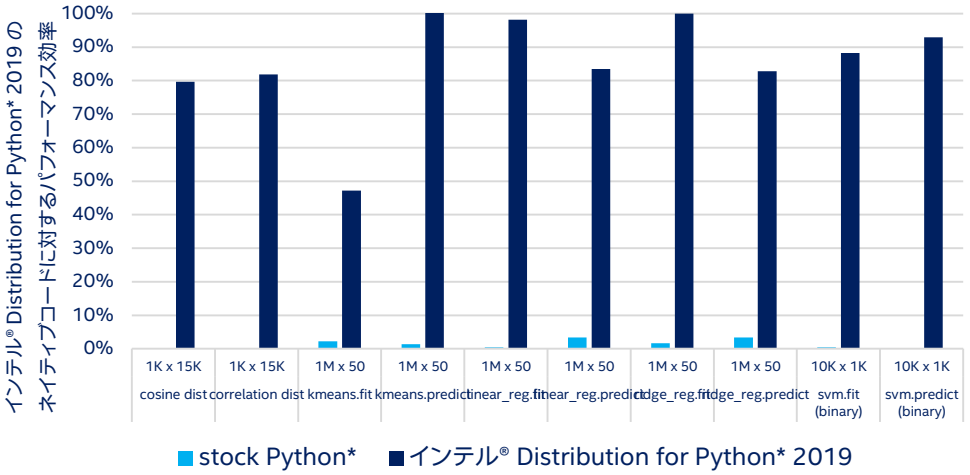
- 科学計算、マシンラーニング、データ解析向けに NumPy*/SciPy*/scikit-learn を高速化
- 既存の Python* から簡単に移行可能 - コード変更不要
- 最新のインテル® プロセッサー向けの高度な最適化

バージョン 2019 の新機能

- scikit-learn を利用した高速なマシンラーニング: インテル® DAAL により高速化されたサポート・ベクトル・マシン (SVM) と K 平均法の予測
- マシンラーニング XGBoost ライブラリーを含む (Linux* のみ)
- コマンドラインのスタンドアロン・インストールとしても利用可能

インテル® Distribution for Python* 2019 はネイティブコード scikit-learn に匹敵するパフォーマンスを達成

stock Python* パッケージとの比較 - インテル® Xeon® プロセッサー



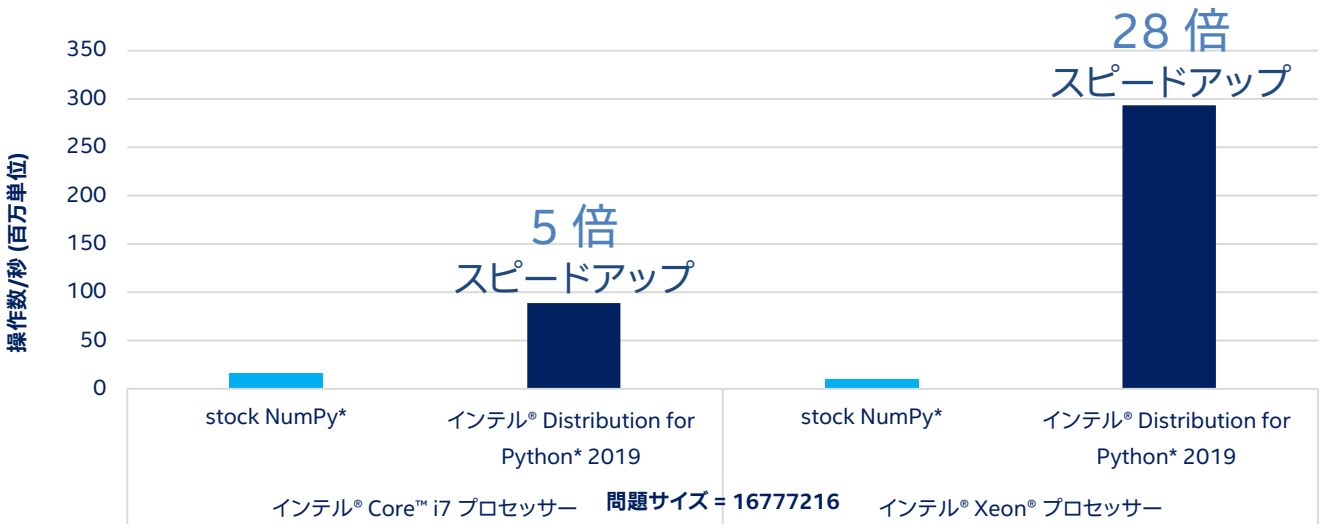
パフォーマンス結果は **2018 年 7 月 9 日**時点のテスト結果に基づいたものであり、公開されている利用可能なすべてのセキュリティー・アップデートが適用されていない可能性があります。詳細については、構成の開示を参照してください。絶対的なセキュリティーを提供できる製品はありません。
性能に関するテストに使用されるソフトウェアとワークロードは、性能がインテル® マイクロプロセッサー用に最適化されていることがあります。SYSmark* や MobileMark* などの性能テストは、特定のコンピューター・システム、コンポーネント、ソフトウェア、操作、機能に基づいて行ったものです。結果はこれらの要因によって異なります。製品の購入を検討される場合は、他の製品と組み合わせた場合の本製品の性能など、ほかの情報や性能テストも参考にして、パフォーマンスを総合的に評価することをお勧めします。詳細については、パフォーマンス・ベンチマーク・テストの開示 (英語) を参照してください。2018 年 7 月 9 日時点のインテルによるテスト。**システム構成:** stock Python*: python 3.6.6 hc3d631a_0 を conda でインストール。numpy 1.15、numba 0.39.0、llvmlite 0.24.0、scipy 1.1.0、scikit-learn 0.19.2 を pip でインストール。インテル® Distribution for Python* 2019 Gold: python 3.6.5 intel_11、numpy 1.14.3 intel_py36_5、mkl 2019.0 intel_101、mkl_fft 1.0.2 intel_np114py36_6、mkl_random 1.0.1 intel_np114py36_6、numba 0.39.0 intel_np114py36_0、llvmlite 0.24.0 intel_py36_0、scipy 1.1.0 intel_np114py36_6、scikit-learn 0.19.1 intel_np114py36_35。Linux* OS: CentOS* 7.3.1611、カーネル 3.10.0-514.el7.x86_64。ハードウェア: インテル® Xeon® Gold 6140 プロセッサー @ 2.30GHz (2 ソケット、18 コア/ソケット、ハイパースレッディング無効)、256GB DDR4 RAM、16 DIMM (16GB @ 2666MHz)。

インテル® コンパイラーでは、インテル® マイクロプロセッサーに限定されない最適化に関して、他社製マイクロプロセッサー用に同等の最適化を行えないことがあります。これには、インテル® ストリーミング SIMD 拡張命令 2、インテル® ストリーミング SIMD 拡張命令 3、インテル® ストリーミング SIMD 拡張命令 3 補足命令などの最適化が該当します。インテルは、他社製マイクロプロセッサーに関して、いかなる最適化の利用、機能、または効果も保証いたしません。本製品のマイクロプロセッサー依存の最適化は、インテル® マイクロプロセッサーでの使用を前提としています。インテル® マイクロアーキテクチャーに限定されない最適化のなかにも、インテル® マイクロプロセッサー用のものがあります。この注意事項で言及した命令セットの詳細については、該当する製品のユーザー・リファレンス・ガイドを参照してください。注意事項の改訂#20110804

© 2019 iSUS *その他の社名、製品名などは、一般に各社の表示、商標または登録商標です。

インテルの最適化によりアプリケーションのパフォーマンスを向上

インテル® Distribution for Python* — ブラックショールズ方程式で優れたパフォーマンスを達成



パフォーマンス結果は **2018 年 7 月 9 日**時点のテスト結果に基づいたものであり、公開されている利用可能なすべてのセキュリティー・アップデートが適用されていない可能性があります。詳細については、構成の開示を参照してください。絶対的なセキュリティーを提供できる製品はありません。性能に関するテストに使用されるソフトウェアとワークロードは、性能がインテル® マイクロプロセッサー用に最適化されていることがあります。SYSmark* や MobileMark* などの性能テストは、特定のコンピューター・システム、コンポーネント、ソフトウェア、操作、機能に基づいて行ったものです。結果はこれらの要因によって異なります。製品の購入を検討される場合は、他の製品と組み合わせた場合の本製品の性能など、ほかの情報や性能テストも参考にして、パフォーマンスを総合的に評価することをお勧めします。詳細については、パフォーマンス・ベンチマーク・テストの開示 (英語) を参照してください。
2018 年 7 月 9 日時点のインテルによるテスト。**システム構成:** stock Python*: python 3.6.6 hc3d631a_0 を conda でインストール。numpy 1.15、numba 0.39.0、llvmlite 0.24.0、scipy 1.1.0、scikit-learn 0.19.2 を pip でインストール。インテル® Distribution for Python* 2019 Gold: python 3.6.5 intel_11、numpy 1.14.3 intel_py36_5、mkl 2019.0 intel_101、mkl_fft 1.0.2 intel_np114py36_6、mkl_random 1.0.1 intel_np114py36_6、numba 0.39.0 intel_np114py36_0、llvmlite 0.24.0 intel_py36_0、scipy 1.1.0 intel_np114py36_6、scikit-learn 0.19.1 intel_np114py36_35。Linux* OS: CentOS* 7.4.1708、カーネル 3.10.0-693.el7.x86_64。ハードウェア: インテル® Core™ i7-7567U プロセッサー @ 3.50GHz、32GB DDR4 RAM、2 DIMM (16GB @ 2133MHz)。

インテル® コンパイラーでは、インテル® マイクロプロセッサーに限定されない最適化に関して、他社製マイクロプロセッサー用に同等の最適化を行えないことがあります。これには、インテル® ストリーミング SIMD 拡張命令 2、インテル® ストリーミング SIMD 拡張命令 3、インテル® ストリーミング SIMD 拡張命令 3 補足命令などの最適化が該当します。インテルは、他社製マイクロプロセッサーに関して、いかなる最適化の利用、機能、または効果も保証いたしません。本製品のマイクロプロセッサー依存の最適化は、インテル® マイクロプロセッサーでの使用を前提としています。インテル® マイクロアーキテクチャーに限定されない最適化のなかにも、インテル® マイクロプロセッサー用のものがあります。この注意事項で言及した命令セットの詳細については、該当する製品のユーザー・リファレンス・ガイドを参照してください。注意事項の改訂 #20110804

© 2019 iSUS *その他の社名、製品名などは、一般に各社の表示、商標または登録商標です。

ほかに確認すべき点

最適化された Python* 以外の Python*/C/C++ アプリケーション･コードの効率は?

パフォーマンス低下の原因が明らかでないものはないか?

パフォーマンス解析により解明できる

© 2019 iSUS *その他の社名、製品名などは、一般に各社の表示、商標または登録商標です。

パフォーマンス向上のための Python* + ネイティブコードのチューニング

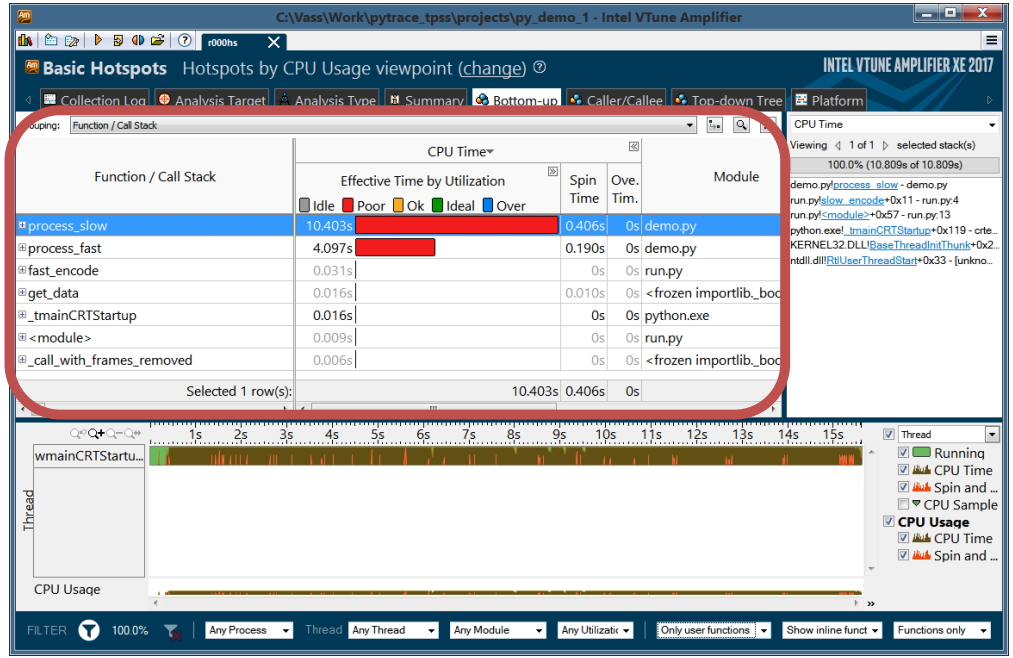
インテル® VTune™ Amplifier によるパフォーマンス解析 (インテル® Parallel Studio XE に含まれる)

課題

- Python* とネイティブコードが混在するアプリケーションを 1 つのツールでプロファイル
- 非効率的なランタイム実行を検出

ソリューション

- Python*/C/C++ 混在コードと拡張を自動検出
- パフォーマンス hotspot を行レベルで正確に識別
- 低オーバーヘッド、実行中のアプリケーションへのアタッチ/デタッチ
- パフォーマンスへの影響が大きい個所をチューニング



Python* とネイティブ関数の自動検出およびパフォーマンス解析

インテル® VTune™ Amplifier とインテル® Parallel Studio XE で利用可能

© 2019 iSUS *その他の社名、製品名などは、一般に各社の表示、商標または登録商標です。

問題のコードを素早く正確に診断

Basic Hotspots Hotspots by CPU Usage viewpoint (change)

INTEL VTUNE AMPLIFIER XE 2017

Collection Log | Analysis Target | Analysis Type | Summary | Bottom-up | Caller/Callee | Top-down Tree | Platform | demo.py

Source | Assembly | Assembly grouping: Address

S. Li.	Source	CPU Time: Total	CPU Time: Self
8	def __init__(self, input):		
9	self.input = input		
11	def process_slow(self, use_trans):		
12	if use_trans:		
13	table = maketrans('.join(self.CHAR.		
14	return self.input.translate(table)		
15	result = ''		
16	for ch in self.input:	2.0%	2.0%
17	result += self.CHAR_MAP.get(ch, ch)	69.2%	69.2%
18	return result		
19			
20	def process_fast(self):		
21	result = []		
22	for ch in self.input:		
23	result.append(self.CHAR_MAP.get(ch, .		
24	return ''.join(result)		
25			
Sele..		69.2%	

CPU Time

Viewing 1 of 1 selected stack(s)

100.0% (10.809s of 10.809s)

demo.pyprocess_slow - demo.py

run.pyslow_encode+0x11 - run.py:4

run.py<module>+0x57 - run

python.exe_tmainCRTStart

KERNEL32.DLLBaseThrea

ntdll.dllRtlUserThreadStart

ボトルネックのコード行を正確に特定

Python* からネイティブ関数への呼び出しの詳細

Basic Hotspots Hotspots by CPU Usage viewpoint (change)

INTEL VTUNE AMPLIFIER XE 2017

Collection Log | Analysis Target | Analysis Type | Summary | Bottom-up | Caller/Callee | Top-down Tree | Platform | demo.py

CPU Time: Total

Function	Effective Time by Utilization (Idle / Poor / Ok / Ideal / Over)	Spin Time	Ove.. Time
_tmainCRTStartup	96.0%	4.0%	0.0%
BaseThreadInitThunk	96.0%	4.0%	0.0%
RtlUserThreadStart	96.0%	4.0%	0.0%
Py_Main	96.0%	4.0%	0.0%
PyEval_EvalCodeEx	96.0%	4.0%	0.0%
call_function	95.9%	4.0%	0.0%
fast_function	95.9%	4.0%	0.0%
PyEval_EvalCode	95.9%	4.0%	0.0%
<module>	95.9%	4.0%	0.0%
run_file	95.9%	4.0%	0.0%
PyRun_AnyFileExFlags	95.9%	4.0%	0.0%
PyRun_SimpleFileExFlags	95.9%	4.0%	0.0%
PyRun_FileExFlags	95.9%	4.0%	0.0%
run_mod	95.9%	4.0%	0.0%
slow_encode	68.5%	2.7%	0.0%
process_slow	68.5%	2.7%	0.0%
unicode_concatenate	47.0%	1.4%	0.0%
Highlighted 1 row(s):	68.5%	2.7%	0.0%

CPU Time: Total

Calle...	Effective Time by Utilization (Idle / Poor / Ok / Ideal / Over)	Spin Time	Ove.. Time	Idle
proce	96.2%	3.8%	0.0%	0.8
fast_	96.2%	3.8%	0.0%	0.8
Selec...	96.2%	3.8%	0.0%	

Callees	CPU Time: Total
process_slow	10.809s
unicode_concatenate	7.337s
PyUnicode_Append	7.229s
unicode_resize	6.929s
resize_compact	6.799s
realloc	6.279s
RtlReAllocateHeap	6.230s
func@0x18002f110	6.091s
RtlCopyMemory	4.100s
RtlFreeHeap	1.195s
Selected 1 row(s):	6.091s

FILTER 100.0% | Any Process | Thread Any Thread | Any Module | Any Utilizatic | User/system functior | Show inline funct | Functions only

© 2019 iSUS *その他の社名、製品名などは、一般に各社の表示、商標または登録商標です。

詳細な解析により詳しい情報を提供

C:\Vass\Work\pytrace_tpss\projects\py_demo_1 - Intel VTune Amplifier

Basic Hotspots Hotspots by CPU Usage viewpoint (change)

INTEL VTUNE AMPLIFIER XE 2017

Collection Log | Analysis Target | Analysis Type | Summary | Bottom-up | Caller/Callee | Top-down Tree | Platform | demo.py

Grouping: Function / Call Stack

Function / Call Stack	CPU Time: Effective Time by Utilization (Idle, Poor, Ok, Ideal, Over)	Spin Time	Ove. Tim.	Module
RtlCopyMemory	4.360s	0s	0s	ntdll.dll
NtFreeVirtualMemory	1.289s	0s	0s	ntdll.dll
process_fast	0.907s	0s	0s	demo.py
process_slow	0.857s	0s	0s	demo.py
PyObject_GenericGetAttrWithDict	0.598s	0s	0s	python34.dll
call_function	0.531s	0s	0s	python34.dll
func@0x18002f110	0.518s	0s	0s	ntdll.dll
Selected 1 row(s):	0.857s	0s	0s	

CPU Time

Viewing 1 of 1 selected stack(s)

100.0% (0.857s of 0.857s)

demo.py!process_slow - demo.py
python34.dll!fast_function+0xd7 - ceval.c...
python34.dll!call_function+0x310 - ceval...
run.py!slow_encode+0x11 - run.py:4
python34.dll!fast_function+0xd7 - ceval.c...
python34.dll!call_function+0x310 - ceval...
run.py!<module>+0x57 - run.py:13
python34.dll!PyEval_EvalCodeEx+0x65b...
python34.dll!PyEval_EvalCode+0x2d - c...
python34.dll!run_mod+0x52 - pythonrun...
python34.dll!PyRun_FileExFlags+0xb4 -...
python34.dll!PyRun_SimpleFileExFlags...
python34.dll!PyRun_AnyFileExFlags+0x...
python34.dll!run_file+0xb9 - main.c:319
python34.dll!Py_Main+0x86d - main.c:7...
python.exe!_tmainCRTStartup+0x119 - ...

1s 2s 3s 4s 5s 6s 6.667s 8s 9s 10s 11s 12s 13s 14s 15s

Thread: wmainCRTStartu...

CPU Usage

wmainCRTStartup (TID: 2304)
CPU Time 100.0%
Spin and Overhead Time 0.0%

Thread: Running, CPU Time, Spin and ..., CPU Sample

CPU Usage: CPU Time, Spin and ...

FILTER 100.0% | Any Process | Thread Any Thread | Any Module | Any Utilizatic | User/system functior | Show inline funct | Functions only

詳細なタイムライン解析

Python* と
ネイティブコードの
コールスタックの
リスト

© 2019 iSUS *その他の社名、製品名などは、一般に各社の表示、商標または登録商標です。

インテル® MKL による高速でスケーラブルなコード

- 高度に最適化、スレッド化、ベクトル化された数学関数により、科学、工学、金融、マシンラーニング・アプリケーションにおける計算を高速化
- 密/スパース線形代数 (BLAS、LAPACK、PARDISO)、FFT、ベクトル演算、サマリー統計、ディープラーニング、スプラインなどの主な機能を提供
- コードを分岐せずに各プロセッサー向けに最適化されたコードを自動ディスパッチ
- シングルコアのベクトル化およびキャッシュ効率を向上する最適化
- マルチコアとメニーコアの自動並列化
- コアからクラスターにスケーリング
- 無料およびロイヤルティー・フリーで利用可能
- 最小限の労力で優れたパフォーマンスを達成

インテル® MKL に含まれる機能

- 密/スパース線形代数
- 高速フーリエ変換
- ベクトル演算
- ベクトル RNG
- 高速ポアソンソルバー
- その他

© 2019 iSUS *その他の社名、製品名などは、一般に各社の表示、商標または登録商標です。

インテル® MKL 2019 の新機能

JIT 高速小行列乗算

- JIT 機能によりインテル® AVX2 およびインテル® AVX-512 向け S/DGEMM の速度を向上

スパース QR ソルバー

- スパース線形方程式、スパース線形最小 2 乗問題、固有値問題、ランクと null 空間の決定などのソルバー

多項式向け乱数生成

- 金融、地質学、生物学分野向けに高度に最適化された多項式乱数ジェネレーター

© 2019 iSUS *その他の社名、製品名などは、一般に各社の表示、商標または登録商標です。

インテル® MKL の機能

線形代数

- BLAS
- LAPACK
- ScaLAPACK
- スパース BLAS
- 反復法スパース ソルバー
- PARDISO
- クラスター・スパース・ソルバー

FFT

- 多次元
- FFTW インターフェイス
- クラスター FFT

ベクトル RNG

- 合同数
- Wichmann-Hill
- メルセンヌツイスター
- Sobol
- Neiderreiter
- 非決定的

サマリー統計

- 尖度
- 変化係数
- 順序統計量
- 最小/最大
- 分散/共分散

ベクトル演算

- 三角
- 双曲線
- 指数
- 対数
- 累乗
- 累乗根

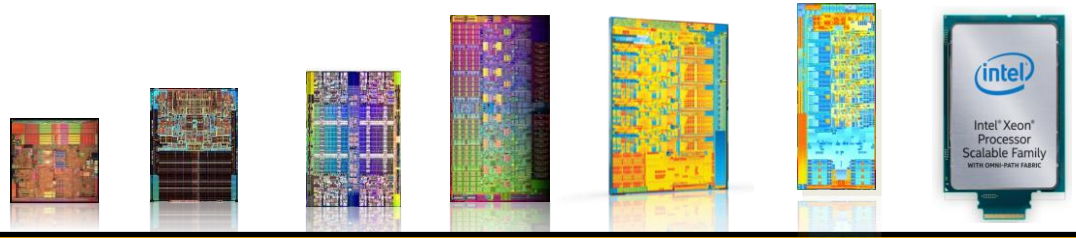
その他

- スプライン
- 補間
- 信頼領域
- 高速ポアソン ソルバー

© 2019 iSUS *その他の社名、製品名などは、一般に各社の表示、商標または登録商標です。

チューニングされた ISA 固有のコードパスに自動ディスパッチ

コア数の増加 → スレッド数の増加 → ベクトル幅の増加



	インテル® Xeon® プロセッサー 64 ビット	インテル® Xeon® プロセッサー 5100 番台	インテル® Xeon® プロセッサー 5500 番台	インテル® Xeon® プロセッサー 5600 番台	インテル® Xeon® プロセッサー E5-2600 v2 製品ファミリー	インテル® Xeon® プロセッサー E5-2600 v3 製品ファミリー v4 製品ファミリー	インテル® Xeon® スケーラブル・プロセッサー[1]
最大コア数	1	2	4	6	12	18-22	28
最大スレッド数	2	2	8	12	24	36-44	56
SIMD 幅	128	128	128	128	256	256	512
ベクトル ISA	インテル® SSE3	インテル® SSE3	インテル® SSE4、インテル® SSE4.1	インテル® SSE4.2	インテル® AVX	インテル® AVX2	インテル® AVX-512

1. 発売済みおよび出荷済み製品の製品仕様は ark.intel.com/ja を参照してください。

© 2019 iSUS *その他の社名、製品名などは、一般に各社の表示、商標または登録商標です。

コア、マルチコア、メニーコア、さらにその先へ 自動でパフォーマンスをスケーリング

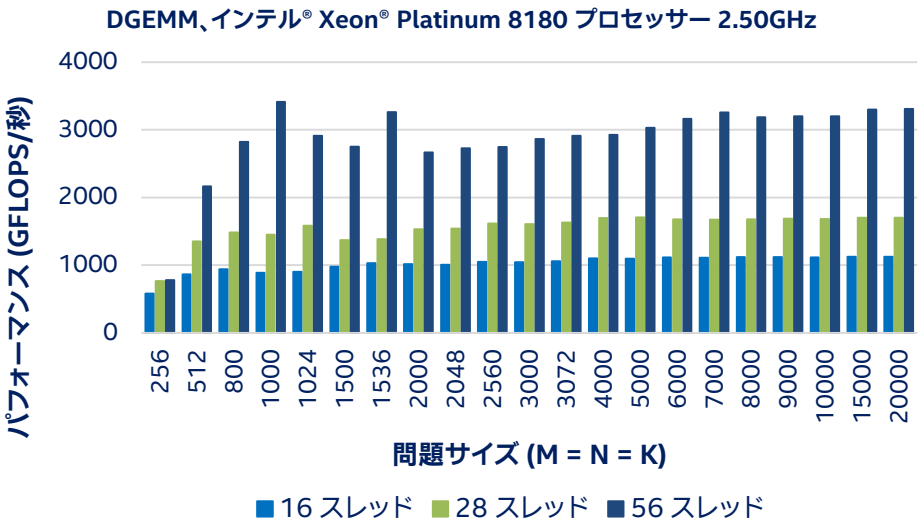
インテル® MKL

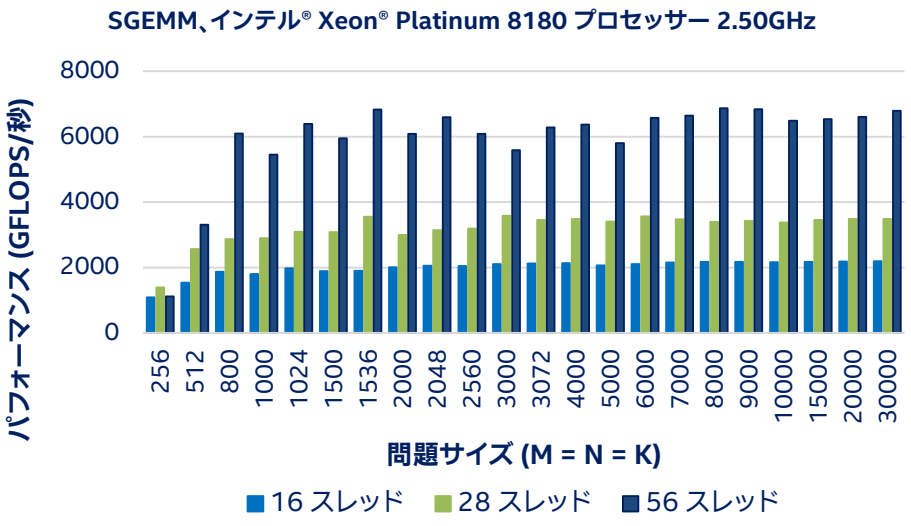
計算リソースのパフォーマンスを引き出す

- コア: **ベクトル化**、プリフェッチ、キャッシュ効率
- マルチコア/メニーコア (プロセッサー/ソケット) レベルの**並列化**
- マルチソケット (ノード) レベルの**並列化**
- クラスター**・スケーリング**

© 2019 iSUS *その他の社名、製品名などは、一般に各社の表示、商標または登録商標です。

インテル® MKL によりインテル® Xeon® プロセッサー向けに最適化された DGEMM、SGEMM





パフォーマンス結果は **2018 年 7 月 9 日**時点のテスト結果に基づいたものであり、公開されている利用可能なすべてのセキュリティー・アップデートが適用されていない可能性があります。詳細については、構成の開示を参照してください。絶対的なセキュリティーを提供できる製品はありません。性能に関するテストに使用されるソフトウェアとワークロードは、性能がインテル® マイクロプロセッサー用に最適化されていることがあります。SYSmark* や MobileMark* などの性能テストは、特定のコンピューター・システム、コンポーネント、ソフトウェア、操作、機能に基づいて行ったものです。結果はこれらの要因によって異なります。製品の購入を検討される場合は、他の製品と組み合わせた場合の本製品の性能など、ほかの情報や性能テストも参考にして、パフォーマンスを総合的に評価することをお勧めします。詳細については、パフォーマンス・ベンチマーク・テストの開示 (英語) を参照してください。
2018 年 7 月 9 日時点のインテルによるテスト。**システム構成:** インテル® Xeon® Platinum 8180 プロセッサー、2x28 コア、2.50GHz、192GB DDR4-2666。インテル® コンパイラーでは、インテル® マイクロプロセッサーに限定されない最適化に関して、他社製マイクロプロセッサー用に同等の最適化を行えないことがあります。これには、インテル® ストリーミング SIMD 拡張命令 2、インテル® ストリーミング SIMD 拡張命令 3、インテル® ストリーミング SIMD 拡張命令 3 補足命令などの最適化が該当します。インテルは、他社製マイクロプロセッサーに関して、いかなる最適化の利用、機能、または効果も保証いたしません。本製品のマイクロプロセッサー依存の最適化は、インテル® マイクロプロセッサーでの使用を前提としています。インテル® マイクロアーキテクチャーに限定されない最適化のなかにも、インテル® マイクロプロセッサー用のものがあります。この注意事項で言及した命令セットの詳細については、該当する製品のユーザー・リファレンス・ガイドを参照してください。注意事項の改訂 #20110804

© 2019 iSUS *その他の社名、製品名などは、一般に各社の表示、商標または登録商標です。

インテル® IPP により画像、ビジョン、信号、セキュリティー、ストレージ・アプリケーションのパフォーマンスを向上

画像、信号、データ処理および暗号化計算を高速化

- マルチコア、マルチ OS、マルチプラットフォームに対応した計算集約型の高度に最適化された関数
- ハイパフォーマンスで、使いやすい、プロダクション環境対応の API により素早くアプリケーション・パフォーマンスを向上
- ソフトウェアの開発と保守にかかる費用を軽減し、開発期間を短縮

バージョン 2019 の新機能

- 大規模データストレージの課題への対応を支援する ZFP 浮動小数点データ圧縮関数 (石油/ガス分野向け)
- bzip2 ソース 1.0.6 の最適化パッチファイル
- エントロピーが高いデータの LZ4 圧縮/展開パフォーマンスを向上
- RBG イメージから CIE Lab 色モデル (およびその逆) に変換する新しい色変換関数
- インテル® AVX-512 (英語) およびインテル® AVX2 (英語) 命令セット向けの最適化を拡張
- インテル® IPP Cryptography ライブラリーのオープンソース・ディストリビューション

詳細: isus.jp/intel-ipp/

© 2019 iSUS *その他の社名、製品名などは、一般に各社の表示、商標または登録商標です。

インテル® IPP

ハイパフォーマンスで、使いやすい、プロダクション環境対応の API

画像ドメイン	信号ドメイン	データドメイン
画像処理	信号処理	データ圧縮
コンピューター・ビジョン	ベクトル演算	暗号化
カラー変換		文字列処理

オペレーティング・システム: Windows*、Linux*、macOS*[1]

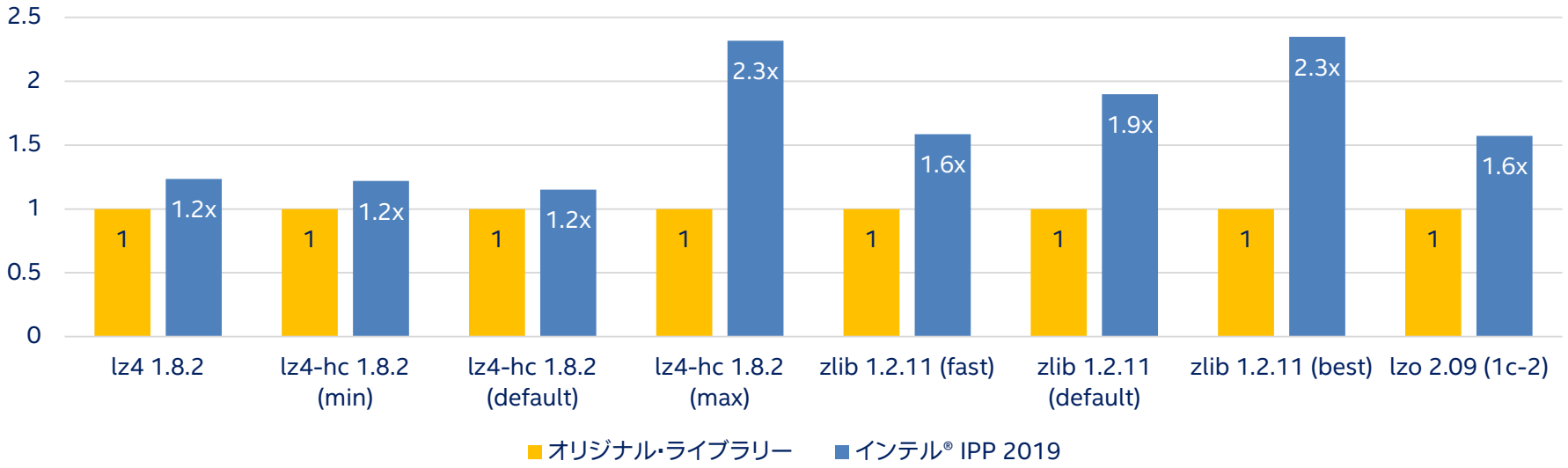
インテル® アーキテクチャー・ベースのプラットフォーム

[1]インテル® Parallel Studio XE Composer Edition でのみ利用可能

© 2019 iSUS *その他の社名、製品名などは、一般に各社の表示、商標または登録商標です。

データ圧縮のパフォーマンスを向上

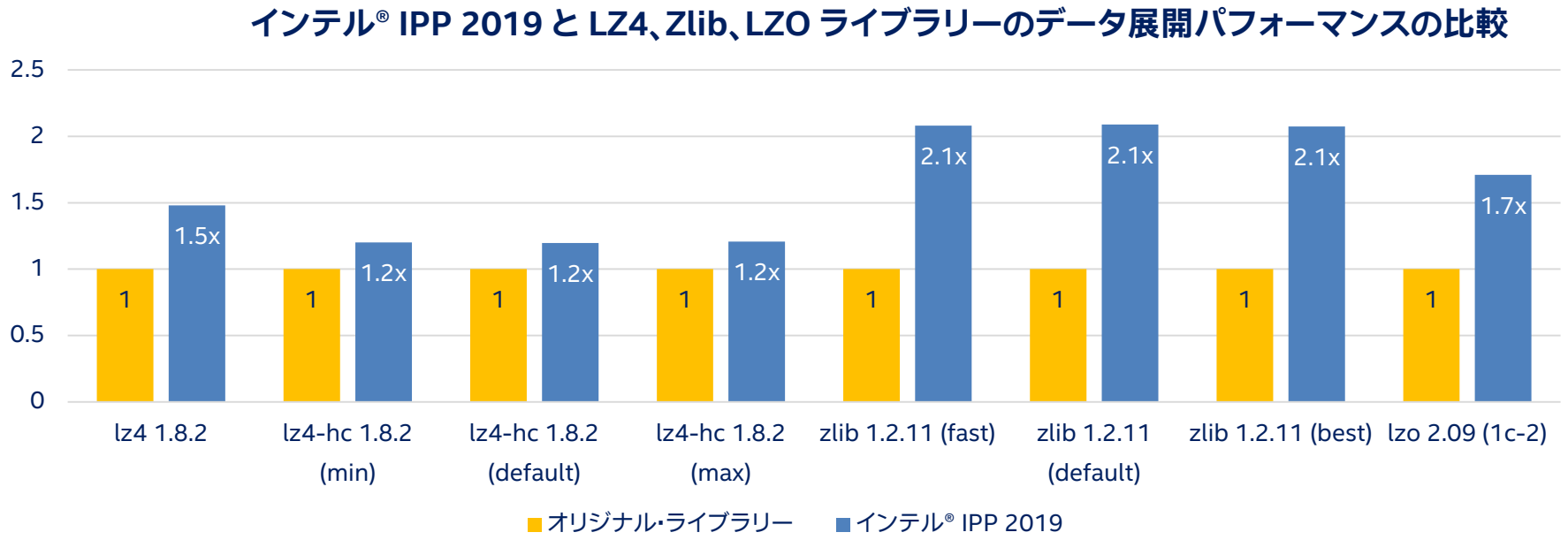
インテル® IPP 2019 と LZ4、Zlib、LZO ライブラリーのデータ圧縮パフォーマンスの比較



パフォーマンス結果は 2018 年 8 月 15 日時点のテスト結果に基づいたものであり、公開されている利用可能なすべてのセキュリティー・アップデートが適用されていない可能性があります。詳細については、構成の開示を参照してください。絶対的なセキュリティーを提供できる製品はありません。性能に関するテストに使用されるソフトウェアとワークロードは、性能がインテル® マイクロプロセッサー用に最適化されていることがあります。SYSmark* や MobileMark* などの性能テストは、特定のコンピューター・システム、コンポーネント、ソフトウェア、操作、機能に基づいて行ったものです。結果はこれらの要因によって異なります。製品の購入を検討される場合は、他の製品と組み合わせた場合の本製品の性能など、ほかの情報や性能テストも参考にして、パフォーマンスを総合的に評価することをお勧めします。詳細については、パフォーマンス・ベンチマーク・テストの開示 (英語) を参照してください。
2018 年 8 月 15 日時点のインテルによるテスト。システム構成: ハードウェア: インテル® Core™ i5-7600 プロセッサー @ 3.50GHz、4 コア、ハイパースレッディング無効。キャッシュ: L1=32KB、L2=256KB、L3=6MB。メモリー: 64GB。Linux* OS: Red Hat* Enterprise Linux* Server 7.2。インテル® コンパイラーでは、インテル® マイクロプロセッサーに限定されない最適化に関して、他社製マイクロプロセッサー用に同等の最適化を行えないことがあります。これには、インテル® ストリーミング SIMD 拡張命令 2、インテル® ストリーミング SIMD 拡張命令 3、インテル® ストリーミング SIMD 拡張命令 3 補足命令などの最適化が該当します。インテルは、他社製マイクロプロセッサーに関して、いかなる最適化の利用、機能、または効果も保証いたしません。本製品のマイクロプロセッサー依存の最適化は、インテル® マイクロプロセッサーでの使用を前提としています。インテル® マイクロアーキテクチャーに限定されない最適化のなかにも、インテル® マイクロプロセッサー用のものがあります。この注意事項で言及した命令セットの詳細については、該当する製品のユーザー・リファレンス・ガイドを参照してください。注意事項の改訂 #20110804

© 2019 iSUS *その他の社名、製品名などは、一般に各社の表示、商標または登録商標です。

データ展開のパフォーマンスを向上



パフォーマンス結果は 2018 年 8 月 15 日時点のテスト結果に基づいたものであり、公開されている利用可能なすべてのセキュリティー・アップデートが適用されていない可能性があります。詳細については、構成の開示を参照してください。絶対的なセキュリティーを提供できる製品はありません。性能に関するテストに使用されるソフトウェアとワークロードは、性能がインテル® マイクロプロセッサー用に最適化されていることがあります。SYSmark* や MobileMark* などの性能テストは、特定のコンピューター・システム、コンポーネント、ソフトウェア、操作、機能に基づいて行ったものです。結果はこれらの要因によって異なります。製品の購入を検討される場合は、他の製品と組み合わせた場合の本製品の性能など、ほかの情報や性能テストも参考にして、パフォーマンスを総合的に評価することをお勧めします。詳細については、パフォーマンス・ベンチマーク・テストの開示 (英語) を参照してください。
2018 年 8 月 15 日時点のインテルによるテスト。システム構成: ハードウェア: インテル® Core™ i5-7600 プロセッサー @ 3.50GHz、4 コア、ハイパースレッディング無効。キャッシュ: L1=32KB、L2=256KB、L3=6MB。メモリー: 64GB。Linux* OS: Red Hat* Enterprise Linux* Server 7.2。インテル® コンパイラーでは、インテル® マイクロプロセッサーに限定されない最適化に関して、他社製マイクロプロセッサー用に同等の最適化を行えないことがあります。これには、インテル® ストリーミング SIMD 拡張命令 2、インテル® ストリーミング SIMD 拡張命令 3、インテル® ストリーミング SIMD 拡張命令 3 補足命令などの最適化が該当します。インテルは、他社製マイクロプロセッサーに関して、いかなる最適化の利用、機能、または効果も保証いたしません。本製品のマイクロプロセッサー依存の最適化は、インテル® マイクロプロセッサーでの使用を前提としています。インテル® マイクロアーキテクチャーに限定されない最適化のなかにも、インテル® マイクロプロセッサー用のものがあります。この注意事項で言及した命令セットの詳細については、該当する製品のユーザー・リファレンス・ガイドを参照してください。注意事項の改訂 #20110804

© 2019 iSUS *その他の社名、製品名などは、一般に各社の表示、商標または登録商標です。

インテル® TBB による高度なスレッド化

スレッド化を利用してマルチコア・パフォーマンスとヘテロジニアス・コンピューティングを活用

- CPU、GPU、FPGA にわたって計算負荷の高い作業を並列処理 — C++ で高レベルのシンプルなソリューションを提供
- 並列プログラミング向けの最も機能が豊富で包括的なソリューション
- 優れた移植性、構成の容易性、コスト、アプローチを備えた将来にわたるスケーラビリティー

バージョン 2019 の新機能

- フローグラフの新機能により並列性とヘテロジニアスへの対応が向上 (タスク・アナライザーと OpenCL* デバイス選択を向上)
- 新しいテンプレートにより C++11 多次元配列を最適化
- C++17 Parallel STL、OpenCL*、Python* conda 言語をサポート
- Windows*、Linux*、Android*、macOS* のサポートを拡張

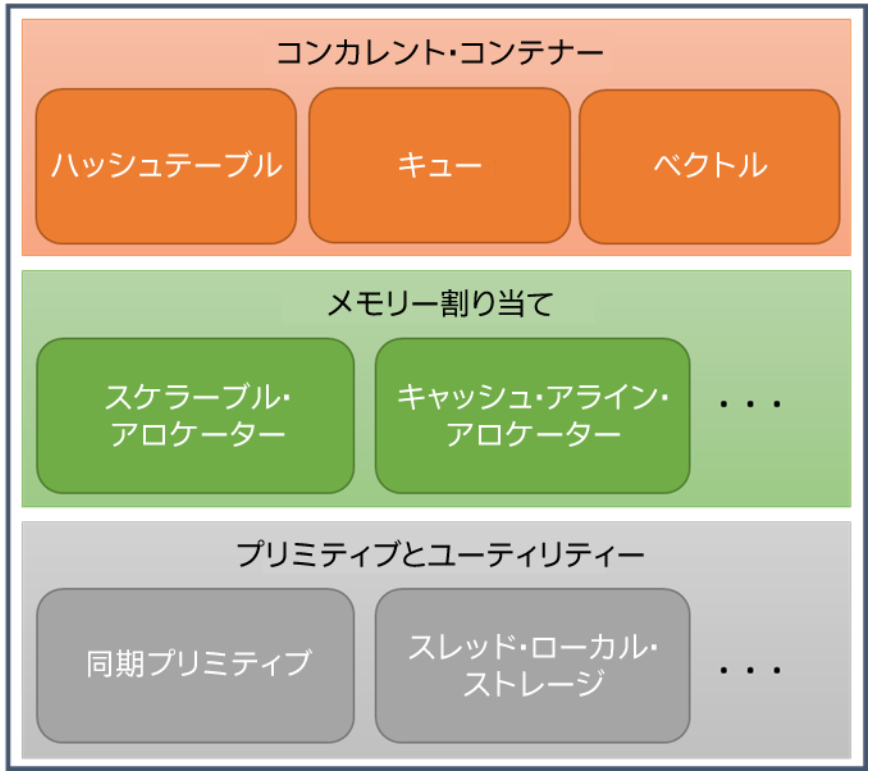
詳細: isus.jp/intel-tbb/

© 2019 iSUS *その他の社名、製品名などは、一般に各社の表示、商標または登録商標です。

インテル® TBB のコンポーネント

並列実行インターフェイス

実行モデルに依存しないインターフェイス

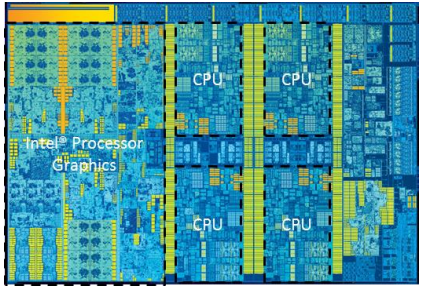


© 2019 iSUS *その他の社名、製品名などは、一般に各社の表示、商標または登録商標です。

ヘテロジニアス・サポート

インテル® TBB

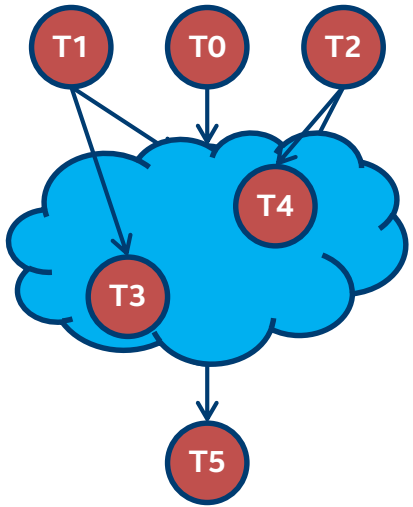
ヘテロジニアス調整レイヤーとしてインテル® TBB のフローグラフは最適化の可能性を保持し、既存のモデルで構成



CPU、統合 GPU など

＋

インテル® TBB
OpenVX*
OpenCL*
COI/SCIF
…

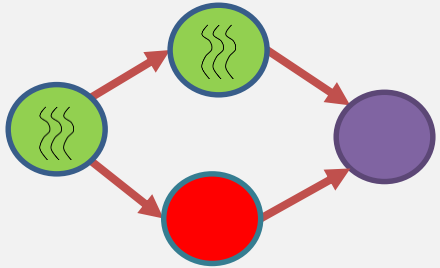


ライブラリー実装の**構成レイヤー**としてのインテル® TBB

- 1 つのスレッドエンジンで CPU 側の全処理に対応

調整レイヤーとしてのインテル® TBB フローグラフ

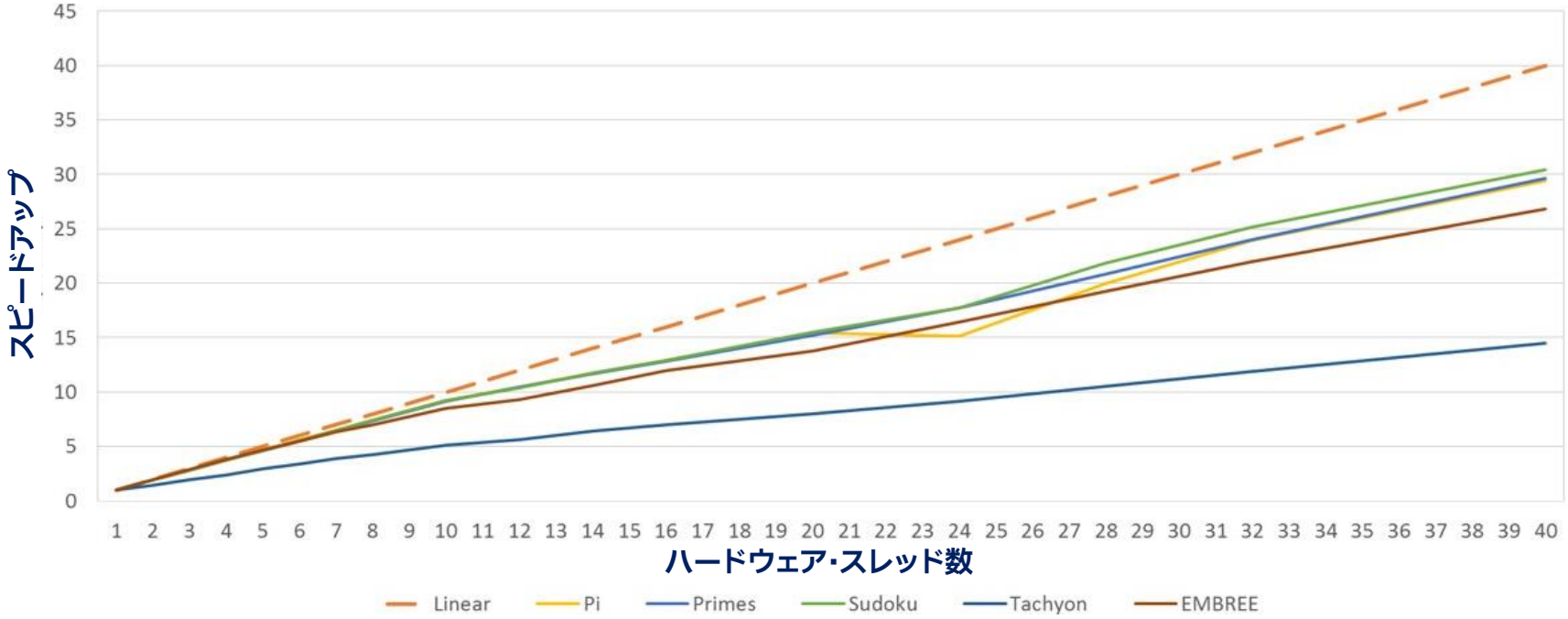
- ヘテロジニアス・ハードウェアとソフトウェアを繋ぎ合わせる
- ブロック間の並列性を示し、統合を容易にする



© 2019 iSUS *その他の社名、製品名などは、一般に各社の表示、商標または登録商標です。

インテル® TBB 2019 の優れたパフォーマンス・スケーラビリティー

インテル® Xeon® プロセッサー



パフォーマンス結果は 2018 年 7 月 31 日時点のテスト結果に基づいたものであり、公開されている利用可能なすべてのセキュリティー・アップデートが適用されていない可能性があります。詳細については、構成の開示を参照してください。絶対的なセキュリティーを提供できる製品はありません。性能やベンチマーク結果について、さらに詳しい情報をお知りになりたい場合は、http://www.intel.com/benchmarks/ (英語) を参照してください。性能に関するテストに使用されるソフトウェアとワークロードは、性能がインテル® マイクロプロセッサー用に最適化されていることがあります。SYSmark* や MobileMark* などの性能テストは、特定のコンピューター・システム、コンポーネント、ソフトウェア、操作、機能に基づいて行ったものです。結果はこれらの要因によって異なります。製品の購入を検討される場合は、他の製品と組み合わせた場合の本製品の性能など、ほかの情報や性能テストも参考にして、パフォーマンスを総合的に評価することをお勧めします。詳細については、www.intel.com/benchmarks (英語) を参照してください。2018 年 7 月 31 日時点のインテルによるテスト。システム構成: ソフトウェア: インテル® 64 対応インテル® C++ コンパイラー 18.0、インテル® TBB 2019。ハードウェア: 2x インテル® Xeon® Gold 6152 プロセッサー @ 2.10GHz、192GB RAM。Linux* OS: CentOS* 7.4 1708 (Core)、カーネル 3.10.0-693.e17.x86_64。注: sudoku、primes、tachyon はインテル® TBB に含まれているコードを使用。

インテル® コンパイラーでは、インテル® マイクロプロセッサーに限定されない最適化に関して、他社製マイクロプロセッサー用に同等の最適化を行えないことがあります。これには、インテル® ストリーミング SIMD 拡張命令 2、インテル® ストリーミング SIMD 拡張命令 3、インテル® ストリーミング SIMD 拡張命令 3 補足命令などの最適化が該当します。インテルは、他社製マイクロプロセッサーに関して、いかなる最適化の利用、機能、または効果も保証いたしません。本製品のマイクロプロセッサー依存の最適化は、インテル® マイクロプロセッサーでの使用を前提としています。インテル® マイクロアーキテクチャーに限定されない最適化のなかにも、インテル® マイクロプロセッサー用のものがあります。この注意事項で言及した命令セットの詳細については、該当する製品のユーザー・リファレンス・ガイドを参照してください。注意事項の改訂 #20110804

© 2019 iSUS *その他の社名、製品名などは、一般に各社の表示、商標または登録商標です。

インテル® DAAL による解析とマシンラーニングのスピードアップ

- インテル® アーキテクチャー・ベースのデバイスで動作しているデータセンターからエッジまで古典的なマシンラーニングと解析パフォーマンス向けに高度にチューニングされた関数
- 最高のスループット・パフォーマンスを引き出せるように、データの取り込みと結果の計算を同時に処理
- 広範なアプリケーションのニーズを満たす、バッチ、ストリーミング、分散型使用モデルをサポート
- Python*、C++、Java* API および Spark* や Hadoop* などの一般的なデータソースへのコネクターを含む

バージョン 2019 の新機能

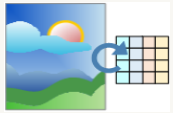
新しいアルゴリズム

- **ロジスティック回帰**、最も広く利用されている分類アルゴリズム
- **拡張勾配ブースティング機能**、不正確な分割計算とユーザー定義のコールバックのキャンセルなどの柔軟性を拡張
- **ユーザー定義のデータ変更プロシージャー**、広範な特徴抽出および変換手法をサポート

詳細: isus.jp/intel-daal/

前処理 → 変換 → 解析 → モデリング → 検証 → 意思決定

展開、
フィルタリング、正規化

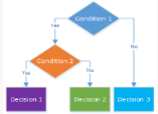


集計、
次元縮小

サマリー統計
クラスタリングなど

マシンラーニング (訓練)、
パラメーター推定、
シミュレーション

仮説検証、
モデルエラー



予測、
決定木など

© 2019 iSUS *その他の社名、製品名などは、一般に各社の表示、商標または登録商標です。

アルゴリズム、データ変換、解析

インテル® DAAL

- データセットの基礎統計
  - 低次モーメント
  - 四分位数
  - 順序統計量
- 相関と依存関係
  - コサイン距離
  - 相関距離
  - 分散共分散行列
- 行列の因数分解
  - SVD
  - QR
  - コレスキー
- 次元縮小
  - PCA
  - 相関ルールマイニング(アプリオリ)
  - 最適化ソルバー(SGD、AdaGrad、lBFGS)
- 外れ値検出
  - 単変量
  - 多変量
  - 数学関数(exp、log、…)

バッチ処理をサポートするアルゴリズム

バッチ、オンライン、分散処理をサポートするアルゴリズム

© 2019 iSUS *その他の社名、製品名などは、一般に各社の表示、商標または登録商標です。

インテル® DAAL のパフォーマンス・スケーリング

インテル® DAAL

CPU コア内

- SIMD ベクトル化: 最新の命令セット、インテル® AVX2、インテル® AVX-512 向けに最適化
- シーケンシャルなインテル® MKL に内部的に依存

マルチコアまたはメニーコアにスケーリング

- インテル® TBB のスレッド化

クラスターにスケーリング

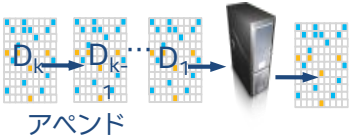
- ユーザー・アプリケーション (MPI、MapReduce など) で分散処理を実行
- インテル® DAAL で提供
  - 部分結果および中間結果向けのデータ構造
  - 部分結果または中間結果をグローバル結果に統合する関数

© 2019 iSUS *その他の社名、製品名などは、一般に各社の表示、商標または登録商標です。

処理モード

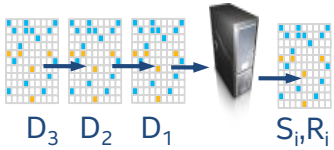
インテル® DAAL

バッチ処理



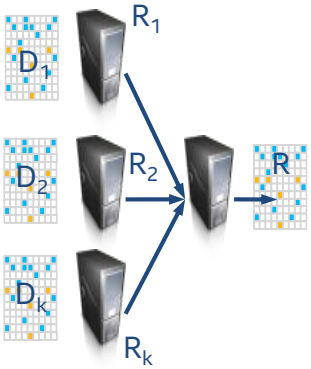
$R = F(D_1,\cdots,D_k)$

オンライン処理



$S_{i+1} = T(S_i,D_i)$
$R_{i+1} = F(S_{i+1})$

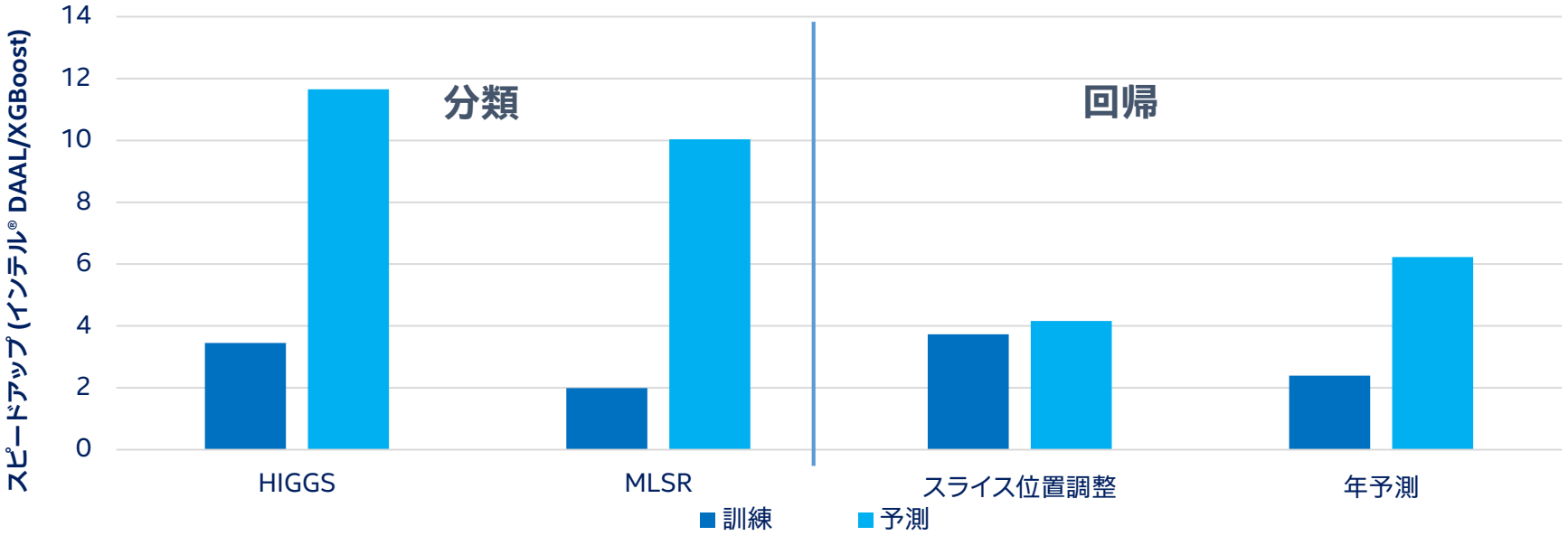
分散処理



$R = F(R_1,\cdots,R_k)$

© 2019 iSUS *その他の社名、製品名などは、一般に各社の表示、商標または登録商標です。

インテル® DAAL 2019 と XGBoost のパフォーマンスの比較

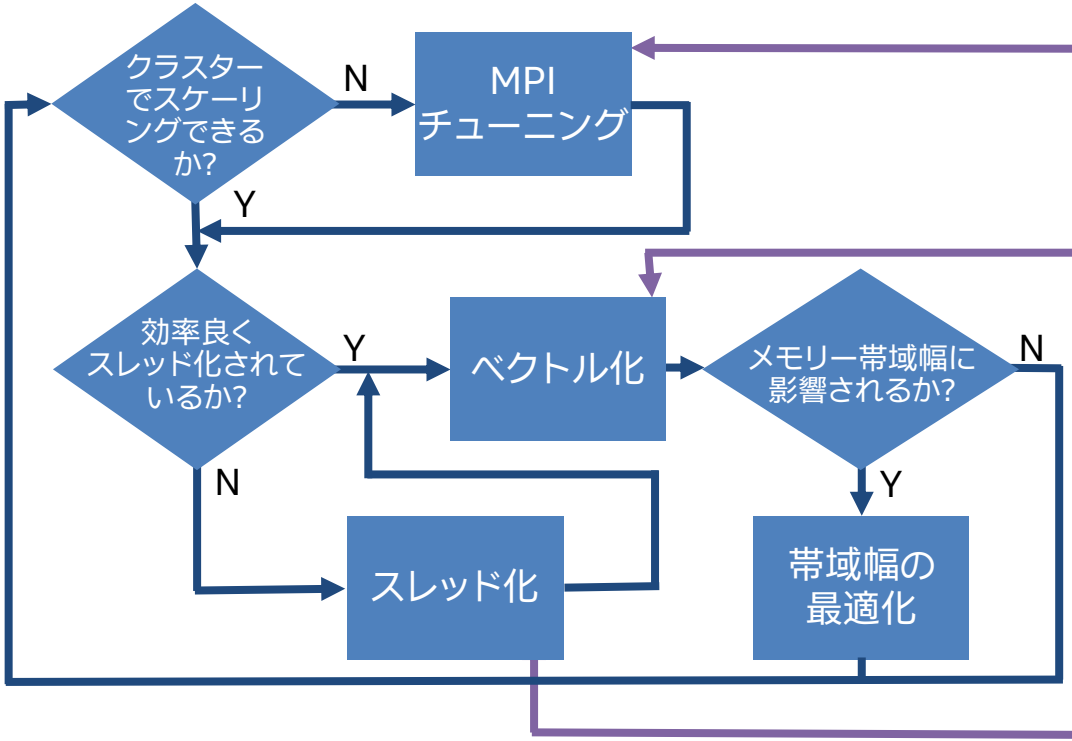


XGBoost オープンソース・プロジェクト
パフォーマンス結果は **2018 年 7 月 9 日**時点のテスト結果に基づいたものであり、公開されている利用可能なすべてのセキュリティー・アップデートが適用されていない可能性があります。詳細については、構成の開示を参照してください。絶対的なセキュリティーを提供できる製品はありません。性能に関するテストに使用されるソフトウェアとワークロードは、性能がインテル® マイクロプロセッサー用に最適化されていることがあります。SYSmark* や MobileMark* などの性能テストは、特定のコンピューター・システム、コンポーネント、ソフトウェア、操作、機能に基づいて行ったものです。結果はこれらの要因によって異なります。製品の購入を検討される場合は、他の製品と組み合わせた場合の本製品の性能など、ほかの情報や性能テストも参考にして、パフォーマンスを総合的に評価することをお勧めします。詳細については、パフォーマンス・ベンチマーク・テストの開示 (英語) を参照してください。
2018 年 7 月 9 日時点のインテルによるテスト。システム構成: インテル® Xeon® Platinum 8180 プロセッサー、2x28 コア、2.50GHz、192GB、12x16GB DDR4-2666。インテル® DAAL 2019。Red Hat* Enterprise Linux* 7.2。インテル® コンパイラーでは、インテル® マイクロプロセッサーに限定されない最適化に関して、他社製マイクロプロセッサー用に同等の最適化を行えないことがあります。これには、インテル® ストリーミング SIMD 拡張命令 2、インテル® ストリーミング SIMD 拡張命令 3、インテル® ストリーミング SIMD 拡張命令 3 補足命令などの最適化が該当します。インテルは、他社製マイクロプロセッサーに関して、いかなる最適化の利用、機能、または効果も保証いたしません。本製品のマイクロプロセッサー依存の最適化は、インテル® マイクロプロセッサーでの使用を前提としています。インテル® マイクロアーキテクチャーに限定されない最適化のなかにも、インテル® マイクロプロセッサー用のものがあります。この注意事項で言及した命令セットの詳細については、該当する製品のユーザー・リファレンス・ガイドを参照してください。注意事項の改訂 #20110804

© 2019 iSUS *その他の社名、製品名などは、一般に各社の表示、商標または登録商標です。

ハイパフォーマンスな実装を支援するツール

インテル® Parallel Studio XE



インテル® MPI ライブラリーと
インテル® MPI Benchmarks

インテル® コンパイラー

インテル® MKL
インテル® IPP
(メディアおよびデータ・ライブラリー)
インテル® DAAL
OpenMP*
インテル® TBB
(スレッド・ライブラリー)

© 2019 iSUS *その他の社名、製品名などは、一般に各社の表示、商標または登録商標です。

関連情報

- ハイパフォーマンス･ビッグデータ･コンピューティング向け Harp-DAAL
https://jp.xlsoft.com/documents/intel/magazine/Intel_ParallelUniverse_Issue32_JPN.pdf#page=31
- インテル® MKL のベクトル化された圧縮行列関数による代数計算の高速化
https://jp.xlsoft.com/documents/intel/magazine/Intel_ParallelUniverse_Issue31_JPN.pdf#page=29
- インテル® Advisor の Python* APIによりパフォーマンスの詳細を得る
https://jp.xlsoft.com/documents/intel/magazine/Intel_ParallelUniverse_Issue31_JPN.pdf#page=47
- インテル® DAAL による R の線形回帰の高速化
https://jp.xlsoft.com/documents/intel/magazine/Intel_ParallelUniverse_Issue29_JPN.pdf#page=58
- インテル® DAAL を利用して実際のマシンラーニングの問題を解く
https://jp.xlsoft.com/documents/intel/magazine/Intel_ParallelUniverse_Issue28_JPN.pdf#page=26
- インテル® IPP を利用した画像識別の最適化
https://jp.xlsoft.com/documents/intel/magazine/Intel_ParallelUniverse_Issue24_JPN.pdf#page=62

© 2019 iSUS *その他の社名、製品名などは、一般に各社の表示、商標または登録商標です。

iSUS