PCクラスタワークショップ in 柏2015

Crayが提供するプログラミング環境

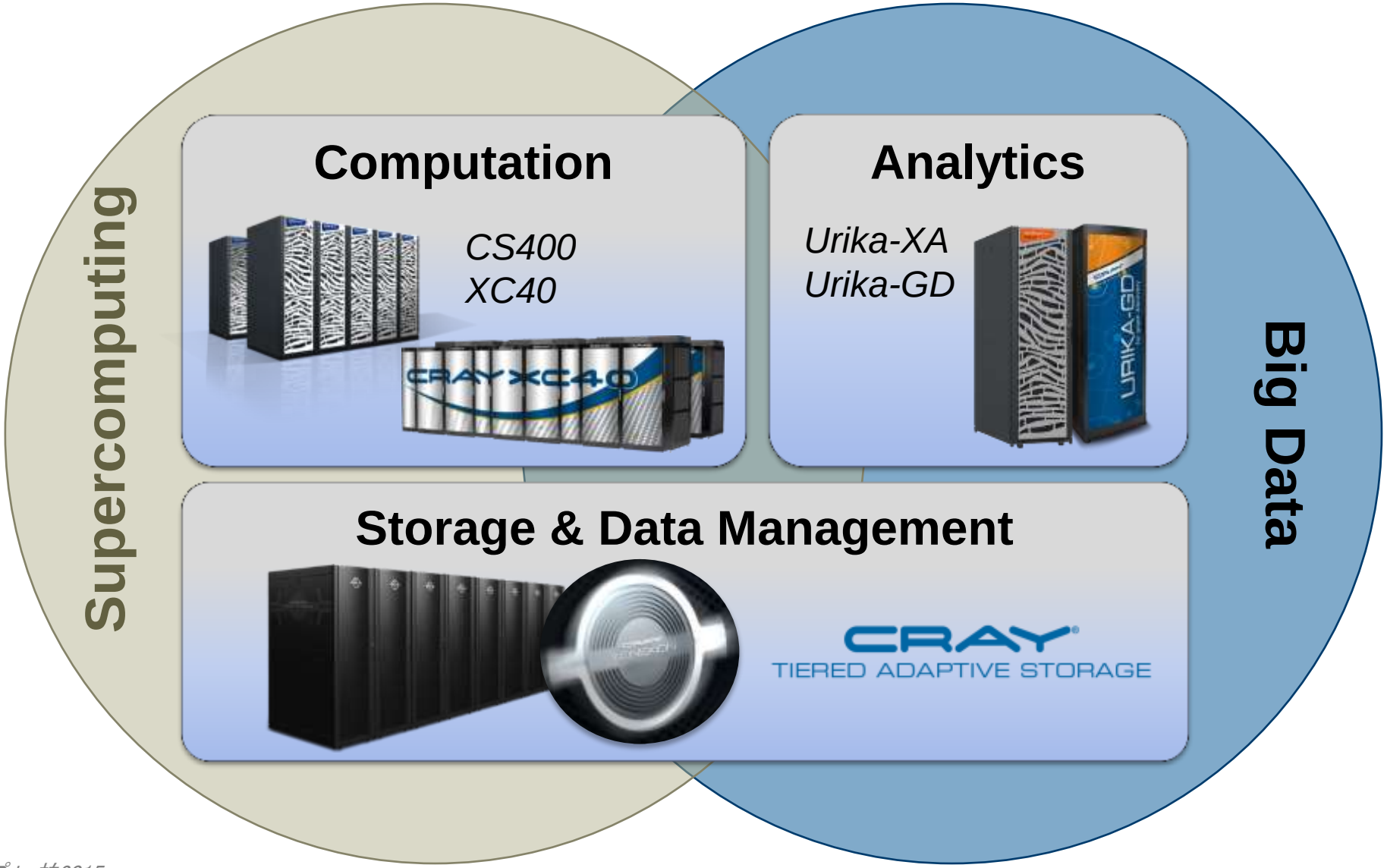
クレイ・ジャパン・インク

June 24th, 2015

アジェンダ

- クレイ・アダプティブ・スーパーコンピューティング
- プログラミング環境
- CCE：Cray Compiling Environment
- CPMAT：Cray Performance Measurement & Analysis Tools
- Cray Reveal：マルチスレッド化支援ツール
- COE：アプリケーションプログラムの移植・最適化・開発支援
- Cray Libsci_ACCライブラリ
- おわりに

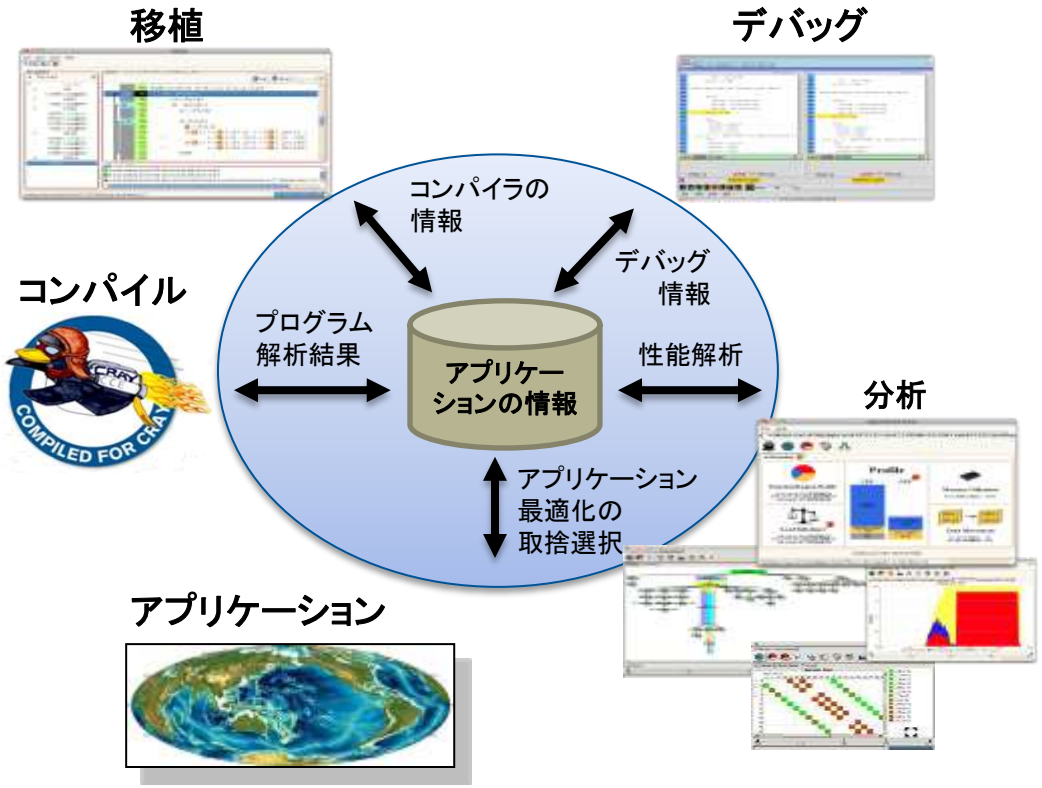
クレイ・アダプティブ・スーパーコンピューティング



プログラミング環境

- **パフォーマンスとプログラマビリティ**
  - アプリケーション性能を最大限に引き出すこと
  - プログラミングの生産性を向上させること
  - 役割は、HW性能（Achievable）と観察される性能とのギャップを埋めること

- **コンパイラ、ライブラリ、ツールが密接に連携することにより、HPCプログラミングの複雑さを克服**
  - HPCシステムの持つ複雑性とスケールの問題を解決すること
  - 機能の拡張と自動化によるシステムの使い易さをの向上
  - インタラクティブなツールでソースコードの変更を実行性能にフィードバックし最適化を支援
  - ユーザの意見を積極的に取り込むことによる機能改善



プログラミング環境（Cray CS400システム）

- **コンパイラ：CCE（Cray Compiling Environment）**
  - OpenMP3.1, OpenACC2.0をサポートするFortran, C/C++コンパイラ
- **性能解析・移植ツール：CPMAT（Cray Performance Measurement & Analysis Tools）**
  - CrayPAT
  - CrayPAT-light
  - Cray Apprentice2
  - Cray Reveal
- **科学技術計算ライブラリ：CSML（Cray Scientific & Math Libraries）**
  - 最適化されたBLAS, LAPACK, ScaLAPACK（MVAPICH2）：LibSci
  - Iterative Refinement Toolkit（IRT）
  - GPU向けBLAS, LAPACKをサポートするCray LibSci_ACC

プログラミング環境（Cray CS400システム）

Programming Languages
- Fortran
- C
- C++
- Python

Programming models
- Distributed Memory
  - MVAPICH 2.0
- Shared Memory
  - OpenMP 3.1
  - OpenACC 2.0

Compilers
- Cray Compiling Environment (CCE)
- GNU
- 3rd party Compilers

Environment
- Modules

Tools

Debuggers
- TotalView
- DDT
- lgdb

Debugging Support Tools
- STAT

Performance Analysis
- CrayPat & Cray Apprentice2

Porting Tools
- Reveal
- CCDB

Optimized Scientific Libraries
- LAPACK
- ScaLAPACK
- BLAS
- Iterative Refinement Toolkit
- FFTW

I/O Libraries
- NetCDF
- HDF5

Cray 純正
他社製ソフトウェア（Licensed ISV）
サードパーティソフトウェア
サードパーティソフトウェア（Crayによる改良済み）

CCE：Cray Compiling Environment

- **科学技術計算のアプリケーション最適化に特化**
  - 自動ベクトル化（SIMD化）
  - 自動並列化による共有メモリ型並列処理
- **標準化規格に準拠**
  - Fortran2008（Coarrayは除く）
  - C++98/2003（C++11/14への対応が進行中）
  - OpenMP 3.1（OpenMP 4.0への対応が進行中）
  - OpenACC 2.0

- **Crayアーキテクチャへ特化した自動最適化機能により、プラットフォームが代わっても再コンパイルするのみで性能を享受できる**
  - システムの複雑さを隠蔽

CCEの特長

- **Crayコンパイラはマルチノード、大規模並列ジョブを前提に開発**
  - 高水準の自動ベクトル化
  - 多重ループのベクトル化に対応： 多重ループの外側で並列性が存在する場合、外側ループでもベクトル化を実施
  - 標準化への対応（OpenMP, OpenACC, …）
  - 2001年からクロスコンパイルをサポート：CPUアーキテクチャ（IvyBridge, Haswellなど）を指定することにより自動で最適化を実施
  - コンパイラによる様々の出力データをツールに読み込ませる事で更なる最適化が可能
  - bit単位での結果の再現性を保証するための機能を提供
- **ユーザコード実行時に不具合が見つかった場合には速やかなバグ対応**

CPMAT：Cray Performance Measurement & Analysis Tools

- **ユーザアプリケーションの性能データを解析、最適化へ導くツール群**
  - CCEが出力する中間表現、最適化情報の活用
  - 膨大な性能データから重要な部分を特定
- **使い易いユーザインターフェース、自動化の向上**
  - GUI
  - プログラム性能の自動計測と自動解析
  - スケーラビリティの強化：多ノードへの対応とレスポンスの向上
  - 性能ツールに慣れていない初心者ユーザ向けのCrayPat-lite
- **複数のプログラミングモデルに対応**
  - MPI, OpenMP, OpenACC, OpenCL, CUDA

CrayPat-lite（性能データへの簡易アクセス）

```
CrayPat/X:  Version 6.1.4.12457 Revision 12457 (xf 12277)  02/26/14 13:58:24
Experiment:                         lite  lite/sample_profile
Number of PEs (MPI ranks):          8164
Numbers of PEs per Node:              16  PEs on each of  510  Nodes
                                       4  PEs on    1  Node
Numbers of Threads per PE:             1
Number of Cores per Socket:            8
Execution start time:  Fri Feb 28 23:06:31 2014
System name and speed:  hera2 2100 MHz

Wall Clock Time:     999.595275 secs
High Memory:             475.52 MBytes
MFLOPS (aggregate):   806112.33 M/sec
I/O Read Rate:            33.57 MBytes/Sec
I/O Write Rate:          215.40 MBytes/Sec

Table 1:  Profile by Function Group and Function (top 7 functions shown)
 Time%  |       Time  |     Imb.  |   Imb.  |      Calls  |Group
        |             |     Time  |  Time%  |             | Function
        |             |           |         |             |  PE=HIDE
 100.0% | 101.961423  |       --  |      --  | 5315211.9  |Total
|-------------------------------------------------------------------------
|  92.5% |  94.267451  |       --  |      --  | 5272245.9  |USER
|-------------------------------------------------------------------------
|  75.8% |  77.248585  | 2.356249  |    3.0%  |     1001.0 |LAMMPS_NS::PairLJCut::compute
|   6.5% |   6.644545  | 0.105246  |    1.6%  |       51.0 |LAMMPS_NS::Neighbor::half_bin_newton
|   4.1% |   4.131842  | 0.634032  |   13.5%  |        1.0 |LAMMPS_NS::Verlet::run
|   3.8% |   3.841349  | 1.241434  |   24.8%  |  5262868.9 |LAMMPS_NS::Pair::ev_tally
|   1.3% |   1.288463  | 0.181268  |   12.5%  |     1000.0 |LAMMPS_NS::FixNVE::final_integrate
|=========================================================================
|   7.0% |   7.110931  |       --  |      --  |    42637.0 |MPI
|-------------------------------------------------------------------------
|   4.8% |   4.851309  | 3.371093  |   41.6%  |    12267.0 |MPI_Send
|   1.5% |   1.536106  | 2.592504  |   63.8%  |    12267.0 |MPI_Wait
|=========================================================================
```

CrayPatとCrayPat-liteは相互補完

- 同一フォーマットの性能解析ファイル
- 両ツールを簡単に切り替えながら使い分ける事が可能
- CrayPat-liteに慣れたユーザはすぐにCrayPatへ移行

CrayPatの使用例：アクセラレータコンピューティング

- **スケーラビリティ**
  - 多数のノードでも短いレスポンス時間（性能結果は1ファイルに集約）
- **アプリケーション全体の性能情報をユーザに提供**
  - 性能データをソースコードにマッピング
  - 性能データをディレクティブ毎にグループ化
  - CPU側、アクセラレータ側の両方の性能解析が可能
- **CPUとGPUの性能情報を一括に管理**
  - 性能情報：アクセラレータの実行時間、CPU実行時間、CPUとGPU間のデータ転送の評価と解析
  - カーネル単位の性能データ
  - アクセラレータのハードウェアカウンタを利用

アーキテクチャの今後

- **ノード内並列処理の増大**
  - ノード内プロセッサ数の増大
  - プロセッサ毎のスレッド数の増大
  - ベクトル（＝SIMD）長の増大
  - メモリ階層はより複雑に
  - スカラ性能の向上は見込めず、むしろ低下の方向へ
- **当面のHPCシステムは同一のアーキテクチャを基に構成される事に（？）**
  - ノード間はメッセージパッシング
  - ノード内はマルチスレッディング（ピュアMPIは好ましくない？）
  - 最も低レベルな部分でベクトル（SIMD）化を活用

ツールによるノード内並列化の作業の流れ

並列化が可能な箇所を探す

- x86システム上でMPIプログラムが正しく動作している
- CCEの自動スレッド化を試す

  （コンパイラがスレッド化可能なループを検知）
- 計算量の多いループを探す

  （CrayPatとCCEの両方を使うと各ループの実行時間が分かる）

ループ内の計算を複数のスレッドに配分

- ループの並列化解析とリビルド

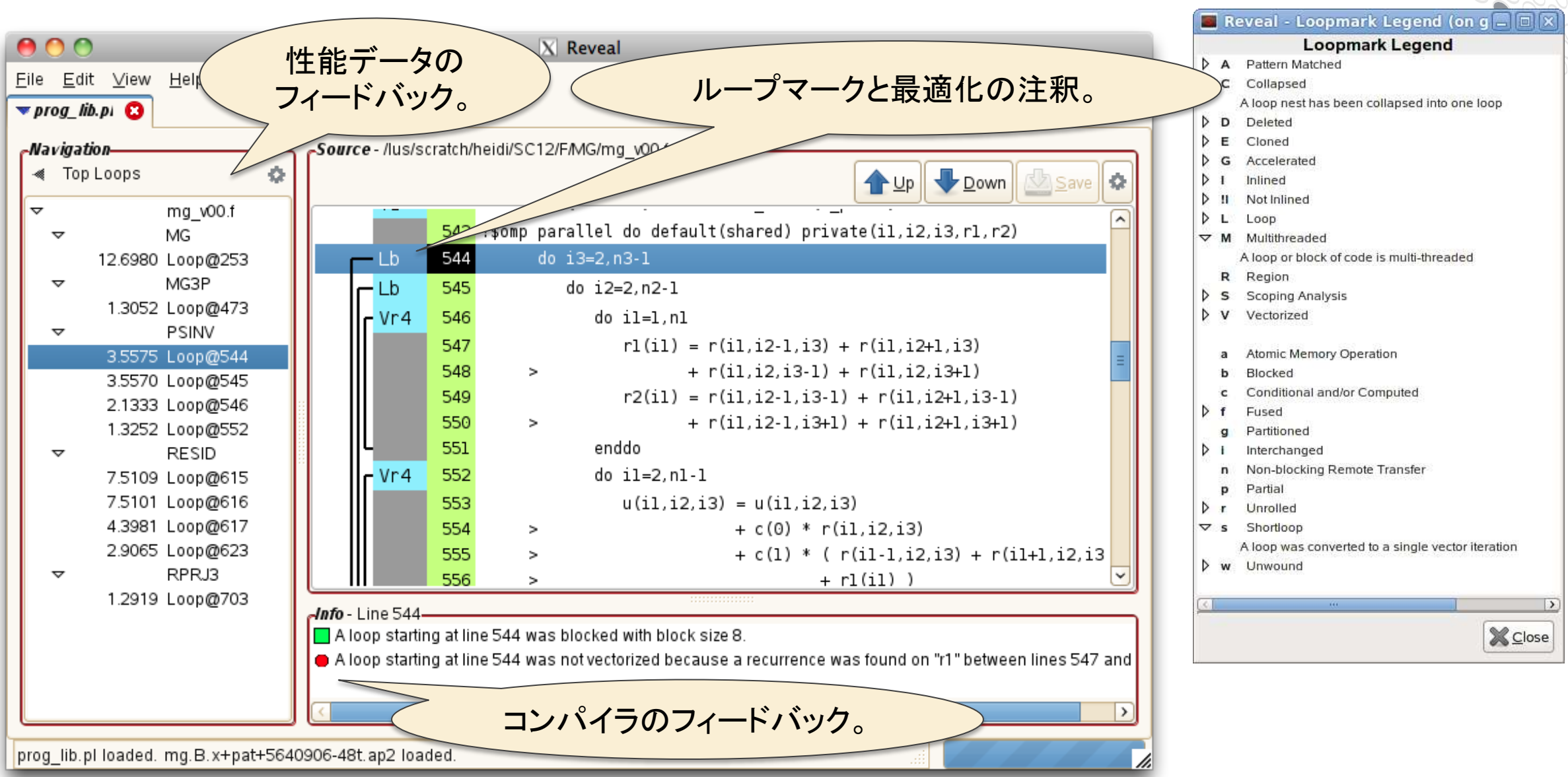
  （Cray RevealとCCEを使うことにより、各ループ情報とそれに対応するソースコードをGUI環境で操作）

並列化ディレクティブの追加、アクセラレータ化

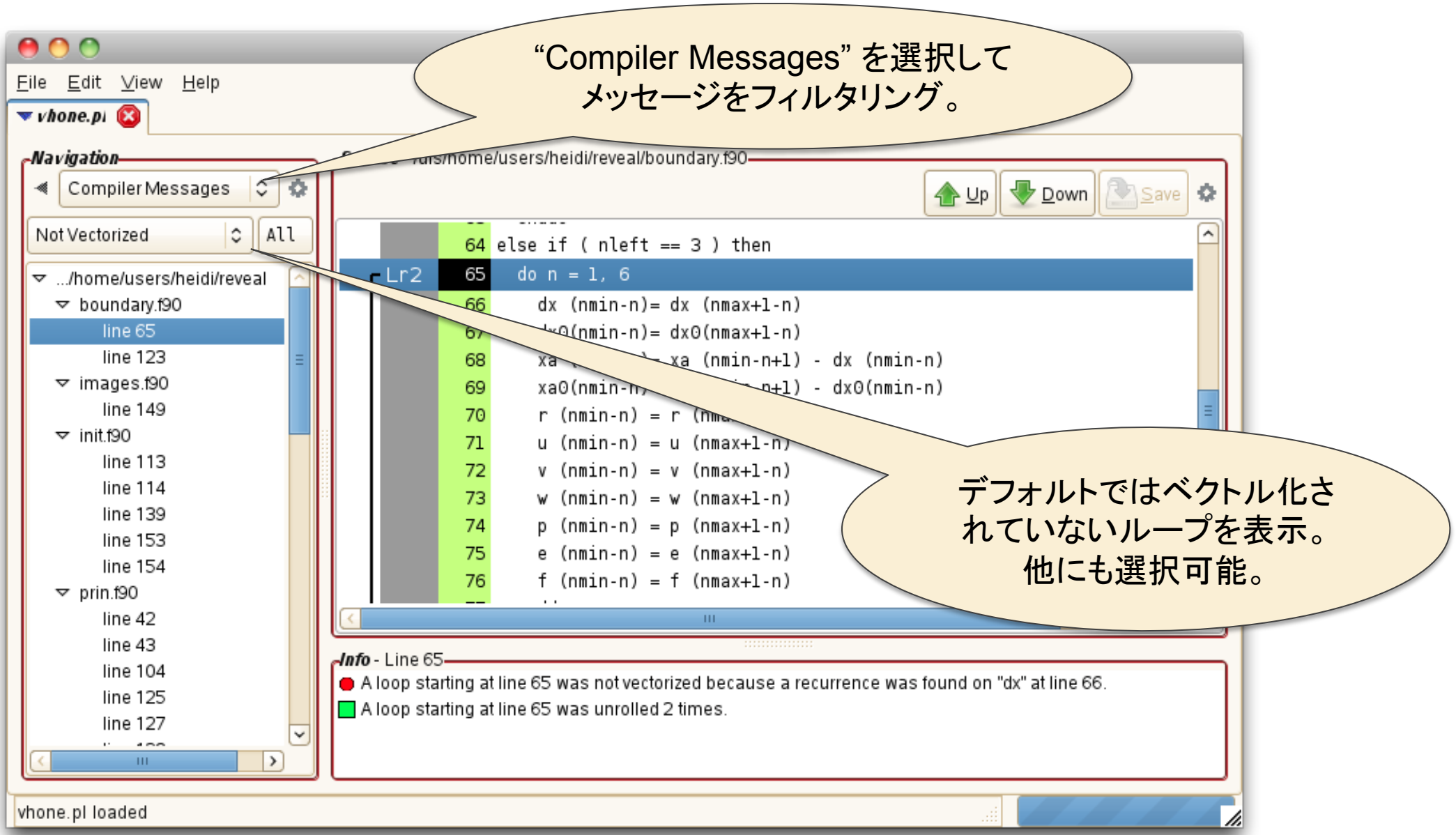
- OpenMPディレクティブを挿入

  （Cray Revealのスコーピング機能）
- x86システム上で動作の確認、性能をチェック
- OpenMPディレクティブをOpenACCディレクティブに書換える

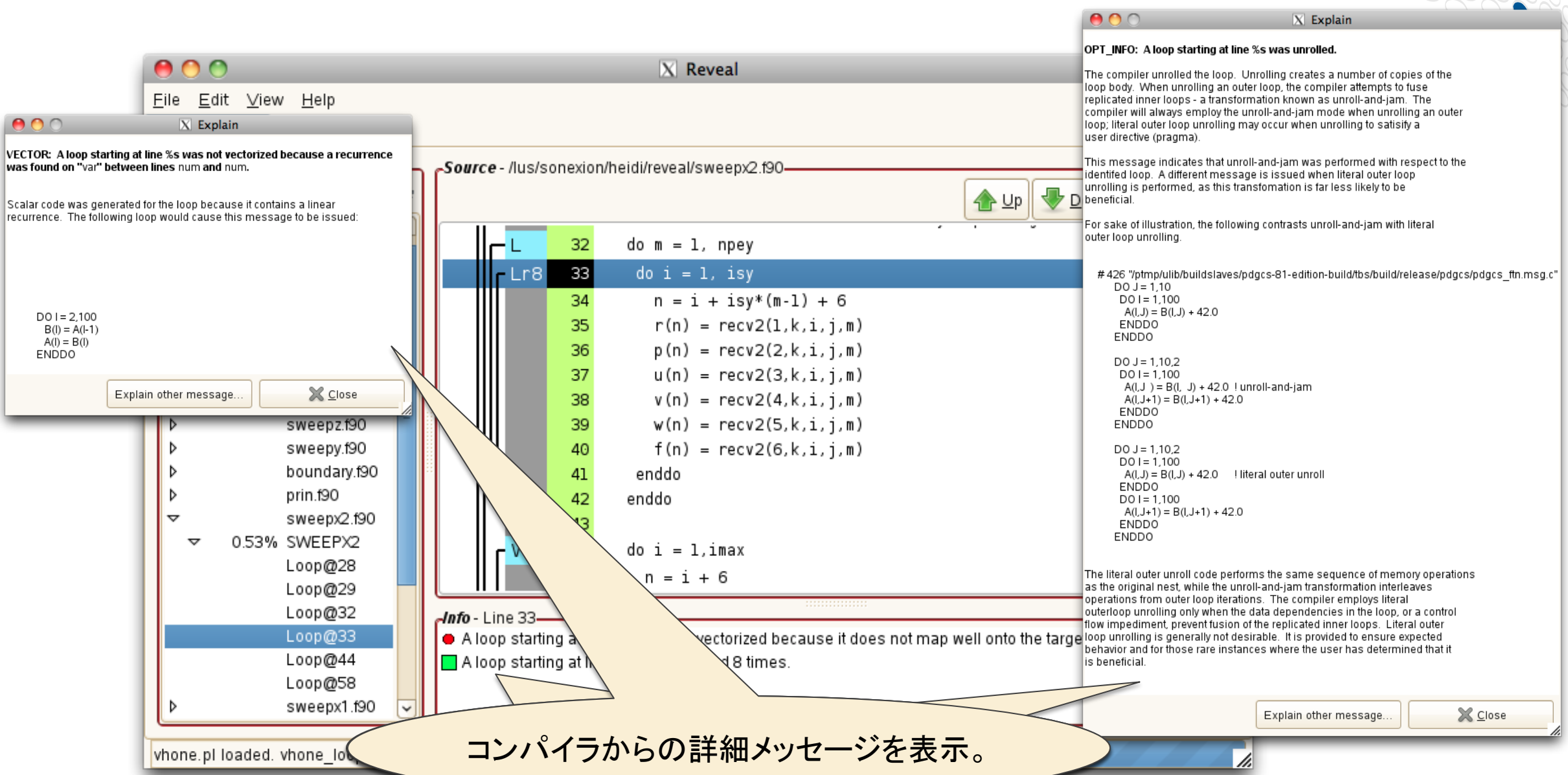
Cray Reveal：性能データと合わせてループマークを可視化



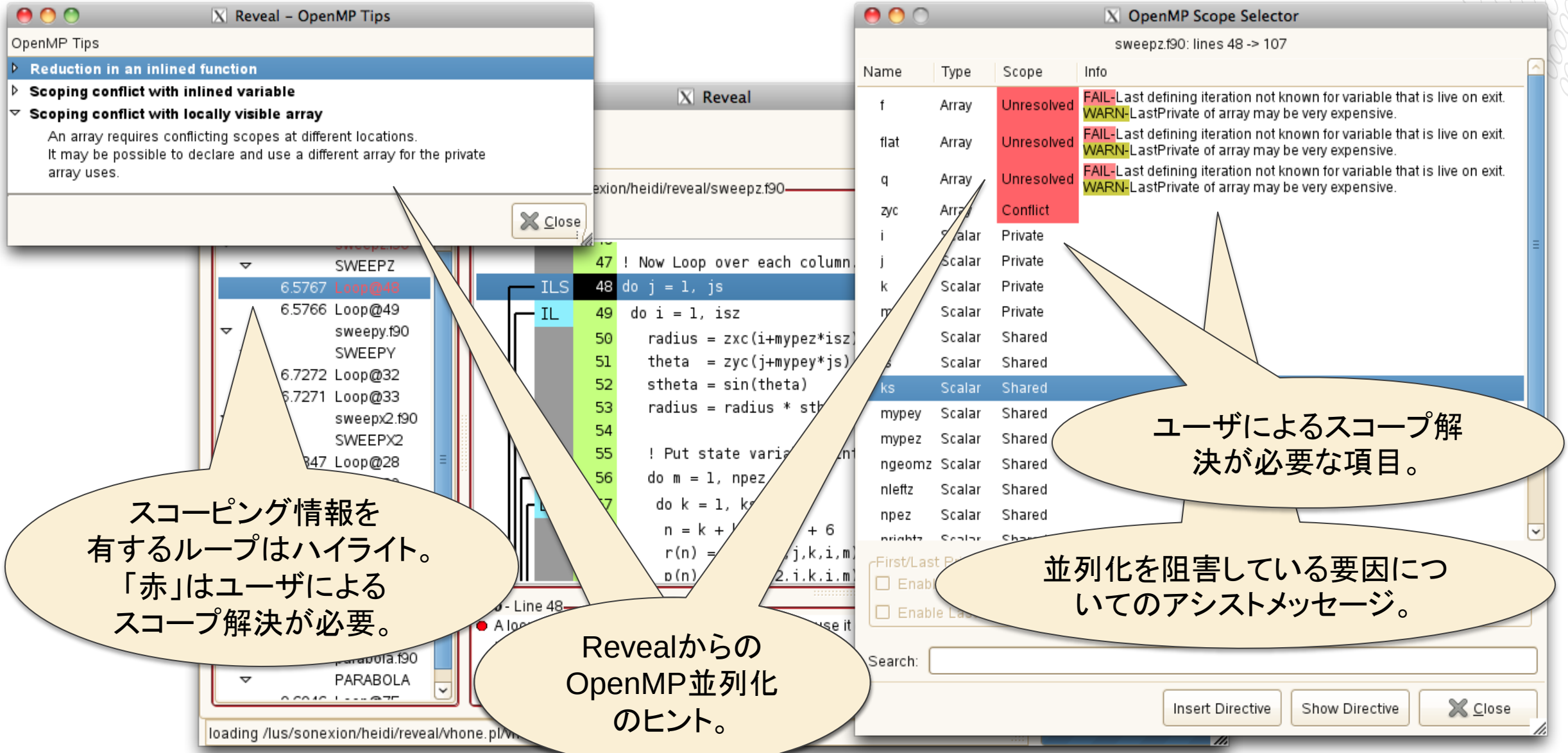
Cray Reveal：コンパイラからのメッセージによるナビゲーション



Cray Reveal：コンパイラからの詳細メッセージを表示



Cray Reveal：スコーピング解析のアシスタンス



Example: Cray porting tool ‘Reveal’

NPB/MG benchmark test

Time in Seconds	1 Thread	2 Threads	4 Threads
Original	2.79		
Using Reveal for high level loops, no assistance (full automation)		2.16	1.13
Using Reveal with Scope all and Threshold		2.12	1.15
Using Reveal with Scope all		2.10	1.08

The way to use Reveal

- The original version was running 64 MPI tasks on 16 nodes.
- The node had 16 cores/node.
- The OpenMP version ran with OMP_NUM_THREADS = 1, 2, 4.

Cray Supercomputing Center of Excellence (COE)

- アプリケーションプログラムの移植を支援
- アプリケーションプログラムの最適化を支援
- アプリケーションプログラムの開発に関して、必要な技術情報を提供し作業を支援
- システムの有効利用ためのコンサルテーションの提供
- ユーザに対する利用相談窓口を設け、システム有効利用をサポート
- カスタムツール、ライブラリ等の作成
- ユーザ向けシステム利用講習会の実施

Cray Libsci_ACC

- 最小限の努力で最大限の性能を得られる手段の提供
- 基本のCPU用ライブラリであるCray Libsciに加えて、CPUとGPUとハイブリッドシステムに対応するCray Libsci_ACCを提供
- 2つのインタフェース

シンプルインターフェース

自動適合型、最小限のソースコード変更（もしくは変更無し）でGPUの基本性能を得る
全てのユーザを対象：Non-GPUユーザ、Non-GPUエキスパート

エキスパート向け

CPU、GPU間のデータ転送を最適に制御することによりGPUの性能を得る
CUDA、OpenACC、およびGPUエキスパートを対象

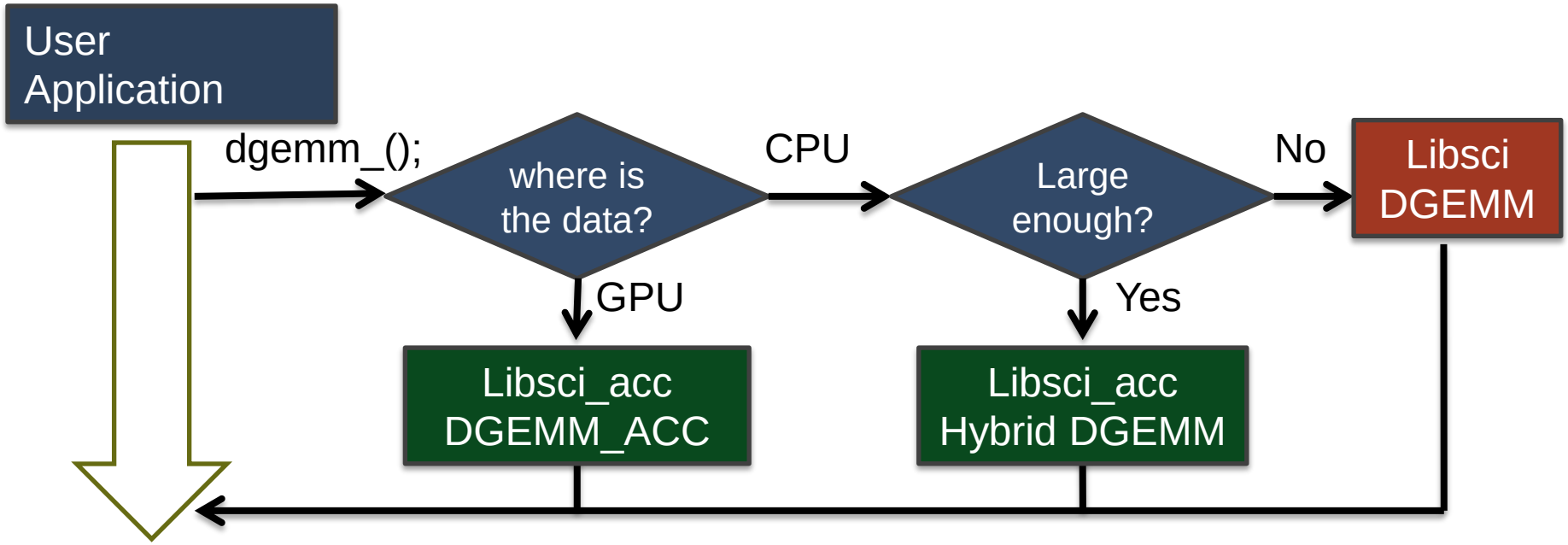
※最大性能を得るのに、必ずしもエキスパートの使用法が必要なわけではありません。

Cray Libsci_ACC （Cont.）

- CPU、GPUのハイブリッドシステムに対する最適ライブラリの自動適合

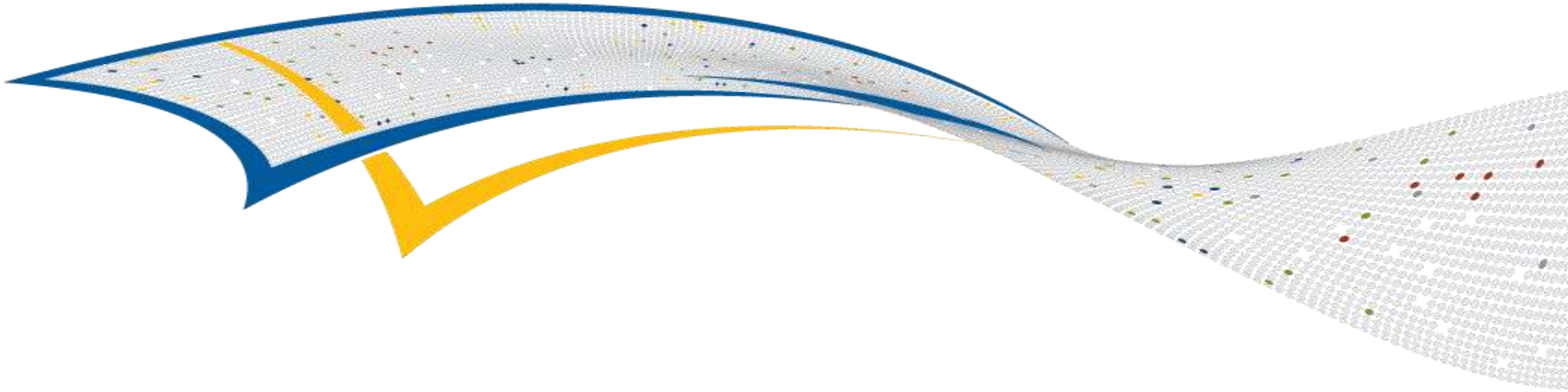
  ランタイム解析により、ベストなライブラリ／カーネルを自動選択
  ホスト側メモリ、GPUメモリ双方のポインタを扱うことが可能

  CPU、GPU間のデータコピーとどちらのデバイスで計算するかを自動で対応
  GPU、CPU、どちらで計算した方が良いかを、ライブラリの自動選択で対応



おわりに

- マルチノード、大規模並列計算処理のためプログラミング環境を提供
- CrayコンパイラはGPUやMICといったアクセラレータを含むヘテロ環境への最適化機能を統合
- ユーザアプリケーション実行時に不具合が見つかった場合には速やかなバグ対応
- ユーザからの性能改善・機能改善の要望にも対応
- Cray Revealによるメニーコア型プロセッサのマルチスレッド化を支援
- 自動並列化だけではスレッド並列できないプログラムの半自動並列化を支援
- CrayPatはマルチGPU、MPI、OpenMPによる並列化に対応した統合型プロファイリングツール
- LibSciに加えてCPUとGPUのハイブリッドシステムに対応するLibSci_ACCの提供
- COEによるアプリケーションプログラム支援と、システム有効利用のサポート



CRAY®

THE SUPERCOMPUTER COMPANY