

次世代メニーコア型スパコンのための システムソフトウェア

堀 敦史

システムソフトウェア技術部会 部会長（理研 AICS）

清水 正明

システムソフトウェア技術部会 副部会長（日立）

2017/12/15



- McKernel
 - 最新情報
- Processes In a Process (PIP)
 - 新しい並列実行環境の紹介

McKernel

背景

- HPCシステムの複雑化への対応

- 並列性の増大

- コア数とノード数



少数のコアをOS専用としても、そのオーバヘッドは無視できる（1/32コアで3パーセント）

- メモリ階層数の増大

- 電力制約



スケーラブルで安定した性能の提供と新ハードウェアへの迅速な対応がカギ

- アプリケーションの複雑化

- 新しいプログラミングモデルの導入

- 例：In-situデータアナリティクス、ワークフロー



Linux API で開発されることが多いため、Linux との ABI 互換性が重要

- 周辺ソフトの複雑化



既存の膨大なツールチェーンがそのまま使えることが重要

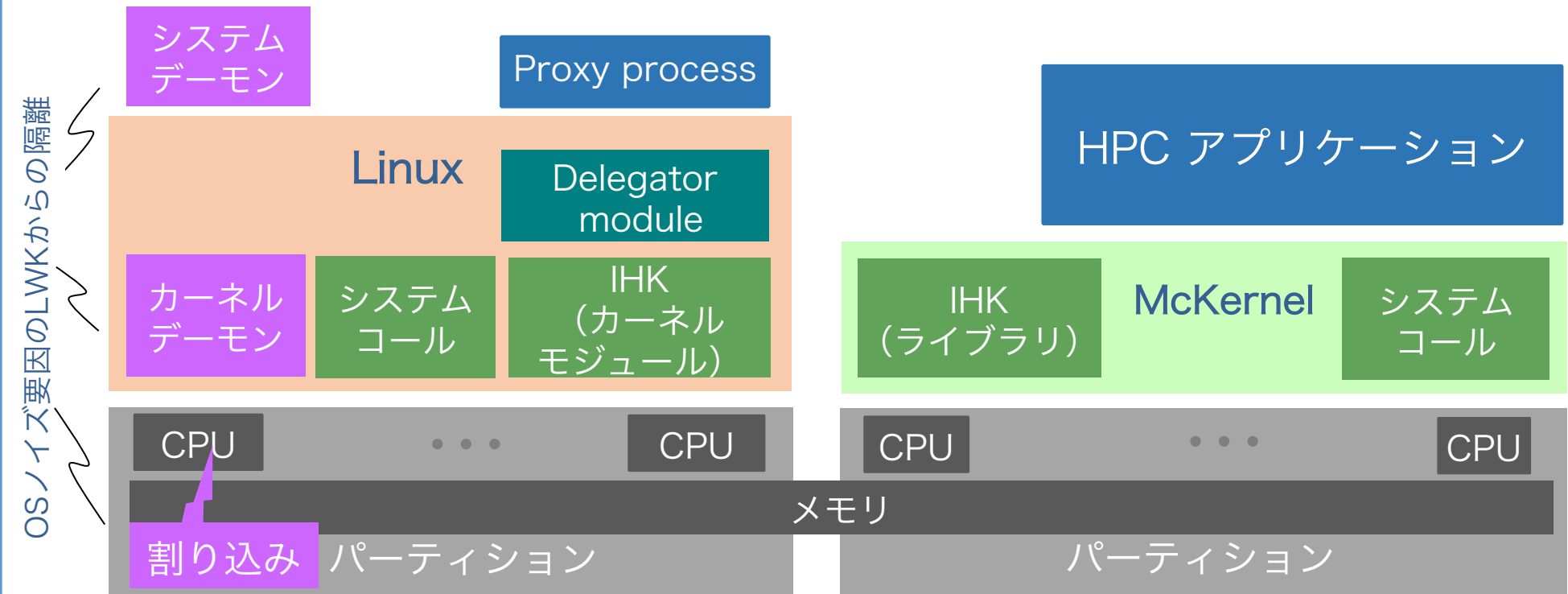
 これらの要件を同時に満たせるか？

Linuxカーネルの問題点

- **Linux** はころころ変わる
 - 頻繁かつ大胆に変わる
 - Linux カーネルを改変し、かつ、catch-up するのは非常に大変
 - 多様なハードウェア
 - 次々に現れるデバイスハードウェア
 - デバイスドライバの作成にはH/Wの詳細な情報が不可欠
- **Linux** カーネルを改造するのは非常に大変
 - プロダクトとしては不可能に近い
- しかしながら、言語環境、実行環境は **Linux** を継承したい
 - 環境も作るのは大変
 - Linux API/ABI との互換性をどのように実現するか？

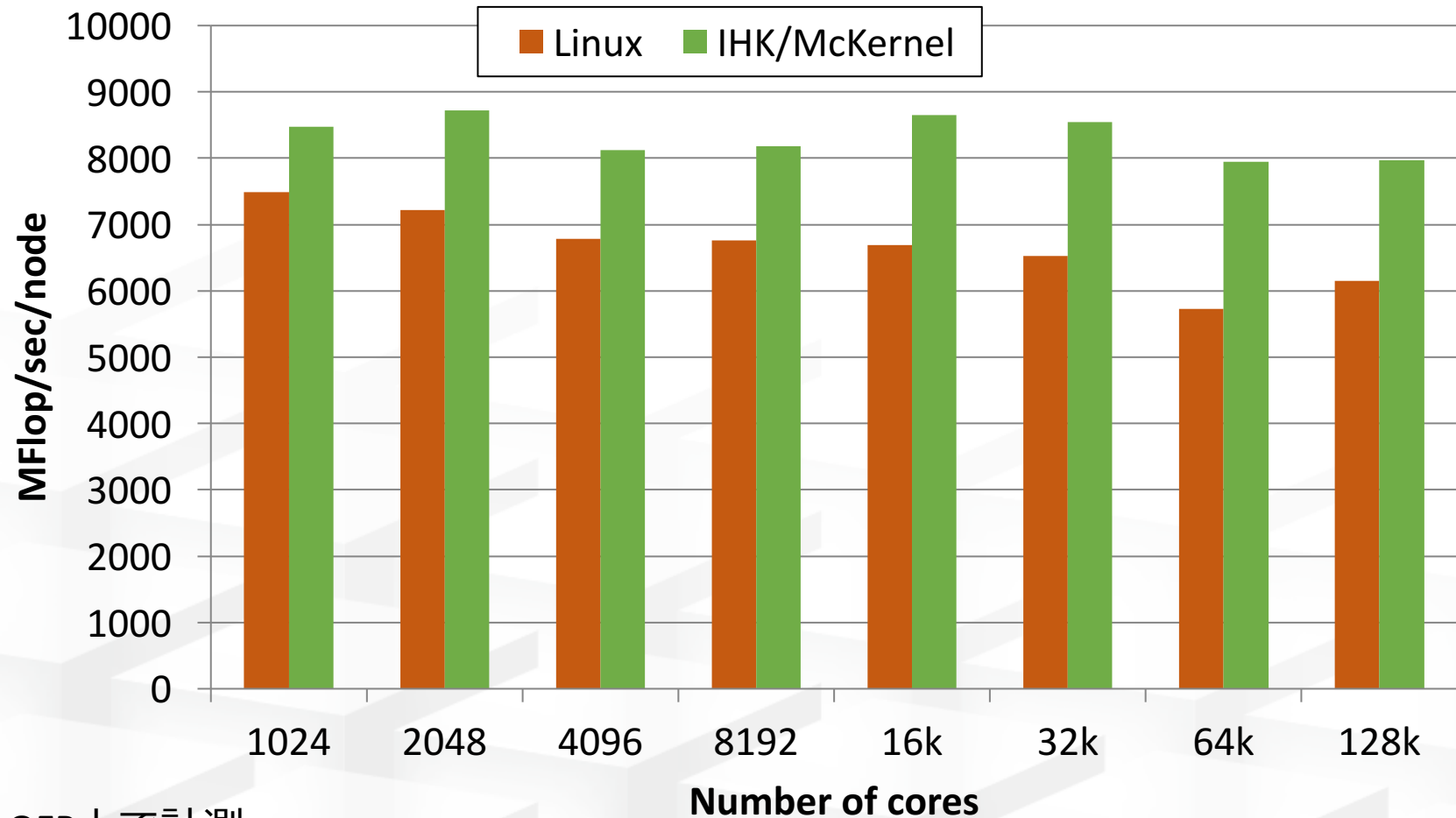
McKernel のアーキテクチャ

- Interface for Heterogeneous Kernels (IHK): 複数の異種OSを持つためのフレームワーク
 - ノード資源のパーティショニング
 - LWKの管理（例：カーネルの起動、停止）
 - LWKとLinuxの間の通信機構（Inter-kernel communication, IKC）の提供
- McKernel: スクラッチから開発されたHPC向けLWK
 - ノイズレス
 - 性能クリティカルなシステムコールのみMcKernelで動作、他はLinuxにオフロード



CCS-QCD (Hiroshima University)

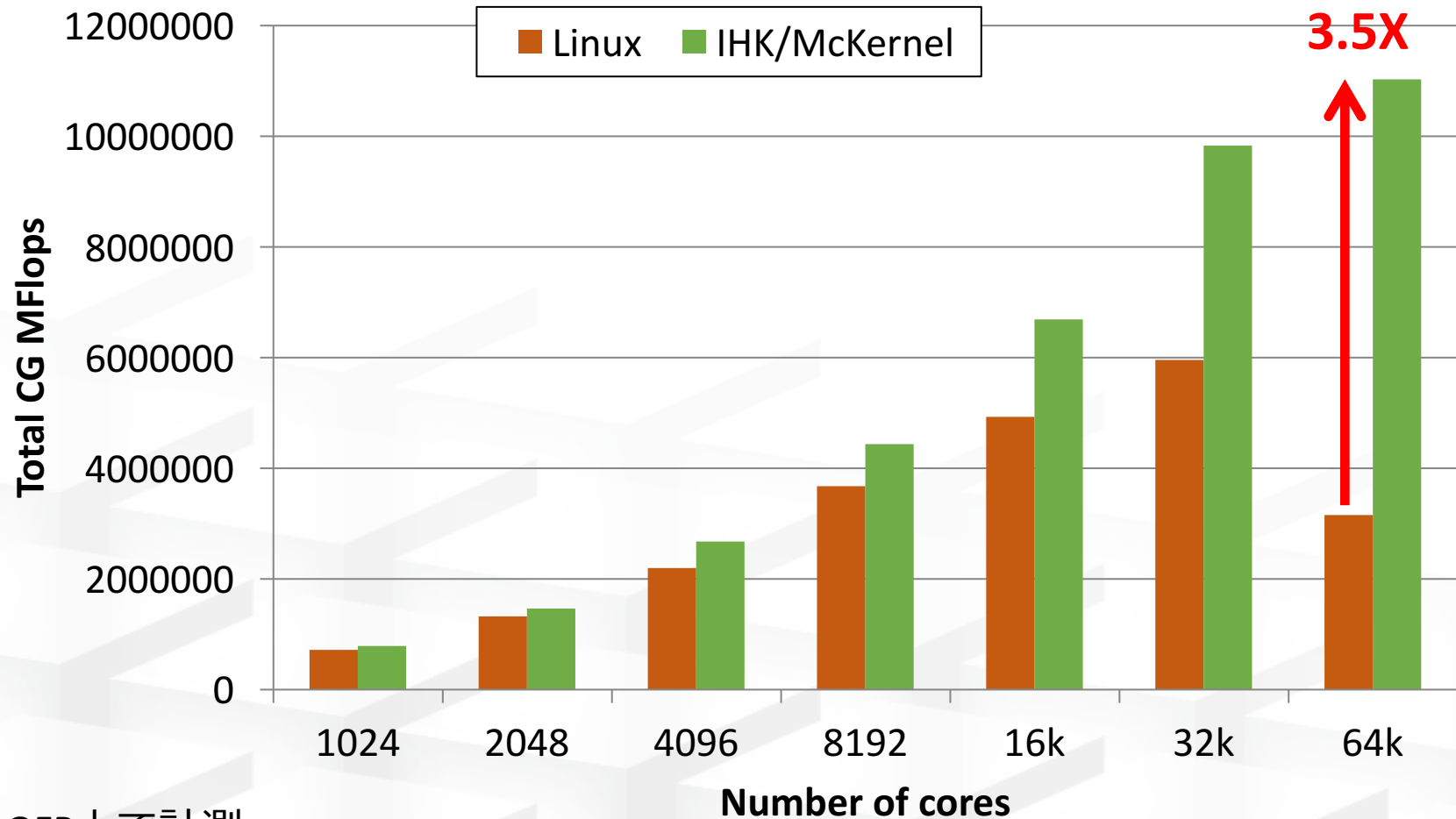
- Lattice quantum chromodynamics code – weak scaling
- Up to 38% improvement



OFP上で計測

miniFE (CORAL benchmark suite)

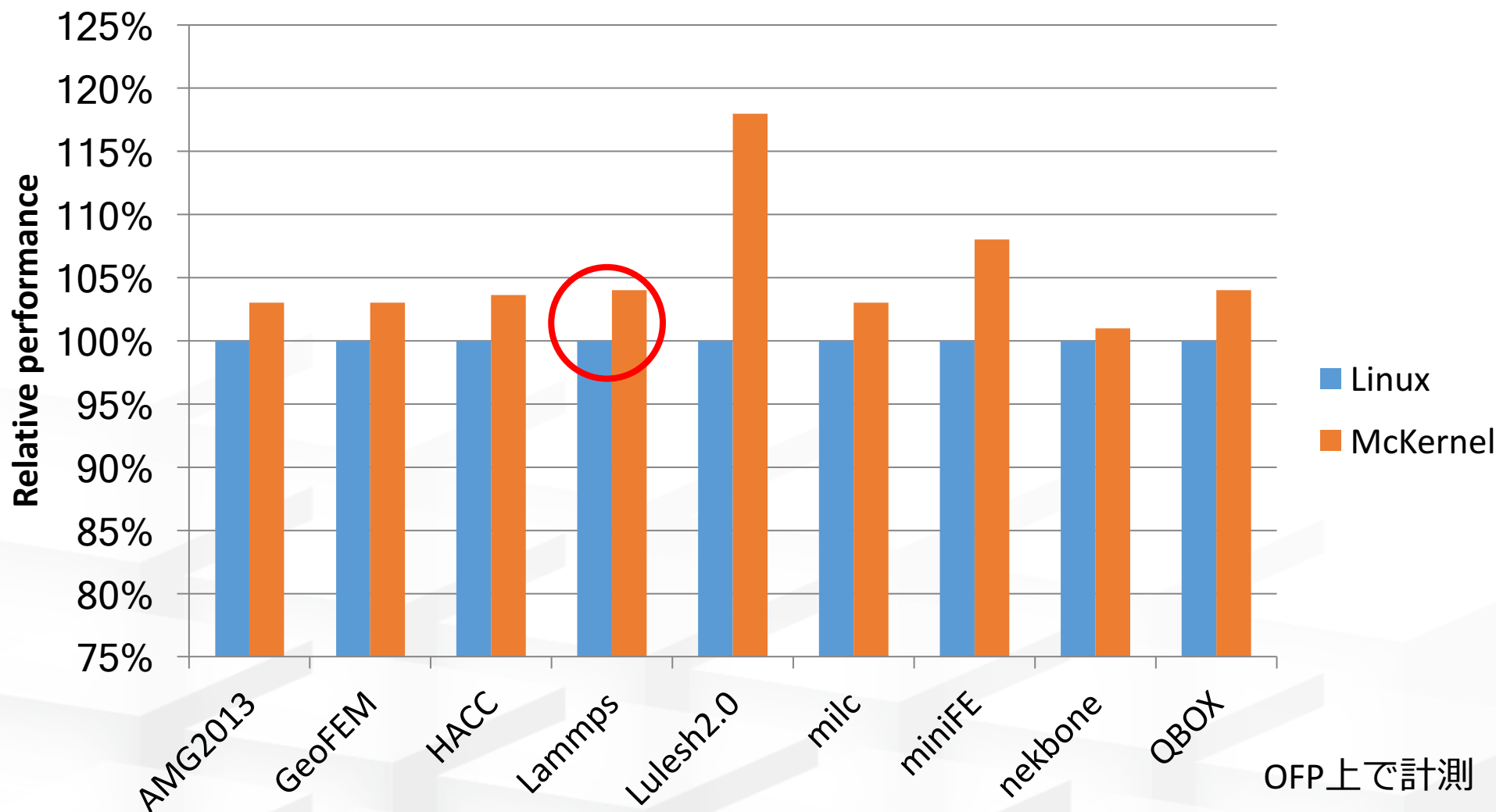
- Conjugate gradient - strong scaling
- Up to 3.5X improvement (Linux falls over..)



OFP上で計測

Single node: McKernel outperforms Linux across the board

→ multi-node Lammmps suffers from network offloading..



- lammmps, HACC, QBOX ~4% better, as opposed to being slower than Linux on 8 nodes
- OmniPath offload overhead??

McKernel - まとめ

- **Xeon Phi** 上でも動作
 - Knights Landing (KNL) 最新メニーコアCPU
- **Linux+McKernel**
 - LinuxコアでMapReduce, McKernelコアで並列アプリという動作でも, Linux だけよりも性能独立性が高い
 - Linuxよりはるかに単純なので, 変更, 機能追加が容易
 - McKernel (+IHK) : 約 8 万行
 - Linux : 2100万行!! (<http://news.mynavi.jp/news/2016/04/11/130/>)
- より高い**Linux**コンパチビリティ
 - Procfs および numa 機能の実装など
- **実戦配備**
 - Oakforest-PACS (東大・筑波大) で採用
 - ポスト京コンピュータでも採用される予定

McKernel に関する論文リスト

- 1) Balazs Gerofi, Masamichi Takagi, Gou Nakamura, Tomoki Shirasawa, Atsushi Hori and Yutaka Ishikawa "On the Scalability, Performance Isolation and Device Driver Transparency of the IHK/McKernel Hybrid Lightweight Kernel", IEEE International Parallel & Distributed Processing Symposium (IPDPS), 2016, Chicago, USA
- 2) Takagi Masamichi, Norio Yamaguchi, Balazs Gerofi, Atsushi Hori and Yutaka Ishikawa: "Adaptive Transport Service Selection for MPI with InfiniBand Network", International Workshop on Exascale MPI (ExaMPI), held in conjunction with ACM/IEEE International Conference for High Performance Computing, Networking, Storage and Analysis (SC), 2015, Austin, TX, USA Balazs Gerofi, Takagi Masamichi and Yutaka Ishikawa: "Toward Operating System Support for Scalable Multithreaded Message Passing", In Proc. EuroMPI, 2015
- 3) Balazs Gerofi, Masamichi Takagi, Yutaka Ishikawa, Rolf Riesen, Evan Powers and Robert W. Wisniewski: "Exploring the Design Space of Combining Linux with Lightweight Kernels for Extreme Scale Computing", In Proc. ROSS, 2015
- 4) Masamichi Takagi, Balazs Gerofi, Norio Yamaguchi, Takahiro Ogura, Toyohisa Kameyama, Atsushi Hori and Yutaka Ishikawa: "Operating System Design for Next Generation Many-core based Supercomputers", IPSJ SIG Technical Reports, 2015-ARC-215(1), pp. 1-8, 2015
- 5) Balazs Gerofi, Akio Shimada, Atsushi Hori, Takagi Masamichi and Yutaka Ishikawa: "CMCP: A Novel Page Replacement Policy for System Level Hierarchical Memory Management on Many-cores", In Proc. HPDC, 2014
- 6) Taku Shimosawa, Balazs Gerofi, Masamichi Takagi, Gou Nakamura, Tomoki Shirasawa, Yuji Saeki, Masaaki Shimizu, Atsushi Hori and Yutaka Ishikawa "Interface for Heterogeneous Kernels: A Framework to Enable Hybrid OS Designs targeting High Performance Computing on Manycore Architectures", In Proc. HiPC, 2014
- 7) Balazs Gerofi, Akio Shimada, Atsushi Hori and Yutaka Ishikawa: "Partially Separated Page Tables for Efficient Operating System Assisted Hierarchical Memory Management on Heterogeneous Architectures", In Proc. CCGRID, 2013
- 8) 佐伯 裕治, 清水 正明, 白沢 智輝, 中村 豪, 高木 将通, Balazs Gerofi, 思 敏, 石川 裕, 堀 敦史, 「ヘテロジニアス計算機上のOS機能委譲機構」, 情報処理学会, 2013-OS-125(15), pp. 1-7, 2013
- 9) Min Si and Yutaka Ishikawa, "Design of Communication Facility on Heterogeneous Cluster," 情報処理学会, 2012-HPC-133, 2012
- 10) Min Si, "An MPI Library Implementing Direct Communication for Many-Core Based Accelerators," SC'12, poster, 2012
- 11) Yuki Matsuo, Taku Shimosawa and Yutaka Ishikawa, "A File I/O System for Many-Core Based Clusters," in conjunction with ICS2012, 2012
- 12) Min Si and Yutaka Ishikawa, "Design of Direct Communication Facility for Manycore-based Accelerators," CASS2012 in conjunction with IPDPS2012, 2012
- 13) 下沢, 石川, 堀, 並木, 辻田, 「メニーコア向けシステムソフトウェア開発のための実行環境の設計と実装」, 情報処理学会, 2011-OS-118(1), 1-7, 2011
- 14) Taku Shimosawa and Yutaka Ishikawa, "Inter-kernel Communication between Multiple Kernels on Multicore Machines," IPSJ Transactions on Advanced Computing Systems Vol.2, No.4 (ACS 28), pp. 64-82, 2009
- 15) Taku Shimosawa, Hiroya Matsuba and Yutaka Ishikawa, "Logical Partitioning without Architectural Supports," 32nd IEEE Intl. Computer Software and Applications Conference (COMPSAC 2008), pp. 335-364, 2008



Processes in Process (PIP)

背景

- メニーコアの台頭
 - これまでのノード内並列モデルのままで良いのか？
 - マルチプロセス (MPI) 、マルチスレッド (OpenMP)

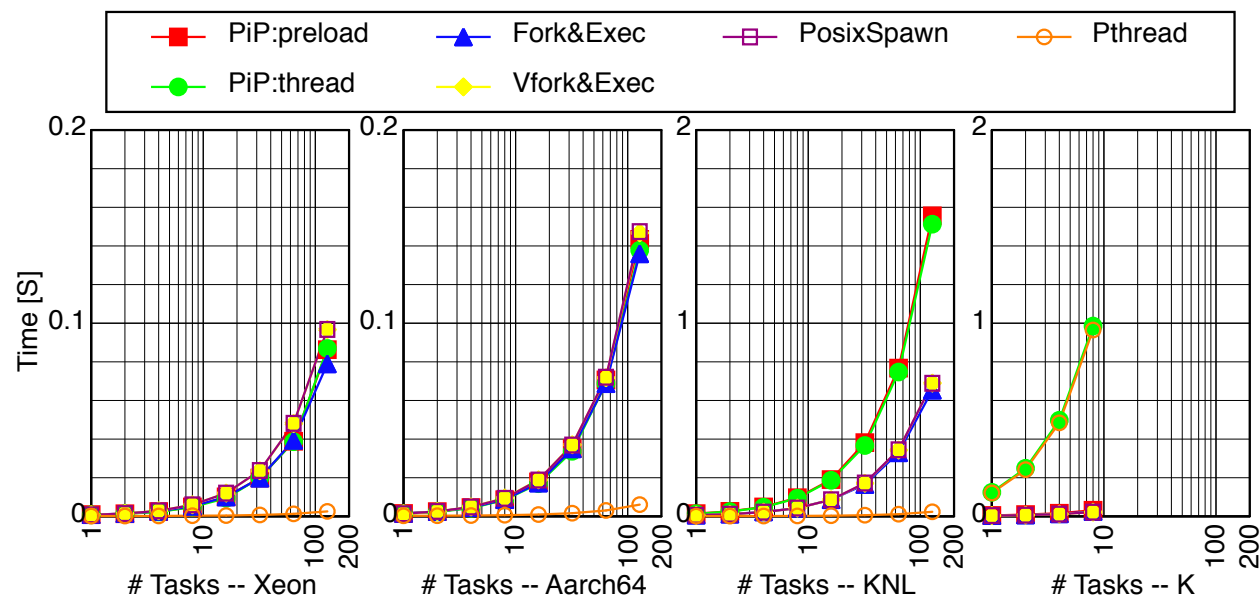
		Address Space	
		Isolated	Shared
Variables	Privatized	Multi-Process (MPI)	3rd Exec. Model
	Shared	??	Multi-Thread (OpenMP)

アイデア

- プロセス間でデータを互いに直接アクセスできるようにしよう！
- 既存研究
 - Smartmap (米SNL)
 - **Kitten OS**
 - X86 アーキテクチャに依存
 - Xpmem (SGI ?)
 - **Linux カーネルモジュール**
 - 直接アクセスするためにシステムコールが必要
 - PVAS (理研)
 - **Linux カーネルに約 5,000 行のパッチ**
 - MPC (CEA@仏)
 - **独自のコンパイラが必要**

PiP is very PORTABLE

	CPU	OS
Xeon and Xeon Phi	x86_64	Linux
	x86_64	McKernel
the K and FX10	SPARC64	XTCOS
ARM (Opteron A1170)	Aarch64	Linux

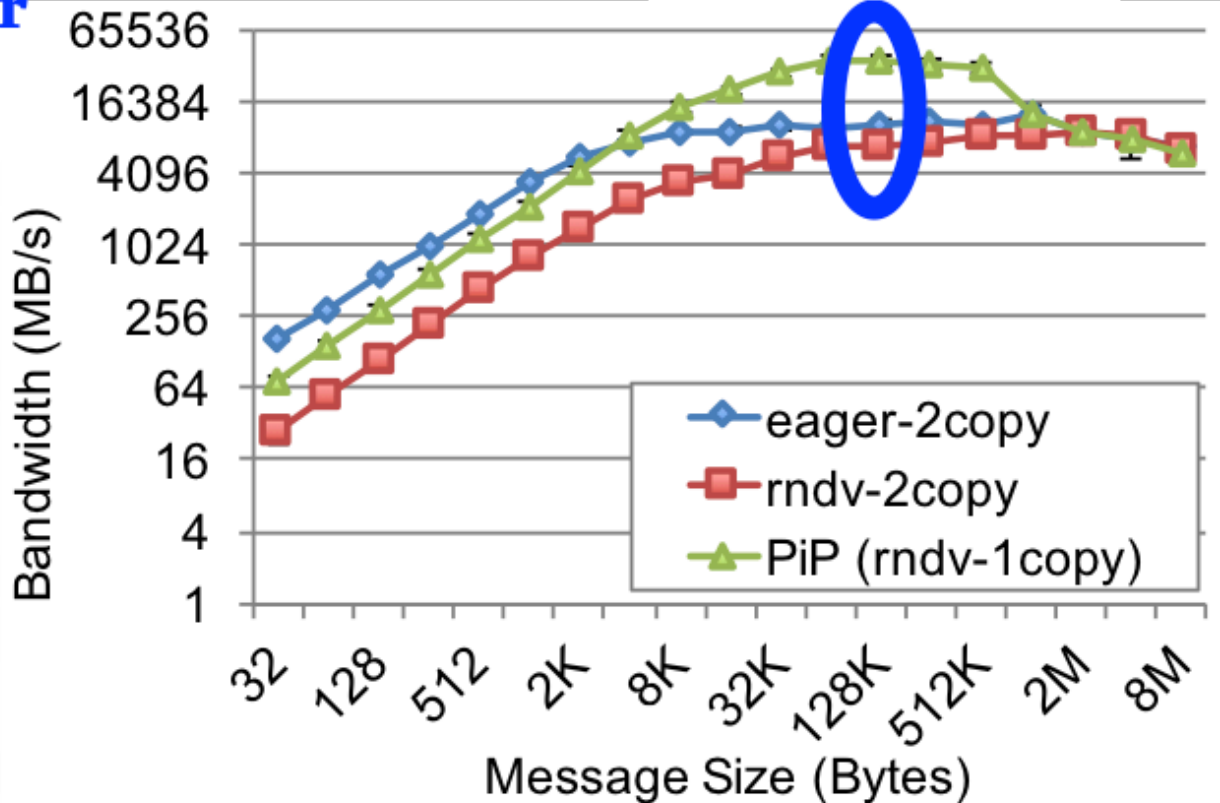


Task Spawning Time

MPICHでの評価

- Current Eager/Rndv. 2 Copies
- PiP Rndv. 1 Copy

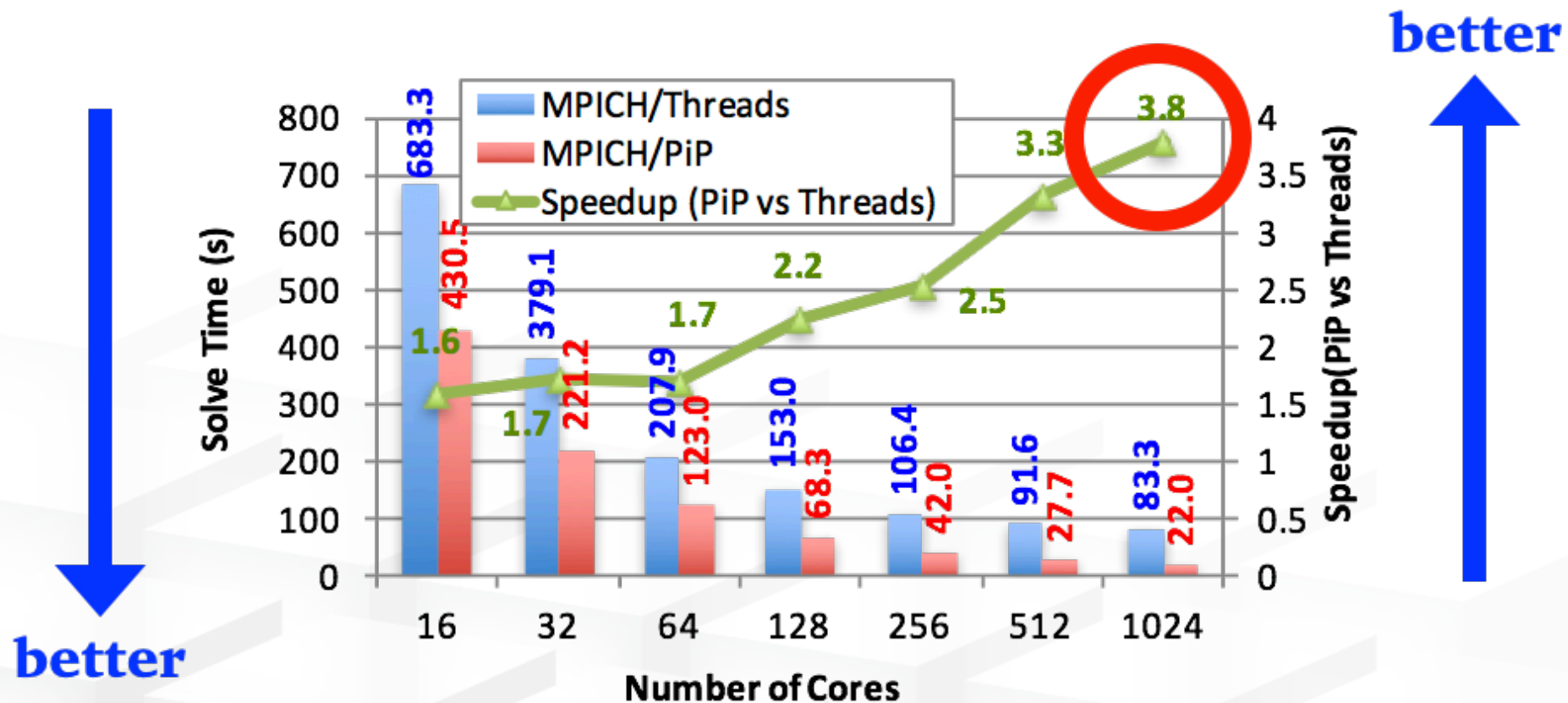
better



(Xeon/Linux)

SNAPでの評価

- (MPI + OpenMP) $\Rightarrow$ (MPI + PiP)



PiP V.S. threads in hybrid MPI+X SNAP
strong scaling on OFP (1-16 nodes, flat mode).

PIP – まとめ

- 従来の2つのノード内並列実行方式、プロセス並列とスレッド並列のいいとこ取り
- これまでに提案された方式と異なり、ユーザレベルで実装される
- **PIP** により、以下の利点が期待される
 - より高速なノード内通信
 - より低メモリ消費な並列実行環境
- 今後
 - 論文発表の後、オープンソースとして公開
 - MPICH、Open MPI の PIP による実装
- 共同研究
 - ANL、テネシー大学、UIUC、仏CEA など