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）

2015/12/17

アウトライン

- McKernel概要（12/14日セミナー資料をベースに）
  - 背景
  - LinuxとLight-Weight Kernelのハイブリッド手法
  - McKernel
  - HPC向けOSプロジェクト比較

※ 12/14 McKernelハンズオン講習会@秋葉原

27名の参加者

背景

- HPCシステムの複雑化
  - 並列性の増大
    - コア数とノード数
  - メモリ階層数の増大
  - 電力制約

➡ 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）

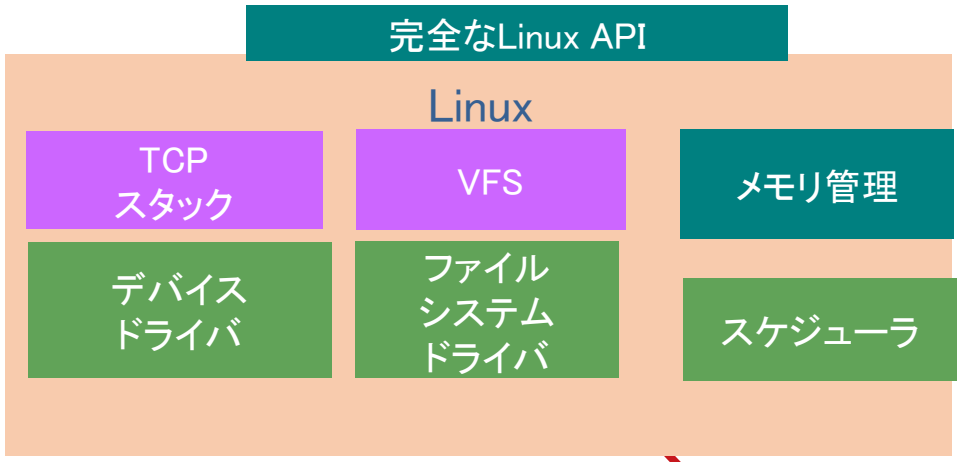
➡ スケーラブルで安定した性能の提供と
新ハードウェアへの迅速な追従が必要
（従来はAPIを絞ったLight-Weight Kernelで対応）

- 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
  - 周辺ソフトの複雑化

➡ Linux APIを用いて開発されることが多いため、Linux との互換性が重要

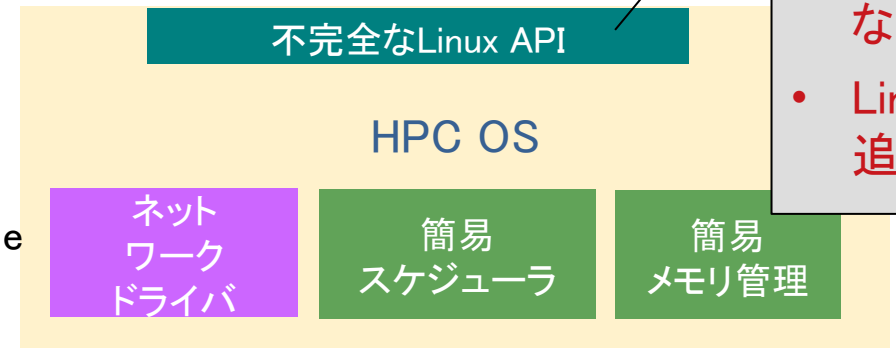
➡ これらの要件を同時に満たせるか？

アプローチ（1）引き算アプローチ



LinuxからHPC性能を阻害する要因を除去

- OSノイズの除去
  - 例：デーモン、タイマ割り込み
- メモリ管理の簡略化
- スケジューラの簡略化



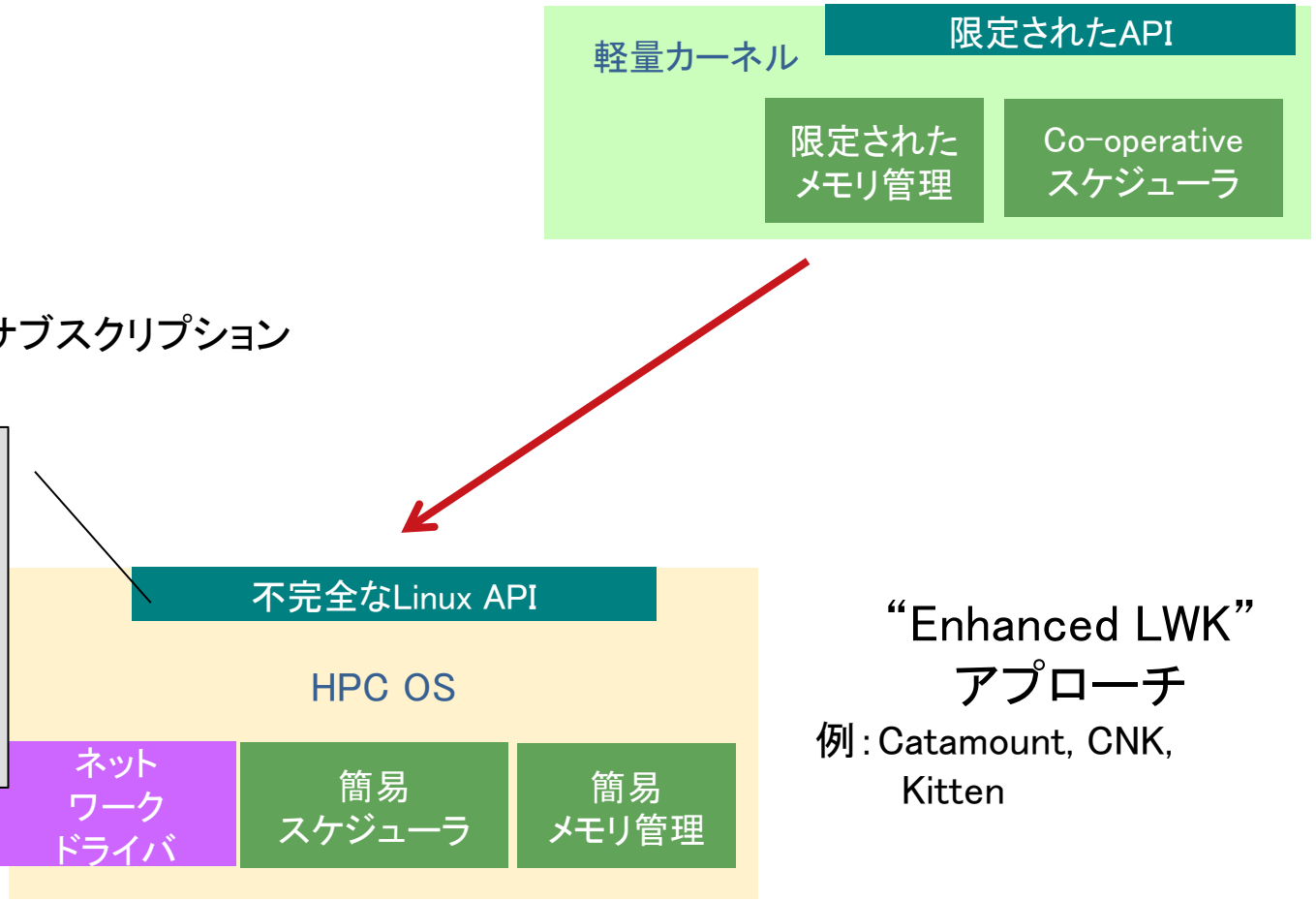
- 阻害要因除去を徹底すると完全なLinux APIサポートができなくなる
- Linuxカーネル変更への追従が困難

“Stripped down Linux”アプローチ

- 例：ZeptoOS, Cray’s Extr. Scale Linux, Fujitsu’s Linux,

アプローチ（2）足し算アプローチ

- スクラッチから作成した軽量カーネル (Light-Weight Kernel, LWK)から始める
- スケーラビリティを損なわないように機能を追加して Linux APIに近づける
  - 例
    - システムコール
    - /procファイルシステム
    - ダイナミックリンクライブラリ
    - スレッドのコアへのオーバーサブスクリプション



- 完全なLinux APIサポートができなくなる
- デバイスドライバの新規開発が必要

“Enhanced LWK” アプローチ

例：Catamount, CNK, Kitten

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？
- **「足し算」や「引き算」アプローリでは無理**

我々のアプローチ　合わせ技アプローチ

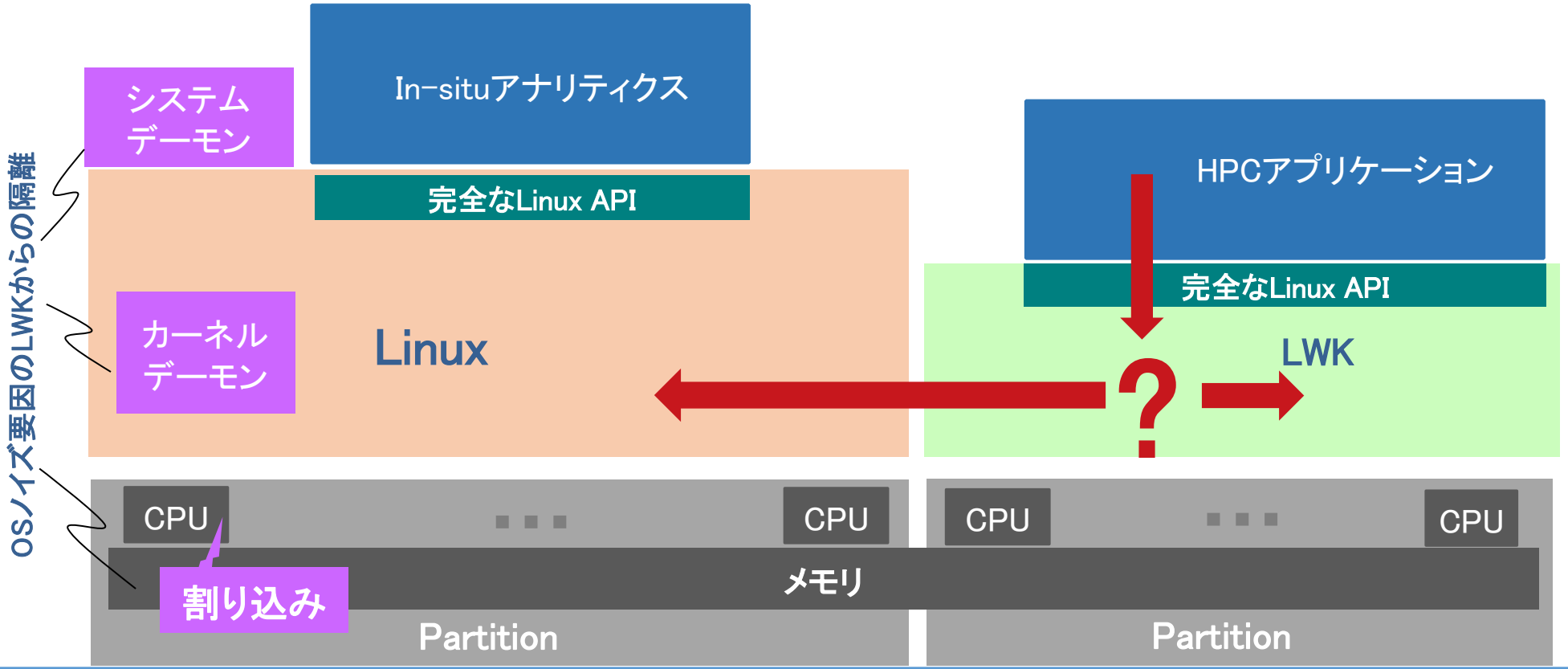
LinuxとLWKのハイブリッド

- 資源を2つのパーティションに分けLinuxとLWKを並列動作、
  またHPCアプリをLWK上で動作

→性能安定性・スケーラビリティ・新ハードウェアへの迅速な追従

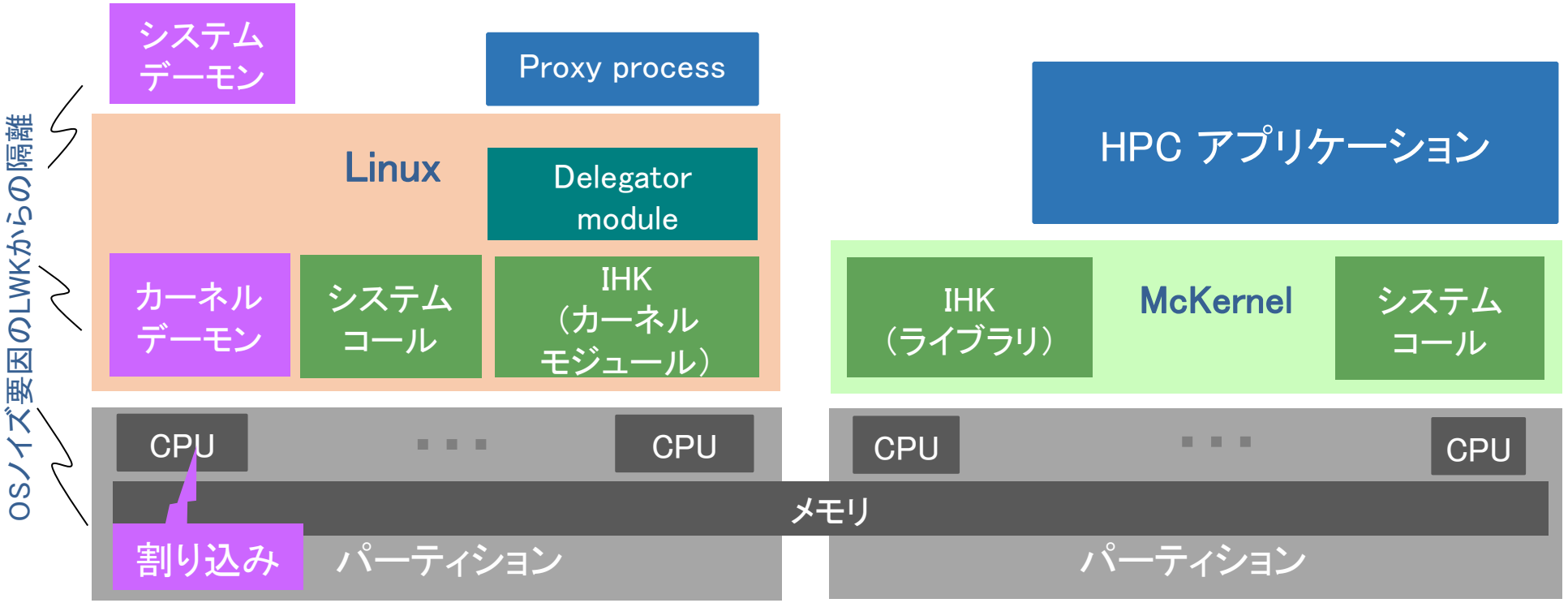
- OSサービスの一部をLWKで実行、他をLinuxにオフロード

→Linux APIサポート



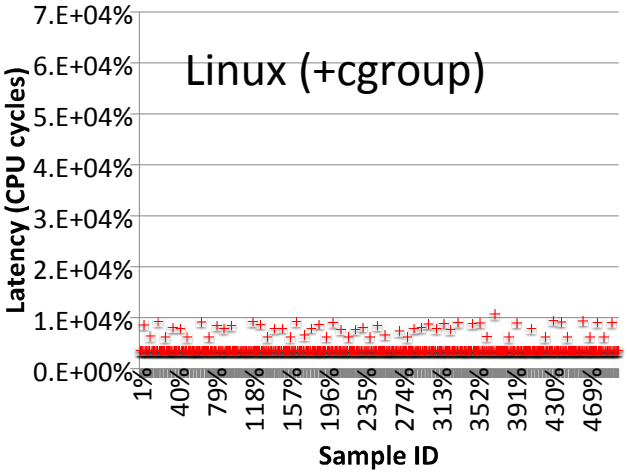
McKernel のアーキテクチャ

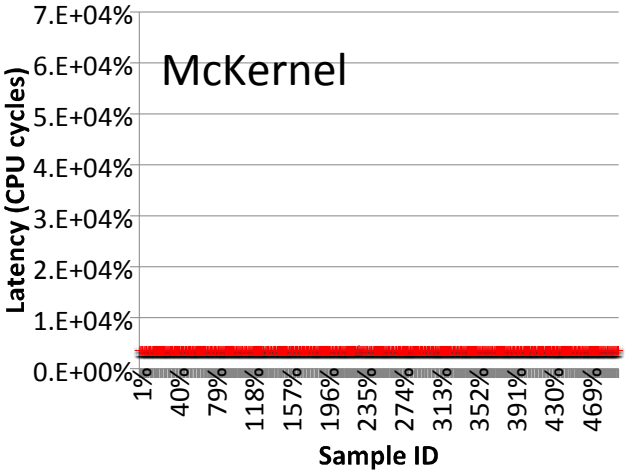
1. Interface for Heterogeneous Kernels (IHK): 複数の異種OSを持つためのフレームワーク
   - ノード資源のパーティショニング
   - LWKの管理(例:カーネルの起動、停止)
   - LWKとLinuxの間の通信機構(Inter-kernel communication, IKC)の提供
2. McKernel: スクラッチから開発されたHPC向けLWK
   - ノイズレス
   - 性能クリティカルなシステムコールのみMcKernelで動作、他はLinuxにオフロード



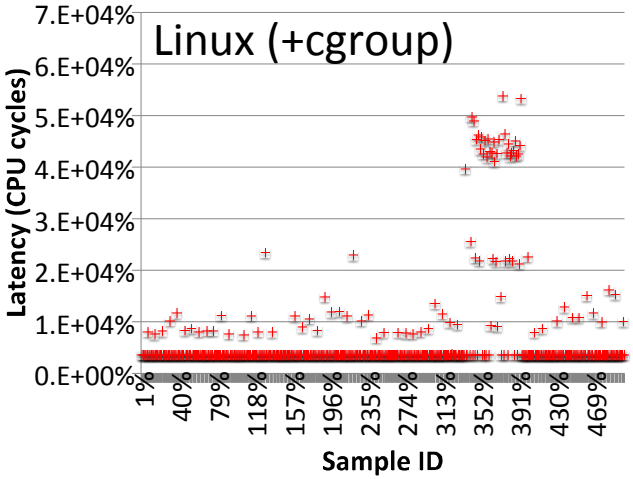
McKernelの評価 (1)

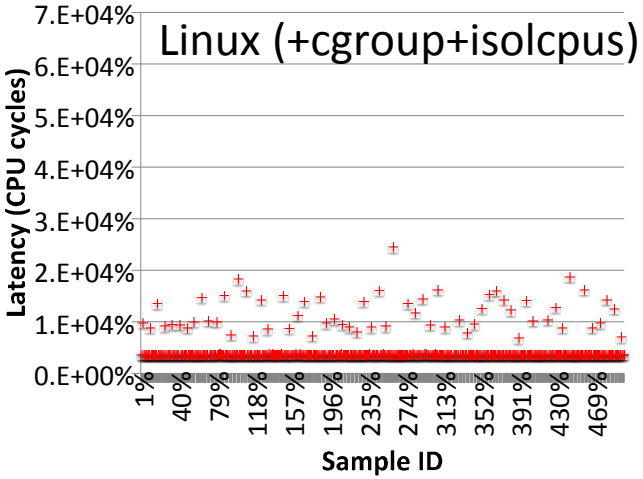
- ノイズの計測（FWQ）

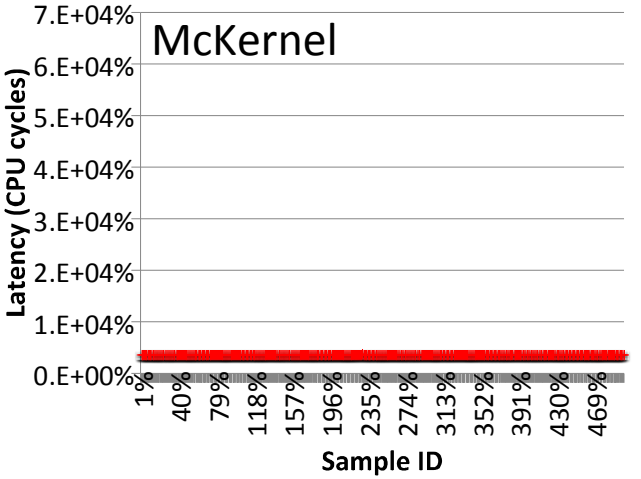




- MapReduce on Linux

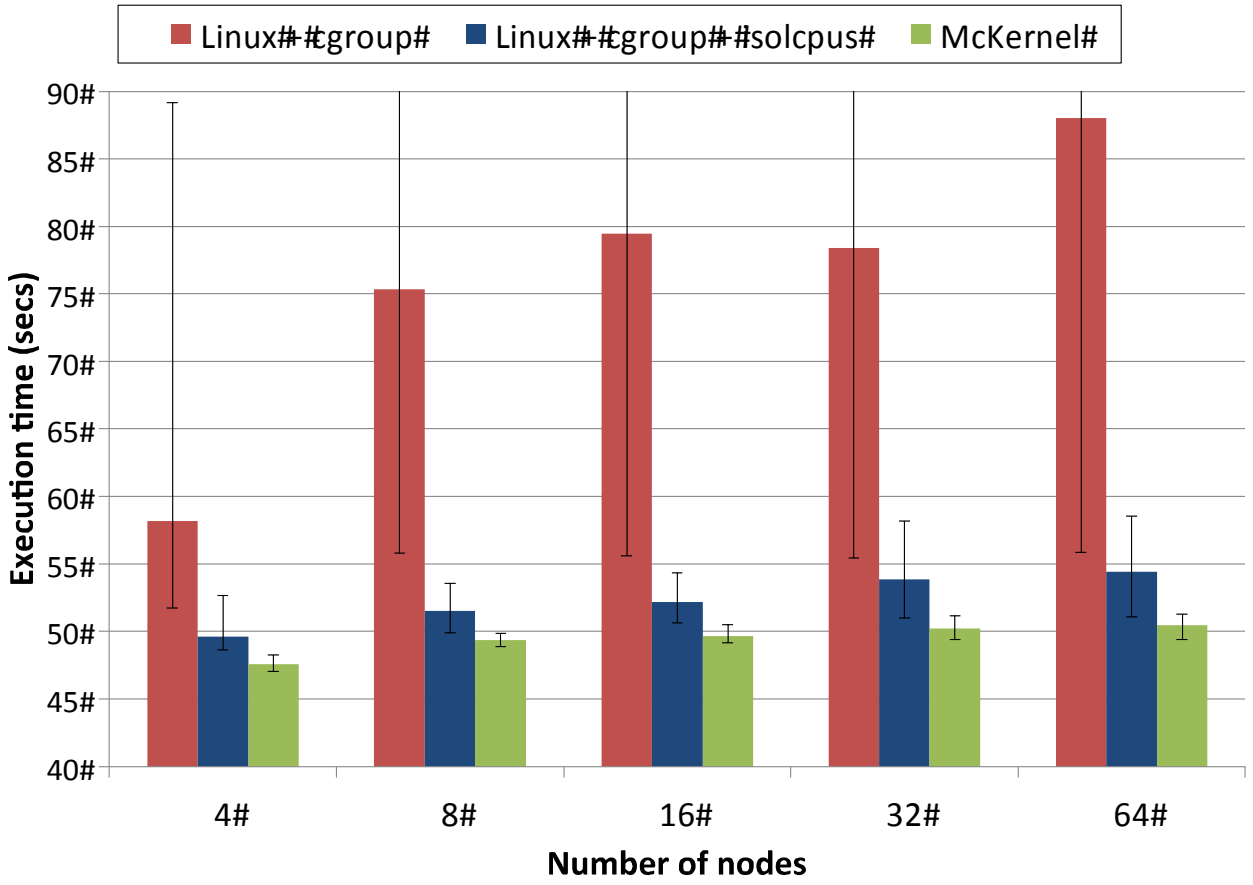






McKernelの評価 (2)

- Linux 側にMapReduce を走らせ，McKernel 側で HPCプログラムを実行した時の HPC プログラムの実行時間



（b）HPC-CG

IPDPS16への投稿論文から

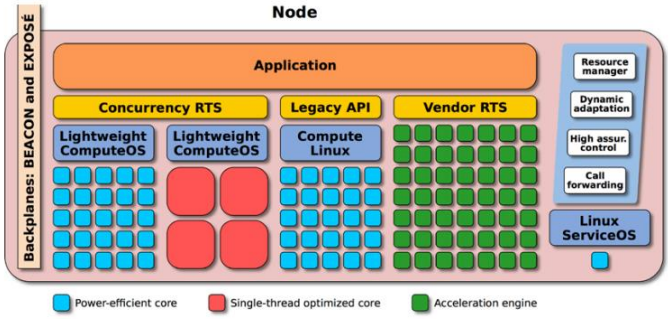
McKernelのAPI/ABI

- HPC向けLWK
- ノイズレス
- Linux API/ABI サポート
  - Linux Test Projectのテストスイートは，ほぼ100%パス

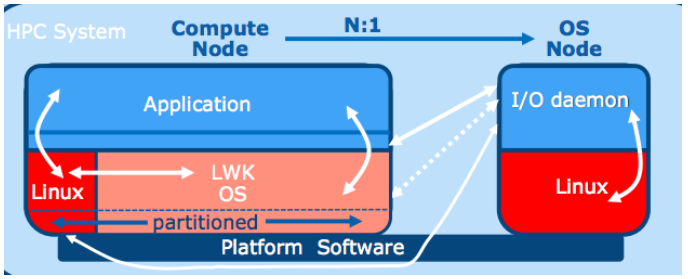
分類	実装済み	実装予定
プロセス管理	arch_prctl, clone, execve, exit, exit_group, fork, futex, get_cpu_id, getpid, get{u,g}id, gete{u,g}id, getres{u,g}id, getppid, gettid, kill, pause, ptrace, rt_sigaction, rt_sigpending, rt_sigprocmask, rt_sigqueueinfo, rt_sigreturn, rt_sigsuspend, set_tid_address, set{u,g}id, setre{u,g}id, setres{u,g}id, setfs{u,g}id, setpgid, sigaltstack, tgkill, vfork, wait4, waittid	{get,set}rlimit, {get,set}_thread_area, rt_sigtimedwait, signalfd, signalfd4
メモリ管理	brk, madvise, mincore, mlock, mmap, mprotect, mremap, msync, munlock, munmap, process_vm_{readv,writev}, remap_file_pages, set_robust_list, shmat, shmctl, shmdt, shmget	{get,set}_mempolicy, {get,set}_robust_list, mbind, migrate_pages, mlockall, modify_ldt, move_pages, munlockall
スケジュール	getcpu, gettimeofday, nanosleep, sched_getaffinity, sched_setaffinity, sched_yield	alarm, {get,set}itimer, settimeofday, time, times
パフォーマンスカウンタ	Original functions: pmc_init, pmc_start, pmc_stop, pmc_reset	PAPI functions

上記以外のシステムコールはLinuxに移譲

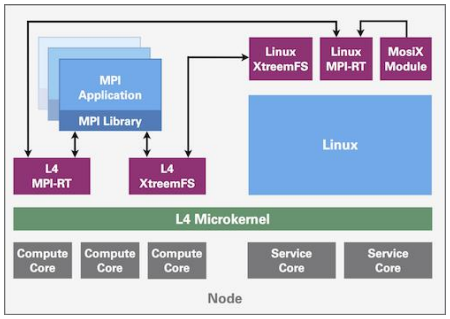
HPC向けOS比較 – 合わせ技が主流



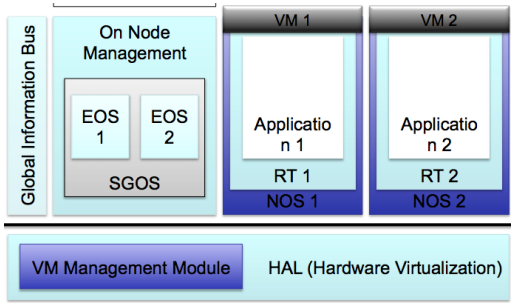
Argo (Argonne National Laboratory)



mOS (Intel)



SPPEXA (TU Dresden)



Hobbes (Sandia National Laboratories)

プロジェクト	Linuxカーネル変更要否	Linux用デバイスドライバの軽量カーネルからの利用	カーネルレベルでの隔離	Linux APIの完全なサポート	開発難易度
Argo	必要	Yes	No	Yes	小
mOS	必要	Yes	No	Yes	小
Hobbes	不要	No	Yes	No	大
SPPEXA (L4+Linux)	?	?	Yes	No?	大?
IHK/McKernel	不要	Yes	Yes	Yes	大

McKernel：まとめ

- **Xeon Phi 上でも動作**
  - Knights Corner (KNC)　　現行のメニーコアCPU
  - Knights Landing (KNL)　次期メニーコア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
- **McKernel**
  - Linuxよりはるかに単純なので，変更，機能追加が容易
- **近い（？）将来**
  - ヘテロなメニーコアへの対応
    - メニーコアは高並列処理
      - カーネルの動作は低い並列性，逐次要素が大きい
    - ヘテロなメニーコア
      - 少数の逐次処理に特化したコア：Linux
      - 多数の並列処理に特化したコア：McKernel
  - 複雑なメモリシステムや電力制限にも対応予定

McKernel に関連する論文リスト

1) Balazs Gerofi, Takagi Masamichi and Yutaka Ishikawa: "Toward Operating System Support for Scalable Multithreaded Message Passing", In Proc. EuroMPI, 2015
2) 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
3) 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
4) 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
5) 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
6) 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
7) 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
8) Min Si and Yutaka Ishikawa, "Design of Communication Facility on Heterogeneous Cluster," 情報処理学会, 2012-HPC-133, 2012
9) Min Si, "An MPI Library Implementing Direct Communication for Many-Core Based Accelerators," SC' 12, poster, 2012
10) Yuki Matsuo, Taku Shimosawa and Yutaka Ishikawa, "A File I/O System for Many-Core Based Clusters," in conjunction with ICS2012, 2012
11) Min Si and Yutaka Ishikawa, "Design of Direct Communication Facility for Manycore-based Accelerators," CASS2012 in conjunction with IPDPS2012, 2012
12) 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
13) Taku Shimosawa and Yutaka Ishikawa, "Inter-kernel Communication between Multiple Kernels on Multicore Machines," IPSJ Transactions on Advanced Comuting Systems Vol.2, No.4 (ACS 28), pp. 64-82, 2009
14) 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

One more thing…

- **MPI規格書の日本語訳**
  - PCCCの支援を受け，理研AICSのメンバーが翻訳に携わる
    - 翻訳にあたり，MPI Forum に確認済み
    - ”Unofficial Version”
    - 見つけた間違い，内容の確認も MPI Forum へ報告，確認済み
  - MPI 2.2 日本語訳初版（652ページ！約1年！）
    - http://www.pccluster.org/ja/mpi.html
    - 印刷？
  - 引き続き，MPI 3.1(?) の翻訳も開始する予定