スーパーコンピュータにおける電力性能最適化フレームワークの評価

稲富雄一、井上弘士

（九州大学大学院システム情報科学研究院）

謝辞

- 九大情報基盤研究開発センターならびに日立SEの皆様には大変お世話になりました
  - 色々ご迷惑もおかけして、すいませんでした・・・

本研究は、本センターの「先端的計算科学研究プロジェクト（ベンチマーク課題）」ならではの成果

一部管理者権限が必要な実験を大規模計算機を用いて行うことは、九大センターを除く他大学計算機センターのシステムではできないこと

おかげさまでSC'15のtechnical paperに採択されました！

Y.Inadomi, T. Patki, K. Inoue, M. Aoyagi, B. Rountree, M. Schulz, D.Lowenthal, Y. Wada, K. Fukazawa, M. Ueda, M. Kondo, I. Miyoshi,
"Analyzing and Mitigating the Impact of Manufacturing Variability in Power-Constrained Supercomputing"

研究概要

- プロセッサ製造ばらつきは電力制約型スパコンにおいて重大な問題である
  - ✓ ４つのスパコンに対して製造ばらつきを解析
  - ✓ 製造ばらつきが電力制約下で最大64%の性能ばらつきに

- 適切な電力配分により製造ばらつき問題が解決出来る
  - ✓ 電力制約下での性能ばらつきを解消するために、低コスト、スケーラブルなばらつきを考慮した電力配分手法を提案
  - ✓ 提案手法により最大5.4倍、平均1.8倍の性能向上を達成

もくじ

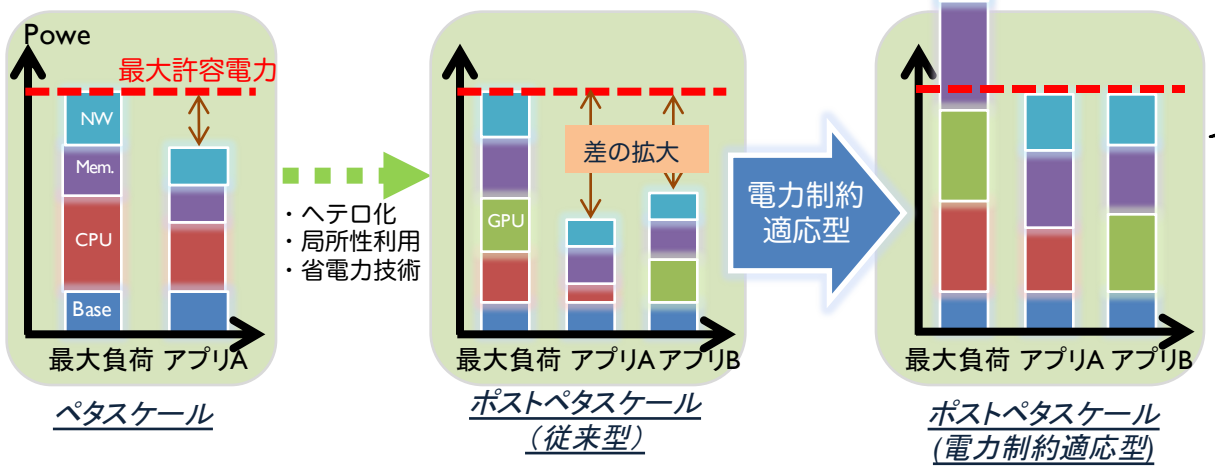
研究背景＝「スパコンの消費電力問題」

- Exa-scaleスパコンでは京コンピュータの２倍程度の消費電力（20-30MW）で100倍の性能向上が必要
  - 電力性能比（FLOPS/W）は「京」の50倍！
- 現状のスパコンシステム設計では与えられた電力バジェットをうまく使えない
  - 計算機へ要求する資源がアプリ依存であることを反映しにくい・・・

ポストペタスケールシステムのあるべき姿

- ハードウェア資源の有効利用から電力資源の有効利用へのパラダイムシフト

ノード数などのハードウェアではなく消費電力こそが最重要資源！



与えられた電力バジェットを効率的に利用してアプリの性能を最適化（＝電力最適化）する必要がある！

電力制約を適用してアプリを動かすと、面倒な問題が生じる・・・

電力制約適応型システム

- 最大負荷時電力が電力制約を超過することを積極的に許容
- 電力性能ノブを自動制御することで実効電力を制約以下に抑制
- 電力資源を計算・記憶・通信へ適応的に配分することで実効性能向上へ

実験環境

Blue=EP type
Red=With Comm. & Sync.

対象アプリ

- HPC Challenge: star DGEMM star STREAM(Triad)
- NPB: BT, SP, EP
- Magneto Hydro-Dynamics(MHD) simulation
  - Typical stencil app. to simulate space plasma
  - Calculations and communications appear in turn
- Fiber benchmark suite: mVMC-mini (mVMC)
  - Variational Monte-Carlo simulation for strongly correlated electron system

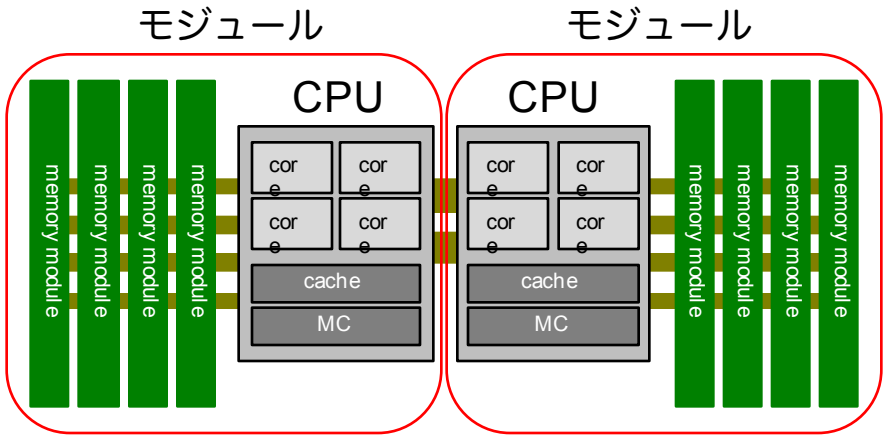
対象計算機

Site	Node Micro-Architecture	Total nodes	Procs. Per Node	Cores Per Procs.	Power Msrmt.
Cab(LLNL)	Intel E5-2670 Sandy Bridge	1,296	2	8	RAPL
BG/Q Vulcan (LLNL)	IBM PowerPC A2	24,576	1	16(compute)	EMON
Teller (SNL)	AMD A10-5800K Piledriver	104	1	4	PI
HA8K(Kyushu Univ.)	Intel E5-2697v2 Ivy Bridge	965	2	12	RAPL

制御対象

- 本発表での用語の定義

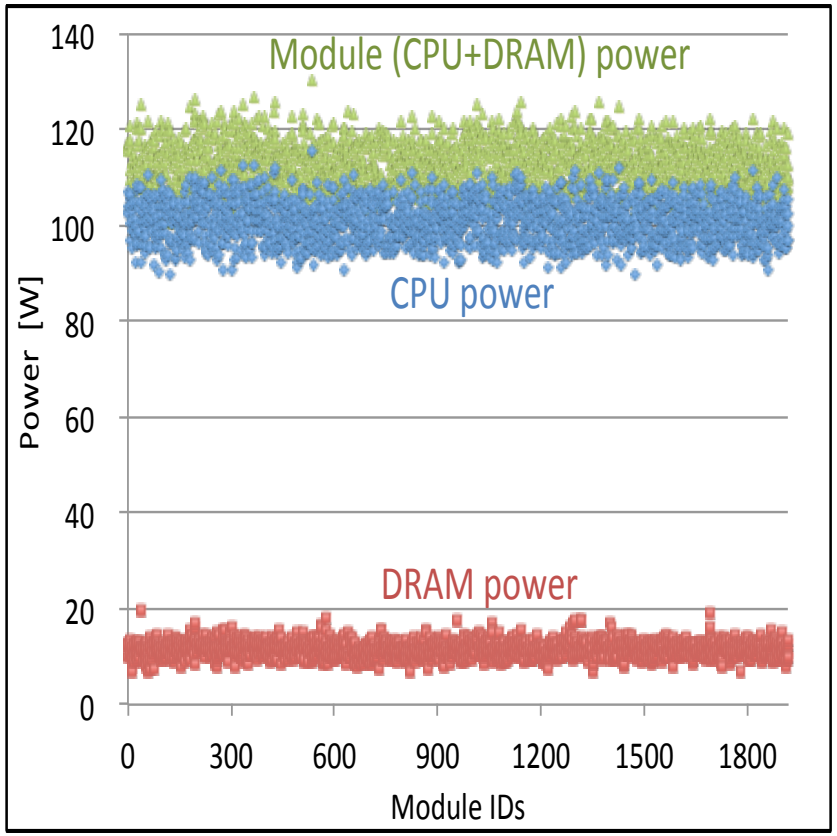
CPU　＝　プロセッサ（チップ）
モジュール　＝　CPU＋「CPUに直接繋がっているDRAM」



- 本来の電力制約対象はモジュール
- 実際に電力制約を適用しているのはCPUのみ
- DRAM消費電力は間接的に制約
  DRAM消費電力はCPU電力制約値から推定

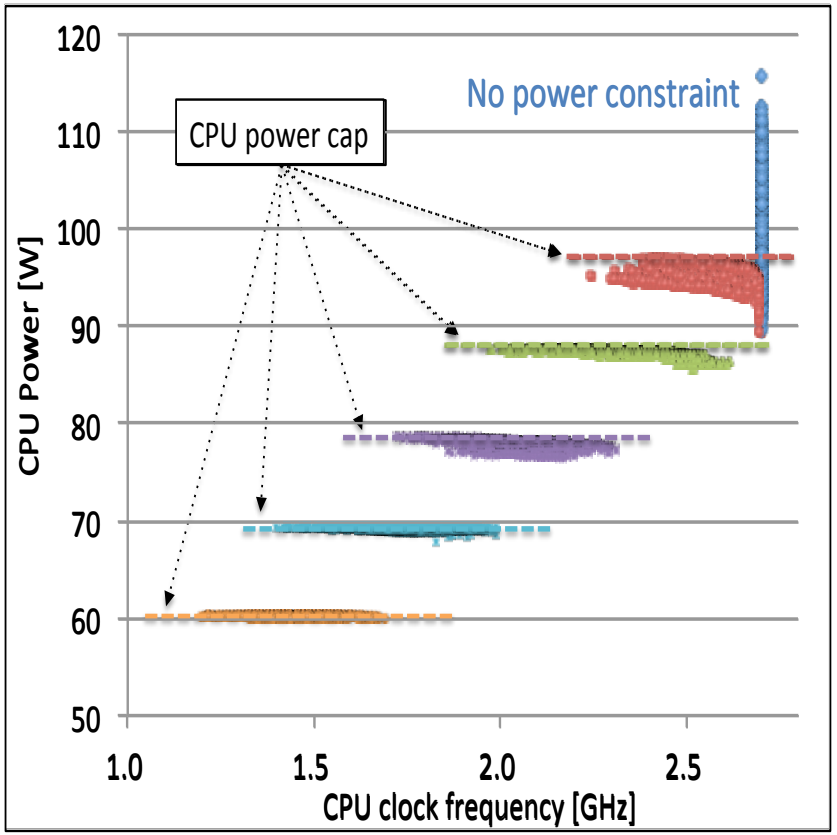
電力制約時のCPU動作周波数ばらつき

出典:Y.Inadomi et al., “Analyzing and Mitigating the Impact of Manufacturing Variability in Power-Constrained Supercomputing”, SC’15 (2015), Austin



30%

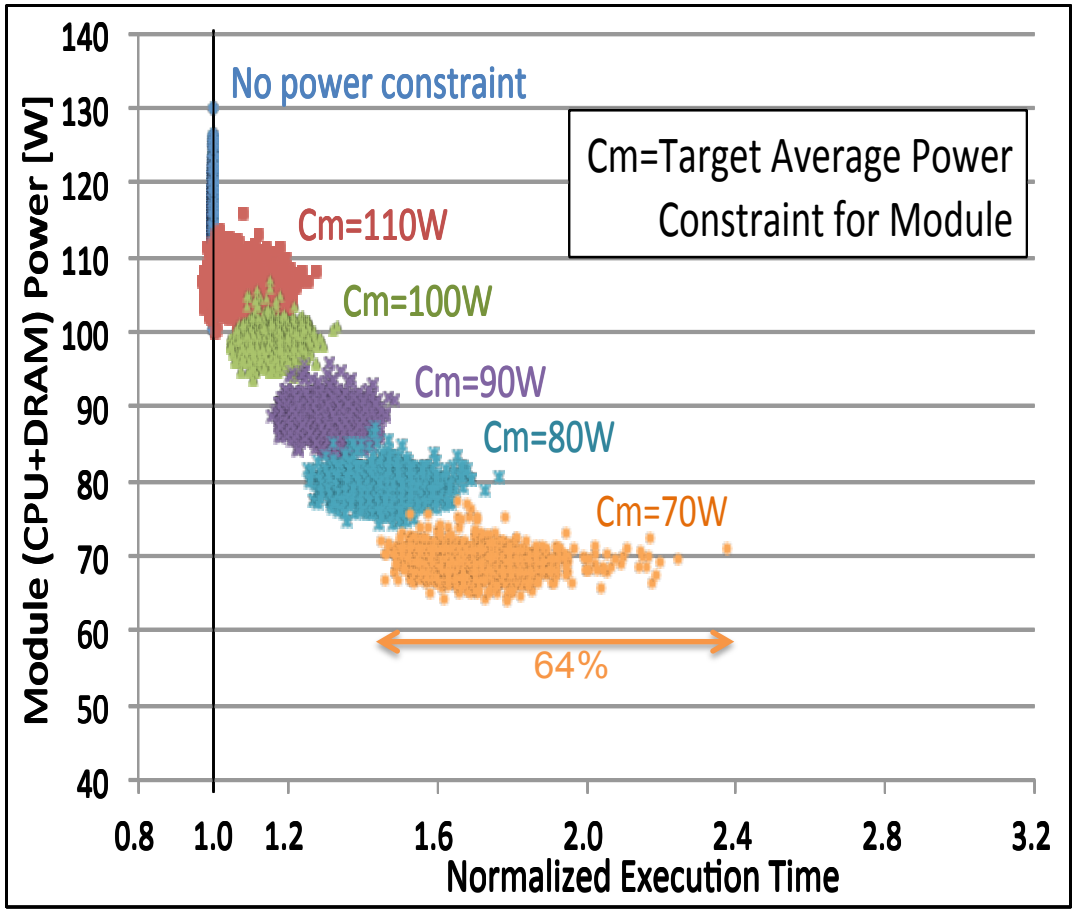
一律電力制約時のCPU消費電力と平均動作周波数
（HA8K, *DGEMM実行時）



非電力制約時の消費電力ばらつきが一律電力制約時の周波数ばらつきに置き換わる

電力制約時の処理性能ばらつき

実行性能とモジュール消費電力（HA8K, *DGEMM実行時）



出典:Y.Inadomi et al., “Analyzing and Mitigating the Impact of Manufacturing Variability in Power-Constrained Supercomputing”, SC’15 (2015), Austin

問題点と目標

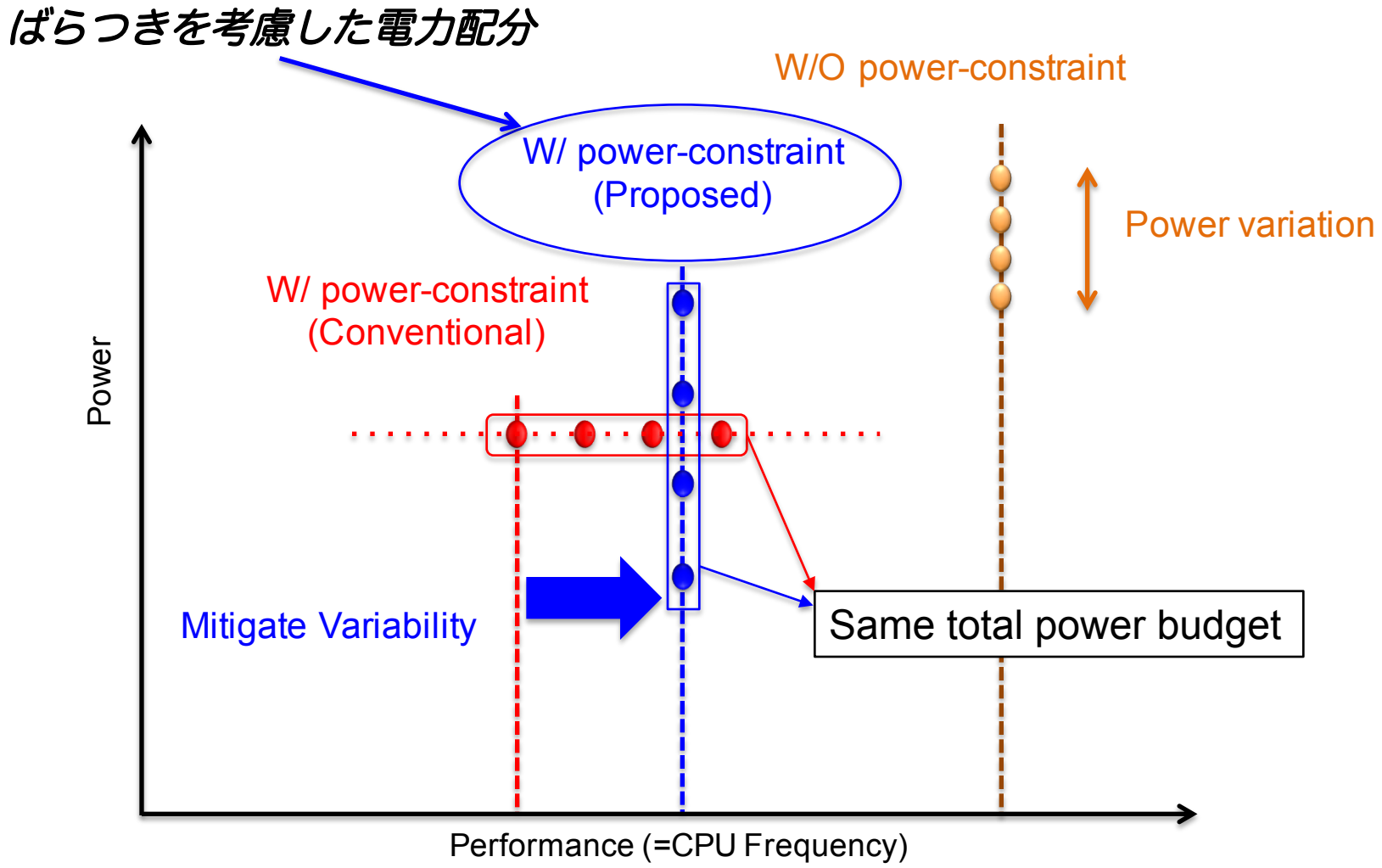
- 電力制約型スーパーコンピューティング
  - ✓ 将来のスパコンは電力制約のもとで運用される可能性あり
- 製造ばらつき
  - ✓ 製造ばらつきが電力制約時の性能ばらつきを引き起こす

将来の電力制約型スパーコンピュータ利用時のHPCアプリの性能が低下する・・・

目標

電力制約型スパコンでのHPCアプリ性能に対する製造ばらつきの影響を小さくする＝電力制約下でのアプリ性能最適化（電力性能最適化）

提案手法の概念


ばらつきを考慮した電力配分
W/O power-constraint
W/ power-constraint
(Proposed)
Power variation
W/ power-constraint
(Conventional)
Power
Mitigate Variability
Same total power budget
Performance (=CPU Frequency)

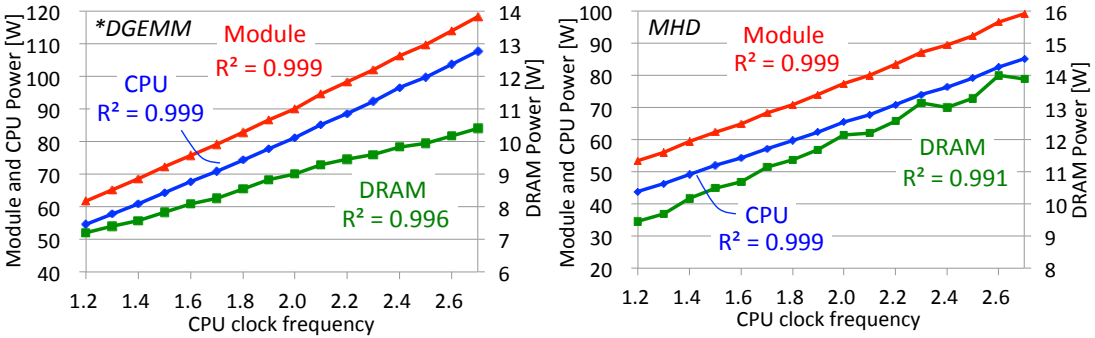
周波数と電力の関係（電力モデル）

仮定

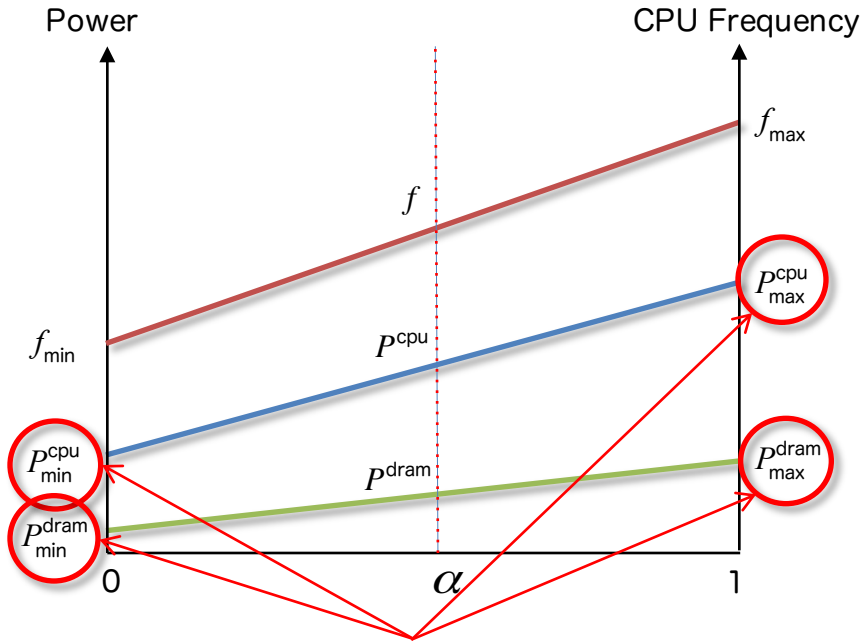
- CPU消費電力、DRAM消費電力は動作周波数に比例
- 動作周波数が同じであれば処理性能も同じ

$$f = \alpha(f_{max} - f_{min}) + f_{min}$$
$$P^{cpu} = \alpha(P^{cpu}_{max} - P^{cpu}_{min}) + P^{cpu}_{min}$$
$$P^{dram} = \alpha(P^{dram}_{max} - P^{dram}_{min}) + P^{dram}_{min}$$
$$(0 \le \alpha \le 1)$$

f = 動作周波数
P^{cpu}, P^{dram} = CPU, DRAM消費電力
f_{max}, f_{min} = 最高動作周波数、最低動作周波数
P^{cpu}_{max}, P^{dram}_{max} = 非電力制約時のCPU, DRAM消費電力
P^{cpu}_{min}, P^{dram}_{min} = 最低動作周波数時のCPU, DRAM消費電力



出典:Y.Inadomi et al., “Analyzing and Mitigating the Impact of Manufacturing Variability in Power-Constrained Supercomputing”, SC’15 (2015), Austin



この４点の情報は何らかの方法で推定

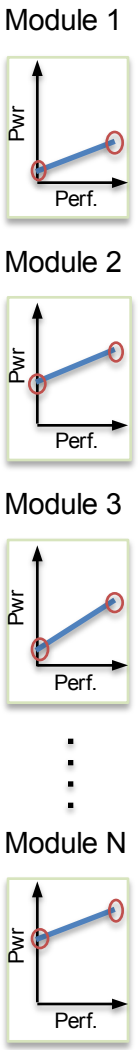
電力モデル補正

アプリ非依存
Power Variation Table (PVT)

Module ID	Normalized Power
1	1.0
⋮	⋮
k	1.2
⋮	⋮
N	0.8

アプリ依存推定消費電力

Module ID	Power Consumption
1	100
⋮	⋮
k	120
⋮	⋮
N	80



モジュール k で得られた消費電力情報

Module ID	Power Consumption
k	120

電力制御ノブの選択肢

２種類の電力制御ノブを検証

- ***Power Capping*** (Pc) using RAPL
- ***Frequency Selection*** (Fs) using CPUFreqlibs

	Power Capping (Pc)	Frequency Selection (Fs)
Power Constraint	◎ Guaranteed	△ Not guaranteed
Performance Equivalence	△ Not guaranteed	◎ Guaranteed

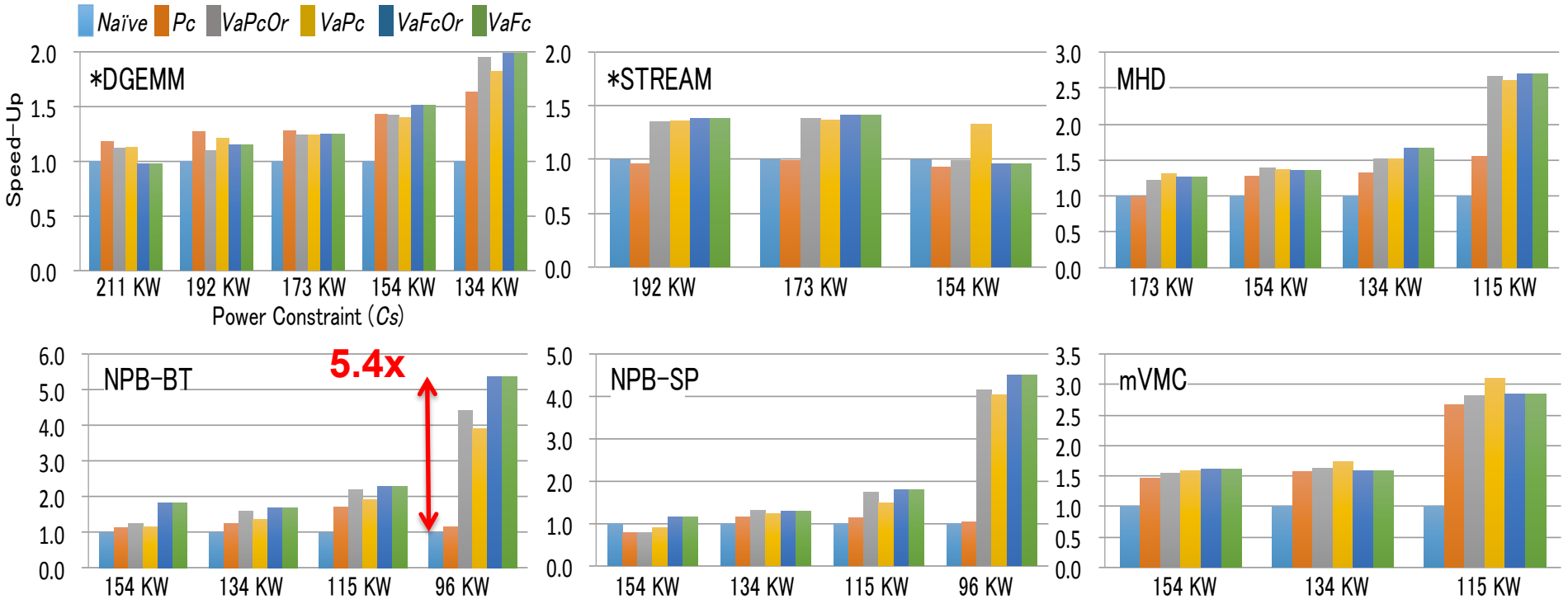
適用した電力配分手法

手法名	アプリ依存？	モジュール依存？	電力モデル補正	制約手法
Naive	No	No	No	Power Cap
Pc	Yes	No	Yes	Power Cap
VaPc	Yes	Yes	Yes	Power Cap
VaFs	Yes	Yes	Yes	Freq. Sel.
VaPcOr	Yes	Yes	No	Power Cap
VaFsOr	Yes	Yes	No	Freq. Sel.

Va=Variation-Aware, Pc=Power Capping, Fs=Frequency Selection
Or=Observed power data are used

速度向上比 (1920モジュール実行時)

様々な電力制約下での各配分手法適用時における性能向上比（対 naïve手法、HA8K）

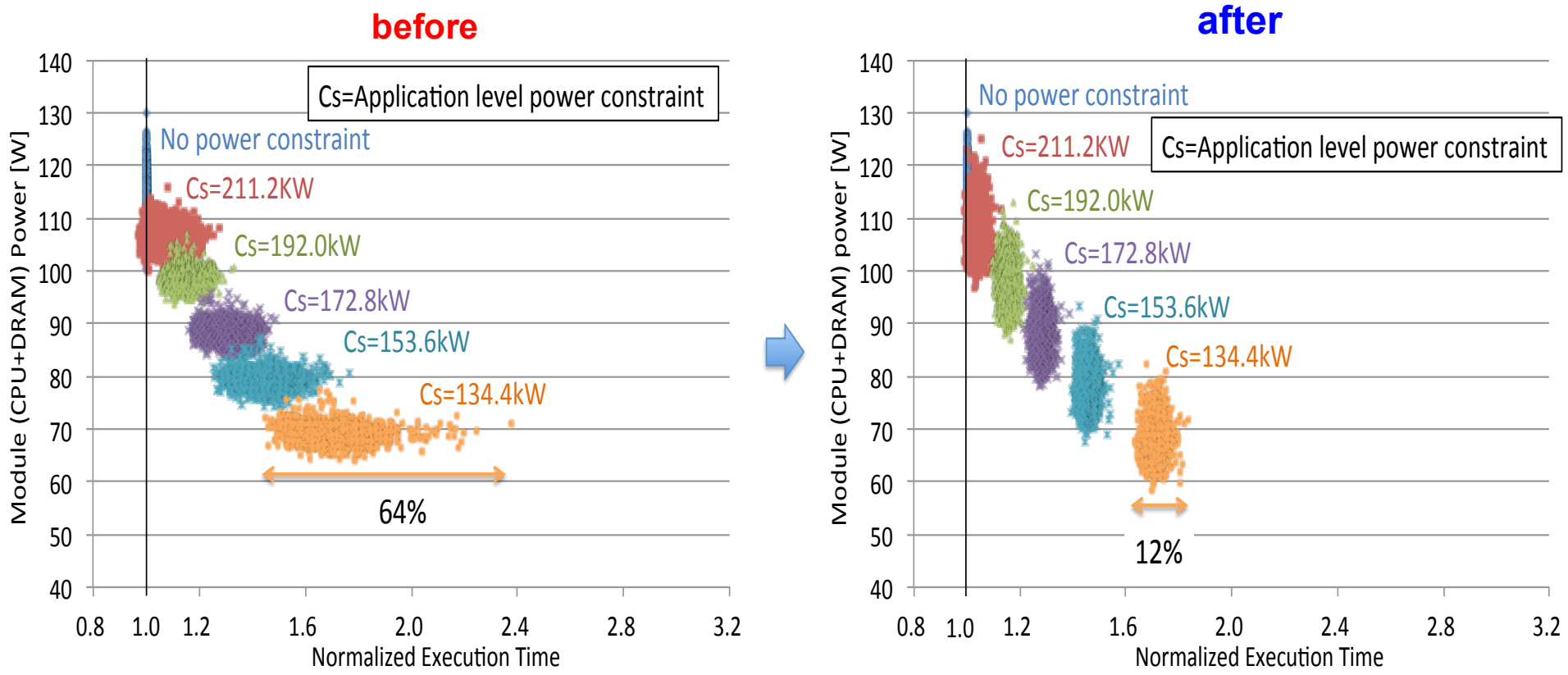


出典:Y.Inadomi et al., “Analyzing and Mitigating the Impact of Manufacturing Variability in Power-Constrained Supercomputing”, SC’15 (2015), Austin

- NPB-BTだと最大5.4倍の速度向上
- ６アプリ平均で1.8倍の速度向上

性能が向上した理由

ばらつきを考慮した電力配分適用前後の実行性能とモジュール消費電力
(HA8K, *DGEMM実行時)



出典:Y.Inadomi et al., "Analyzing and Mitigating the Impact of Manufacturing Variability in Power-Constrained Supercomputing", SC'15 (2015), Austin

- ばらつき考慮電力配分により性能ばらつきが改善

まとめ

- 現代のHPCシステムでは製造ばらつきが原因のモジュール間消費電力ばらつきが見られ、それが電力制約時の処理性能ばらつきを生じることが分かった
- 提案した低コスト、スケーラブルなばらつきを考慮した電力配分によって電力制約下でのHPCアプリ性能が最大で5.4倍、平均でも1.8倍向上した

謝辞

本研究は、JST,CREST の研究領域「ポストペタスケール高性能計算に資するシステムソフトウェア技術の創出」の研究課題「ポストペタスケールシステムのための電力マネージメントフレームワークの開発」の支援を受けています