マルチコア／マルチレーン・クラスタにおける性能評価及び最適化

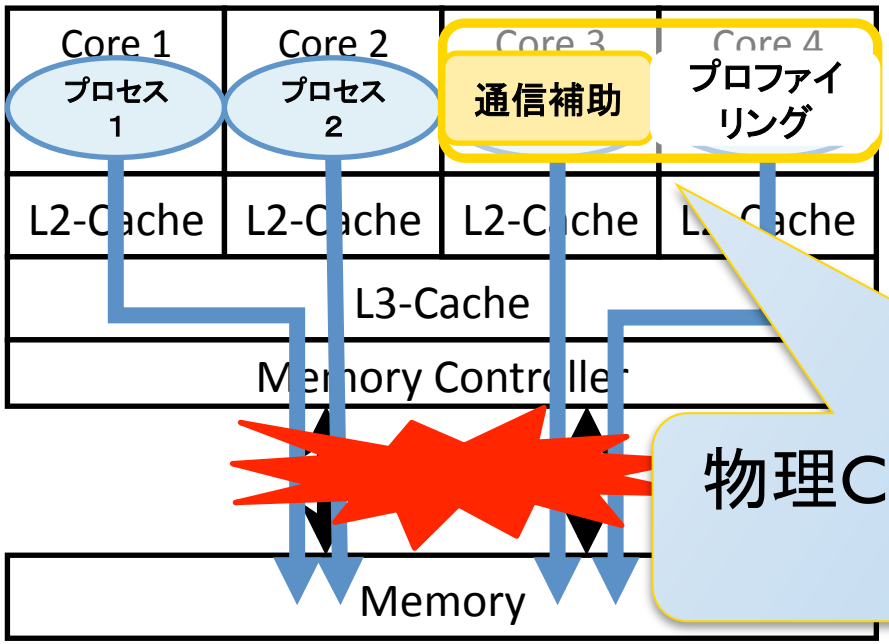
朴　泰祐
筑波大学システム情報工学研究科／
計算科学研究センター

Multi-core/Multi-lane cluster

- T2K Open Supercomputerに代表される fat-node + fat-bandwidth PC cluster
- ノード構成(T2K): 4-core CPU x 4 + Infiniband x 4
- ノード内の総メモリバンド幅及び通信バンド幅は高いが、複数のMPI processで共有する場合、バンド幅が不足
- 「multi-coreは常に100%使うべきか？」
  - バンド幅不足で有効利用不可能なcoreが発生
  - coreの利用方法は計算（演算）以外にないか？
- 「multi-railは常に100%使うべきか？」
  - メッセージ長によってはtrunkingが有効でない
  - アプリケーション特性によって、使用rail数を調整すべき
- これらを実アプリケーション上で評価・解析する
  - メモリバンド幅とアプリケーションの関係
  - trunking とアプリケーション性能の関係
- 学際共同利用研究チーム：筑波大＋自動車工業会（自工会）

multi-coreの有効利用

- メモリウォール問題の解決策
  - 演算並列化に**必要十分なCPUコア数**を設定
    ⇒演算性能とメモリアクセス性能の**バランスのとれた並列化**
  - 「**余剰コア**」の有効利用 ⇒ **総合的な性能向上**



メモリバンド幅がボトルネックとなり

物理CPUコア数－演算CPUコア数
→「余剰コア」と呼ぶ

ベンチマーク

- STREAM Benchmark
  - メモリアクセス性能を測定
  - キャッシュの効果を反映させないよう配列サイズを大きくとる
  -

name	kernel	Memory access
COPY	a(i) = b(i)	1 load, 1 store
SCALE	a(i) = q*b(i)	1 load, 1 store
SUM	a(i) = b(i) + c(i)	2 load, 1 store
TRIAD	a(i) = b(i) + q*c(i)	2 load, 1 store

- Lattice QCD(格子量子色力学): (石川@広島大学)
  - 素粒子物理のカラーゲージ理論QCD を4 次元離散格子に置き換え，格子点に変数を定義したもの
  - 多数の隣接通信、小データかつ多数の集団通信を含む

QCDに必要なデータ供給性能

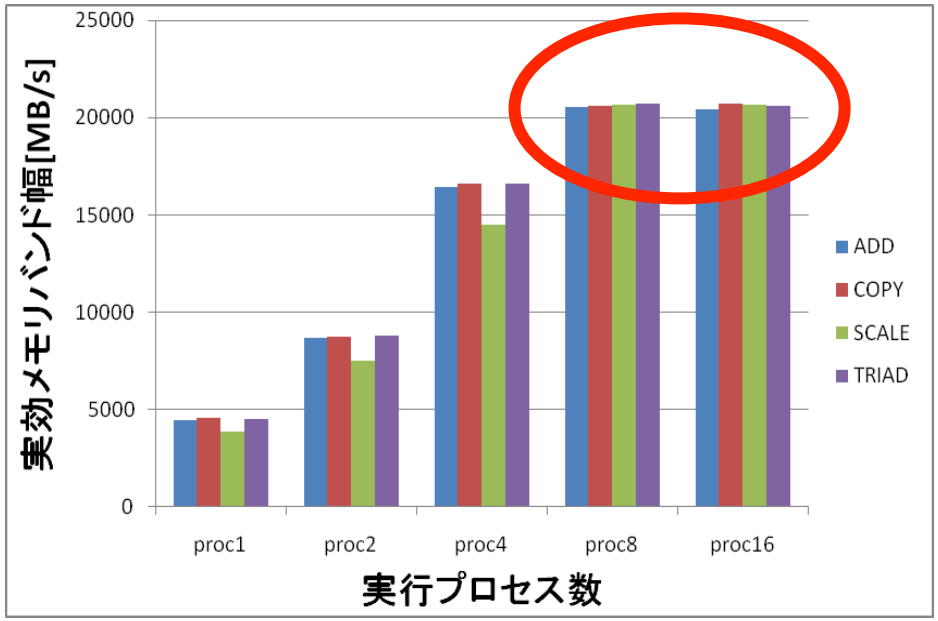
$$\mathrm{Mult}(n,m)^{a,b}_{\alpha,\beta} = \sum_{\mu=1}^{4}\left[(1-\gamma_\mu)_{\alpha,\beta}\,(U_\mu(n))^{a,b}\delta_{n+\hat{\mu},m} + (1+\gamma_\mu)_{\alpha,\beta}\,(U^\dagger_\mu(n-\hat{\mu}))^{a,b}\delta_{n-\hat{\mu},m}\right]$$

dim	computation	load	store	B/flops
t	168(x), 120(+) = 288 flop	21*2 complex = 672B	12 complex = 192B	3.00
z	144(x), 192(+) = 336 flop	21*2+12complex = 864B	12 complex = 192B	3.14
y	144(x), 192(+) = 336 flop	21*2+12 complex = 864B	12 complex = 192B	3.14
x	144(x), 192(+) = 336 flop	21*2+12 complex = 864B	12 complex = 192B	3.14
clover	288(x), 312(+) = 600 flop	21*2+12 complex = 864B	12 complex = 192B	1.76

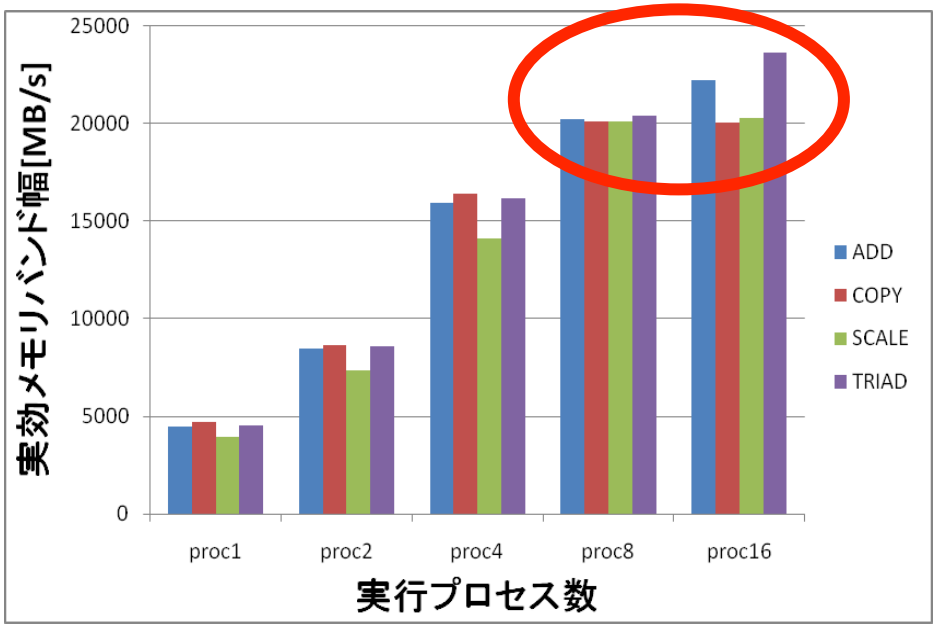
⇒ 5088B / 1896flop = 2.68 Byte/flop

- T2K-Tsukubaの「体力」: 0.13Byte/flop

結果－メモリアクセス性能：
(プロセス配置：各ソケットに分散，メモリバインド：ローカルメモリ)



STREAM benchmark 出力



PAPIによる計測結果

- MPIプロセス数8以上からメモリバンド幅の伸びが低下している

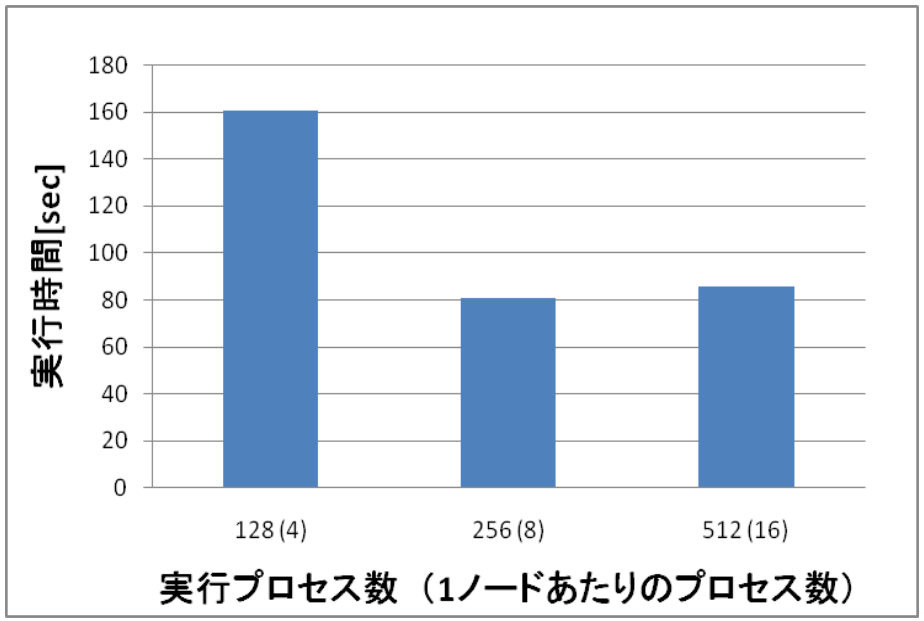
MPIプロセス数8以上でメモリウォール問題の影響
⇒ノード当たり8プロセスで性能向上頭打ち

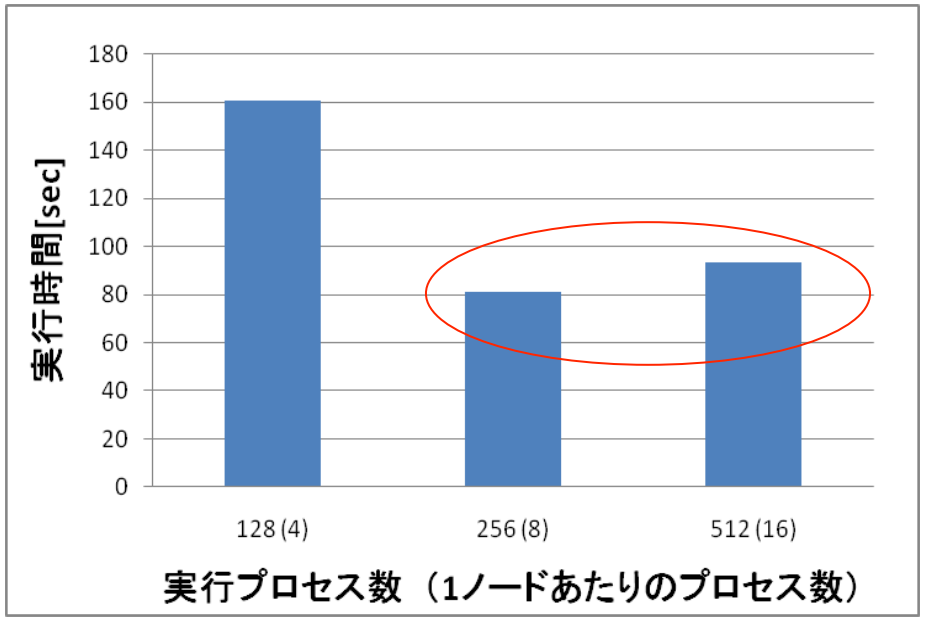
QCDベンチマーク性能評価

- QCDベンチマークを以下の環境で実行
- flat-MPI

プログラム	Solver bench
使用ノード数	32ノード(512CPUコア)固定
MPIプロセス数	128,256,512
MPIプロセス数(1ノードあたり)	4,8,16
QCDの問題サイズ(x,y,z,t)	32,32,32,64
ブロックあたりの格子サイズ	4

QCDにおけるノード内プロセス数の限界





ノード数固定、プロセス数変化の場合で、mem-check daemon なし（左）とあり（右）

- 1ノードあたり8、16プロセスの実行時間がほぼ同程度
  - ⇒ノード当たり8プロセス実行までは性能向上
  - ⇒メモリウォール問題の影響
- memory check daemon を走らせた場合、16proc/nodeの場合の性能が低下（ 8proc/nodeを下回る ）

memory bandwidth: 考察と今後

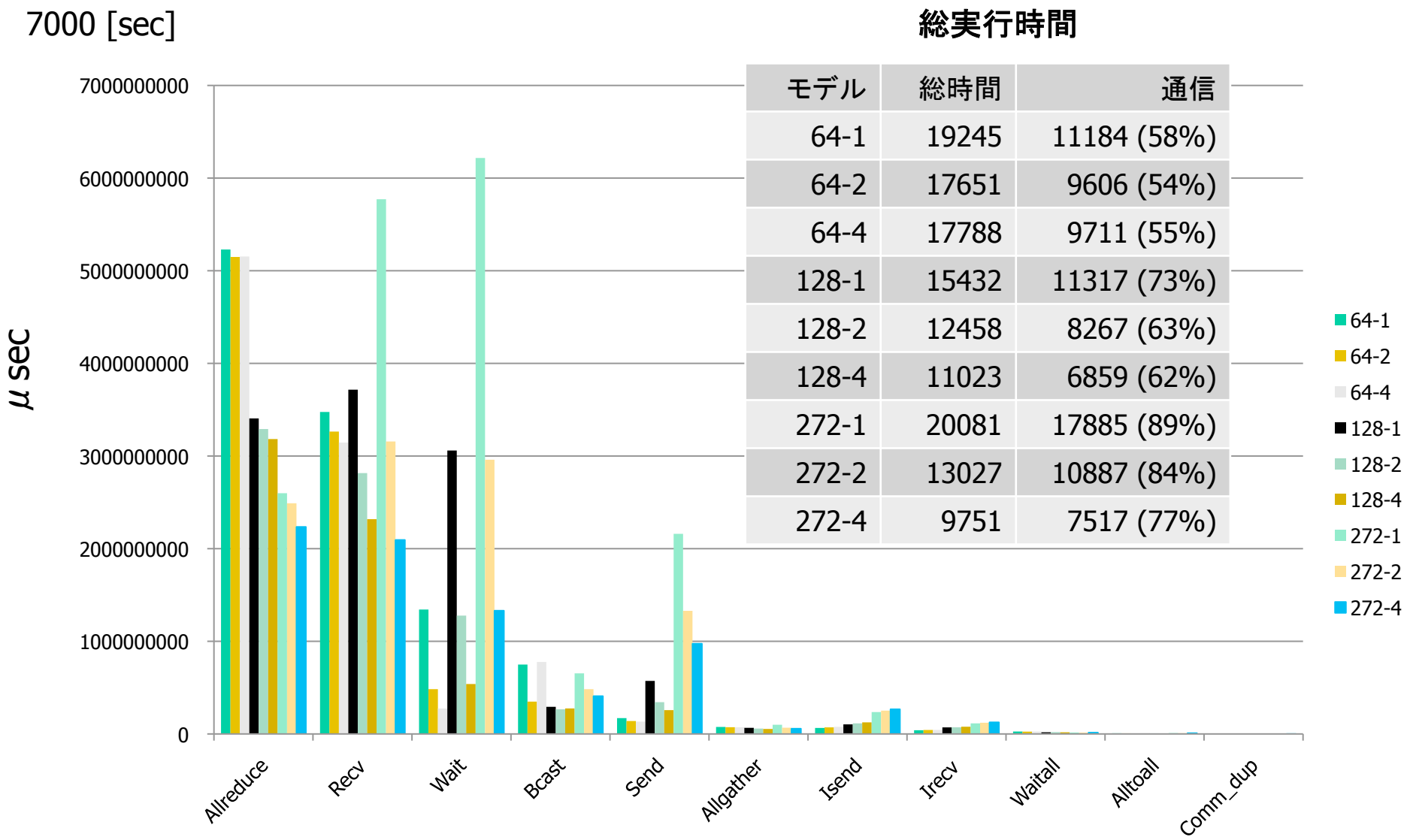
- QCDのようなmemory bandwidth intensiveなアプリケーションでは、有効利用可能なcore数の限界が示された
- memory check daemonのようなプロセスが定常的に存在する場合、余剰coreを設けない場合、かえって性能が低下する現象を確認
- PAPIを用いたバンド幅予測は有効

- 今後、PAPIに基づく性能（バンド幅）プロファイルにより、core数を自動調整するようなシステムを開発

Multi-rail trunkingの性能評価

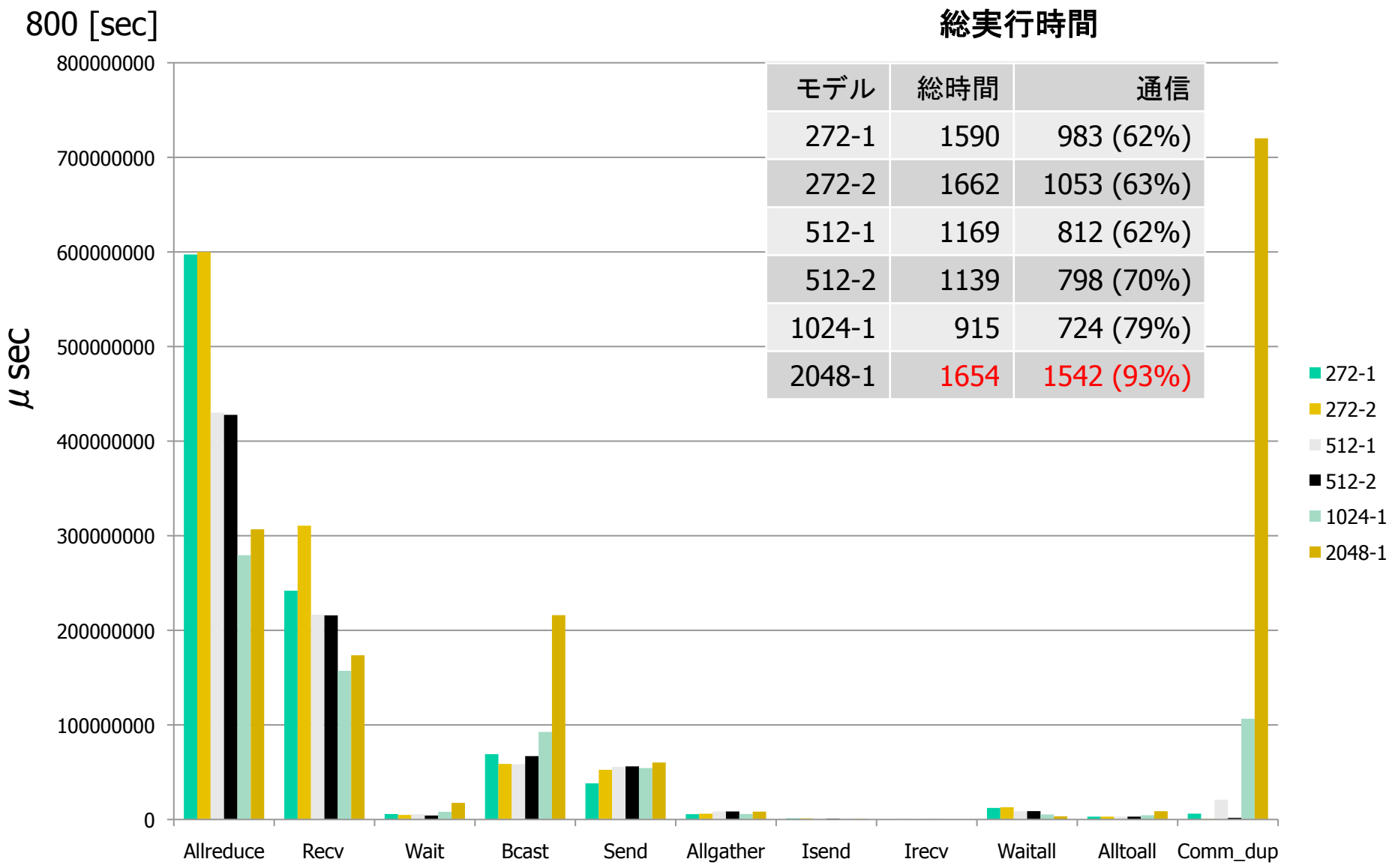
- プロファイリングに基づく性能解析のためのツール開発
- MPI-PROF (MPI profiler)
  - 既存のMPE等は限定的領域の詳細解析に向いているが、memory footprint が大き過ぎて全体実行の解析に向かない
  - light-weight なMPI通信解析を行うツール MPI-PROFを開発
  - 全てのevent logを記録せず、message length, latency, 通信相手等の平均、分布のみを収集⇒ヒストグラム化
  - MPI_Wtimeよりも高い精度
    - 複数の方法で time stamp を取得、可能であれば clock counterで
- アプリケーションコードに手を入れず、通信ライブラリのみで解決
- 商用アプリ（バイナリのみ提供）“LS-DYNA” に適用

MITSUBISHIモデル(top10 + MPI_Comm_dup)



モデル	総時間	通信
64-1	19245	11184 (58%)
64-2	17651	9606 (54%)
64-4	17788	9711 (55%)
128-1	15432	11317 (73%)
128-2	12458	8267 (63%)
128-4	11023	6859 (62%)
272-1	20081	17885 (89%)
272-2	13027	10887 (84%)
272-4	9751	7517 (77%)

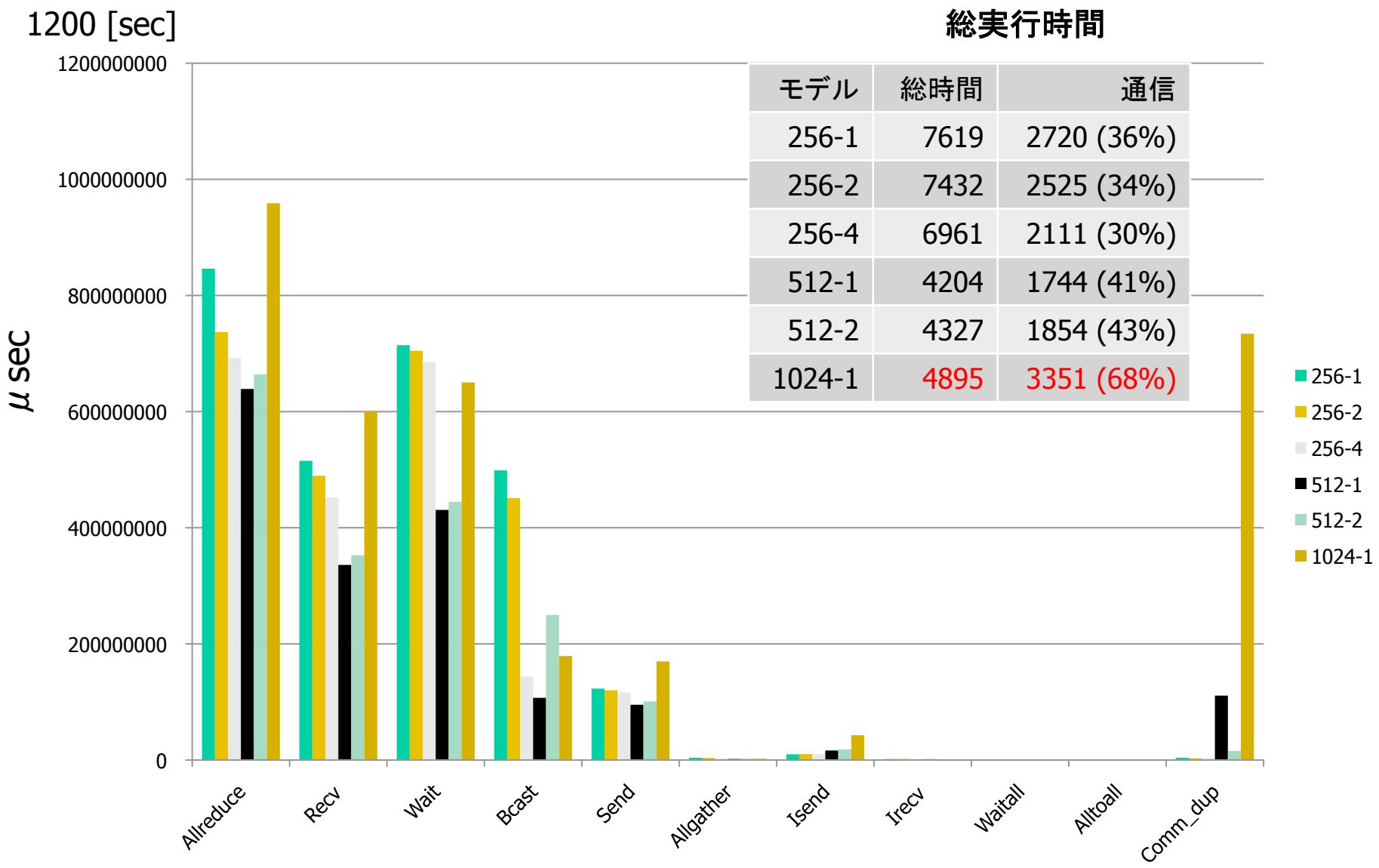
HONDAモデル(top10 + MPI_Comm_dup)



総実行時間

モデル	総時間	通信
272-1	1590	983 (62%)
272-2	1662	1053 (63%)
512-1	1169	812 (62%)
512-2	1139	798 (70%)
1024-1	915	724 (79%)
2048-1	1654	1542 (93%)

TOYOTAモデル（top10 + MPI_Comm_dup)



モデル	総時間	通信
256-1	7619	2720 (36%)
256-2	7432	2525 (34%)
256-4	6961	2111 (30%)
512-1	4204	1744 (41%)
512-2	4327	1854 (43%)
1024-1	4895	3351 (68%)

Message size distribution in NICAM

```
EVENT  2: Isend
  #MESSAGE SIZE HISTOGRAM
  total size   : 76259984
  count        : 24420
  average size : 3122
  max size     : 51072
  min size     : 24
  mode         : histogram[ 9]=7147                ( 29)
(#:141)
     1-|
     2-|
     4-|
     8-|
    16-|#
    32-|
    64-|#
   128-|##
   256-|##########################################
   512-|#####################################################
  1024-|########################
  2048-|########################
  4096-|###################
  8192-|#########
 16384-|####
 32768-|#
 65536-|
131072-|
```

```
EVENT  6: Bcast
  #MESSAGE SIZE HISTOGRAM
  total size   : 2161960
  count        : 155674
  average size : 13
  max size     : 648
  min size     : 4
  mode         : histogram[ 3]=151823     ( 97)
(#:2977)
     1-|
     2-|
     4-|#
     8-|#####################################################
    16-|#
    32-|#
    64-|#
   128-|
   256-|#
   512-|#
  1024-|
  2048-|
  4096-|
  8192-|
 16384-|
 32768-|
 65536-|
131072-|
```

通信プロファイルについて

- 実アプリケーションにおいて、multi-rail trunking が効く場合とそうでない場合の評価はできた
- MPI-PROFの結果に基づき
  - 単純なpoint-to-point通信については平均メッセージ長によるtrunk数調整は可能
  - collective通信に関しては振る舞いが不安定であり、trunk数予測は必ずしも正確でない
- LS-DYNAの大規模化（最大2048 MPI proc.）では通信時間が支配的になり、スケーラビリティが不十分であることが示された
- 一部のMVAPICH関数のオーバヘッドが非常に大きいことが明らかに
  ⇒ 問題として取り上げ中

まとめ

- multi-core/multi-lane clusterにおいて、メモリバンド幅と通信性能を、PAPI & MPI-PROF で効率的にプロファイル可能な環境を構築
  ⇒ T2K-Tsukubaで利用可能に
- 現状ではプロファイリングまで
  ⇒ 今後、自動チューニングまでを回すような実行システムを開発
- QCD, LS-DYNAのような典型的計算科学／計算工学アプリケーションに対し、ユーザに負担をかけずにプロファイル＋自動チューンイングを行うことが理論的に可能
- MPI-PROFについては今後、T2K-Tsukuba上の一般のアプリケーションにおけるジョブ情報付加機能等を検討
- multi-lane の動的制御まで展開（？）