Gfarm Symposium 2016
2016年12月9日東京

Gfarmファイルシステムの概要と最新機能

建部修見
筑波大学

Gfarmファイルシステム

- 2000年より研究開発
  - 国際会議SC03、SC05、SC06で受賞
- オープンソース広域分散ファイルシステム
  - http://oss-tsukuba.org/software/gfarm/
  - 18,000 downloads since March, 2007
- サポート
  - NPO法人つくばOSS技術支援センター（日本他）
  - Libre Solutions Pty Ltd（オーストラリア）
- 特徴
  - 性能・容量がスケールアウト
    - データアクセス局所性、ファイル複製
  - 単一障害点なし
    - 複製数維持機能,、ホットスタンバイMDSサーバ
  - 無停止で拡張、更新可能
  - データ完全性を保証しサイレントデータ損傷も対応可

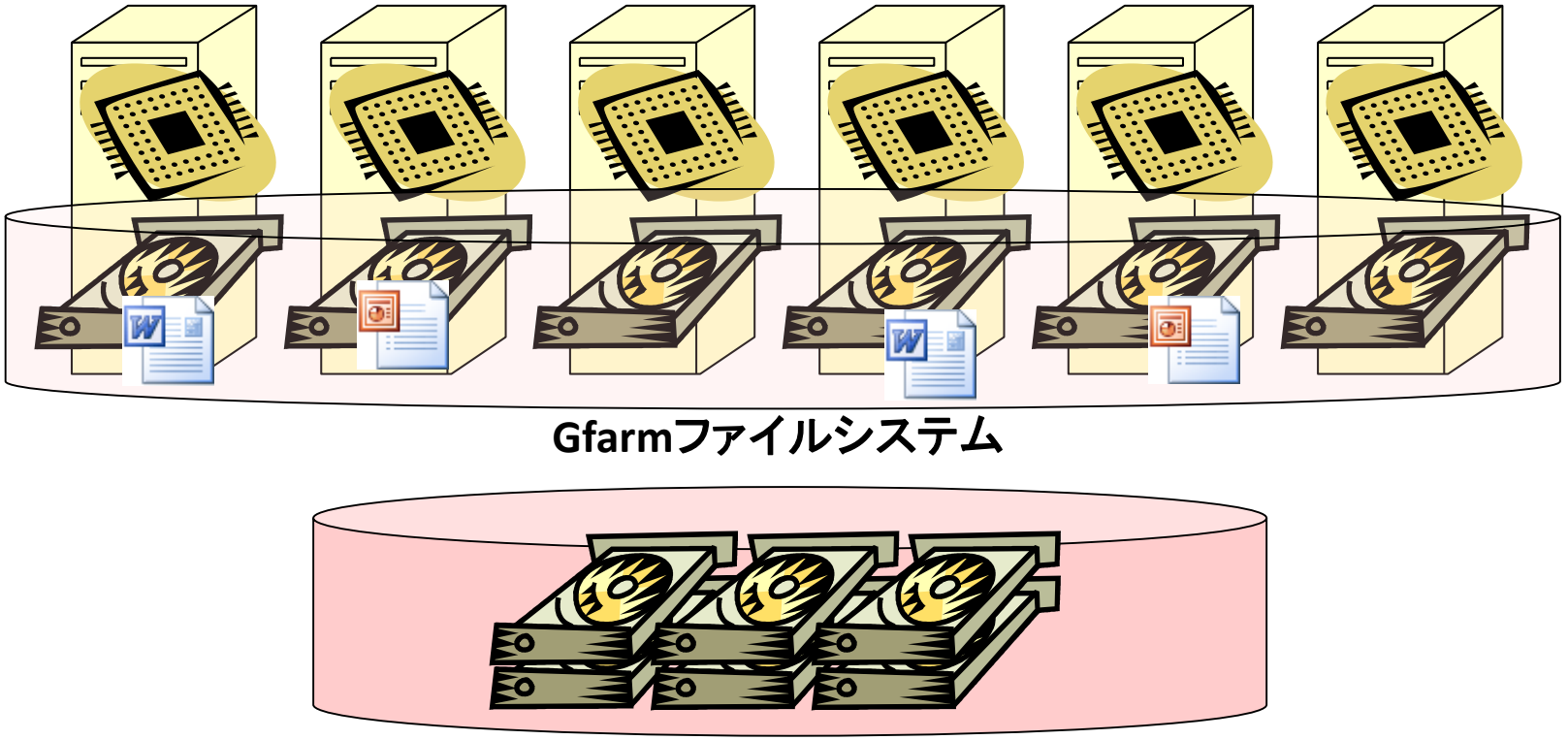
oss-tsukuba.org

Gfarmファイルシステム（2）

- JLDG（7PB、8拠点）、HPCI共用ストレージ（22.5PB、3拠点）、NICTサイエンスクラウド、（株）クオリティアActive! worldで実運用
- 計算ノードのローカルディスクを利用するデータ解析
- Pwrakeワークフローシステム、MapReduce、MPI-IO、バッチキューイングシステム
  - データ局所性awareなプロセススケジューリング
  - ディスクキャッシュawareなプロセススケジューリング
  - データ局所性を高めるファイル複製作成

Gfarmファイルシステムの構成

- 一般的なPCのローカルディスクを束ねる
- ユーザには、共有ファイルシステムとしてみえる
- 複数のディスクに分散してデータを保持



HPCI共用ストレージ

- 大学情報基盤センターをはじめ全国からマウント可能な共有ファイルシステム（22.5PB）
- スパコン間のデータ共有、共有データ格納

西拠点（AICS 9.5PB）

東拠点（東京大、東工大 13.0PB）

最新機能・状況紹介

主なリリース

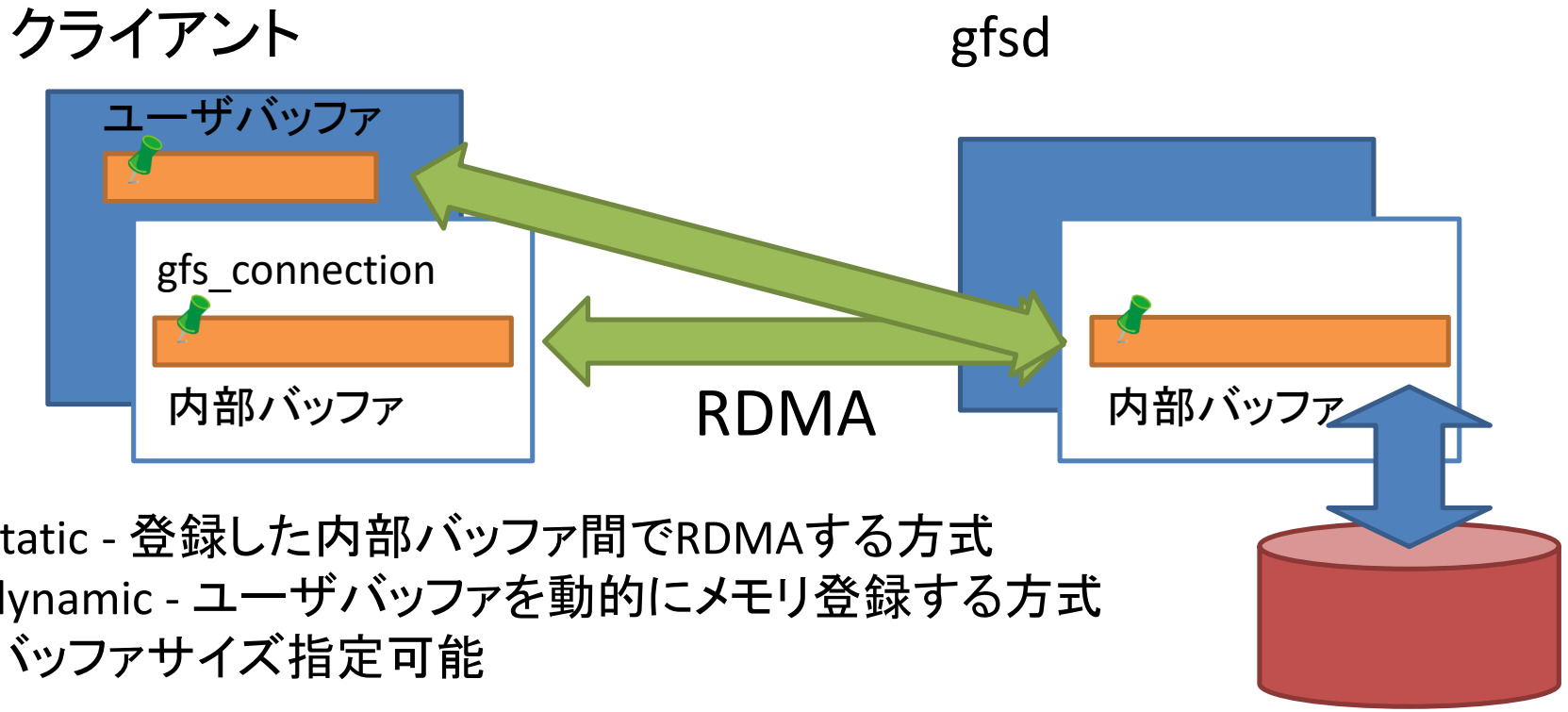
日付	version	新機能、更新機能
2016/12/8	2.7.0	• InfiniBand RDMAサポート • ディレクトリクオータ機能
2016/12/3	2.6.15	• OpenSSL 1.1.0サポート
2016/10/26	2.6.14	• gflsofコマンド – オープン中のファイル表示
2016/1/16	2.6.8	• 書込み後ベリファイ機能
2015/6/25	2.6.5	• 過剰複製削除
2014/12/17	2.6.0	• End-to-endデーター貫性サポート • クライアント透過なフェイルオーバ

Gfarm 2.7.0の新機能

- InfiniBand RDMAサポート
  - IB VerbsによるRDMA転送
- ディレクトリクオータ機能

InfiniBand RDMAによるアクセス

- ファイルアクセスをRDMA（読込はRemote Write, 書込みはRemote Read）で行う



static - 登録した内部バッファ間でRDMAする方式
dynamic - ユーザバッファを動的にメモリ登録する方式
バッファサイズ指定可能

- RDMA通信に失敗したとき、IP通信へフォールバック

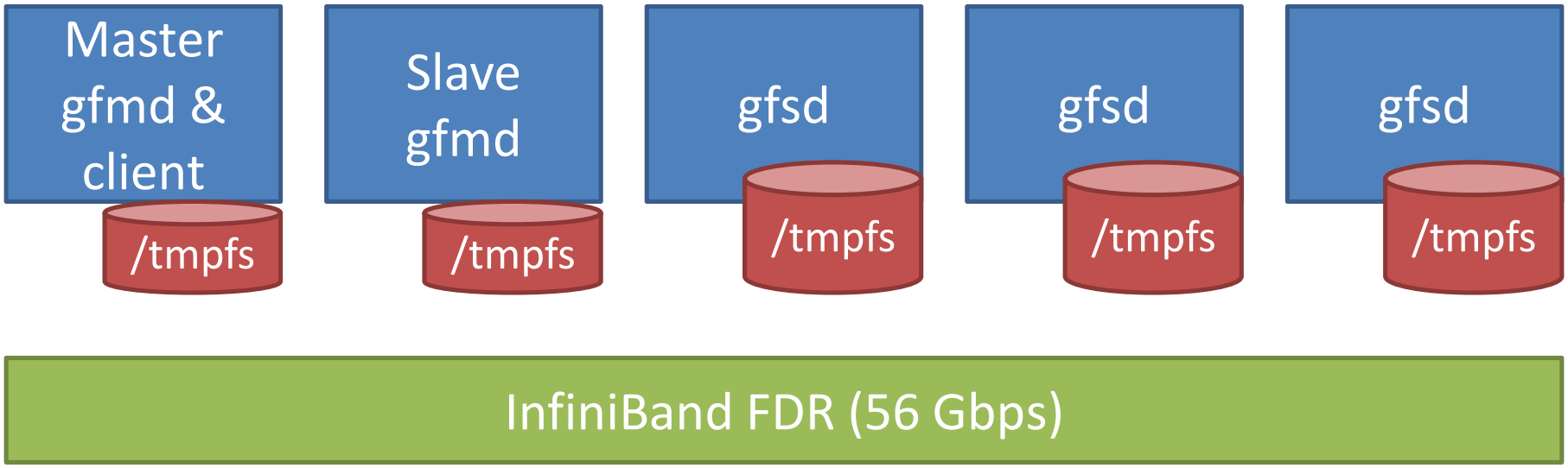
設定ファイル（１）

- ib_rdma
  - enable – configure --with-infinibandした場合のデフォルト
  - disable – RDMAを利用しない場合
- rdma_device_name
  - デバイス名の指定（オプション）
- rdma_port
  - ポート番号の指定（オプション）
- rdma_min_size
  - RDMA転送を行う最小サイズ（デフォルトは65536バイト）

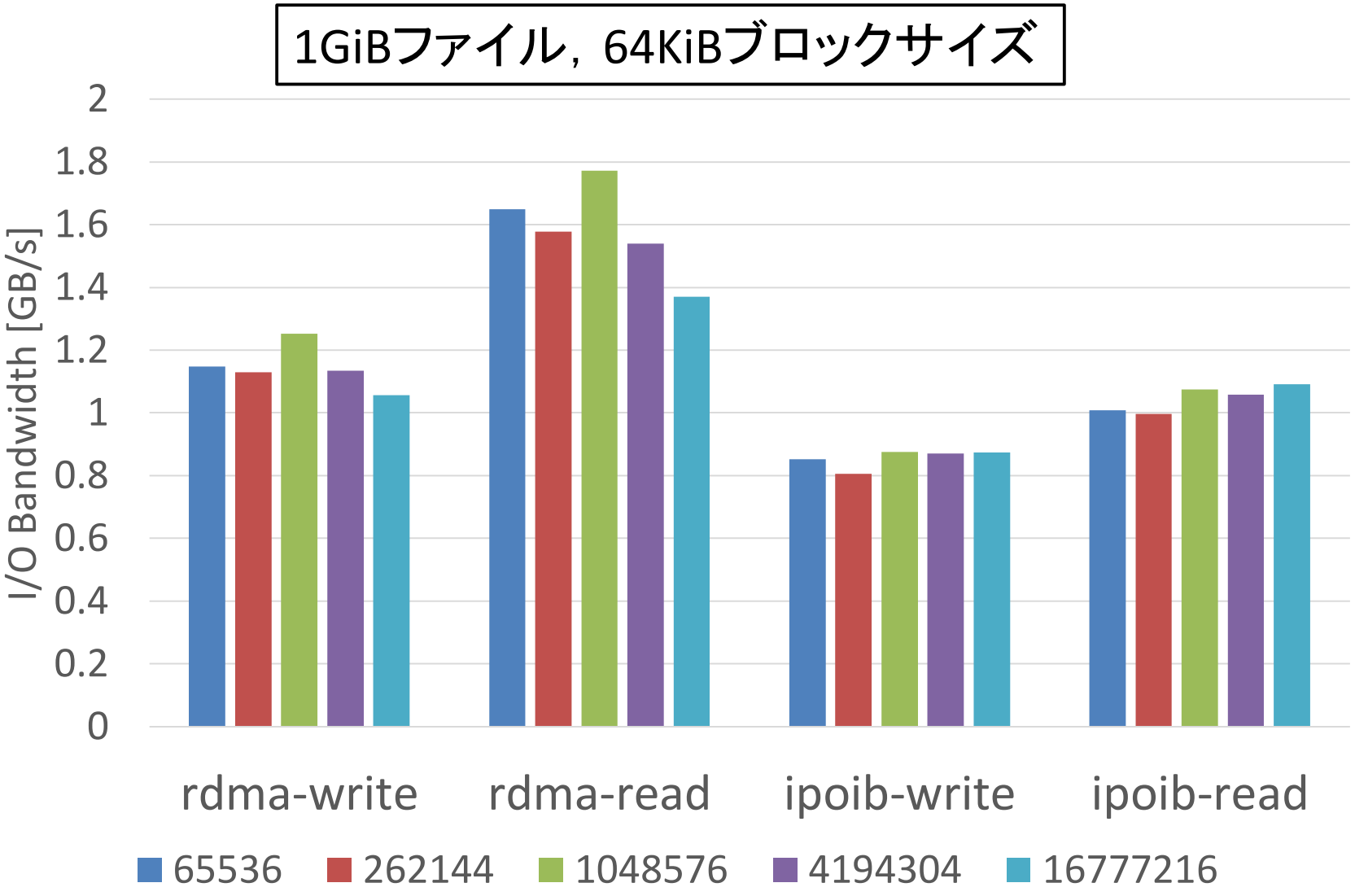
設定ファイル（２）

- rdma_mr_reg_mode
  - static - RDMA用バッファを接続毎に静的に確保
  - dynamic - RDMA毎にバッファを動的にピンダウン
  - static dynamic – RDMA用バッファを確保し、バッファ以上の転送は動的にピンダウン
- rdma_mr_reg_static_min_size
  - 静的確保するバッファの最小サイズ（デフォルトは1MiB）
- rdma_mr_reg_static_max_size
  - 静的確保するバッファの最大サイズ（デフォルトは16MiB）

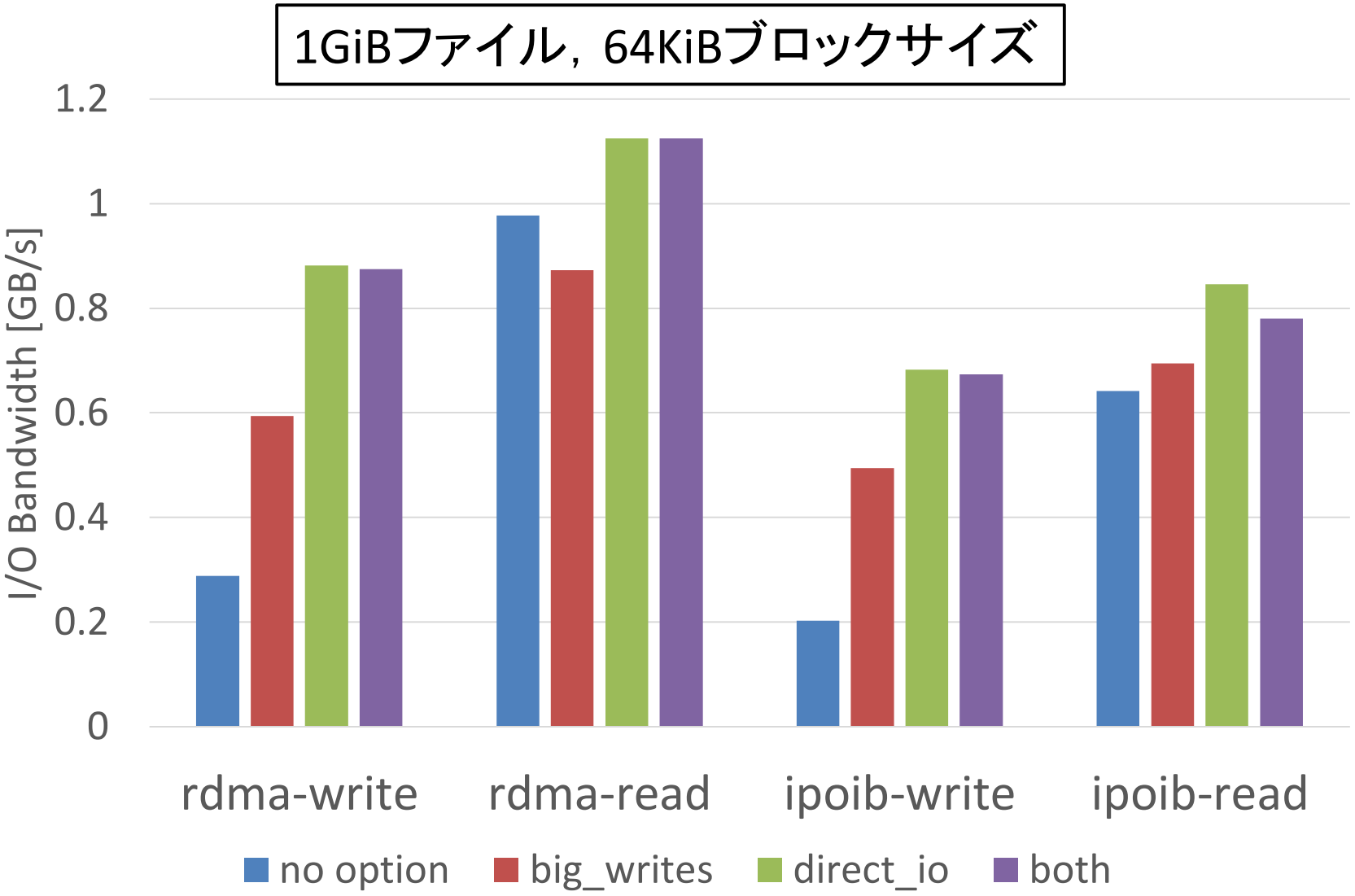
評価環境


Master
gfmd &
client
/tmpfs
Slave
gfmd
/tmpfs
gfsd
/tmpfs
gfsd
/tmpfs
gfsd
/tmpfs
InfiniBand FDR (56 Gbps)

Gfarm APIによるファイルアクセス性能



POSIX APIによるファイルアクセス性能



ディレクトリクオータ機能（１）

- ディレクトリ単位でファイル数、利用サイズの制限をユーザ権限で行う

- gfdirquota – ディレクトリセット作成、設定
- gfedquota, gfquota – クオータ設定、表示

gfdirquotaコマンド

- ディレクトリセット表示

```
% gfdirquota [-l] [-A [-u username]]
```

  - -Aは全ユーザのディレクトリセットを表示（要gfarmroot権限）
  - -lはディレクトリセットと登録されているディレクトリを表示

- ディレクトリセット作成

```
% gfdirquota -c ディレクトリセット
```

- ディレクトリセットにディレクトリを登録

```
% gfdirquota -a ディレクトリセット ディレクトリ ...
```

- ディレクトリセット削除

```
% gfdirquota -d ディレクトリセット
```

```
% gfdirquota -c projectX
% gfdirquota -a projectX /work/projectX
```

gfedquota，gfquotaコマンド

- オプションでディレクトリセット、ディレクトリを指定
  - -D ディレクトリセット
  - -d ディレクトリ

```
% gfedquota –D projectX –H 1000000
                                    # ハードリミット設定
% gfquota –D projectX
```

ディレクトリクオータ機能（２）

- 異なるディレクトリセット間でのディレクトリ移動はできない
- 登録するディレクトリは空ディレクトリ
  - いずれも管理者は可能だがその間Gfarmはロックされる（メンテナンスなどに行うことを想定）
- 異なるディレクトリセット間でハードリンクはできない
- ディレクトリセットからディレクトリを除くためにはそのディレクトリを削除する

設定ファイル

- directory_quota_count_per_user_limit
  - ユーザ毎の最大ディレクトリセット数（デフォルトは100）

データ完全性（1）

- サイレントデータ障害の検知
- digestを書込時に計算しメタデータに保存
- 逐次書込（読込）時にgfsdでdigest検査
  - 破損ファイルは読込時にEIO (checksum error)を返し、読込失敗。lost+foundへ移動させ自動修復
- ファイル複製作成時のdigest検査
- 書込み後ベリファイによるdigest検査
- クライアントからのEnd-to-endのデータ完全性

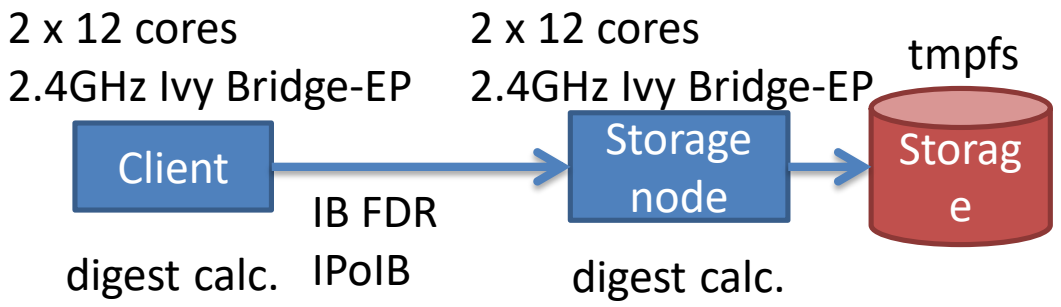
データ完全性（２）

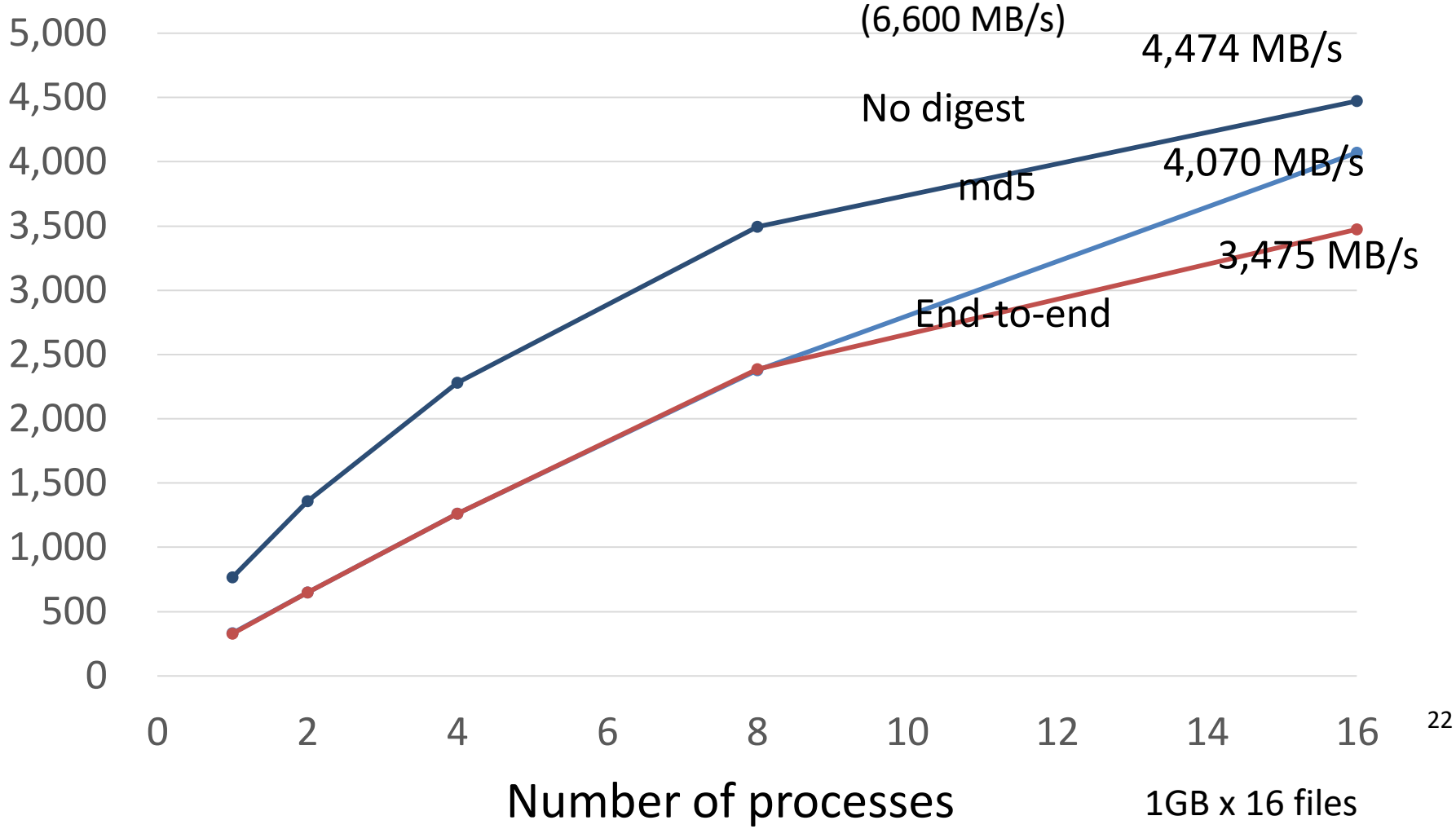
- データ完全性の保証
  digest *digest_type*　　// gfmd.conf
  md5, sha1, sha256, …をサポート
- End-to-endデータ完全性の保証
  client_digest_check enable　　// gfarm2.conf

% **gfcksum** *file*
  – ファイルのdigestを表示
% **gfcksum** *-c [-h host] file*
  – （hostに格納されている）ファイルのdigestを計算し確認

書込み性能

Write Bandwidth [MB/s]



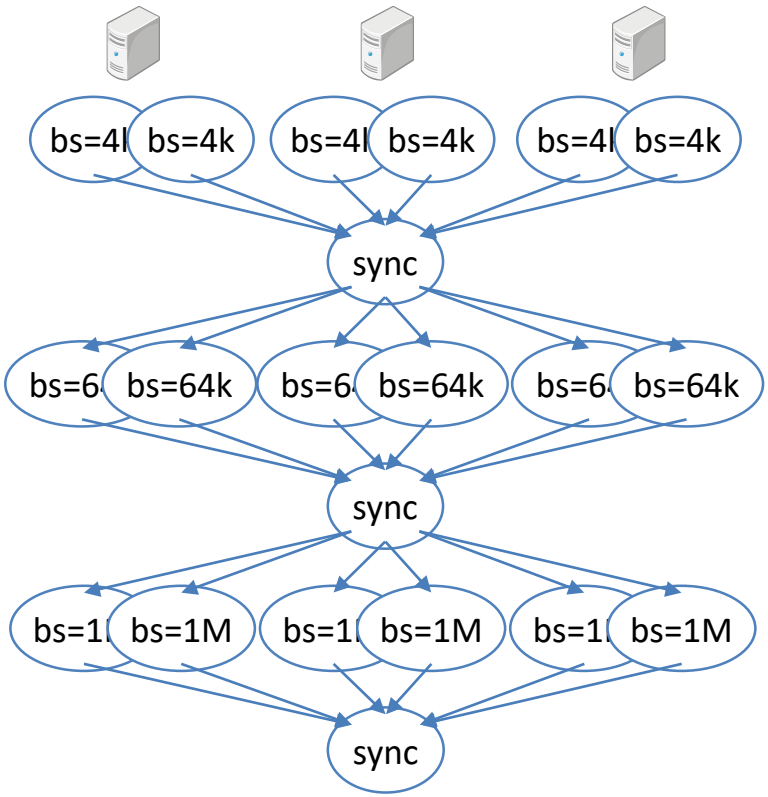


JLDGにおける運用例

- 7.8 PB, 7 拠点, 39 ファイルサーバ
  - 物理学研究者による全国規模のストレージ
  - 7.2 PB利用, 109 Mファイル
- md5によるEnd-to-end一貫性チェックと書込み後ベリファイ利用
- 2016年8月19~22日
  - 書込み後ベリファイで5ファイル、複製作成で1ファイルの損傷ファイルを検出
  - I/Oエラーは起こっていない

並列I/Oベンチマーク by Pwrake

ベンチマークDAG



ベンチマーク Rakefile の例

```
NCORE = ENV['NCORE'].to_i
PARAMS = ENV['PARAMS'].split
SH = ENV['SH']

sync = "sync_begin"
task sync

PARAMS.each do |param|
  a = []
  NCORE.times.map do |core|
    name = "task_#{param}_#{core}"
    cmd = eval("¥"#{SH}¥"")
    task name => sync do
      sh cmd
    end
    a << name
  end
  sync = "sync_#{param}"
  task sync => a
end

task :default => sync
```

Pwrakeの起動：パラメータとコマンドを引数にセット

```
$ pwrake -F hosts NCORE=6 PARAMS="4k 64k 1M"
  SH="dd if=/dev/zero of=#{name} bs=#{param} count=1000"
```

まとめ

- Gfarmファイルシステム
  - NPO法人つくばOSS技術支援センターによるサポート
- Gfarm 2.7.0をリリース
  - InfiniBand RDMAサポート
  - ディレクトリクオータ機能
- データ完全性、サイレントデータ損傷対応
- HPCI共用ストレージ、JLDGなど実運用実績
- ビッグデータ処理、並列I/Oベンチマークツール