I/Oベンチマークと Lustre性能

DDN Storage

Shuichi Ihara

2022/12/09

I/O性能とベンチマーク

- I/O性能
  - ストレージシステム選定の１つの要件
  - I/O性能といってもその範囲は非常に多岐にわたる
- I/Oベンチマークは複雑
  - ストレージだけでなく、ネットワーク、クライアントに関連
  - 多くのベンチマークツールが存在し、I/Oワークロードを再現は手間
- ストレージシステムの性能を正しく理解
  - どのようにI/Oすればアプリケーションが速くなるか
  - 得意、不得意なI/Oは何か。どのように改善できるか

© 2022 DDN

IO500

- HPCのワークロードを想定したI/Oベンチマークスイート
  - 標準化されたベンチマークツールで簡単にベンチマークできることを想定
  - テストの種類はI/O帯域、メタデータ操作およびメタデータ検索
  - それぞれのテストに“Hero”(Easy)と”Anti-Hero”(Hard)が存在
  - 単一性能基準に依存しない
- 登録されたベンチマーク結果は公開しランキング
  - SC17(2017, 11月)が最初のリスト
  - 他のxxx500同様、リストは年２回更新
  - “Full“リストと”10クライアント”のリストが存在
  - プロダクション、テストシステム、オンプレ、クラウド等環境は問わない

© 2022 DDN

IO500

- ベンチマークはIOR, mdtestおよびfindから構成される
  - IOR Easy (WriteとRead)
    - プロセス毎のファイル、IOサイズは自由に設定可能
  - IOR Hard (WriteとRead)
    - 全プロセスが共通単一ファイルに同時アクセス(Stride IO)
    - IOサイズはページアラインされていない47,008バイト
  - MDTEST Easy (Write, Stat, Delete)
    - プロセス毎にユニークなディレクトリ、0byteのファイルサイズ
  - MDTEST Hard (Write, Stat, Read, Delete)
    - 全プロセスが共通単一ディレクトリに同時アクセス
    - ファイルサイズは3901バイト
  - Find
    - 生成された全ファイルから3901 byteだけのファイル数を検索
    - 自由なツール利用を許可
- その他
  - 書き込みはそれぞれ5分以上。作成されたファイルは、永続メディアへの保存必須
  - クライアントキャッシュは利用できない

相乗平均 Bandwidth スコア

相乗平均 IOPS(Metadata) スコア

相乗平均 合計スコア

© 2022 DDN

IO500トレンドと注意点

- IO500のスコアが高ければ、すべての性能問題が解決するわけではない
  - あくまでもHPCにおけるI/Oワークロードのいくつかをピックアップしたもの
  - それぞれのテストが同時に行われるわけでなく一つ一つ個別に実行
    - プロダクションでは、不特定多数のユーザが不特定のIOを実施
- どのようなファイルシステムでもIO500が正しく実行できれば登録申請可能
  - ストレージシステムの構成に大きく依存
    - Top100等と比較してIO性能実行効率(実測値/理論値)が分かりづらい
  - 聞きなれないファイルシステムも多く存在
- IO500のアイデアを利用できるところは多数存在
  - 自分のワークロードに当てはめてみる

    例) 単一の共有ファイルに対して多くのプロセスから同時書き込み、読み込み

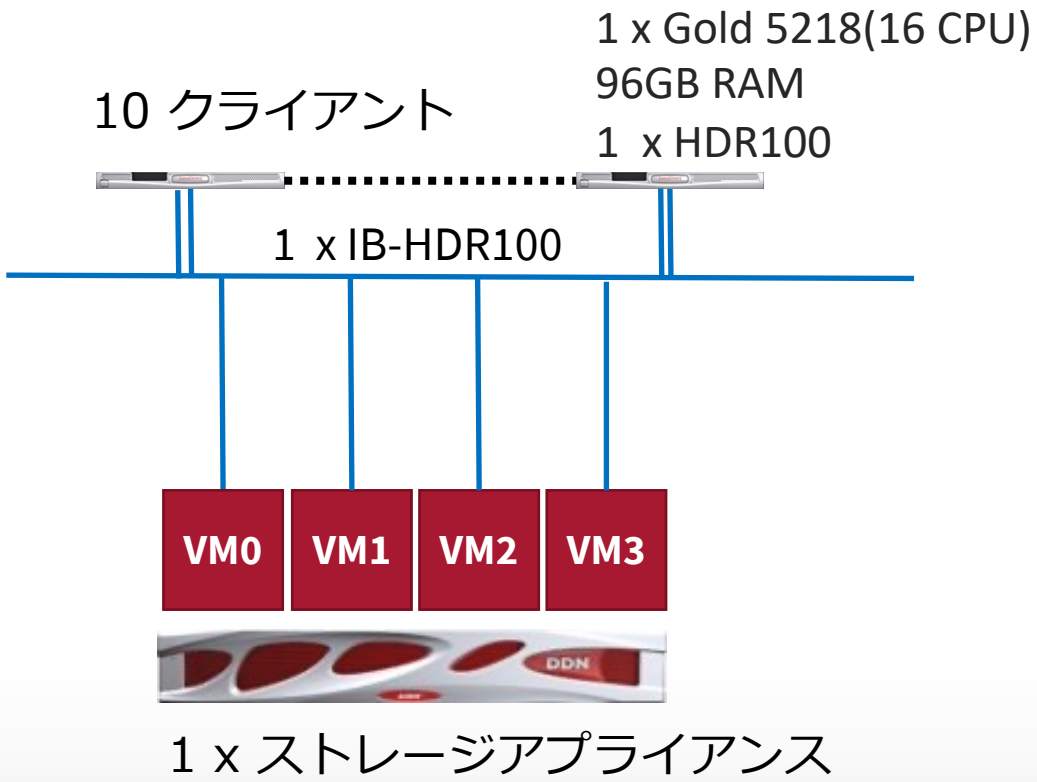
    ただしIOサイズは47008バイトではなく1Mバイト

© 2022 DDN

IO500をどう活用するか？

- 一貫性のあるベンチマーク
  - 構成を最小化し持続的ベンチマーク
    - 性能のリグレッションテスト
    - 性能課題の明確化
    - 性能改善のデモンストレーション、製品反映
- 性能効率の改善
  - 少ないハードウェアリソースで高いI/O性能
  - HPC以外のワークロードにも有用
- IO500を通じてI/O性能の理解
  - ボトルネックの特定
  - リアルな性能問題への適応
  - I/Oトレーサビリティ

IO500 ベンチマーク基本構成



© 2022 DDN

継続的な性能改善による成果

プラットフォーム	ES400NV	ES400NVX	ES400NVX2
	4 x Lustre Server(VM) • 8 x CPU, 150GB RAM • PCI Gen3, 20 x NVMe	10 x CPU	12 x CPU PCI Gen4, 200Gbps

	PreSC19	SC19	ISC20	ISC22	SC22
IOR Easy Write	25.88	28.62	37.56	55.95	58.07
IOR Easy Read	39.94	41.72	45.95	83.86	77.56
IOR Hard Write	2.78	2.96	2.77	5.02	5.27
IOR Hard Read	8.99	42.19	40.81	39.73	49.36
Find	1,735.41	810	1,698.00	6,248.55	12628.78
Mdtest Easy Write	143.88	152.84	157.22	270.04	312.9
Mdtest Easy Stat	455.03	451.97	453.51	740.01	1,278.50
Mdtest Easy Delete	88.52	132.76	135.09	223.61	272.64
Mdtest Hard Write	32.33	79.65	90.47	119.41	157.4
Mdtest hard Read	44.92	172.59	169	194.33	238.82
Mdtest Hard Stat	20.41	449.93	446.75	514.36	1,214.03
Mdtest Hard Delete	16.35	75.15	76.94	101.98	122.44
Bandwdith	12.68	19.65	21.02	31.10	32.90
IOPS	91.41	207.62	232.69	368.48	544.23
Score	34.05	63.87	69.93	107.05	133.81

© 2022 DDN

性能改善およびチューニング(1)

- Lustreの機能強化によるもの
  - ReadAhead for StrideRead (Lustre-2.14/EXA5.1)
    - Single shared fileおよびStrideReadの性能改善
    - ior-hard-read
  - Lustre OverStripe (Lustre-2.13/EXA5.1)
    - Single shared fileの大幅な性能改善
    - ior-hard-write
  - Batched RPCs for Statahead(Lustre-2.16)
    - 連続したstat()ワークロード性能改善
    - 全体的なnetworkレイテンシーの大幅な削減
    - mdtest-{easy,hard}-stat

© 2022 DDN

性能改善およびチューニング(2)

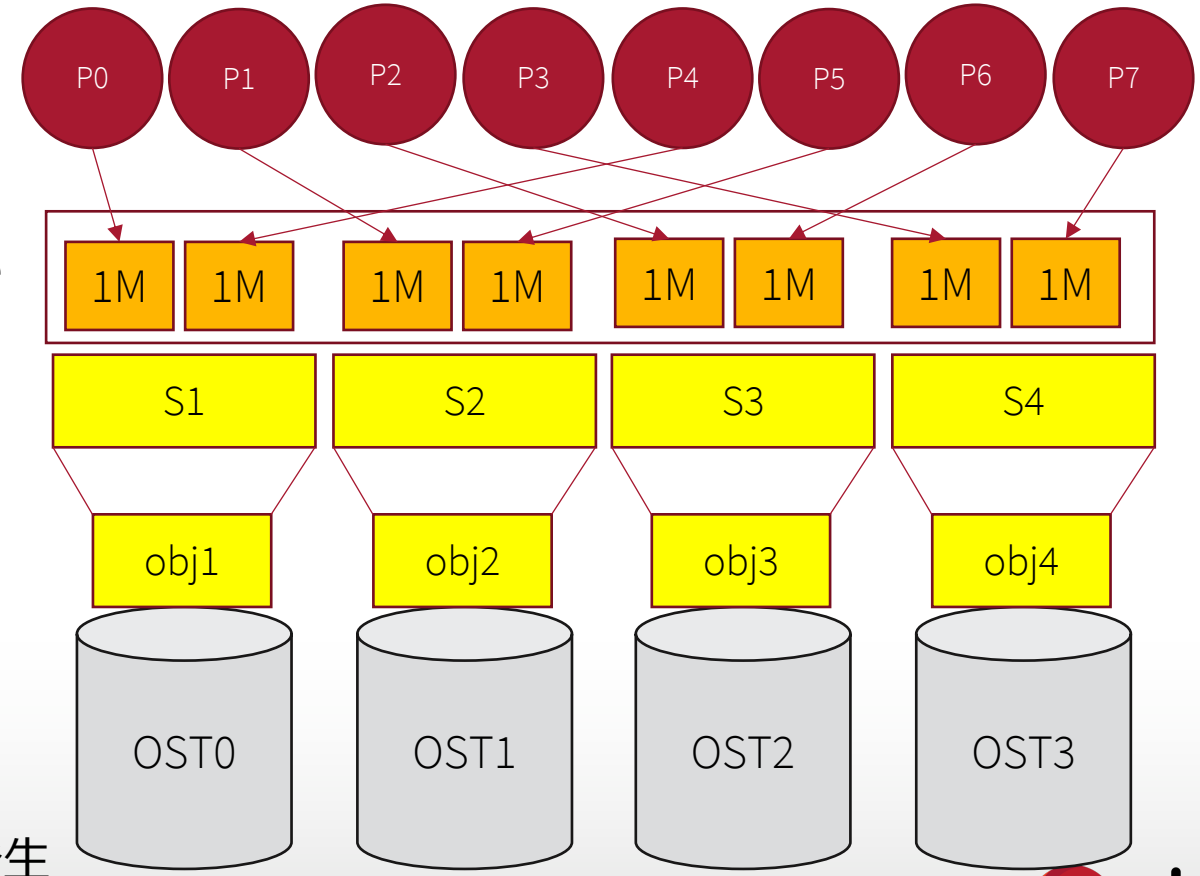
- その他
  - Linux Kernel
    - VFS Parallel Lookup (kenrel-4.7以降でサポート)
      - mdtest-hard-statの性能改善
    - VFS Parallel Updatesの制限
      - クライアント側に１つのファイルシステムに対して複数のマウントポイントを設定
      - Singularityを利用してMPIのランク毎にマウントポイントをラウンドロビン
      - mdtest-hard-{write,delete}の改善
  - メタデータスキャン
    - findコマンドの代替として利用
    - メタデータデバイスを直接スキャン
    - findの性能を20倍向上

© 2022 DDN

Lustre Striping

```
root@ubuntu1804-1:~# lfs setstripe -c -1 /ai200x1/shared-file
root@ubuntu1804-1:~# lfs getstripe /ai200x1/shared-file
/ai200x1/sharedfile
lmm_stripe_count:  4
lmm_stripe_size:   1048576
lmm_pattern:       raid0
lmm_layout_gen:    0
lmm_stripe_offset: 1
obdidx objid objid group
     1             2            0x2             0
     3             2            0x2             0
     0             2            0x2             0
     2             2            0x2             0
```

(”-c -1”は全てのOST)

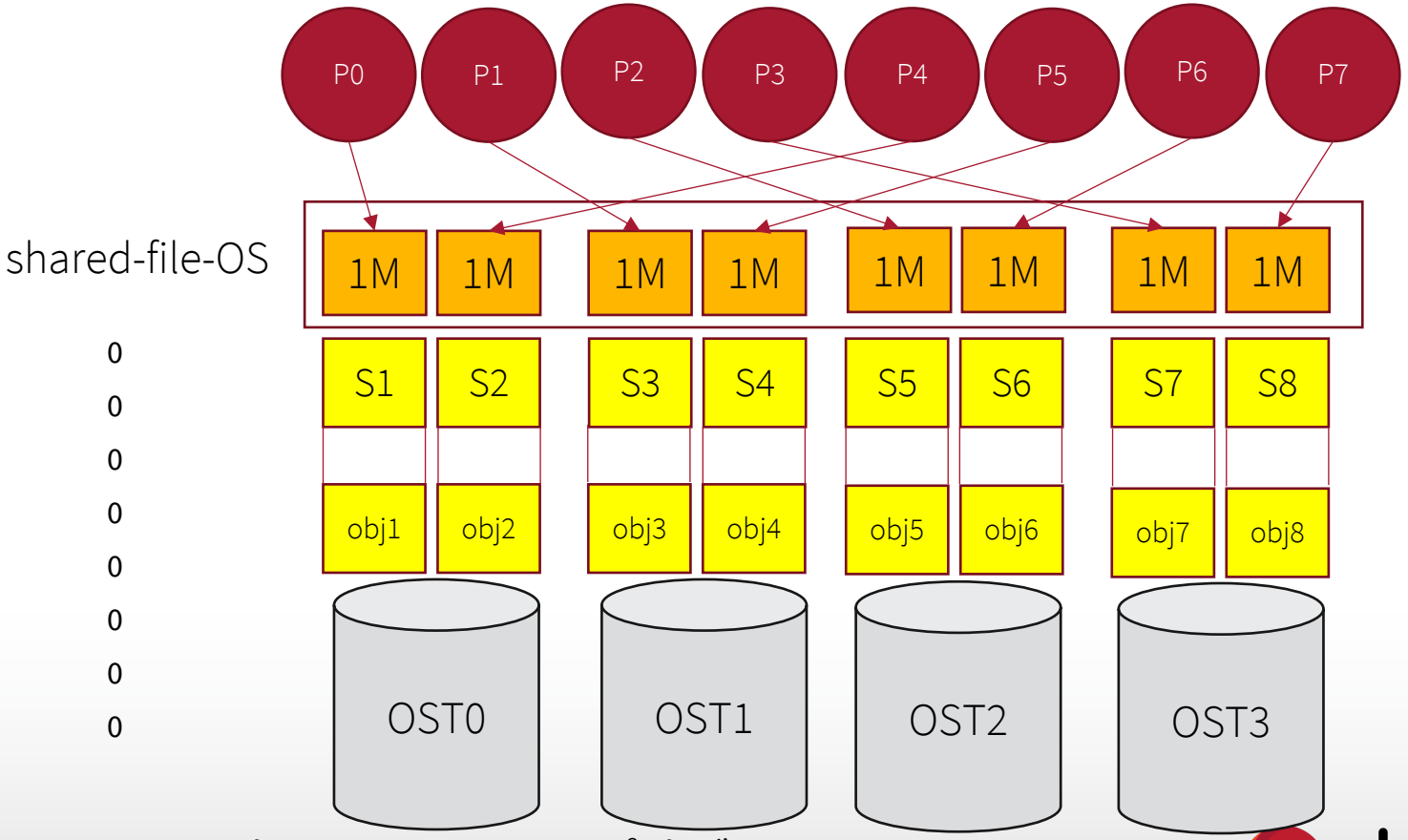


プロセス数 > ストライプ数の場合
OSTオブジェクトに対する競合が発生

© 2022 DDN

Lustre OverStriping (Lustre-2.13/EXA-5.1)

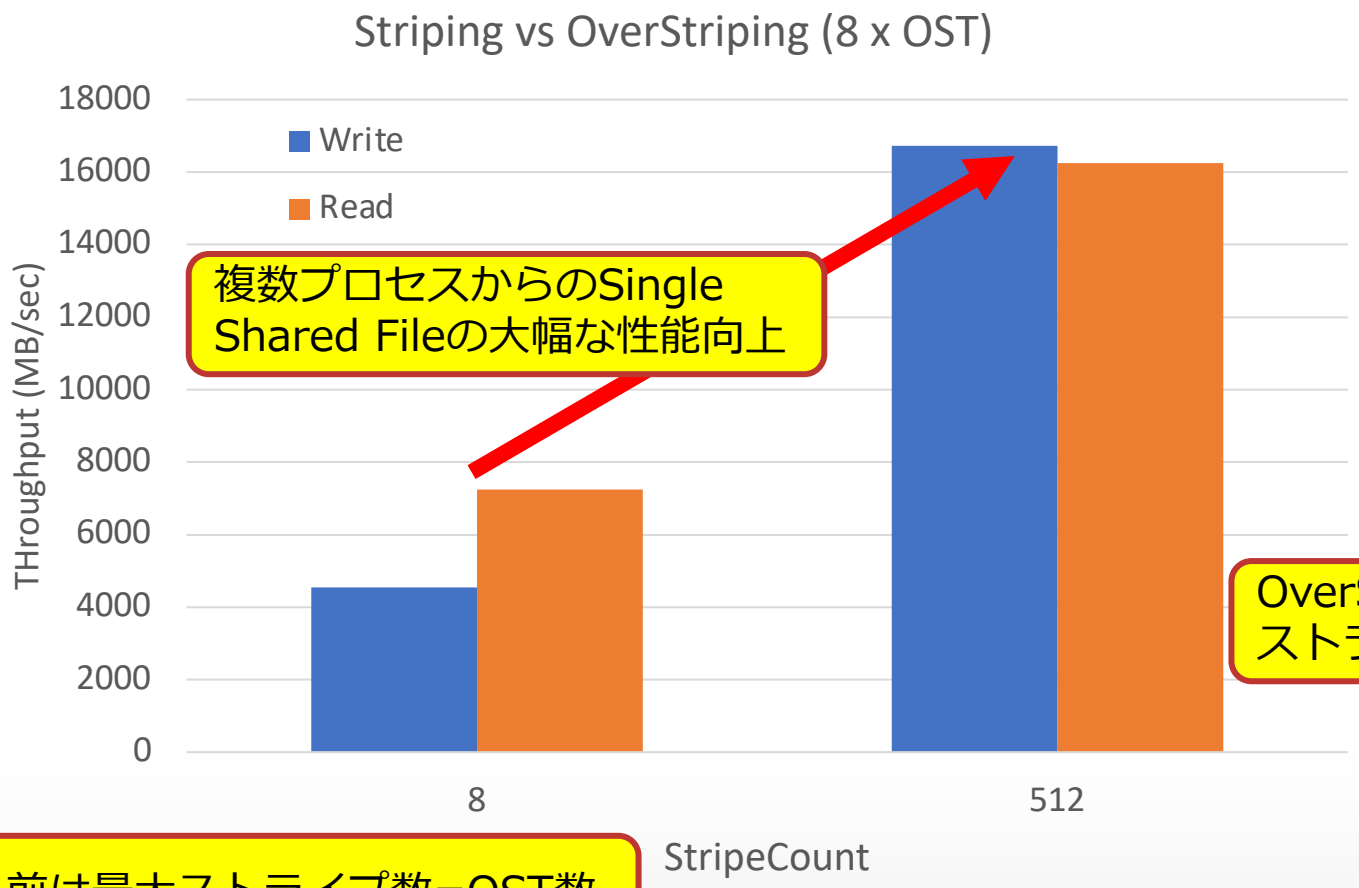
```
root@ubuntu1804-1:~# lfs setstripe -C 8 /ai200x1/shared-file-OS
root@ubuntu1804-1:~# lfs getstripe /ai200x1/shared-file-OS
/ai200x1/shared-file-OS
lmm_stripe_count:  8
lmm_stripe_size:   1048576
lmm_pattern:       raid0,overstriped
lmm_layout_gen:    0
lmm_stripe_offset: 0
obdidx objid objid group
     0             4           0x4             0
     2             4           0x4             0
     1             4           0x4             0
     3             4           0x4             0
     0             5           0x5             0
     2             5           0x5             0
     1             5           0x5             0
     3             5           0x5             0
```



© 2022 DDN

OverStriping機能によってOST数以上のストライプ数が
作成可能になりオブジェクト内における競合を排除

Striping vs OverStriping 性能比較



複数プロセスからのSingle Shared Fileの大幅な性能向上

OverStripeではOST数以上にストライプ数を設定可能

以前は最大ストライプ数=OST数

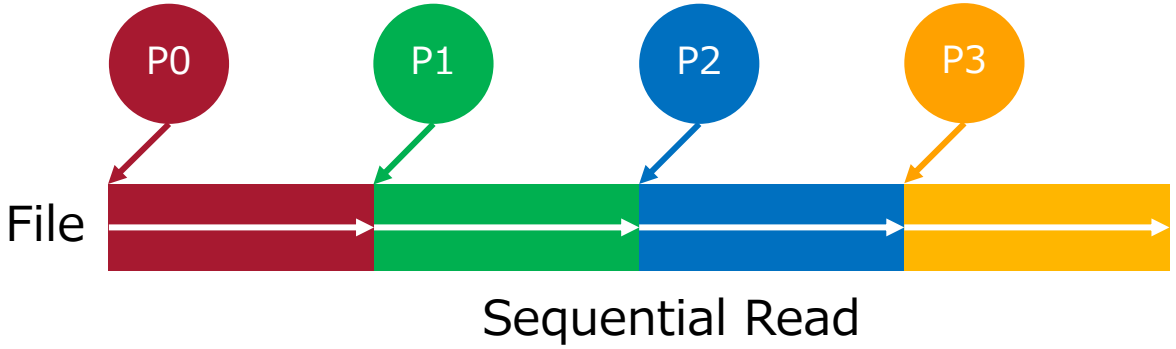
- ES7990(160 x HDD, 2 x OSS, 8 x OST)
- 32クライアント, 512プロセス
- 1MB, SingleShared File

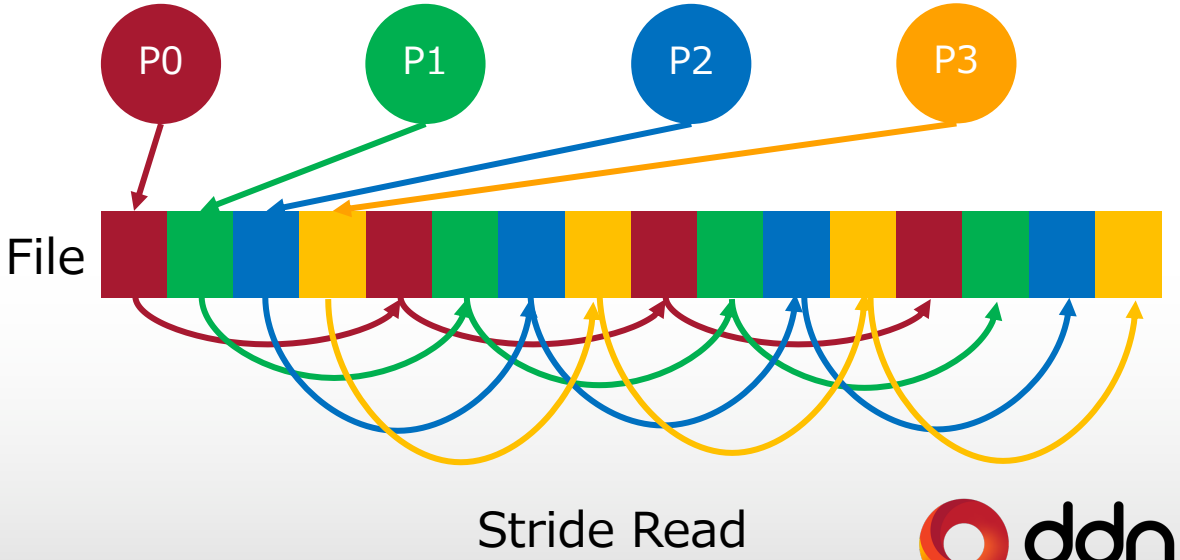
  # ior/src/ior -w -r -C -g -i 3 -vv -s 13000 -b 1m -t 1m -a POSIX -e
- ストライプカウント8と512で比較

© 2022 DDN

Lustre ReadAheadの改善(Lustre-2.14/EXA5.1)

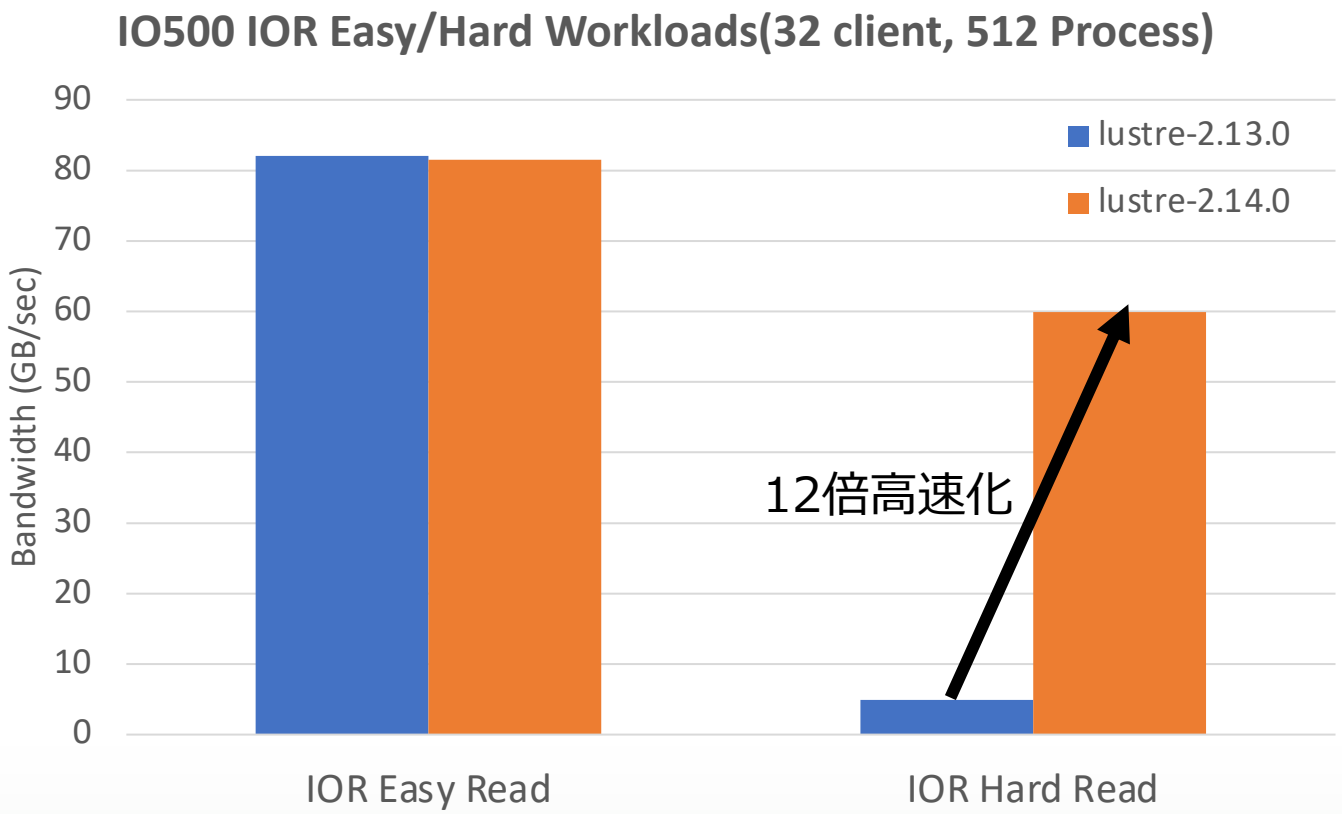
- Lustre ReadAheadの効率向上
  - IOパターンを正しく検知
  - Readahead Windowのリセットの回避
    - Cache Missを減らす
    - ReadするまえのCacheの破棄を減らす
- Stride Readパターンをサポート
  - これまでSequential Readをサポート
- Unaligned ReadAahedサポート
  - ReadAheadをページ単位からByte単位へ変更
  - ページサイズにアラインしていないIOサイズ(例47008byte)でもCacheMissしない





© 2022 DDN

Lustre ReadAheadの改善(Lustre-2.14/EXA5.2.3)



Lustre-2.13

```
# lctl get_param llite.*.read_ahead_stats
llite.exafs-ffff9b96c1349800.read_ahead_stats=
hits                              3340631 samples [pages]
misses                            32901120 samples [pages]
```

Readahedキャッシュヒット率: 9%

Lustre-2.14

```
llite.exafs-ffff9b96b8117000.read_ahead_stats=
hits                              33616605 samples [pages]
misses                            4444696 samples [pages]
```

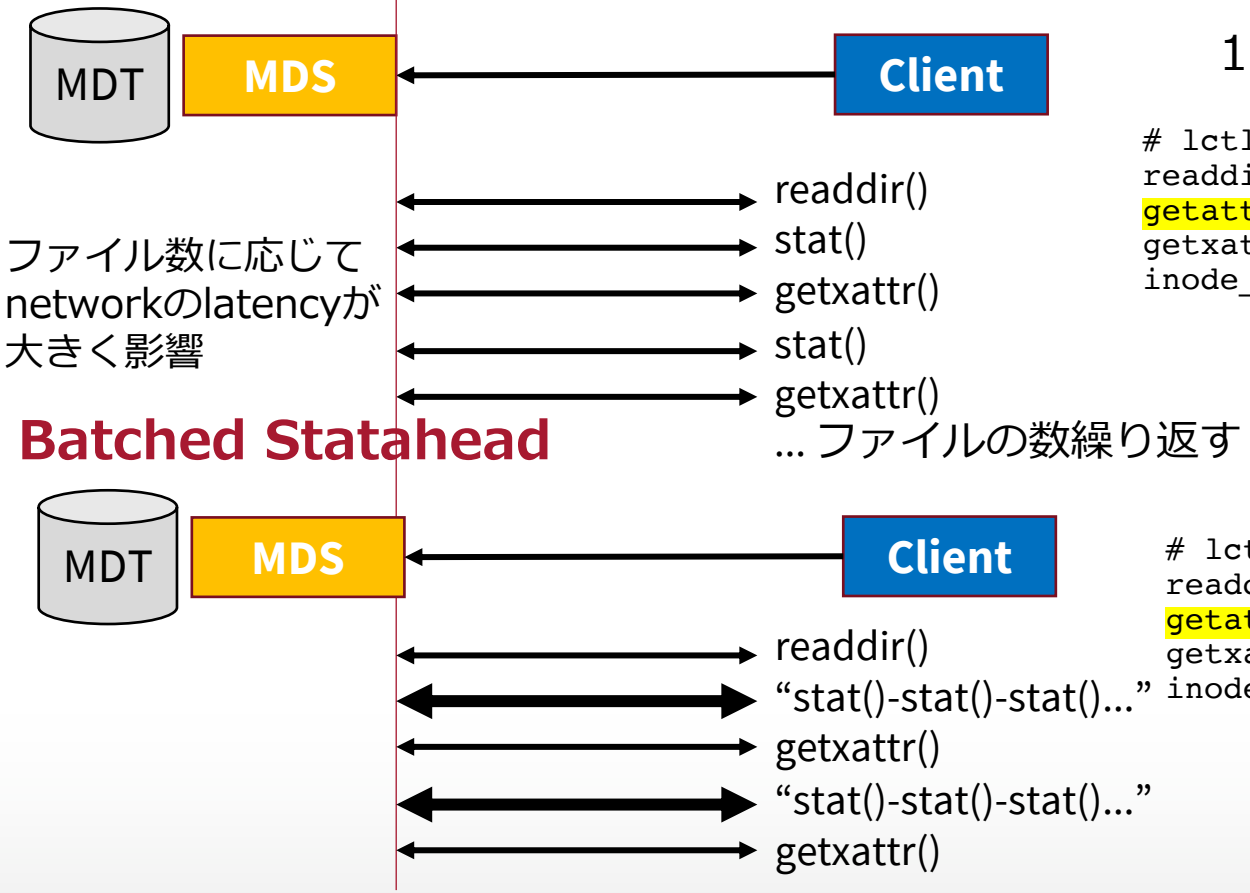
Readahedキャッシュヒット率: 88%

IOR Easy Read: プロセス毎のファイル、16MB IOサイズ、Sequential Read
IOR Hard Read: 全プロセスから単一のファイル、47008 Byte IOサイズ、Stride Read

© 2022 DDN

Lustre Batched Stataheadのサポート(Lustre-2.16)

従来の Statahead



100万ファイルあるDIRに対して”ls –l”を実行した場合

```
# lctl get_param llite.*.stats
readdir           1223 samples [usecs] 0 22518 562923 3229697067
getattr           1000002 samples [usecs] 2 12215 10099392 1764429274
getxattr          1 samples [usecs] 95 95 95 9025
inode_permission 10000021 samples [usecs] 0 197 2527584 3966880
```

100万回のgetattr()に費やした時間10秒

```
# lctl get_param llite.*.stats
readdir           1223 samples [usecs] 0 20710 196717 454495281
getattr           1000002 samples [usecs] 2 13734 5410470 2600319924
getxattr          1 samples [usecs] 96 96 96 9216
inode_permission 10000021 samples [usecs] 0 8814 2289195 80999859
```

100万回のgetattr()に費やした時間5.4秒に短縮

© 2022 DDN

まとめと今後

- IO500が１つのきっかけで、多くのLustre性能が改善
  - IO500のスコアだけではなく、実アプリケーションのI/O性能向上にも貢献
  - 良いLustreのプロダクトサイクル
    - パフォーマンス制限または性能ボトルネックの特定
    - アセスメント(原因の特定、現在の実装、実アプリケーションの影響等)
    - プロトタイプの実装、性能のデモ(IO500、実アプリケーション)、フルパッチの実装
    - EXAScaler/Upstream Lustreへの反映
- 更なるメタデータ性能の改善
  - Batch statahead後のstatオペレーションはCPUの利用率はほぼ100%
  - Batchフレームワークに適応させることで他のメタデータオペレーションも高速化可能
    - 大量の小さなファイルの読み込みおよび作成
    - 将来的には全てのメタデータオペレーション

ご期待下さい！

© 2022 DDN

ddn