コンテナ型仮想化における
ディスクアクセス性能の改善

九州工業大学
大学院 情報工学府
情報創成工学専攻
光来研究室 博士前期課程
2年 佐藤寛文

コンテナ型仮想化

- アプリケーションのための仮想実行環境を提供
  - 例：Docker
  - 計算機全体を仮想化するハイパーバイザ型仮想化より軽量に動作
    - 高速に起動することができる
    - 少ないリソースしか必要としない

ハイパーバイザ型仮想化

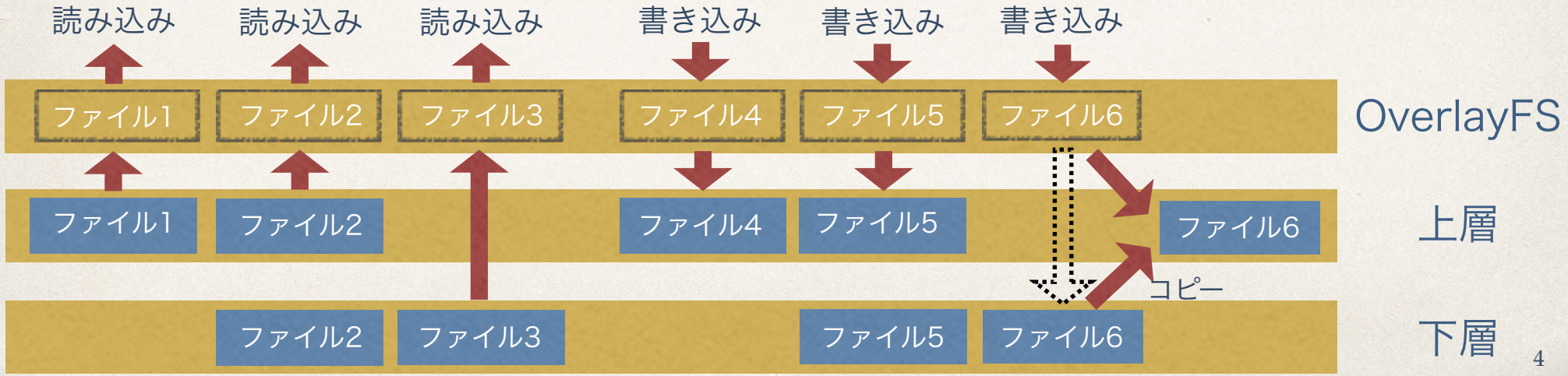
コンテナ型仮想化

コンテナのディスクイメージ

- コンテナが用いるファイルを格納した仮想ディスク
  - 共通ディスクイメージを複数のコンテナで共有
  - コンテナごとに差分ディスクイメージを用意
    - 個別のアプリケーション、ログなどを格納
- OverlayFSを用いて重ね合わせる
  - 下層に共通ディスクイメージ、上層に差分ディスクイメージ

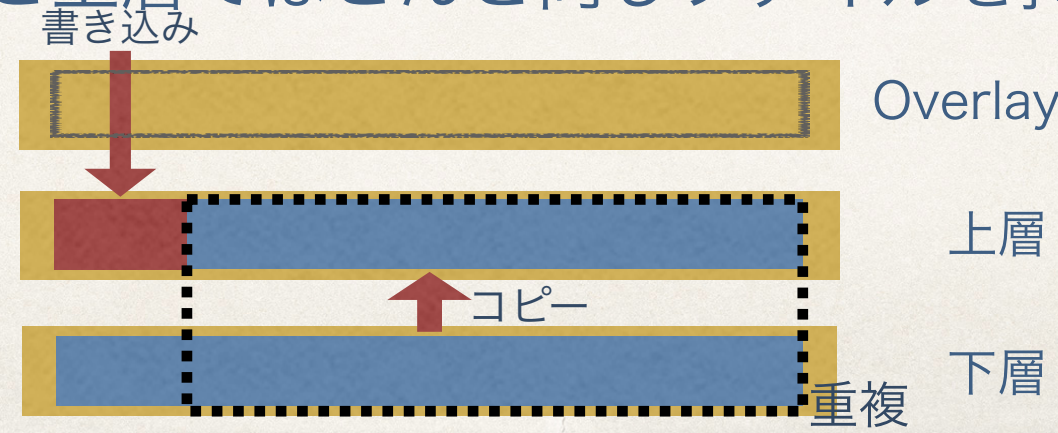
OverlayFSの読み書き処理

- ファイルの読み込み
  - 上層にあれば上層から、なければ下層から読み込む
- ファイルへの書き込み
  - 上層にあれば上層に書き込む
  - 上層になければ下層のファイルを上層にコピーする
    - コピーした上層のファイルに書き込む



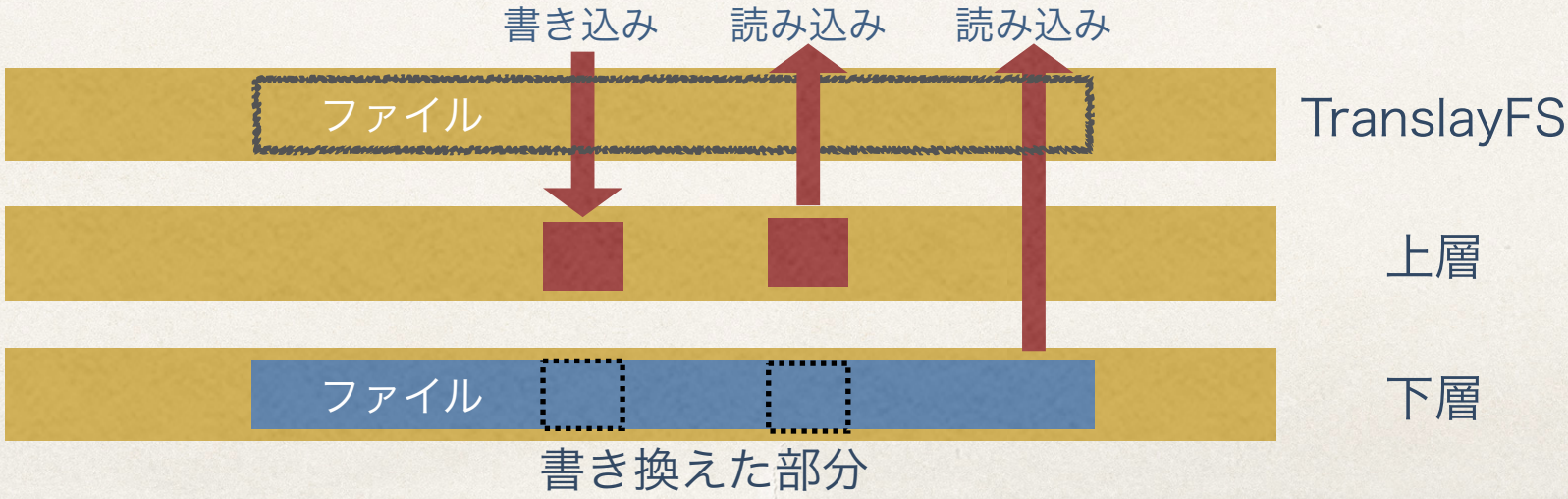
OverlayFSの問題

- 下層のファイルを最初に書き換える際にコピー処理のオーバヘッドが大きい
  - ファイル全体が一括でコピーされる
  - コピーが完了するまでコンテナが停止
    - コピーはシステム全体の性能にも影響を与える
  - ディスク容量を圧迫
    - 下層と上層でほとんど同じファイルを持つことになる



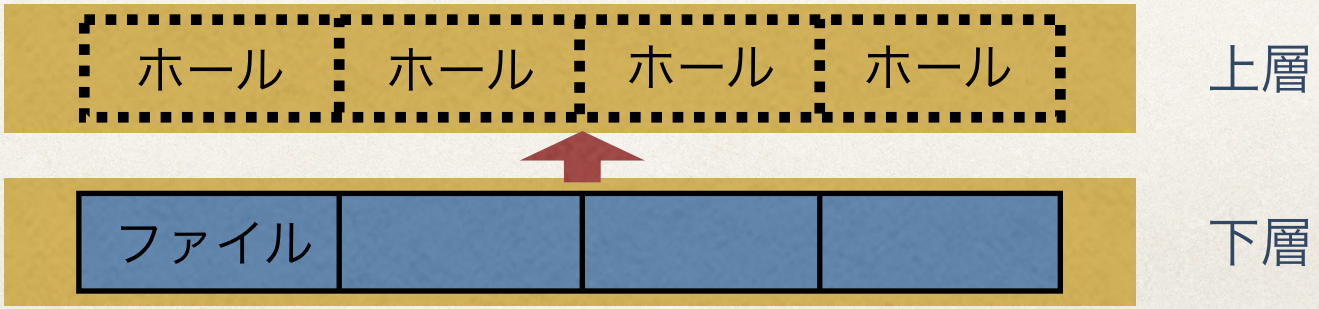
提案：TranslayFS

- OverlayFSを改良し、できるだけ下層から上層へのコピーを行わないようにするファイルシステム
  - 上層には書き換えたデータのみを保持
    - それ以外のデータは下層から読み込む
  - 一括コピーのオーバヘッドを削減
  - ディスク容量を節約



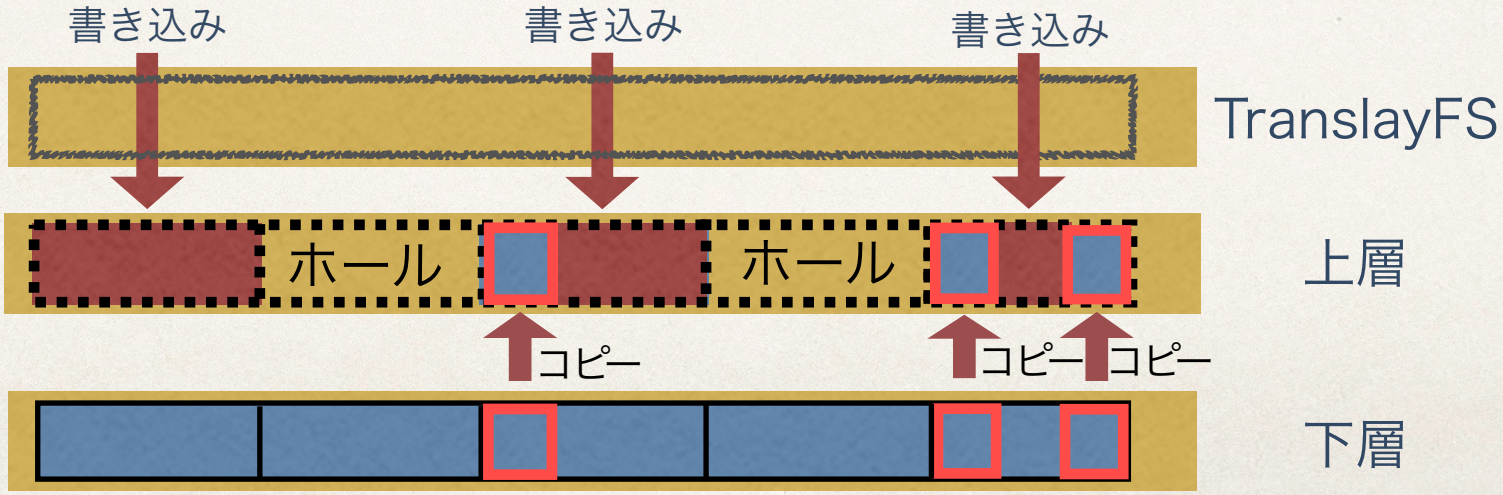
スパースファイルの活用

- ファイルの変更部分を管理するのは煩雑
  - データベースを用いると性能低下の恐れ
- スパースファイルを作成して効率よく管理
  - スパースファイルとは？
    - 実際のデータを持たないファイル
    - ホールと呼ばれる空のブロックから成る
    - ディスク容量をほとんど消費しない
  - ファイルに対して初めて書き込みが行われた時に作成



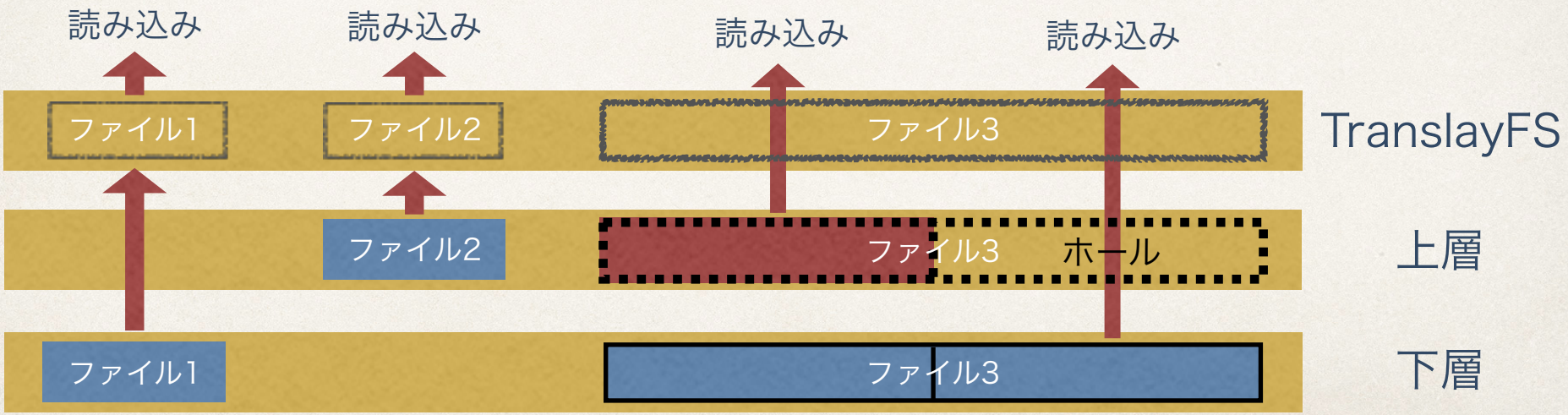
TranslayFSの書き込み処理

- 上層のスパースファイルにブロック単位で書き込む
  - 書き込んだブロックだけが実際のデータを含む
    - その分だけディスク容量を消費
  - 下層のブロックの一部だけを書き換える場合は、書き換えない部分だけを下層からコピー
    - コピーを最小限に抑えることができる



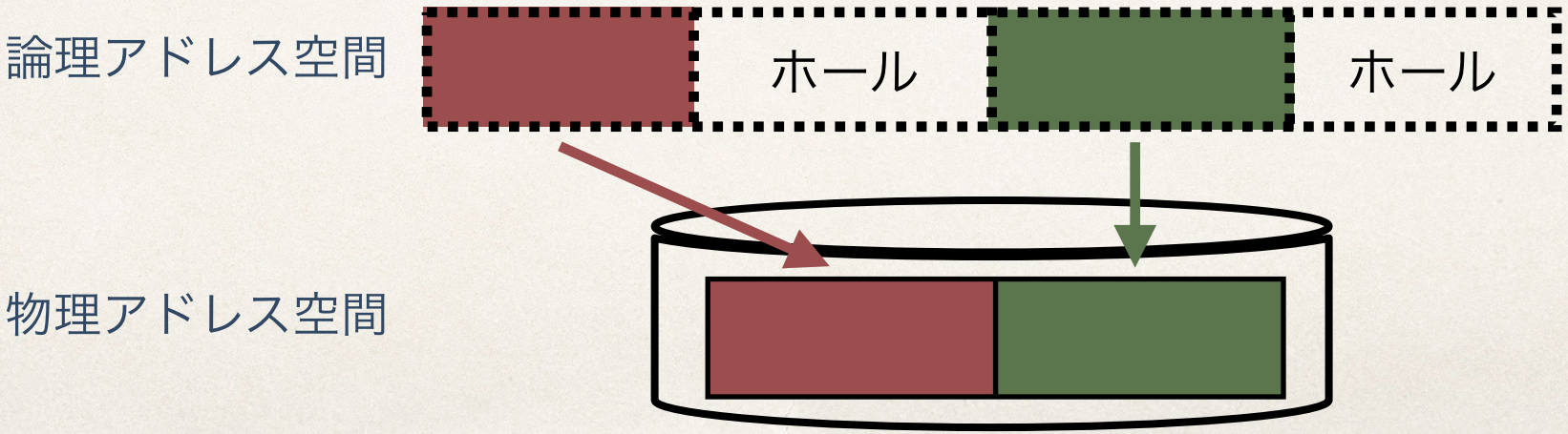
TranslayFSの読み込み処理

- 上層のスパースファイルまたは下層のファイルから読み込む
  - 上層の対象ブロックがデータを含んでいれば上層から
  - 上層の対象ブロックがホールなら下層から
  - データはブロック単位で書き込まれているため、どちらかのブロックだけ読み込めばよい



スパースファイル中のホールの検出

- 読み書きの際にスパースファイルの対象ブロックがホールかどうかを検出する必要がある
  - ファイルの論理ブロック番号をディスクの物理ブロック番号に変換
  - 変換できなければそのブロックはホール
    - 対応するディスク上のブロックがないということ

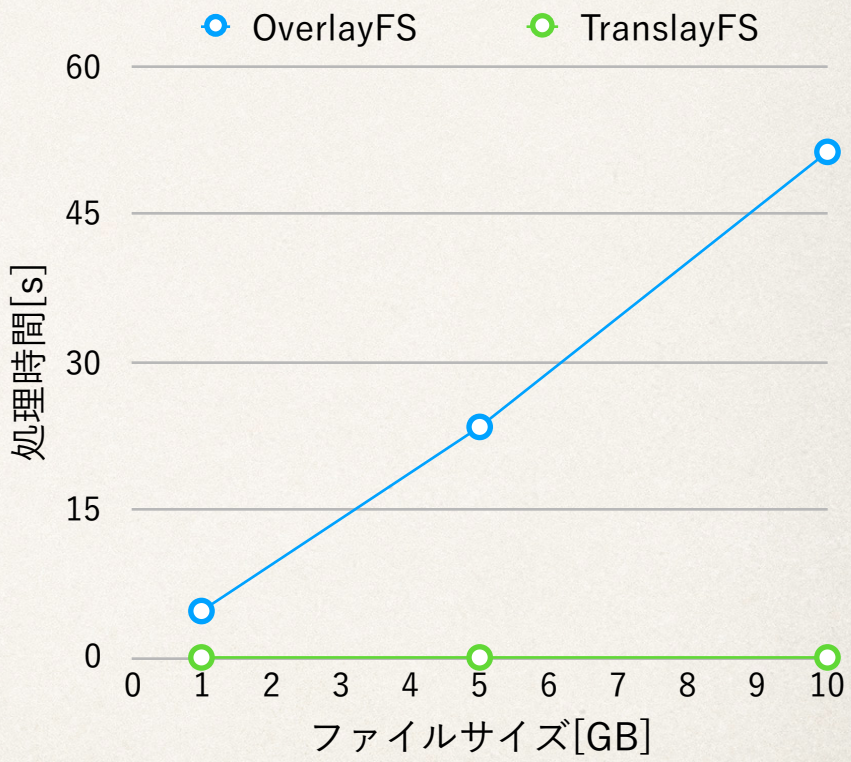


実験

- 目的
  - TranslayFSとOverlayFSとのファイルアクセス性能の比較
  - 初回書き込み、2回目の書き込み、ストライド書き込み、読み込みの性能を実験
- 実験環境
  - CPU：Intel Core i7-3770 CPU @3.40GHz
  - メモリ：8GB
  - ハードディスク：SATA3 HDD
  - OS：Linux 4.4.0

初回書き込み性能の比較

- 様々なサイズのファイルに1バイトの書き込みを行うのにかかる時間を測定

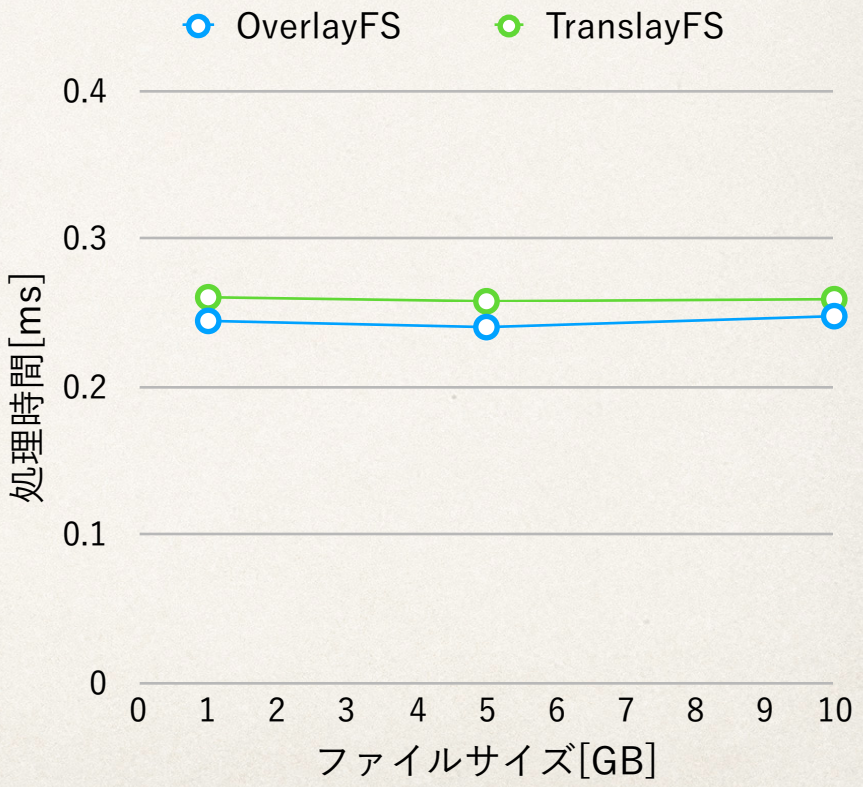


- OverlayFS
  - ファイルサイズに比例して時間が増加
  - 10GBでは51秒
- TranslayFS
  - 10GBでも0.4ミリ秒で完了
  - スパースファイルが作成に4KBだけ書き込まれた

2回目の書き込み性能の比較

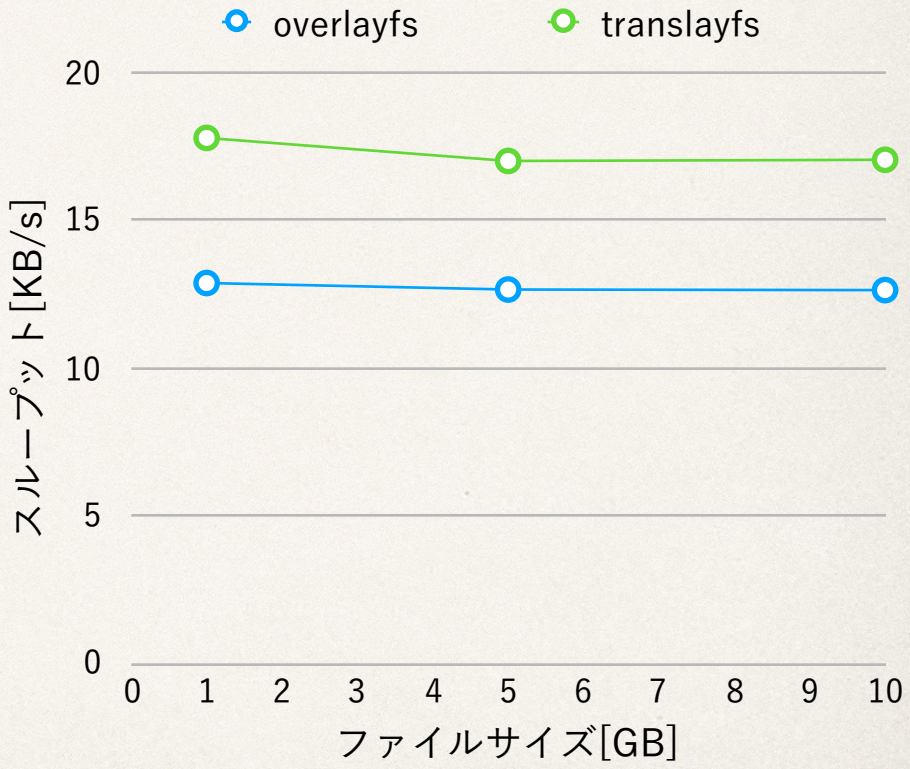
- 1バイトの書き込みを行ったファイルの同じブロックにさらに1バイトの書き込みを行う時間を測定

- TranslayFSの方が4〜7%遅くなった
  - TranslayFSを経由してアクセスするオーバヘッド
- OverlayFSでは上層のファイルに直接アクセス



ストライド書き込み性能の比較

- 様々なファイルの全ブロックに順番に1バイトずつ書き込んだ時の性能を測定

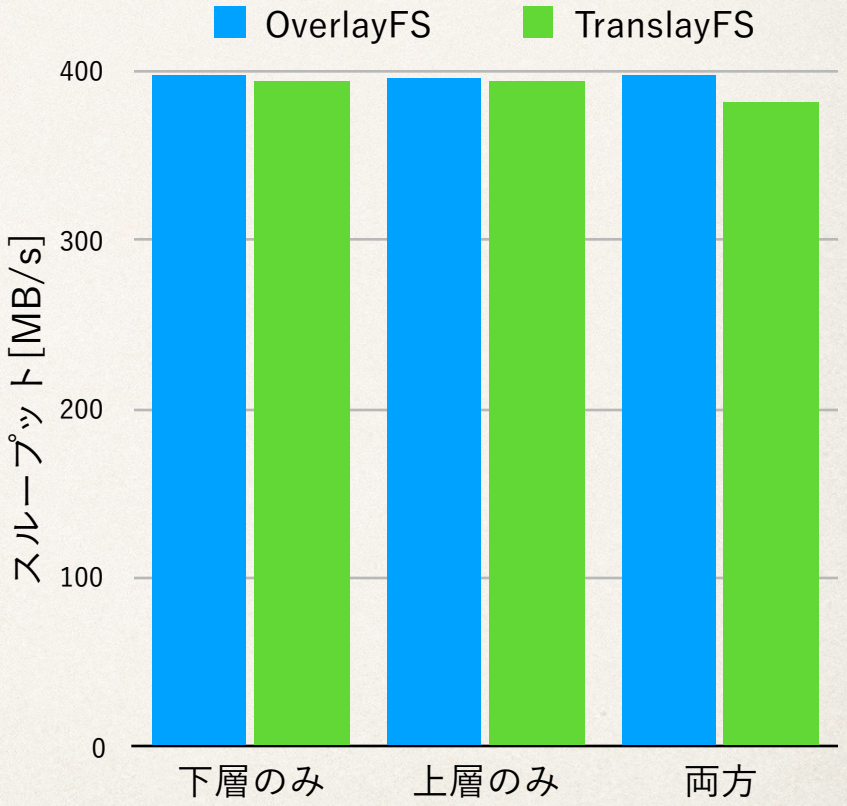


- TranslayFS
  - OverlayFSより36%高速
- OverlayFS
  - ファイルの先読みがうまく行えていない可能性
    - 上層のファイルには読み書きが両方行われる

読み込み性能の比較

- 10GBのファイル全体を読み込んだ時の性能を測定

- 下層と上層の一方にのみファイルがある場合
  - TranslayFSは0.5%低下
  - TranslayFSを経由するオーバヘッド
- 下層と上層の両方にファイルがある場合
  - TranslayFSは3.9%低下
  - ホールを検出するオーバヘッド



関連研究

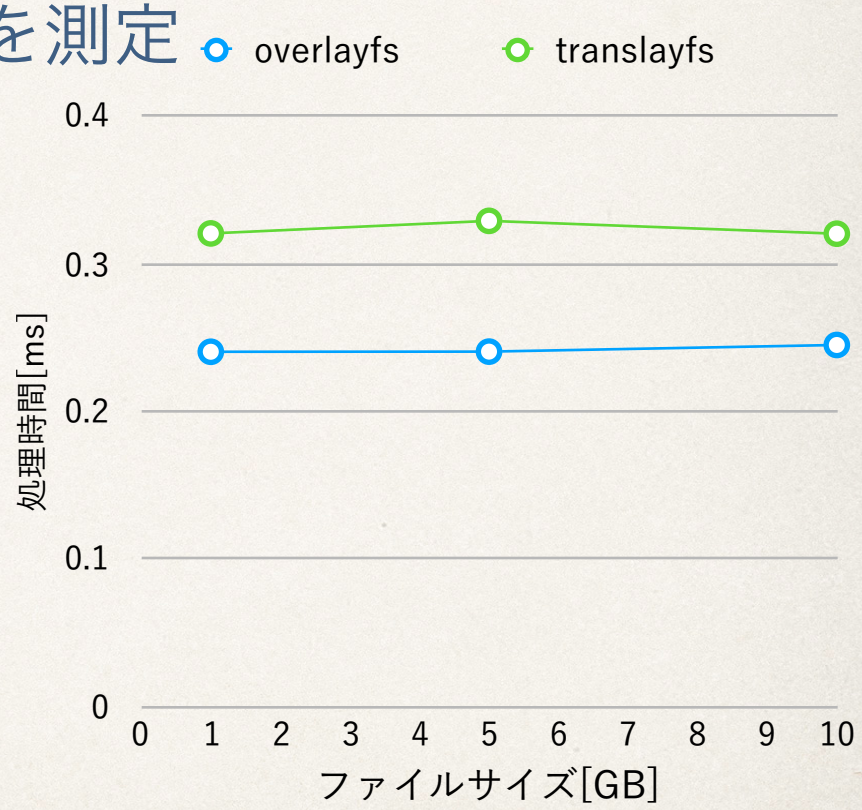
- Unionファイルシステム（OverlayFS、AUFS等）
  - 複数のファイルシステムを重ねるファイルシステム
  - ファイル単位でコピーを行うオーバヘッドが大きい
- ZFS［LLNL］、Btrfs［Oracle］
  - 差分管理などを提供する高機能なファイルシステム
    - ブロック単位でコピーを行う
  - 性能や安定性に問題がある
- devicemapper［Red Hat］
  - 差分管理を提供するブロックデバイス
  - ファイルシステムより扱いにくい

まとめ

- OverlayFSを改良し、できるだけ下層から上層へのコピーを行わないようにするTranslayFSを提案
  - スパースファイルを用いて書き換えたデータのみ上層に保持
  - コピーのオーバヘッドを減らし、ディスクを節約
  - ファイルに対する最初の書き込み性能が大幅に向上
  - それ以外の性能は最大7%低下
- 今後の課題
  - 2回目以降の書き込み性能と読み込み性能の向上
    - ホールの判定回数を減らす
    - 上層のスパースファイルにホールがない場合はTranslayFSを経由せずに直接アクセス

2回目の書き込み性能の比較（別ブロック）

- 1バイトの書き込みを行ったファイルの別のブロックにさらに1バイトの書き込みを行う時間を測定



- TranslayFSの方が30〜37%遅くなった
  - TranslayFSを経由してアクセスするオーバヘッド
  - 下層からコピーするオーバヘッド
- OverlayFSでは上層のファイルに直接アクセス

ランダム書き込み性能の比較

- 10GBのファイルに一度に書き込むサイズを変えながら1%分のランダム書き込みを行い時間を測定

- 書き込みサイズが増加するほど処理時間が低下
- TranslayFSはOverlayFSと比較して処理時間の推移が激しい
- 書き込みサイズが410バイト、4096バイトの時はTranslayFSの方が速くなった