Finding and Analyzing Performance Pitfalls of On-Demand Paging of InfiniBand

(InfiniBandのオンデマンドページングの性能問題の発見と分析)

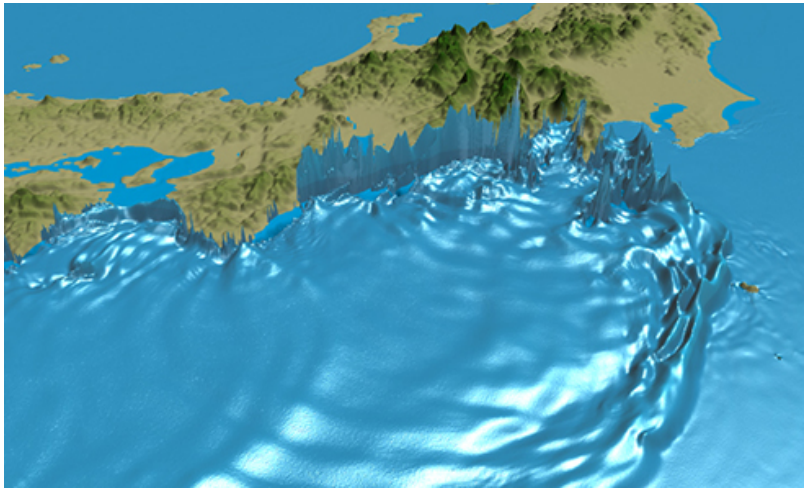
東京大学大学院 情報理工学系研究科 電子情報学専攻 修論審査

2021年2月2日

学籍番号 48-196438 田浦研究室 福岡 拓也

分散コンピューティング

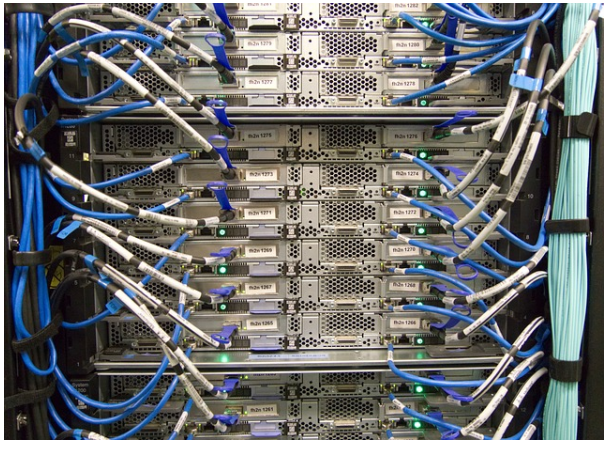
- 分散コンピューティングが不可欠の時代
  - 人工知能 (AI), ビッグデータ, グラフ処理, 科学技術計算
  - データセンター, スーパーコンピュータ
- ノードと呼ばれる複数の計算機を用いて計算

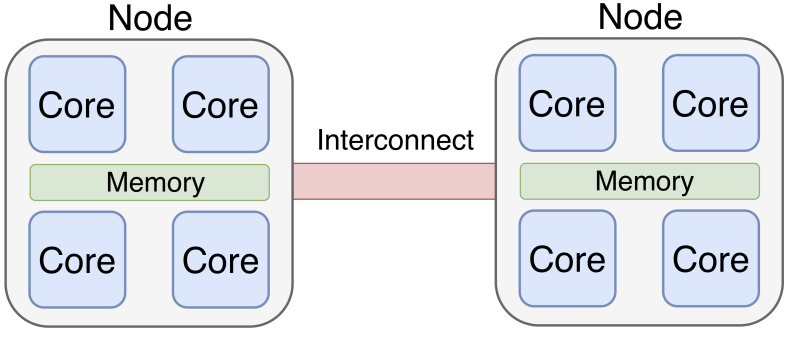


https://www.nikkei.com/article/DGXMZO60655390S0A620C2MM8000/
https://www.itmedia.co.jp/enterprise/articles/1711/27/news048.html

高性能インターコネクトの重要性

- ノード間通信はインターコネクトを通して実行
  - InfiniBand, Ethernet, Omni-Path, Tofuなど
- 全体性能はインターコネクトの性能に大きく影響される
  - 性能は計算よりも通信に律速 [1]



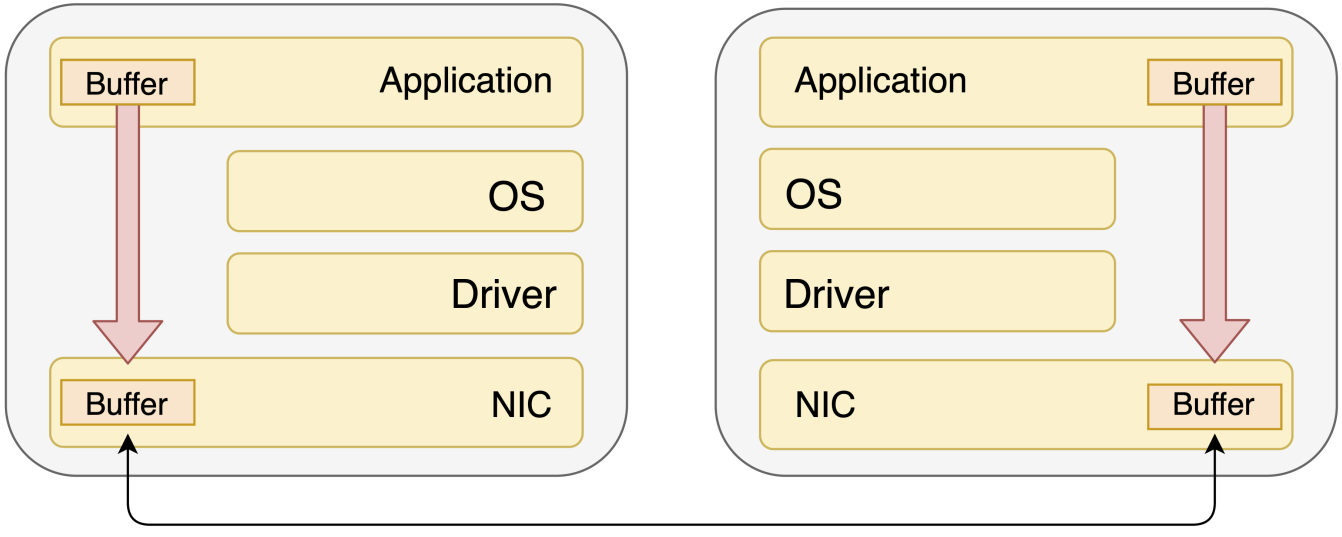


https://www.servants.co.jp/blog/technology/hno-mellanox/1742

[1] Dongarra, J., Heroux, M. A., & Luszczek, P. (2016). High-performance conjugate-gradient benchmark: A new metric for ranking high-performance computing systems. The International Journal of High Performance Computing Applications, 30(1), 3–10.

高性能な通信を支えるRDMA

- Remote Direct Memory Access (RDMA)
- 低遅延と高スループット
- カーネルへのバッファコピーを省き，リモートのCPUをバイパス
- InfiniBand, RoCE, iWARPなどで採用



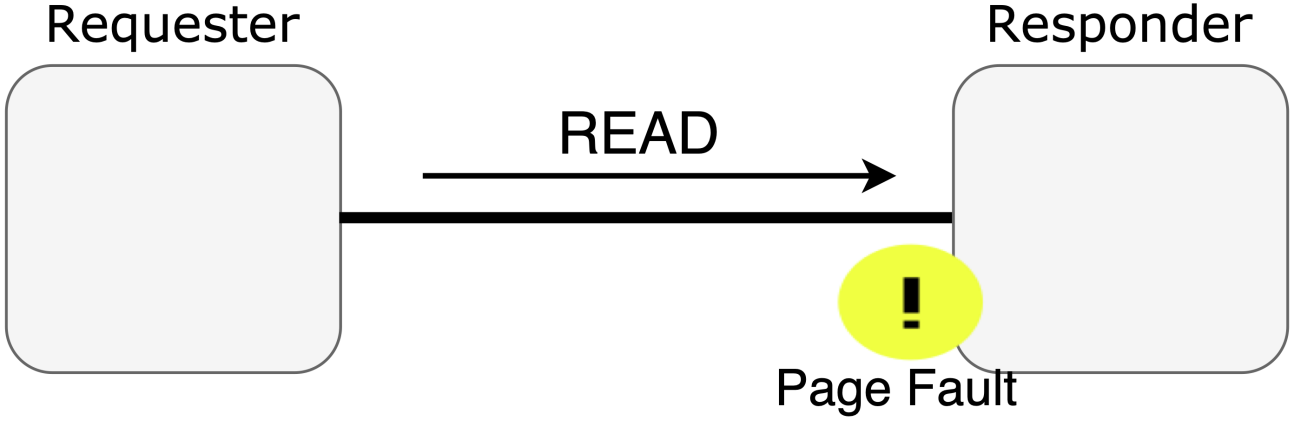
RDMAの問題点

- 通信を発行する前に，通信バッファに対してメモリレジストレーションを行う必要がある
  - バッファのスワップアウトを防ぐために物理メモリにピン留め
  - 仮想アドレスと物理アドレスの変換エントリをNIC (Network Interface Card)に登録
- 2つの大きな問題点
  - スワップアウトせずに計算に使用できるメモリの制限
  - バッファを使い回す際のプログラミングコストの上昇 (cf. Pin-down cache [1])

[1] Tezuka, H., O'Carroll, F., Hori, A., & Ishikawa, Y. (1998). Pin-down cache: A virtual memory management technique for zero-copy communication. IPPS/SPDP 1998.

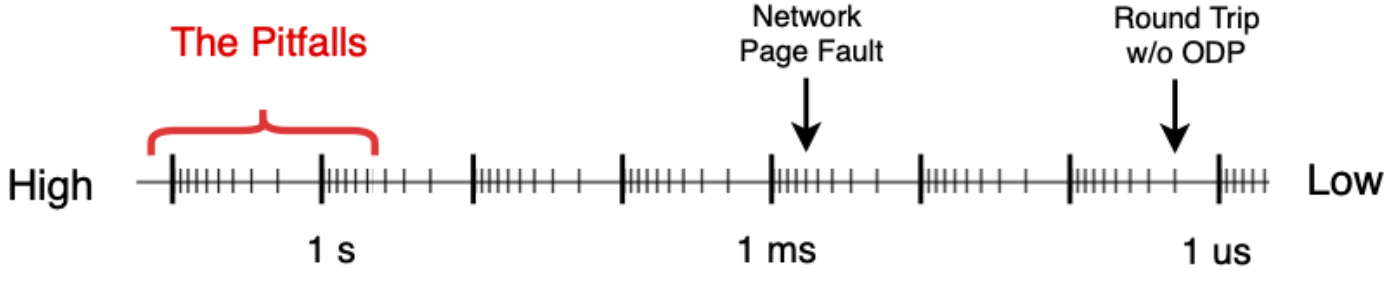
On-Demand Paging (ODP)

- Mellanoxによって最近InfiniBandに導入された前途有望な技術
  - 事前のメモリレジストレーションが不要に
- ネットワーク越しのNICのページフォルトを利用することで，通信バッファのピン留めを自動化
  - ピン留めする通信バッファのメモリ使用量の削減
  - ピン留めの手動管理が不要
- 過去の研究によると，ページフォルトに関わるオーバーヘッドは許容範囲 (数百usほど)



本研究の貢献

- ODPの挙動をパケットキャプチャツールibdumpで分析
- ODPの重大な性能面のバグ (Pitfalls) を二種類発見
  - 驚くべきことに，単純な条件で数百msから数秒のストール
    - cf. インターコネクトのレイテンシは数usほど
  - 1つ目の性能バグ: Packet damming
    - 原因: InfiniBandの異常に長い再送のタイムアウト
  - 2つ目の性能バグ: Packet flood
    - 原因: NICのページ状況の更新の長時間にわたる失敗
- この二つの性能バグが実システムで生じることを確認



発表の流れ

背景

- InfiniBandと再送制御
- On-Demand Paging (ODP)

ODPの関連研究

マイクロベンチマークを用いた実験

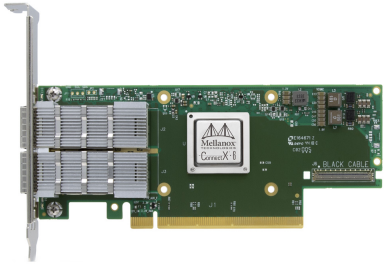
- 1つ目のバグ: Packet Damming
- 2つ目のバグ: Packet Flood

実システムを用いた実験

まとめ

InfiniBand

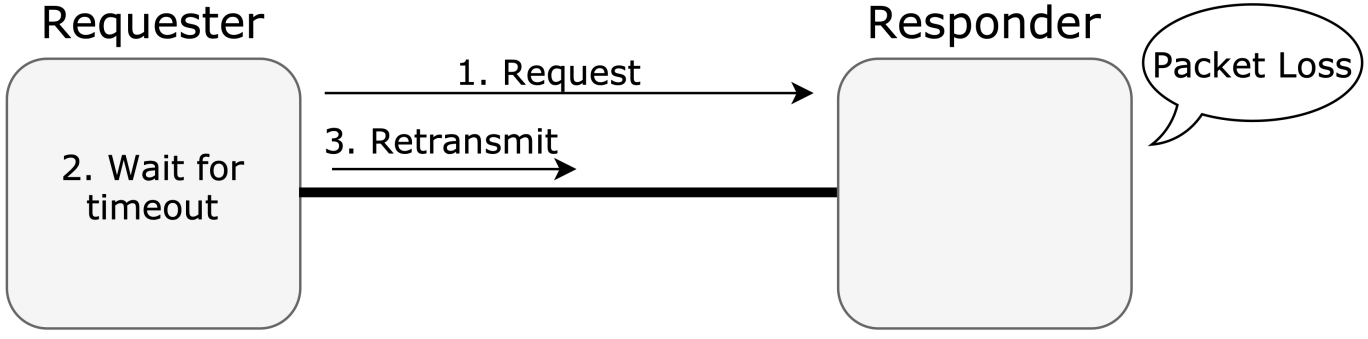
- 主に高性能計算で用いられる，RDMAをサポートした超低遅延インターコネクト
  - 右下の図がRDMAのNIC (Network Interface Card)
- 二種類の通信オペレーション
  - 双方向通信関数: SEND, RECEIVE
  - 片方向通信関数: READ, WRITE
- QP (Queue Pair) と呼ばれる通信リソースごとに通信を投入



https://www.infinibandta.org/tag/roce/
https://jp.mellanox.com/products/ethernet-adapters/connectx-6/

InfiniBandのトランスポート層

- InfiniBandは4種類のトランスポートプロトコルをサポート
  - 代表的なものが，Reliable Connection (RC) と Unreliable Datagram (UD)
- RCは信頼性のあるプロトコルで，途中でエラーが生じた時には再送を行う
  - 例えば，パケットロスが生じてタイムアウトになった場合など



発表の流れ

背景

- InfiniBandと再送制御
- On-Demand Paging (ODP)

ODPの関連研究

マイクロベンチマークを用いた実験

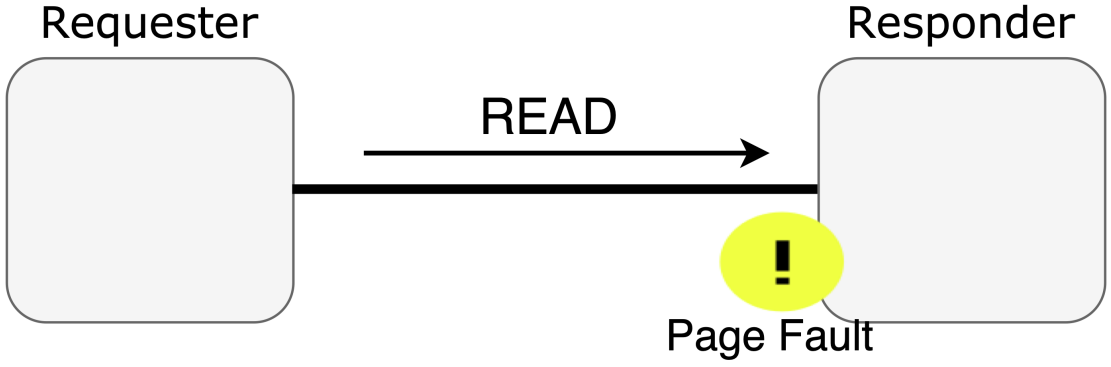
- 1つ目のバグ: Packet Damming
- 2つ目のバグ: Packet Flood

実システムを用いた実験

まとめ

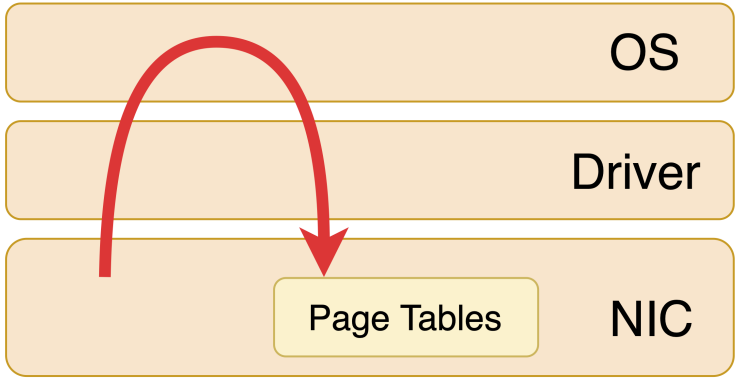
On-Demand Paging (ODP) の概要

- 通信バッファのピン留めをハードウェアでその都度行い，事前のメモリレジストレーションを不要にする技術
- 通信バッファのピン留め，NICへのアドレス変換エントリの登録は，そのページが通信に必要になって初めて行う
- 各種MPIライブラリでの導入例あり



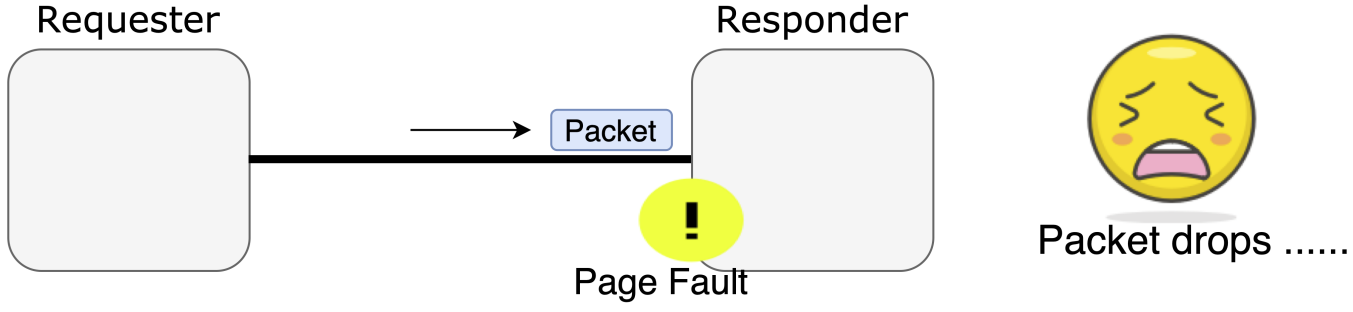
ODPの基本的な実装

- InfiniBandのドライバとNICのファームウェアで実装
- NICのページテーブルにエントリがないページにアクセス要求が来た時，次の手順で対処
  i. NICがOSにページフォールトを要求
  ii. OSは物理ページがない場合は割り当て
  iii. ドライバ経由でNICのページテーブルに，アドレス変換エントリを登録

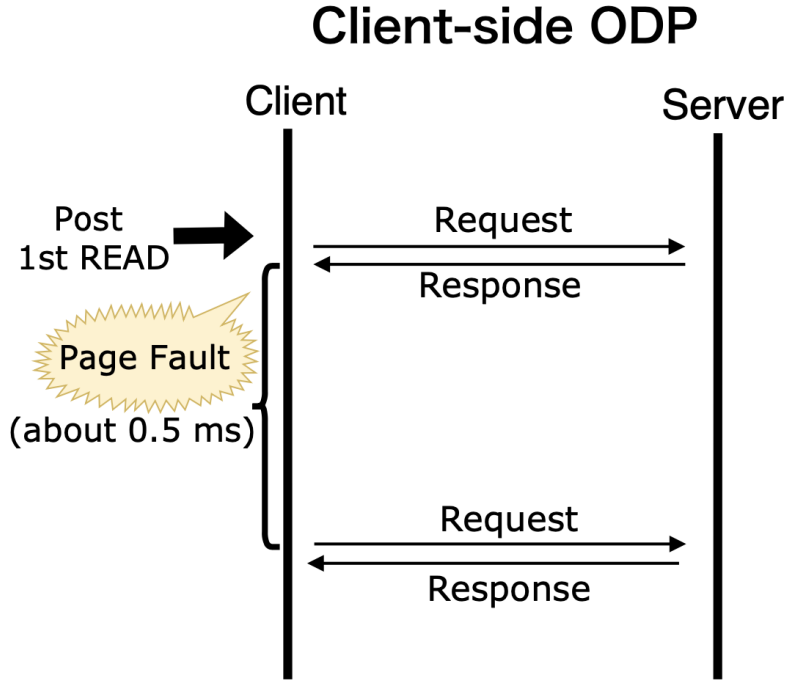


再送を用いたODP実装のサポート

- パケットの受け取り側でページフォルトが発生した場合には困難がある
  - NICのメモリサイズは限られている
  - ページフォルトが解決するまで受け取ったパケットをNICに保持することはできない
- RCの再送を利用して対処
- RDMA READを具体的にどのように対処しているのかibdump（パケットキャプチャツール）で観察



ODPを使用した時のREADの挙動



- 時間の都合上，Client-side ODPのみ解説
- ClientはResponseを受け取った後にページフォルトを引き起こして一定時間後に再送

発表の流れ

背景

- InfiniBandと再送制御
- On-Demand Paging (ODP)

ODPの関連研究

マイクロベンチマークを用いた実験

- 1つ目のバグ: Packet Damming
- 2つ目のバグ: Packet Flood

実システムを用いた実験

まとめ

ODPの関連研究

ODPの実装と性能分析に取り組んだ研究を紹介

- Lesokhinらは，ODPそのものを提唱 [1]
  - ページフォルトのオーバーヘッドは数百usであり，ODPのコストは許容可能
- Liらは，ODPをMPIに取り入れて評価 [2,3]
  - ODPを使用しない時と比較して，少ないメモリ量で同程度の性能

[1] Lesokhin, I., Eran, H., Raindel, S., Shapiro, G., Grimberg, S., Liss, L., … Tsafrir, D. (2017). Page Fault Support for Network Controllers. ASPLOS'17.

[2] Li, M., Hamidouche, K., Lu, X., Subramoni, H., Zhang, J., & Panda, D. K. (2016). Designing MPI Library with On-Demand Paging (ODP) of InfiniBand: Challenges and Benefits. SC'16.

[3] Li, M., Lu, X., Subramoni, H., & Panda, D. K. (2017). Designing Registration Caching Free High-Performance MPI Library with Implicit On-Demand Paging (ODP) of InfiniBand. HiPC'17.

発表の流れ

背景

- InfiniBandと再送制御
- On-Demand Paging (ODP)

ODPの関連研究

マイクロベンチマークを用いた実験

- 1つ目のバグ: Packet Damming
- 2つ目のバグ: Packet Flood

実システムを用いた実験

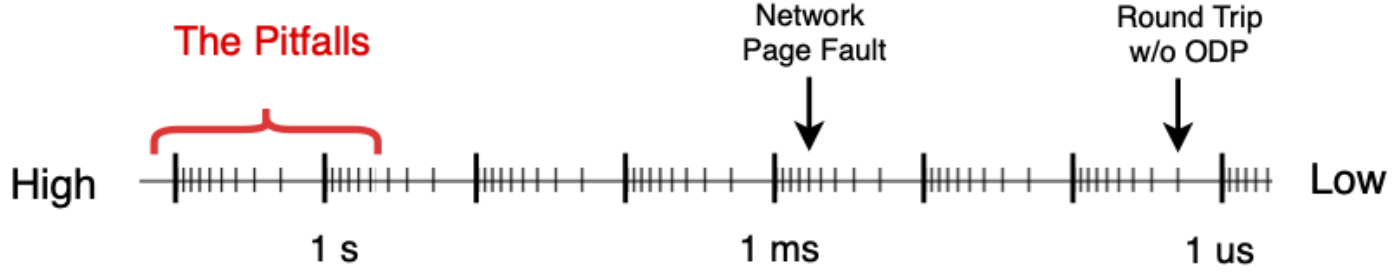
まとめ

実験環境

- ２つのマシンをInfiniBandで直接接続
- Xeon Phi CPU 7250 (1.40 GHz, 272スレッド)
- PC4-19200 196GB, MCDRAM 16GB
- Mellanox MCX456A-FCAT ConnectX-4 VPI adapter
- 再送のタイムアウトの設定は最小値を使用

1つ目の性能バグ Packet damming

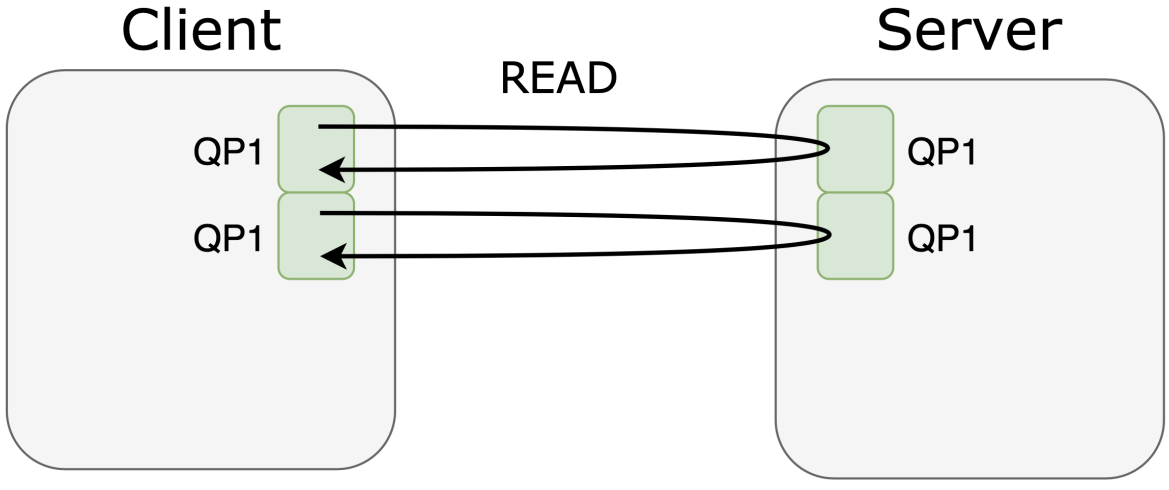
- 状況: READを発行した直後に一定間隔で別の通信を発行
- 特徴: 通信パケットがstuckする (堰き止められる，damming)
- 影響: 数百msの遅延
- 原因: パケットロスとそれに伴う異常に長いタイムアウト



- 発見のきっかけは分散共有メモリライブラリMenpsの分析 [1]
  - ODPが原因であることがわかるまでに数ヶ月を要した

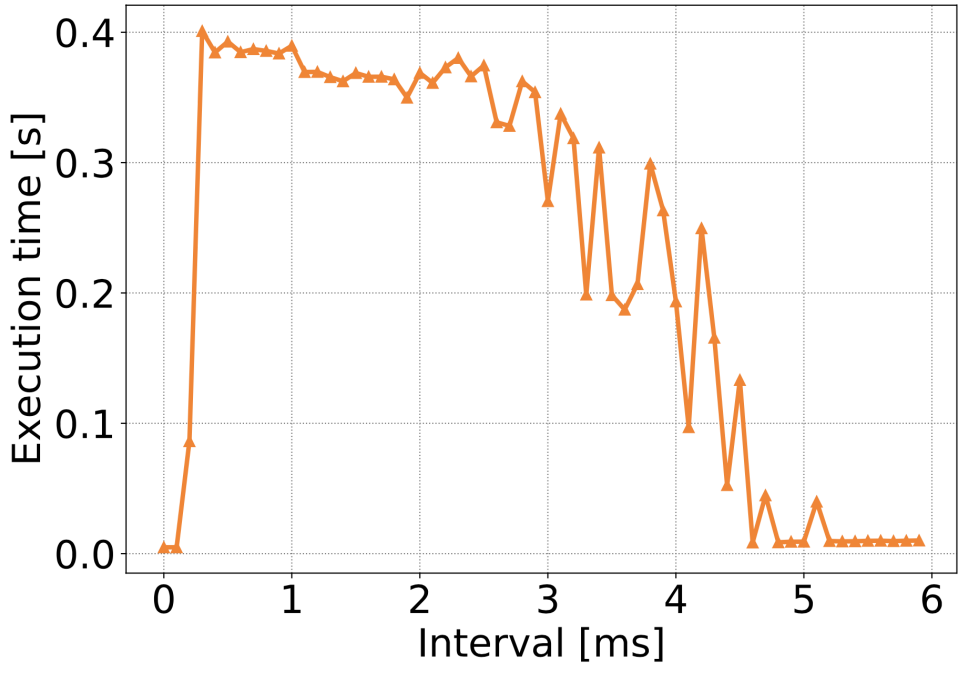
[1] Wataru Endo. A decentralized implementation of software distributed shared memory. Doctoral dissertation.

マイクロベンチ (Packet damming)



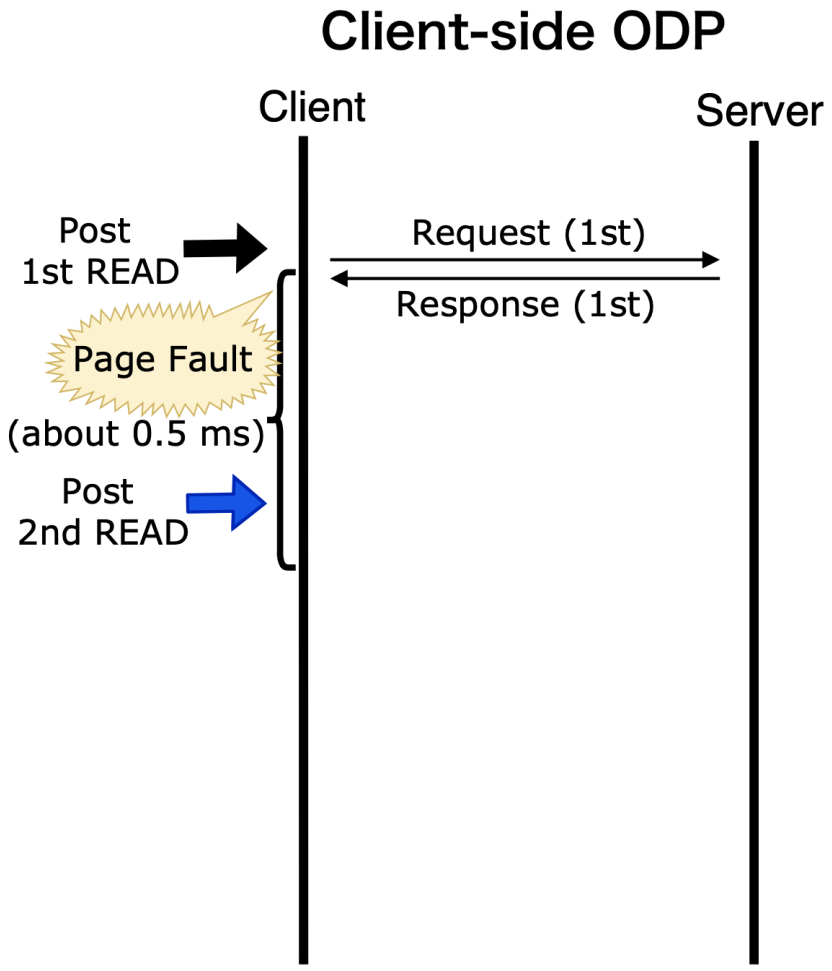
- 片方のマシンがもう一方にRDMA READを2つ発行
  - とてもシンプルなマイクロベンチマーク
- 通信の発行する間隔 (Interval)を変更
- 1つの通信のサイズは100 bytes，QPは1つのみ使用

マイクロベンチの実行時間



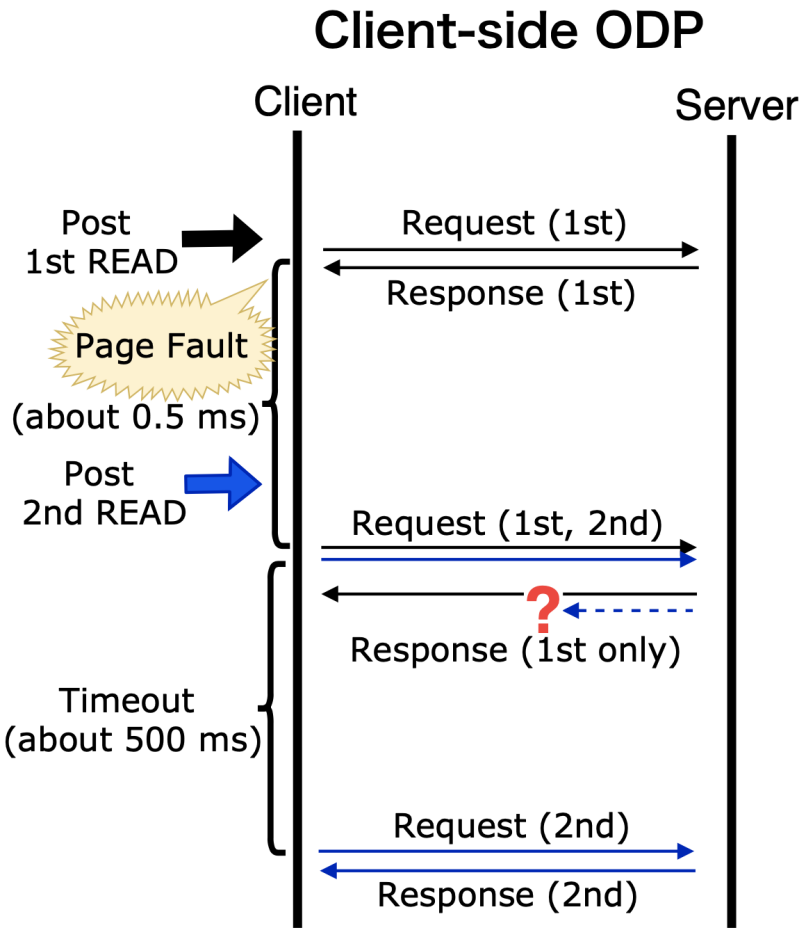
- Both-side ODPの結果
  - ServerとClient両方についてODPを設定
- Intervalを変更して，10回実行した時の実行時間の平均をプロット
- Intervalが500usから4500usの時には，実行時間が数百msほどと非常に長い

遅延が大きい時のClient-side ODPの挙動



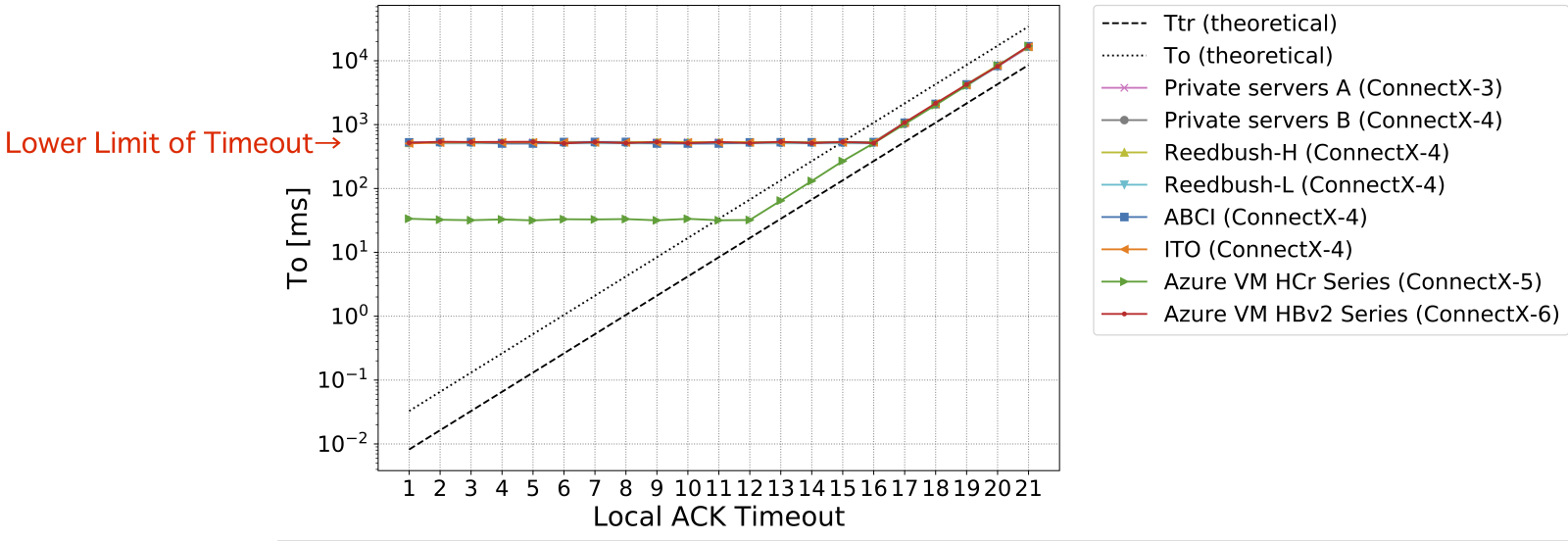
- ibdumpで分析
- (Server-side ODPは割愛)
-
-

遅延が大きい時のClient-side ODPの挙動



- ibdumpで分析
- (Server-side ODPは割愛)
- 再送の待ち時間の間に２つ目のパケットが来ると，その後にパケロス
- タイムアウトまで待って再送，しかしこのタイムアウトが異常に長い（なぜ？）

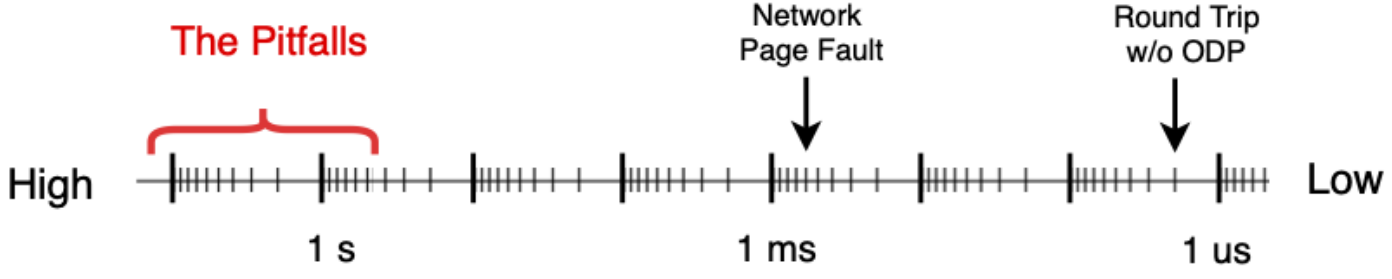
タイムアウトは最小に設定したはずでは？



- タイムアウトの下限値を実際に測定
- 面白いことに，タイムアウトの下限値は最新の機種を含めたほとんどの機種で500msほどと，非常に大きな値だった
  - この下限値はファームウェアで規定，ユーザから変更不可
  - InfiniBandはリンク層でロスレスのフロー制御をするので，通常の使用ではパケロスが生じないことを見越した設定か
- これが性能バグの元凶

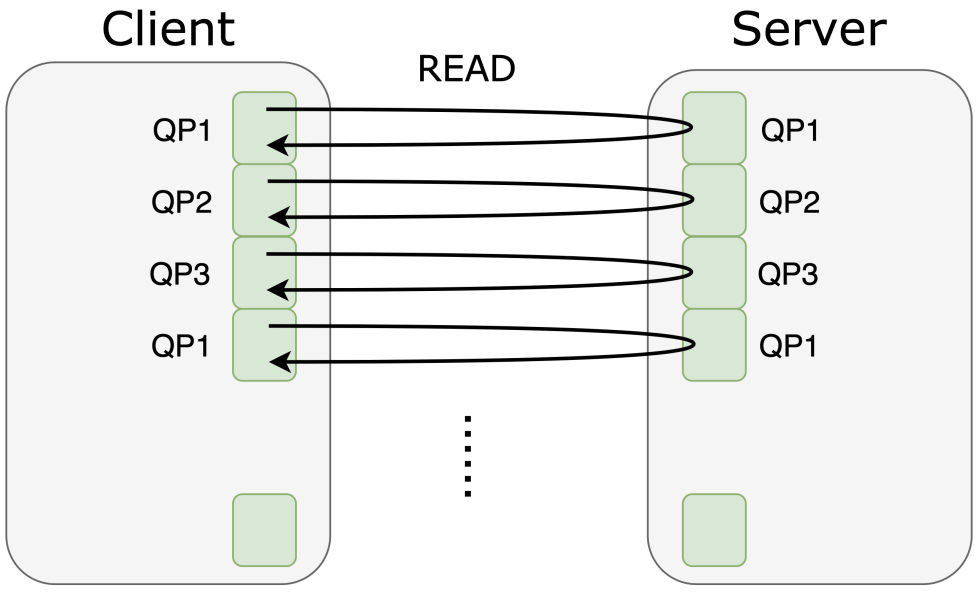
2つ目の性能バグ: Packet flood

- 状況: Client-side ODPで複数のQPに対してREADを同時に発行
- 特徴: 同じリクエストの再送による，パケット数の増加
- 影響: 数百msから数十秒の遅延
- 原因: QP間でのページフォルト解決の伝達の失敗



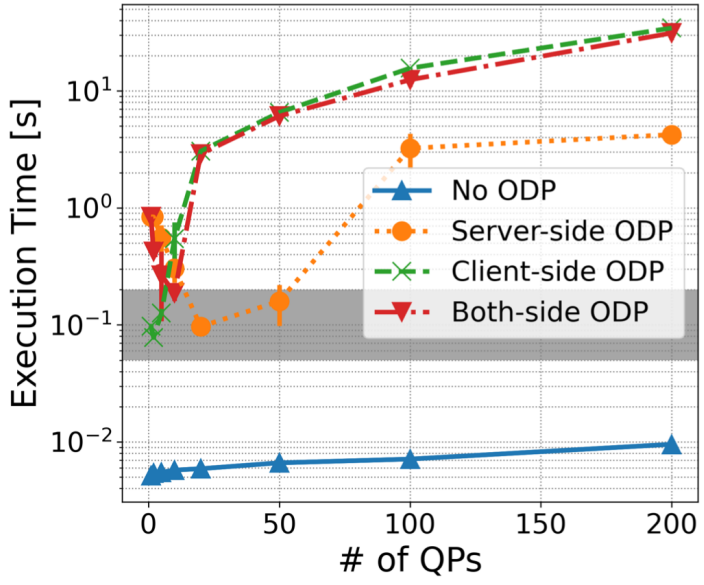
- 発見のきっかけはSparkUCXでのPacket dammingの再現のための実験

マイクロベンチ (Packet flood)

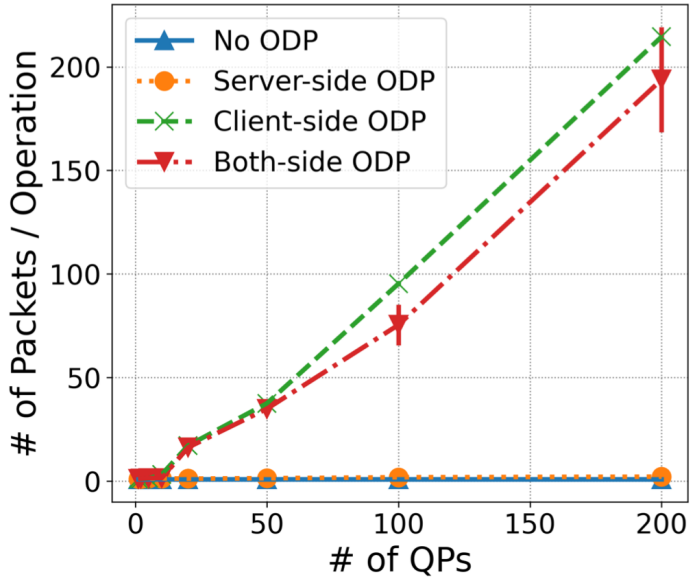


- 通信の個数は8192個，サイズは100バイトに固定
- 発行する通信同士の間隔は空けない
- QPの数を変化させて，ラウンドロビンに割り当て

マイクロベンチの実行時間とパケット数比



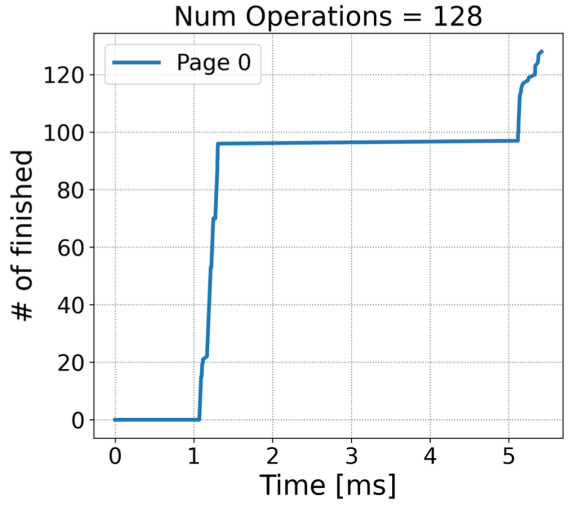
(a) Execution Time (s)



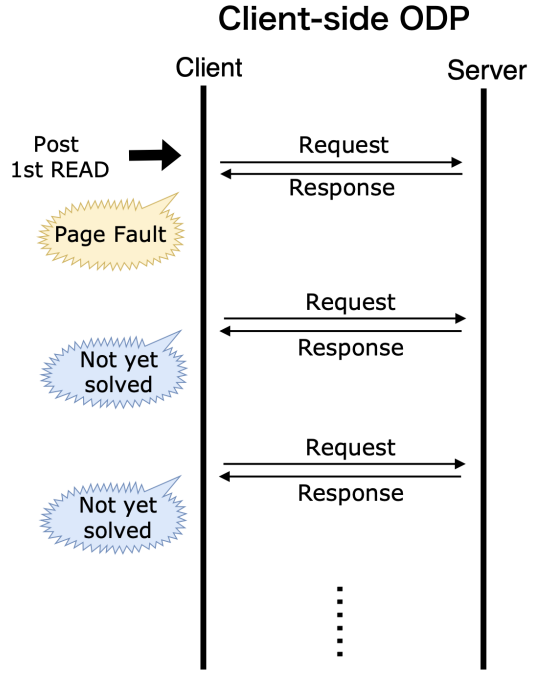
(b) Number of Packets

- 灰色の部分はページフォルトの個数から計算した妥当な実行時間
- Client-side ODPにおいて，最大数十秒のレイテンシとパケットの大量発生を観測

通信終了数の推移



(a) 128 Operations



- ページフォルトはページ (4096バイト) 単位で解決される
- 1つのページ内でページフォルトが解決している時には，原理的には全ての通信がすぐに終わって良いはず
- 見たところ，ページ状態を更新するのが失敗している
  - ページフォルトが解決しないので再送し続けて，パケット数が増大

発表の流れ

背景

- InfiniBandと再送制御
- On-Demand Paging (ODP)

ODPの関連研究

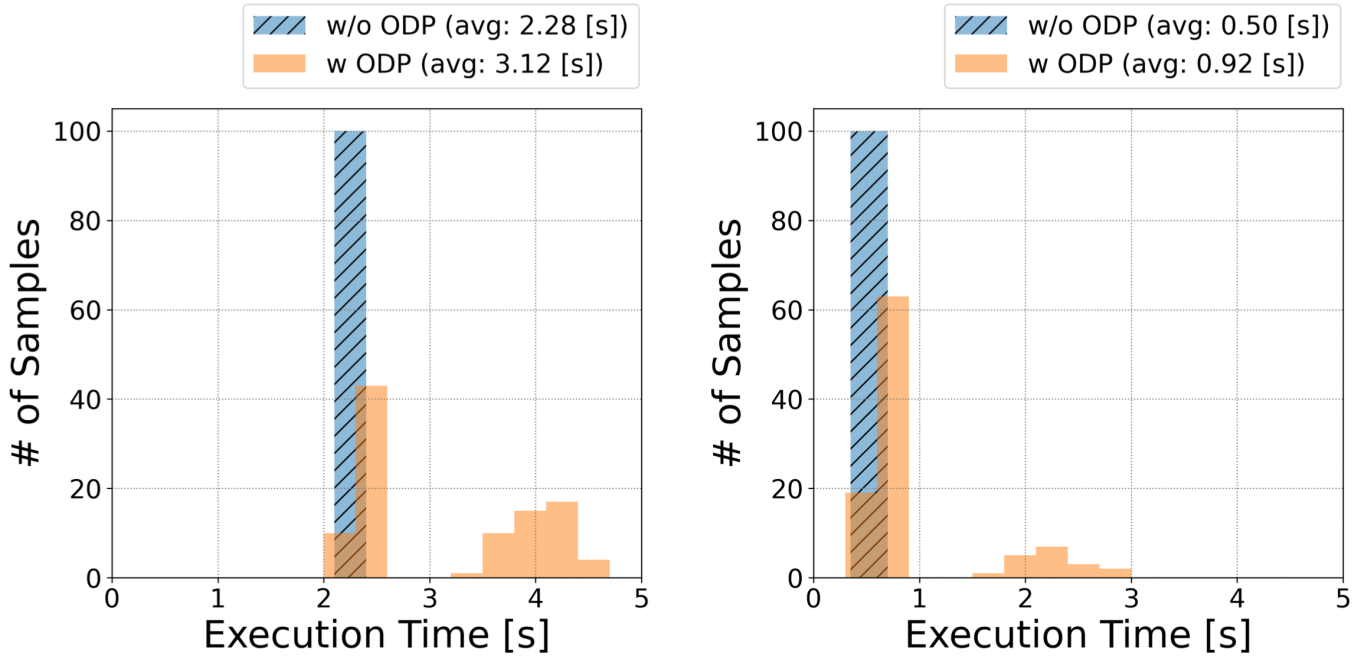
マイクロベンチマークを用いた実験

- 1つ目のバグ: Packet Damming
- 2つ目のバグ: Packet Flood

実システムを用いた実験

まとめ

ArgoDSM* の評価



(a) KNL-4 (2 nodes)　　(b) Reedbush-H (2 nodes)

- 高性能計算向け分散共有メモリライブラリ
- InitとFinalizeのみを含むベンチマークの時間を測定
- ODPを有効にした場合には分布が二極化
  - 実行時間が大きいグループについてはタイムアウトが発生
  - つまりPacket dammingが生じている

* https://argodsm.com/

SparkUCX* の評価

mllib.RankingMetricsExample

	QPs	Disable [s]	Enable [s]	**Enable/Disable**
KNL-4 (2)	389	517 ± 3.7	674 ± 140	1.30
Reedbush-H (2)	980	46.6 ± 1.4	111 ± 56	2.38
ABCI (2)	2191	107 ± 1.8	147 ± 1.6	1.37
ABCI (4)	2667	83.2 ± 3.4	197 ± 4.7	2.37

- 分散フレームワークSparkのバックエンドにRDMAを使用したもの
  - QPを大量に生成
- 走らせたSparkアプリはjoin操作で大量のREADを発行
- ODPを用いた場合には途中でstuck，その際に大量のREADパケットを確認
  - これはまさしくPacket floodの特徴

* https://github.com/openucx/sparkucx/

発表の流れ

背景

- InfiniBandと再送制御
- On-Demand Paging (ODP)

ODPの関連研究

マイクロベンチマークを用いた実験

- 1つ目のバグ: Packet Damming
- 2つ目のバグ: Packet Flood

実システムを用いた実験

まとめ

まとめ

- ODPの重大な性能面のバグ (Pitfalls) を二種類発見
  - NICのページフォルトの単体の3桁から4桁大きい長さのストールが生じる
  - 単純なマイクロベンチマークで再現
- 1つ目の性能バグ: Packet damming
  - 状況: READを発行した直後に一定間隔で別の通信を発行
  - 原因: パケットロスとそれに伴う異常に長いタイムアウト
- 2つ目の性能バグ: Packet flood
  - 状況: Client-side ODPで複数のQPに対してREADを発行
  - 原因: QP間でのページ情報の伝達の失敗
- この二つの性能バグがArgoDSM，SparkUCXという実システムで生じうることを確認

得られた知見

- HWについてのハイレベルな知見
  - SWでやっていたメモリレジストレーションをHWに移植する試みは思っていたよりも困難
  - RCの信頼性と再送に依存した，ある種ハックのような現状の実装方式では，複数の通信が関わった時にボロが出る
- 性能劣化のみならず，原因特定の困難さにも重大性がある
  - Packet dammingの原因特定に要した時間は数ヶ月
  - 通信ハードウェアという最深部に根源，目に見えたエラーが出ない, 再現性がない, など
  - これらのバグの存在を世に出すことは重要

将来の展望

- 発見したバグの改善
  - ハードウェアからのアプローチとしては，ベンダーには報告済みなので調査結果待ち
    - 最新の機種ではPacket dammingは解決されているとのことだが，タイムアウトの下限値は大きいまま
    - タイムアウトの下限値が大きい設定の真の理由は？
    - ODP以外の機能で再送に頼る場面が見当たらないのか調査
  - ソフトウェアからのアプローチ
    - Packet dammingは回避策が考え得るのでそれを評価
    - Packet floodの回避策を模索
- より広範なシステム/アプリでのODPを評価

Publication

国際会議（査読あり）

[1] T. Fukuoka, W. Endo, and K. Taura. An Efficient Inter-Node Communication System with Lightweight-Thread Scheduling. HPCC '19, August. 2019.

[2] T. Fukuoka, S. Sato, and K. Taura. Pitfalls of InfiniBand with On-Demand Paging. ISPASS '21, March. 2021. (投稿中)

国内会議

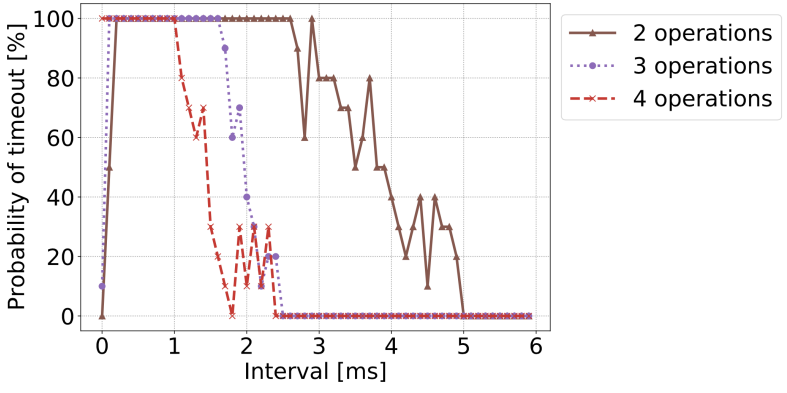
[3] T. Fukuoka, W. Endo, and K. Taura. An Implementation of Inter-node Communication System with Efficient Lightweight-thread Scheduling. xSIG ’19, May. 2019. (Best Undergraduate Student Award)
[4] T. Fukuoka, S. Sato, and K. Taura. Analyzing Performance Pitfalls of On-Demand Paging of InfiniBand. SWoPP ’20, August. 2020. (第150回OS研究会 最優秀若手発表賞)

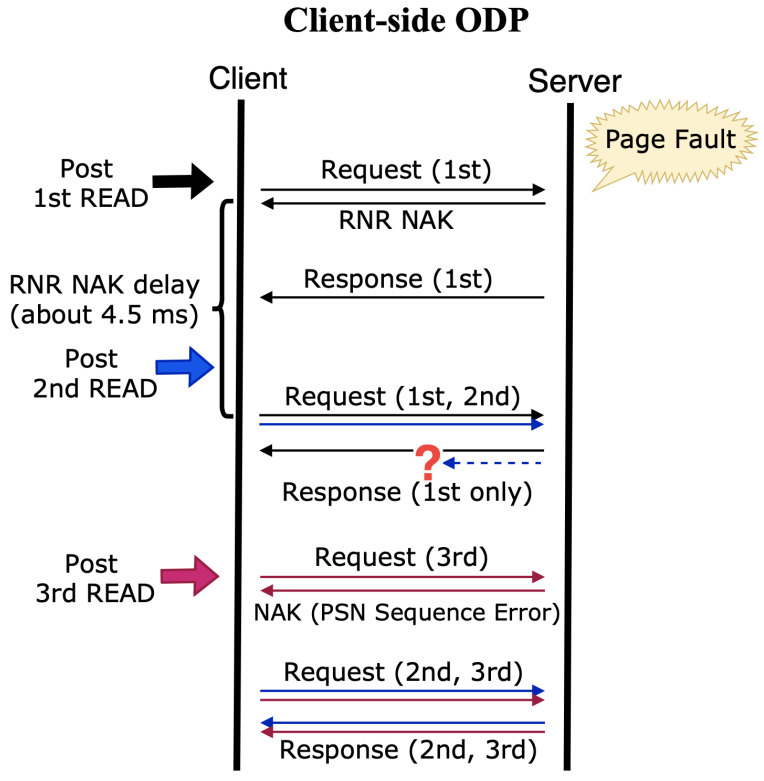
国内会議（ポスター発表）

[5] T. Fukuoka, W. Endo, and K. Taura. An Implementation of Inter-node Communication System with Efficient Lightweight-thread Scheduling. xSIG ’19, May. 2019.

予備スライド

通信が3つ以上の時の評価



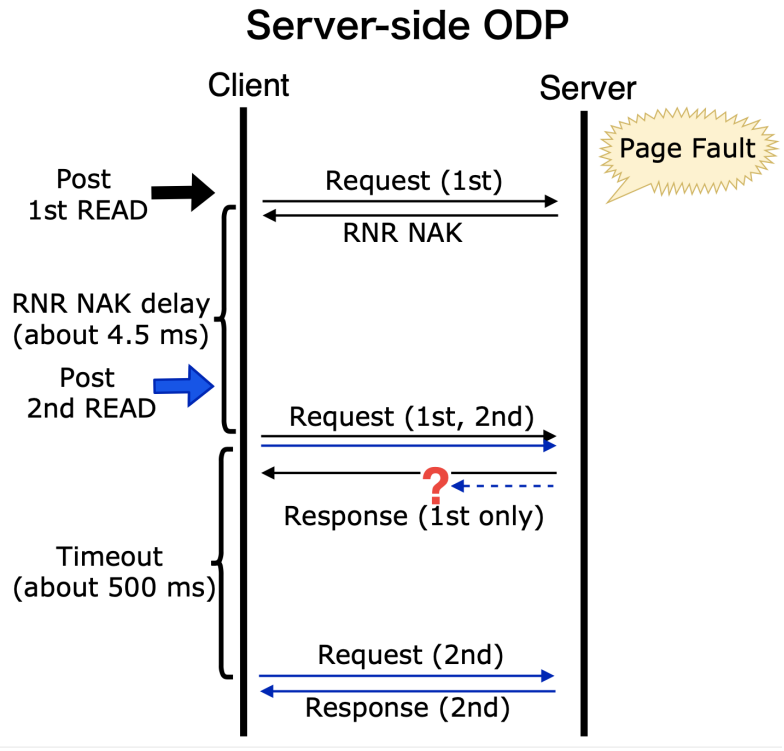


- 3つ以上READを発行した場合にはPacket dammingが生じづらくなる
- タイムアウトの待ち時間中に新しくパケットが投入されると，ServerからClientにNAKが返って即座に再送が行われるため

なぜServer-side ODPでは生じないのか

- Client側は再送を自分で行うので，通信の状態管理が必要
- Server側はRequestを受け取り，ページの状態をチェックして，RNR NAKを返すだけでよいので通信の状態管理が必要ない
- 通信の状態管理が必要ということは，それと共にページの状態がキャッシュされていてもおかしくないはず

Server-side ODPでのPacket damming

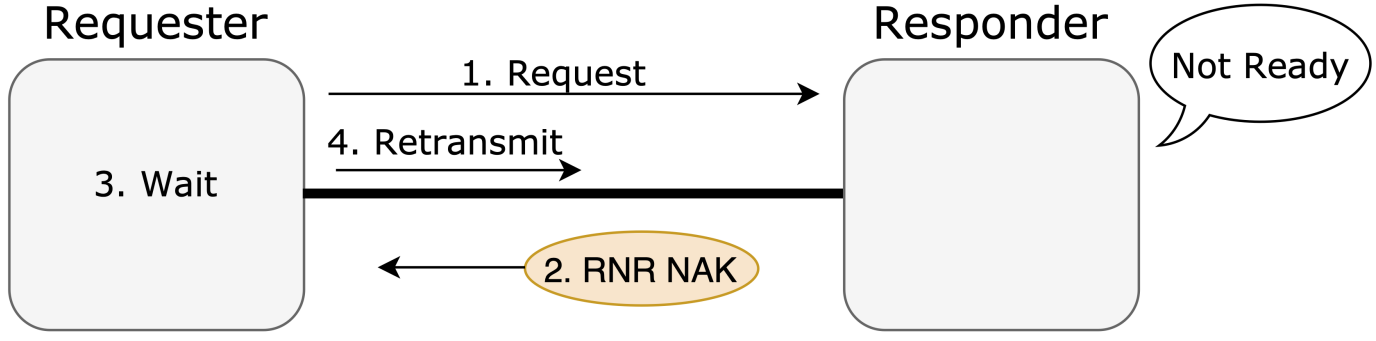


- RNR NAKの待ち時間中に二つ目のREADが投入された場合，同様にパケロスとそれに伴うタイムアウトが生じる

再送が生じる条件 (1)

Responderの準備ができていない場合

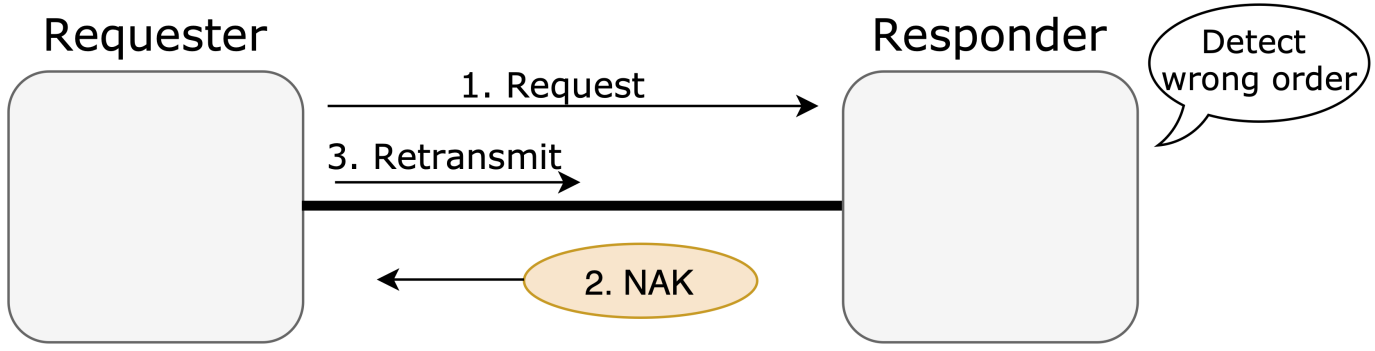
- Responderは，Receiver-Not-Ready (RNR) NAKを返し，Requesterに一定時間後 (RNR NAK delay)の再送を要求
- 例えば双方向通信の際に，RESPONDERがRECEIVEを投入していない場合



再送が生じる条件 (2)

Responderがパケットの到着順序に誤りがあるのを検出した場合

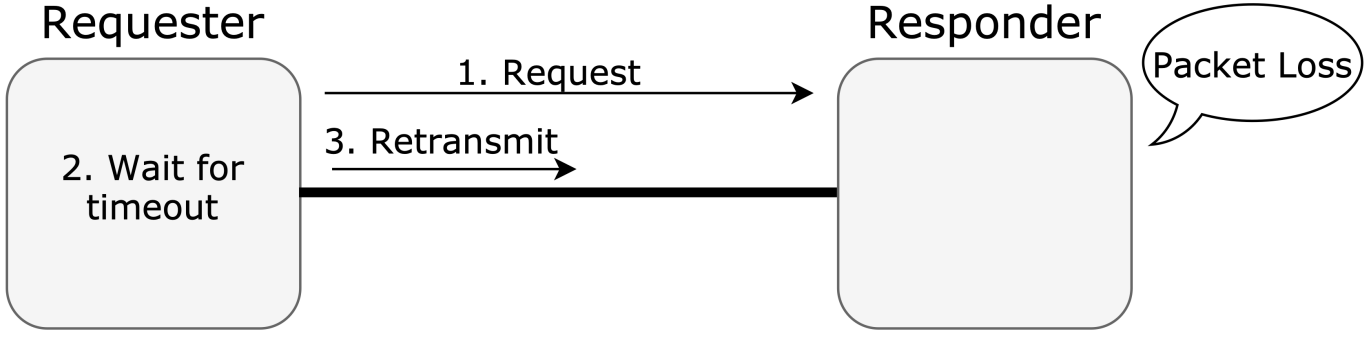
- 例えば，N+1番目のパケットがN番目のパケットより前に到着した場合には，N番目のパケットロスが消失したと考えられる
- ResponderはNAK (PSN Sequence Error) を返して，Requesterに再送を要求



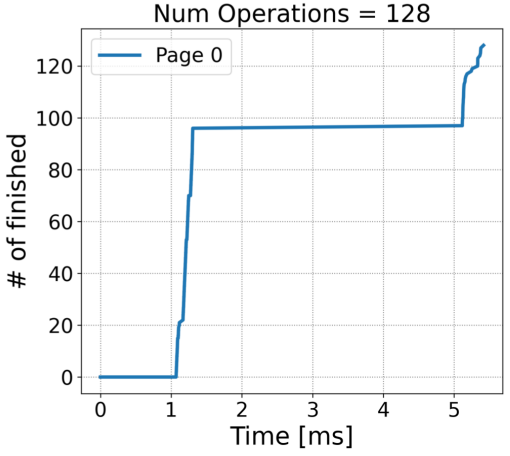
再送が生じる条件 (3)

Requesterが通信を投入してから一定時間，Responseが返ってこなかった場合（これが一番重要）

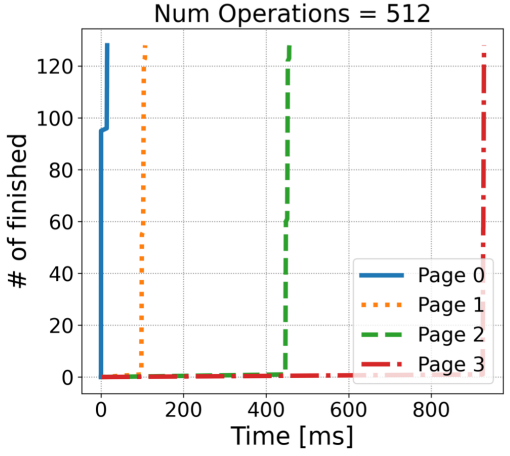
- Requesterはパケットロスが生じたと判断して，再送を行う
- Responseを待つ時間（タイムアウト）は設定可能で，理論上は
  最短で数usに設定することができる
- ただし，下限値はIBカードのベンダーによって規定されるとのこと（伏線）



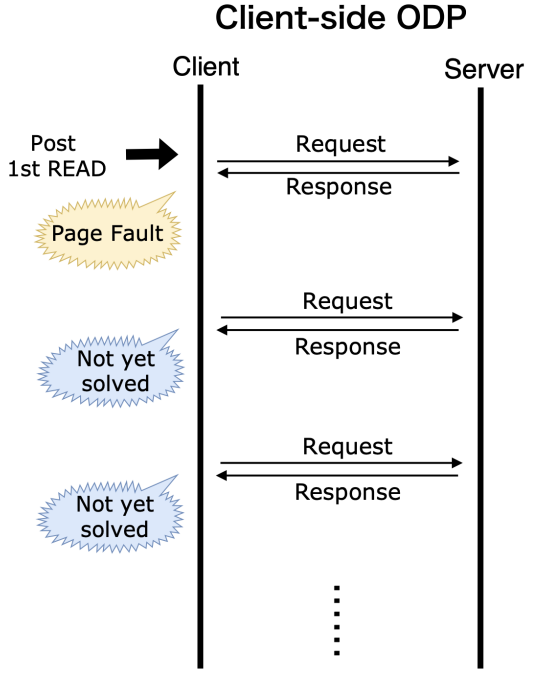
Packet floodの詳細な分析



(a) 128 Operations



(b) 512 Operations



その他修論で行ったこと

- Packet damming
  - 通信が3つ以上の時の評価
- Packet flood
  - WRITEでの評価
- ConnectX-6での評価