

リアルタイムLinux関連の研究活動

李 奕驍(名古屋大学大学院情報学研究科)

背景

- 近年Linuxベースの組み込みシステムが増えている

特に機能が複雑化した制御システムによく使われている

- ロボティクス ROS/ROS2
- 車載IVI AGL (Automotive Grade Linux)
- ものづくり LinuxCNC、Klipper



リアルタイム性と信頼性の課題が多い

- カーネル設計当初は汎用処理を想定
制御処理に使う場合、リアルタイム性の改善が必要
- 共有リソースが多くてアプリ間に干渉が生じやすい
組み込みシステムでは高信頼アプリの保護が求められる

- リアルタイム性と信頼性向上に関する技術

カーネル実装の改善

- Low-Latency Desktop
不要な排他制御等を最適化、短いクリティカルセクション
- Fully Preemptible Kernel (PREEMPT_RT)
割込み等の処理中も完全にPreemptできる拡張



リソース分離の強化

- 仮想化技術で信頼度が低いアプリの干渉を制限
軽量仮想化 (コンテナ)、Hypervisor (QEMU/KVM)
- 独立の専用カーネルでリアルタイム処理を行う
LinuxとRTOSを共存させるコカーネル技術Xenomai



ハードウェアアクセラレータの活用

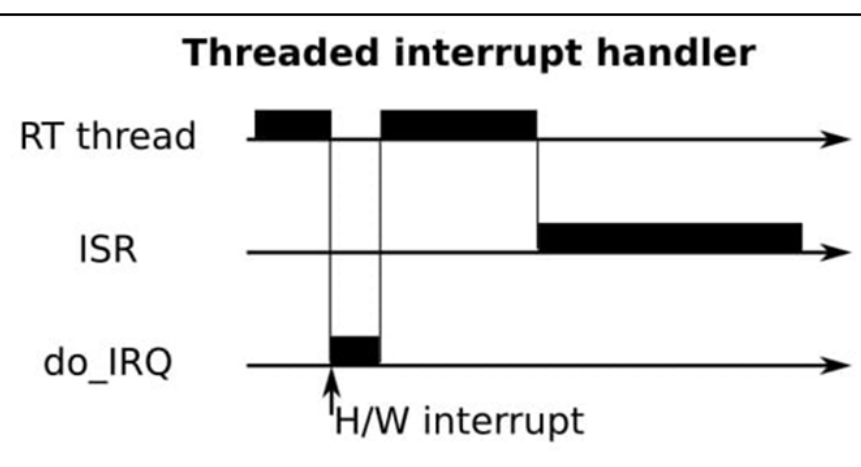
- 遅延要件が厳しい処理 → DSP、FPGA等
- 並列計算が必要な処理 → GPU、NPU等
CPUで処理するとシステム全体の性能が低下



PREEMPT_RTの性能干渉要因の評価と改善

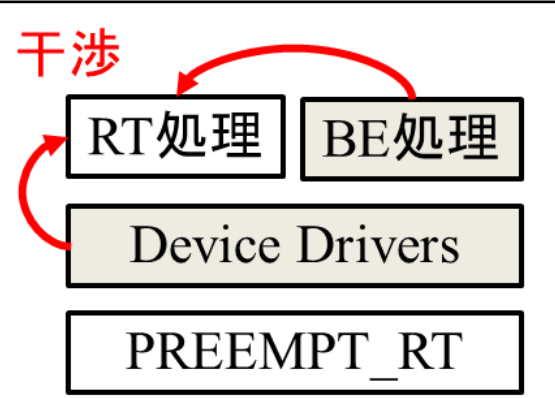
PREEMPT_RTの概要

- カーネルの応答性を向上するパッチ
目標: PREEMPTできない処理を最小限
- 全割込みハンドラのThread化が特徴
ハンドラ本体ISRは低優先度で実行可能
割込み受け付け時はISRの起動だけ行う
RT制御は低優先度のISRより先に処理



既存研究の問題点

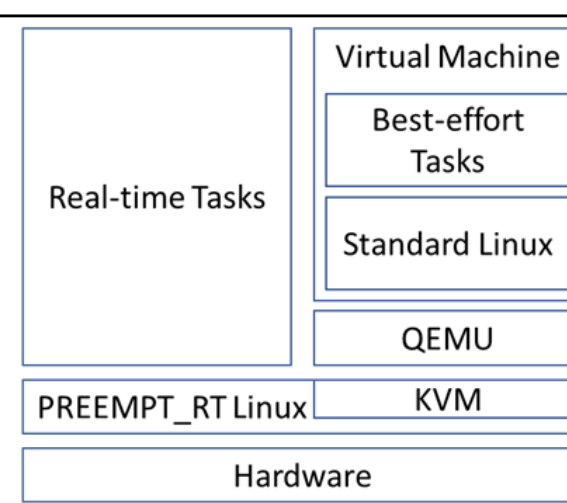
- 評価条件は理想すぎる
殆どはRT処理しかないシミュレーション
実世界は複雑なMCS (Mixed-criticality System)
- カーネル以外の干渉要因は未考慮
Linuxのデバイスドライバは膨大
HW (メモリ、キャッシュ等) も性能に影響



※ A Performance Evaluation of Embedded Multi-core Mixed-criticality System Based on PREEMPT_RT Linux, Journal of Information Processing, 2023 (<https://doi.org/10.2197/ipsjip.31.78>)

MCS評価環境の構築

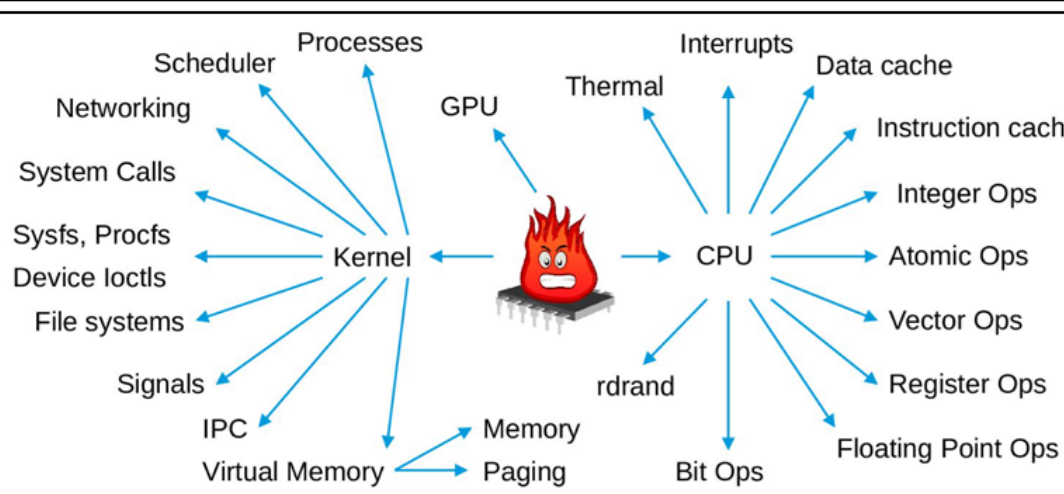
- BEタスクで負荷をかけて干渉を評価
多種多様なストレスが望ましい
- ハードウェアの影響も考慮できる
Performance Counter等の情報で分析
- 仮想マシンで干渉の影響を改善
カーネルの分離による効果を測定



干渉負荷にstress-ngを採用

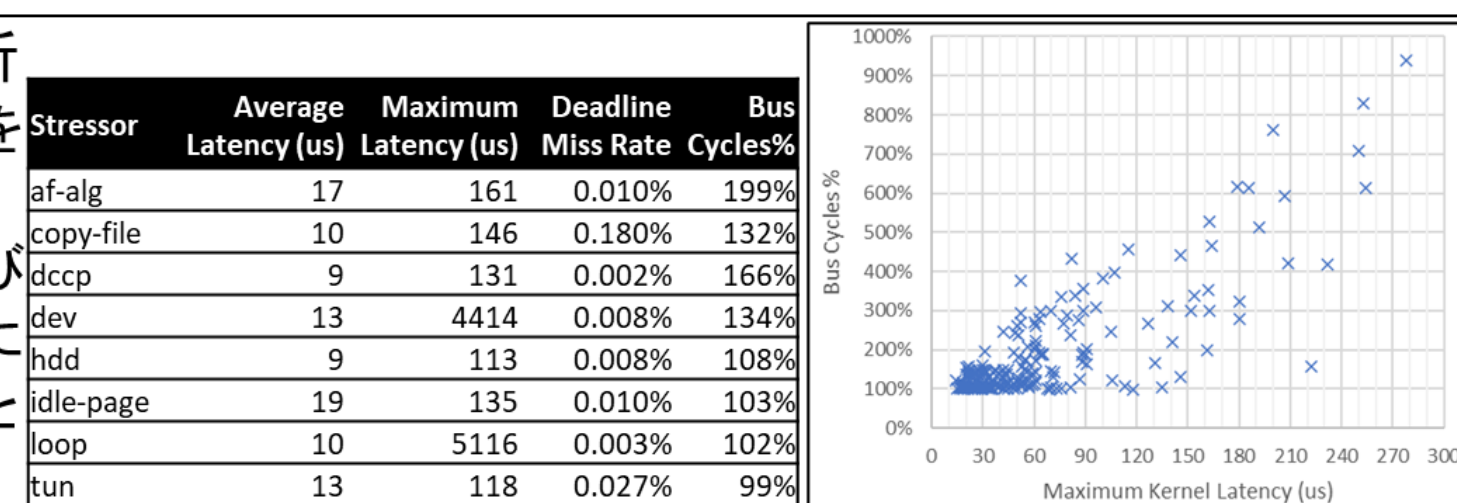
- 大量なストレスを提供
200種類以上のテストプログラム
- システム全体を評価できる
カーネルだけでなく、デバイス
やH/Wの要因も分析可能

※ Kernel Recipes 2023 →



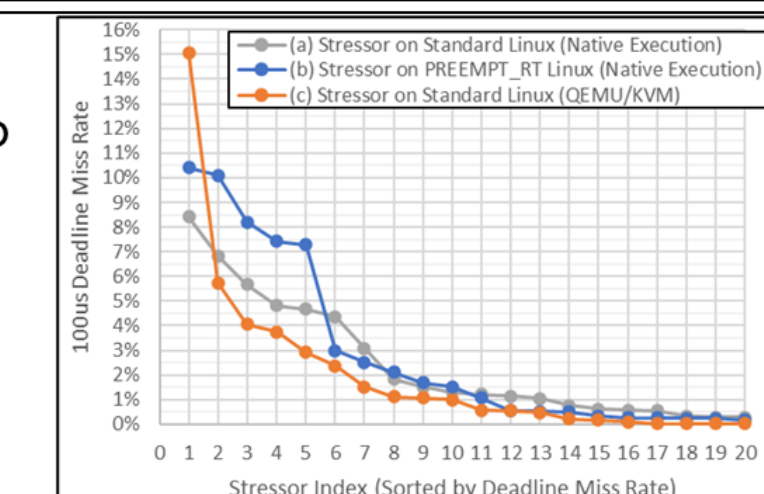
評価データの分析

- 影響大きい負荷を特定できた
- ハードウェア及びデバイスの要因による干渉が多いと判明



仮想マシンによる改善の効果

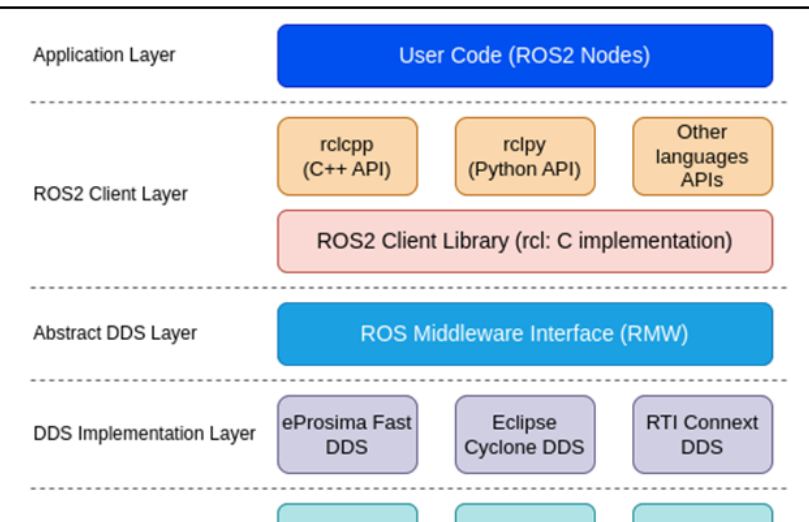
- Native実行だとPREEMPT_RTが悪化の場合もある
Standard Linuxのほうが良いストレス存在
- QEMUによる分離の効果が明らか
殆どのストレスで最も良い性能



ROS2アプリの性能問題解析

ROS2の概要

- ロボット開発向けのフレームワーク
主にLinuxをベースとして構築
- 階層化したアーキテクチャを採用
高度な機能を持つが振舞いも複雑
- 前世代のROS1に比べてRT性能向上
多くのボトルネックを改善
様々な環境で使われる実績があり



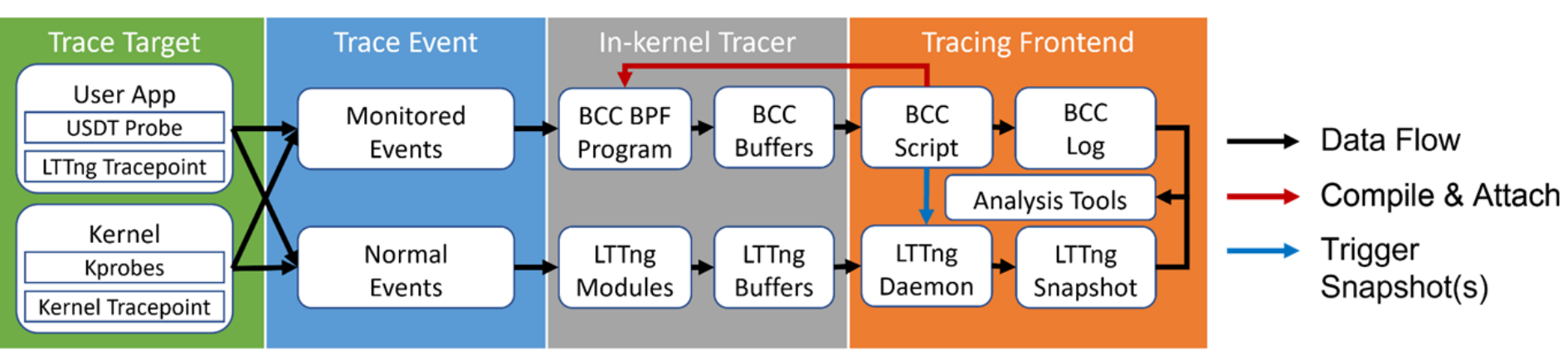
アプリ性能問題の分析が難しい

- 発生頻度が極めて低い場合があり
数日～数週に1回の場合等、テスト時に対処できない
- 原因の特定と解決が非効率
システム的全層をトレースして分析する必要がある

※ Monitor and Analyze Rare ROS2 Performance Issues with A Unified Tracing Framework, The 6th World Symposium on Software Engineering (WSSE 2024), 2024 (Accepted)

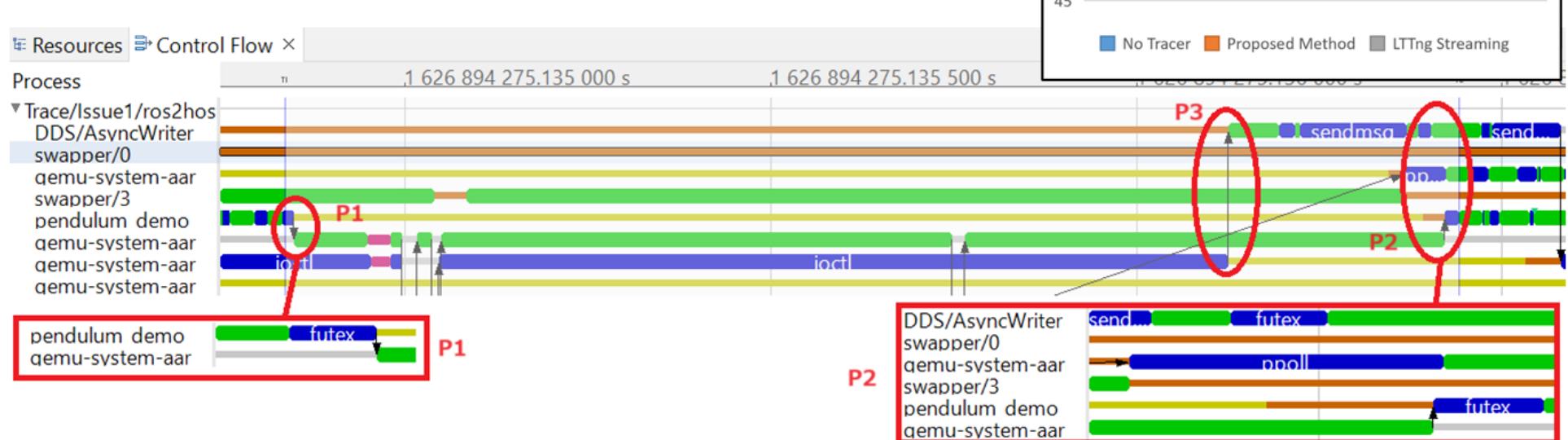
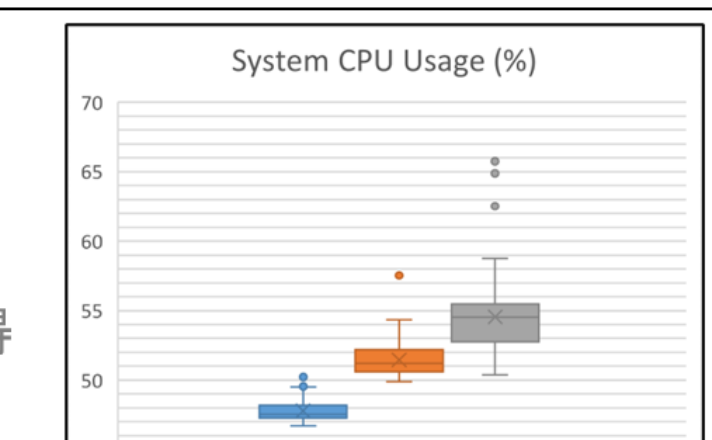
性能解析を効率化できるトレースフレームワーク

- Trace Target
ユーザ空間とカーネル両方をトレースできる
- Trace Event
性能問題の動的検出と詳細分析向け二種類のイベント対応
- In-kernel Tracer
動的に生成モニタリング用プログラム
性能オーバーヘッドが最適化されたLTTngモジュール



提案手法の評価

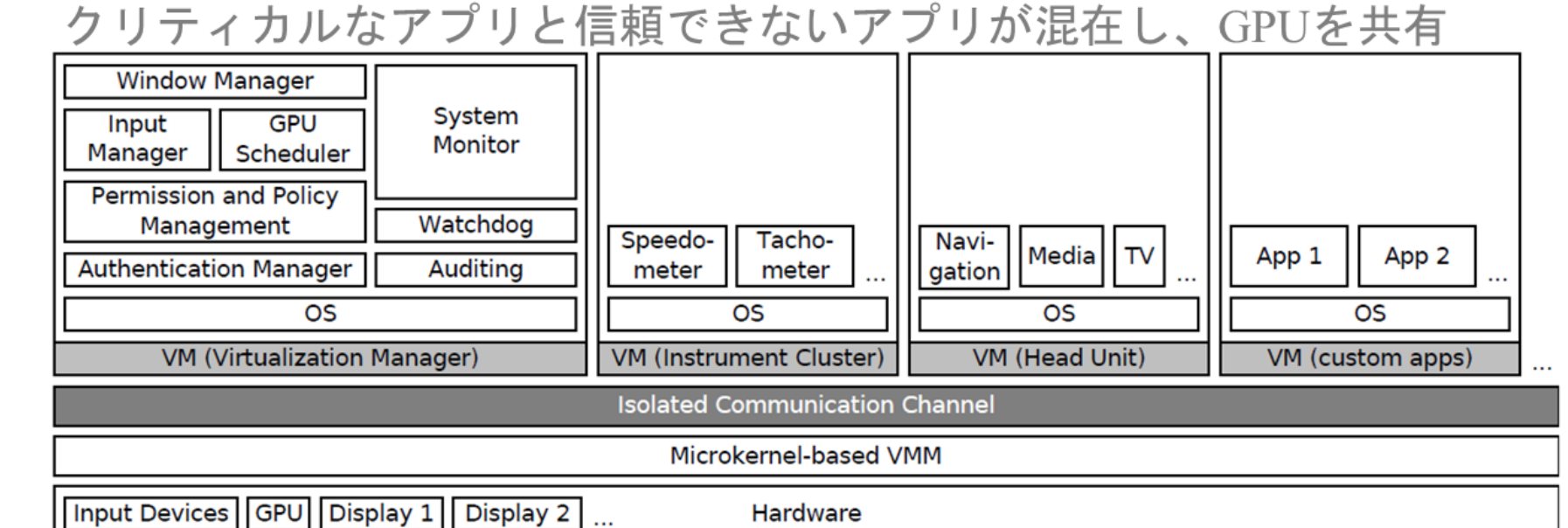
- 既存ツールより低いコストで高機能を実現
CPUコストの絶対値は5%程度
- ROS2フレームワークのボトルネックを発見
性能問題を検出シトレースデータを自動的に取得
DDSミドルウェアによる優先度逆転の例↓



GPU描画のリアルタイム性保証

組み込みシステムにGPU採用が増加

- 車載向けの例

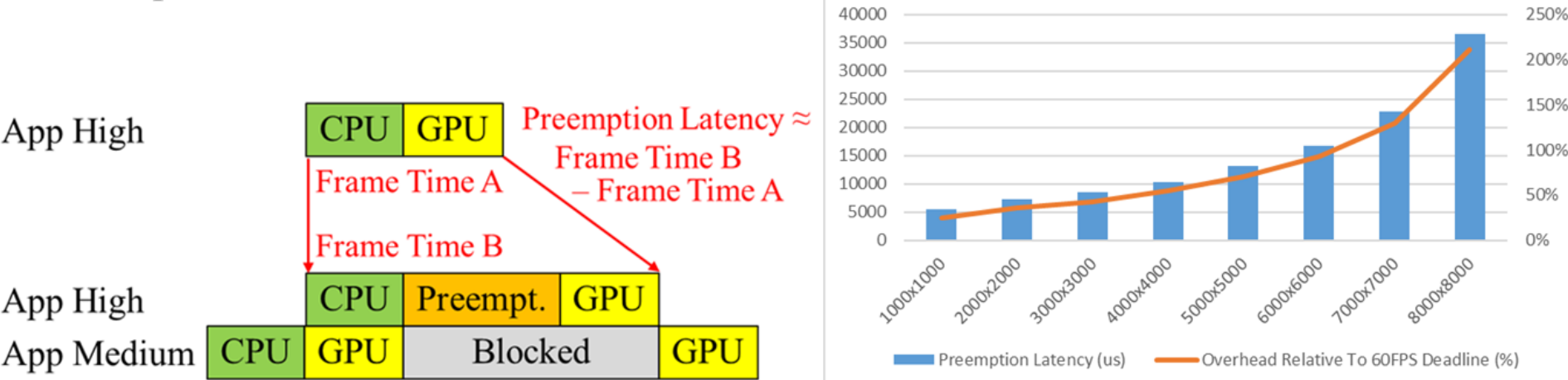
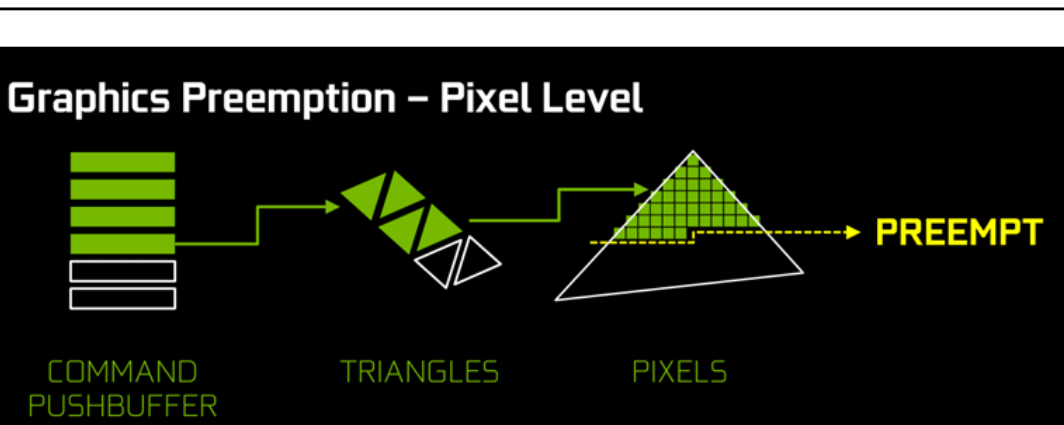


リアルタイム性を保証できるスケジューリング手法が必要

- 既存研究は殆ど描画でなくGPGPU用途と想定
GPGPU処理を呼ぶタスクの応答時間を保証することが目標
- 非信頼アプリの存在を考慮していない場合が多い

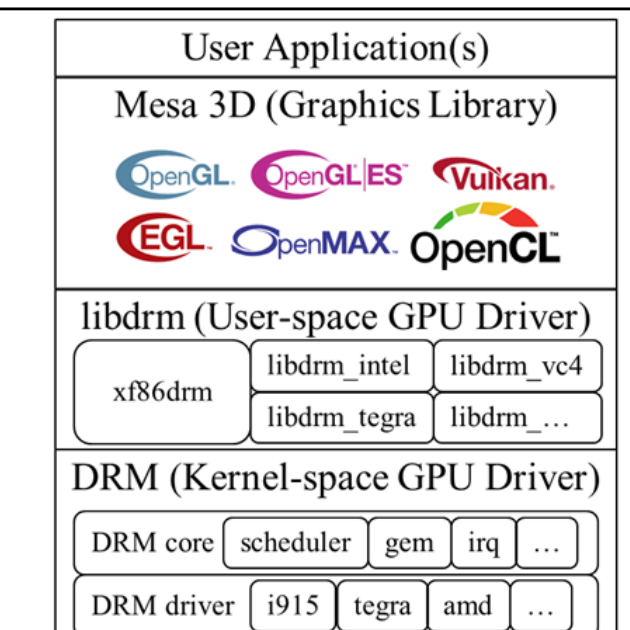
GPUのPreemption機能が普及

- NVIDIAはPixel粒度で可能
- 他のベンダも細粒度Preempt対応
AMD Radeon RX
Intel Iris Xe
Renesas (R-Car Soc)
- 測定では性能コストが大きいと判明
既存のCPU向け手法のように頻繁にPreemptすることができない



描画ライブラリとドライバのオープンソース化

- Linux (Open-source) Graphics Stack (右図→)
IntelとAMDはこのスタックに完全対応
Mesa 3D: OpenGL等の描画ライブラリを実装
libdrm: ユーザ空間GPUドライバ
DRM: カーネル空間GPUドライバ
- NVIDIA Tegra GPUは公式のOSSドライバを提供
現時点、カーネル空間のドライバNVGPUのみ
詳しいドキュメントを提供していない



OSSドライバを拡張して新しいスケジューリング手法を開発中

