小口研究室　研究紹介　(2016年度)
(お茶の水女子大学理学部情報科学科)

ビッグデータ処理基盤のストリーミング機能を利用したセンサデータ解析フレームワークの検討
（研究担当：一瀬　絢衣）

研究背景

- ライフログアプリケーションの普及
  - 防犯・監視用途
  - ペットや子供を遠隔地から見守る

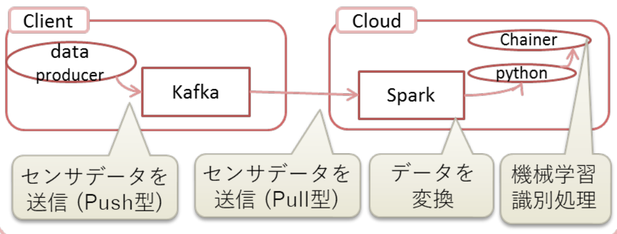
  ✓データ量、計算量の多さ
- ディープラーニング技術の発達

  ✓計算負荷の高さ

クラウドでリアルタイムに動画像解析を行うことは困難

研究概要

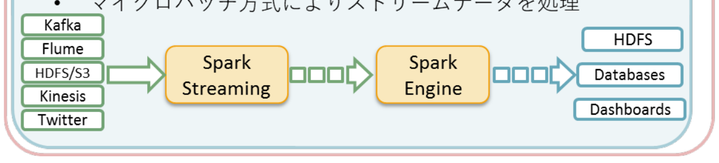
ディープラーニングによるリアルタイムセンサデータ解析フレームワークを検討



関連技術

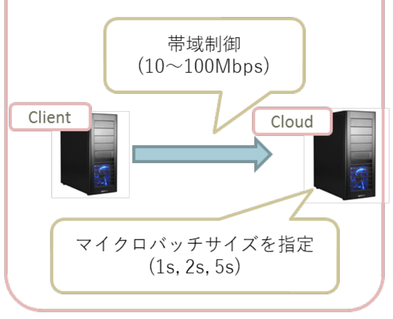
- **Apache Kafka**：分散メッセージングシステム
- **Apache Spark**：分散処理フレームワーク
- **Chainer**：ディープラーニングライブラリ

Spark Streaming
- Sparkのコンポーネントの一つ
- マイクロバッチ方式によりストリームデータを処理



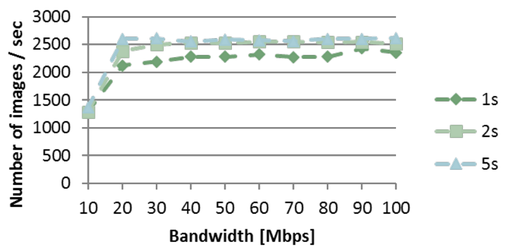
実験環境

MNISTデータセットの画像識別



スループット計測

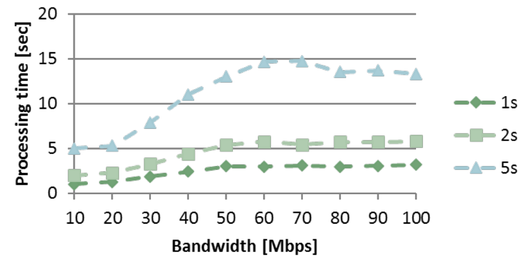
1秒あたりに処理できる画像数を計測



- 20Mbps以上で横ばい
  → クラウド側の処理がボトルネック
- バッチサイズが大きい場合にスループット増加
  Chainer の呼び出しオーバヘッドか削減

リアルタイム性評価

1マイクロバッチの処理時間を計測



- 10Mbps：リアルタイムに処理可能
- 20Mbps以上：　処理時間がマイクロバッチサイズを上回る
- 60Mbps以上：処理時間が横ばい
  →Producer–Kafka間のデータ転送がボトルネック

まとめ

- リアルタイムセンサデータ解析処理における性能要件を検討

今後の課題

- クラスタを用いた分散実行による高速化
- データ転送におけるボトルネックの解析

Android OSにおける協調的輻輳制御手法による異種クライアント端末通信性能の向上
（研究担当：島田 歩実）

研究背景

◆ **近年のロスベースTCP**

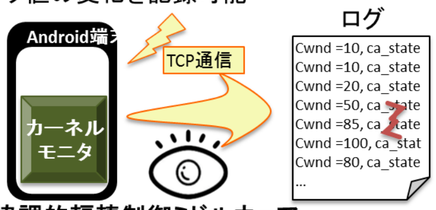
❖高いスループットを得るためにアグレッシブな送信を行う輻輳制御手法

❖無線LANアクセスポイントにおいて多くのACKパケットが蓄積し通信が滞るという問題が発生

先行研究

①カーネルモニタ

通信時におけるLinuxシステムのカーネル内部の処理を解析するシステムツールで、カーネル内部のパラメータ値の変化を記録可能



②協調的輻輳制御ミドルウェア

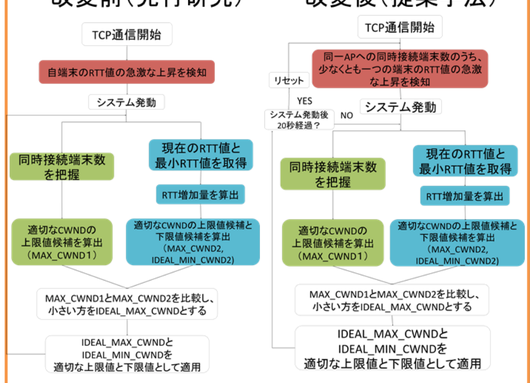
無線LANアクセスポイントにおけるACKパケットの蓄積を回避することで、多数のスマートフォン端末が同時通信するときの全体の通信速度と公平性の向上を実現

研究目的

①と②をスマートフォン端末に加え、タブレット端末にもナイーブに導入し、

さらに、双方にとってより効果的なミドルウェアになるよう、改良していく

輻輳制御手法フローチャート



CWND上限値候補(MAX_CWND1)の計算式

$$MAX\_CWND1 = \frac{\text{Bandwidth (Mbps)} * RTT(sec)}{Segmentsize(1.5Kbyte) * NumberOfTerminals}$$

※Bandwidth = BW * NumberOfTerminals$^{-0.15}$

RTT値によるCWNDの上限値候補(MAX_CWND2), 下限値(IDEAL_CWND)の制御パラメータ

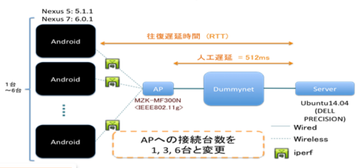
diff_rtt	0～51	51～105	106～161	161～331	331～
MAX_CWND2	555	200	100	10	1
MIN_CWND	200	50	10	2	1

※diff_rtt = current_rtt – min_rtt

改変点
- ❖システム発動タイミング
  - 旧: 自端末のRTT値の急激な増加
  - 新: AP共有端末の少なくとも一つの端末のRTT値急激な増加
- ❖システム発動期間
  - 旧: 一度発動すると制御し続ける
  - 新: 周辺端末の状況により制御手法を自動的にオン・オフ

実験概要

以下の環境において2分間Iperf通信を行い，スマートフォン端末 (Nexus 5)，タブレット端末 (Nexus 7)の性能をそれぞれ評価
(APへの接続台数：1台，3台，6台)

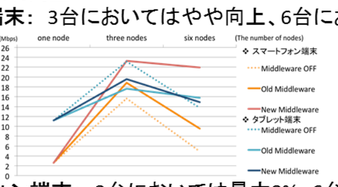


実験結果

◆合計通信速度

スマートフォン端末：3台においては最大48%、6台においては最大347%向上

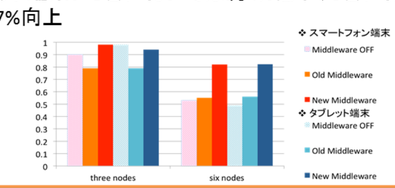
タブレット端末：3台においてはやや向上、6台においては役18%低下



◆公平性

スマートフォン端末：3台においては最大8%、6台においては最大54%向上

タブレット端末：3台においては約4%低下、6台においては最大67%向上



大規模災害時におけるFLAREによるアプリケーション毎のSDN制御手法の実装と評価
（研究担当：平久 紬）

研究背景

❖モバイル端末の高機能化

❖クラウドコンピューティングの発達

様々な種類の膨大なトラフィックが入り交じっている

→ 大規模災害時、緊急情報になかなかアクセスできない

研究目的

ソーシャル情報に基づく経路制御システムの必要性

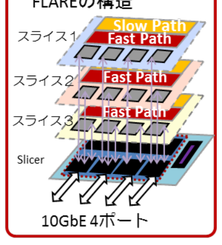
❖実社会の急激な状況変化を検出

❖トラフィック種別を判定し、アプリケーションごとに制御

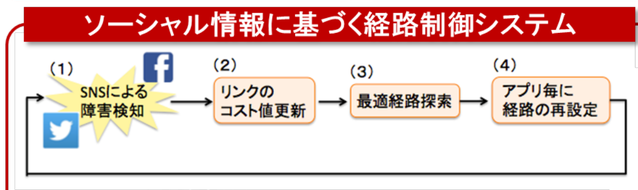
大規模災害時におけるアプリ毎のSDN制御システム実験を行い、
本システムの有効性を示す

FLARE

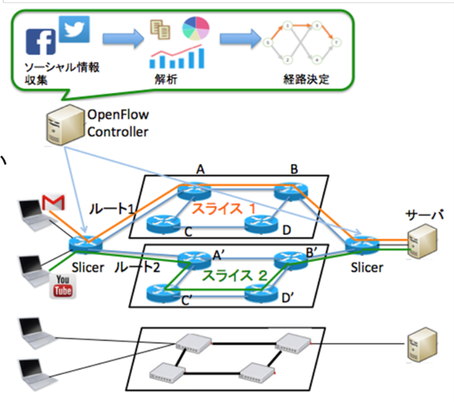
FLAREの構造



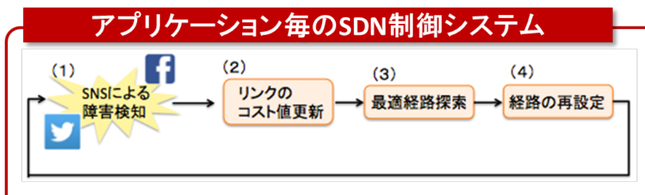
ソーシャル情報に基づく経路制御システム

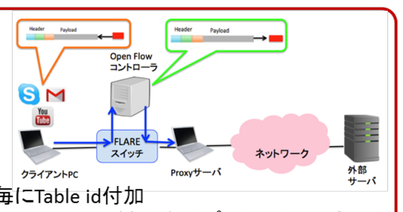


(1) Twitterによる障害検知
　リアルタイムにツイートを監視し,通信障害がどこで起こっているかを検知する.
(2) リンクのコスト値の更新 (60秒ごと)
　初期値:　1
　ツイート中にスイッチと対応させた地名を含むツイートが 20 件以上あったら, + 1
(3) 最適経路探索
　ダイクストラ法による、最適な経路を決定
(4) アプリごとに経路の再設定
　OpenFlowのREST-APIを使用し、コントローラから経路変更



アプリケーション毎のSDN制御システム





(1) Twitterによる障害検知*
(2) アプリのフロー検出
　クライアント PC 上でアプリケーションの識別を行う.
　プロセステーブルと宛先ポート番号からアプリを識別.
　パケットの後ろにアプリケーション名をタグとして付ける.
(3) アプリ毎にTable id付加
　クライアントPC上で付加したアプリケーション名からアプリを識別し、アプリ毎にTable idを付加する.
(4) Table id毎に帯域制限
　OpenFlowのREST-APIを使用し、コントローラからTable id毎に帯域制限

実験①

- 東日本大震災時(2011年3月11日 14:00~15:00)のTweetをもとに実験
- 東北大から東大への通信を想定

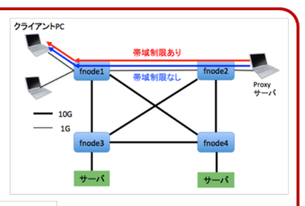
実験結果

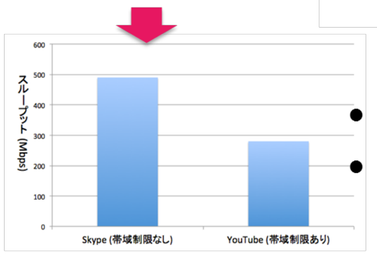
検知前
東北大→東大

検知後
東北大→大手町→名古屋

実験②

- クライアントPC上でYouTubeとSkype起動
- Skypeは災害時に重要となるアプリであると判断
- YouTubeのみ帯域制限





帯域制限していないパケットは約500Mbps
帯域制限しているパケットは300Mbps以下におさえられている.

http://ogl.is.ocha.ac.jp/

oguchi@is.ocha.ac.jp