# Load Shedding in Network Monitoring Applications

Pere Barlet Ros
(pbarlet@ac.upc.edu)

Universitat Politècnica de Catalunya (UPC)
Departament d'Arquitectura de Computadors

Workshop on Video Streaming over MANETs
Barcelona, December 18-19, 2008

# Outline

# Outline

1. Introduction
   - Motivation
   - Related work
   - Contributions
2. Prediction and Load Shedding Scheme
3. Conclusions and Future Work

# Motivation

- Network monitoring is crucial for operating data networks
  - Traffic engineering, network troubleshooting, anomaly detection . . .
- Monitoring systems are prone to dramatic *overload* situations
  - Link speeds, anomalous traffic, bursty traffic nature . . .
  - Complexity of traffic analysis methods
- Overload situations lead to *uncontrolled* packet loss
  - *Severe* and *unpredictable* impact on the accuracy of applications
  - . . . when results are most valuable!!

# Motivation

- Network monitoring is crucial for operating data networks
  - Traffic engineering, network troubleshooting, anomaly detection . . .
- Monitoring systems are prone to dramatic *overload* situations
  - Link speeds, anomalous traffic, bursty traffic nature . . .
  - Complexity of traffic analysis methods
- Overload situations lead to *uncontrolled* packet loss
  - *Severe* and *unpredictable* impact on the accuracy of applications
  - . . . when results are most valuable!!

## Load Shedding Scheme

- Efficiently handle extreme overload situations
- Over-provisioning is not feasible

# Related Work

- Load shedding in data stream management systems (DSMS)
  - Examples: Aurora, STREAM, TelegraphCQ, Borealis, . . .
  - Based on declarative query languages (e.g., CQL)
  - Small set of operators with known (and constant) cost
  - Maximize an aggregate performance metric (utility or throughput)

## Limitations

- Restrict the type of metrics and possible uses
- Assume explicit knowledge of operators' cost and selectivity
- Not suitable for non-cooperative environments

- Resource management in network monitoring systems
  - Restricted to a pre-defined set of metrics
  - Limit the amount of allocated resources in advance

# Contributions

1. Prediction method
   - Operates w/o knowledge of application cost and implementation
   - Does not rely on a specific model for the incoming traffic
2. Load shedding scheme
   - Anticipates overload situations and avoids packet loss
   - Relies on packet and flow sampling (equal sampling rates)
3. Packet-based scheduler
   - Applies different sampling rates to different queries
   - Ensures fairness of service with non-cooperative applications
4. Support for custom-defined load shedding methods
   - Safely delegates load shedding to non-cooperative applications
   - Still ensures robustness and fairness of service

# Outline

# Case Study: Intel CoMo

- CoMo (Continuous Monitoring)[1]
  - Open-source passive monitoring system
  - Framework to develop and execute network monitoring applications
  - Open (shared) network monitoring platform

- Traffic queries are defined as *plug-in* modules written in C
  - Contain complex computations

[1] `http://como.sourceforge.net`

# Case Study: Intel CoMo

- CoMo (Continuous Monitoring)[1]
  - Open-source passive monitoring system
  - Framework to develop and execute network monitoring applications
  - Open (shared) network monitoring platform

- Traffic queries are defined as *plug-in* modules written in C
  - Contain complex computations

**Traffic queries are black boxes**

- Arbitrary computations and data structures
- Load shedding cannot use knowledge of the queries

[1] `http://como.sourceforge.net`

# Load Shedding Approach

## Working Scenario

- Monitoring system supporting multiple arbitrary queries
- Single resource: CPU cycles

## **Approach:** Real-time modeling of the queries' CPU usage

1. Find correlation between **traffic features** and CPU usage
   - Features are **query agnostic** with **deterministic worst case** cost
2. Exploit the correlation to predict CPU load
3. Use the prediction to decide the sampling rate

# System Overview

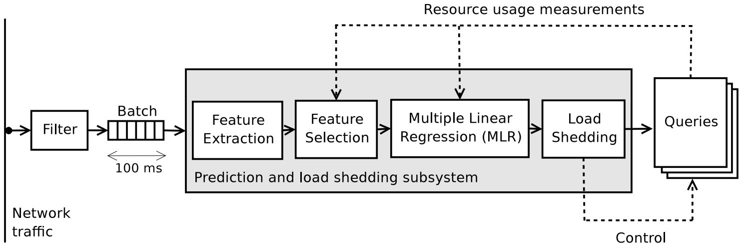



Figure: Prediction and Load Shedding Subsystem

# Traffic Features vs CPU Usage

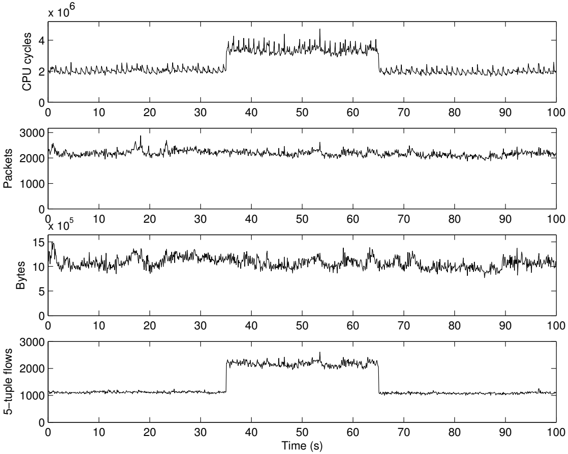



Figure: CPU usage compared to the number of packets, bytes and flows

# Traffic Features vs CPU Usage

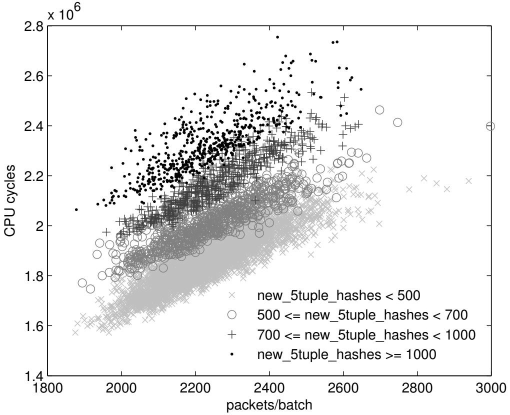



Figure: CPU usage versus the number of packets and flows

# Prediction Methodology[2]

## Multiple Linear Regression (MLR)

$$Y_i = \beta_0 + \beta_1 X_{1i} + \beta_2 X_{2i} + \cdots + \beta_p X_{pi} + \varepsilon_i, \qquad i = 1, 2, \ldots, n.$$

- $Y_i$ = $n$ observations of the response variable (measured cycles)
- $X_{ji}$ = $n$ observations of the $p$ predictors (traffic features)
- $\beta_j$ = $p$ regression coefficients (unknown parameters to estimate)
- $\varepsilon_i$ = $n$ residuals (OLS minimizes SSE)

[2] P. Barlet-Ros et al. "Load Shedding in Network Monitoring Applications", Proc. of USENIX Annual Technical Conference, 2007.

# Prediction Methodology[2]

## Multiple Linear Regression (MLR)

$$Y_i = \beta_0 + \beta_1 X_{1i} + \beta_2 X_{2i} + \cdots + \beta_p X_{pi} + \varepsilon_i, \qquad i = 1, 2, \ldots, n.$$

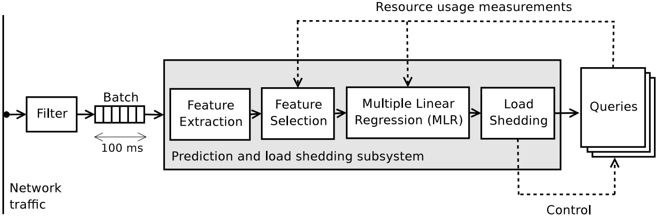
- $Y_i$ = $n$ observations of the response variable (measured cycles)
- $X_{ji}$ = $n$ observations of the $p$ predictors (traffic features)
- $\beta_j$ = $p$ regression coefficients (unknown parameters to estimate)
- $\varepsilon_i$ = $n$ residuals (OLS minimizes SSE)

## Feature Selection

- Variant of the Fast Correlation-Based Filter (FCBF)
- Removes **irrelevant** and **redundant** predictors
- Reduces significantly the cost and improves accuracy of the MLR

[2] P. Barlet-Ros et al. "Load Shedding in Network Monitoring Applications", Proc. of USENIX Annual Technical Conference, 2007.

# Load Shedding Scheme[3]




## Prediction and Load Shedding subsystem

1. Each 100*ms* of traffic is grouped into a *batch* of packets
2. The traffic features are efficiently extracted from the batch (multi-resolution bitmaps)
3. The most relevant features are selected (using FCBF) to be used by the MLR
4. MLR predicts the CPU cycles required by each query to run
5. Load shedding is performed to discard a portion of the batch
6. CPU usage is measured (using TSC) and fed back to the prediction system

[3] P. Barlet-Ros et al. "On-line Predictive Load Shedding for Network Monitoring", Proc. of IFIP-TC6 Networking, 2007.

# Results: Load Shedding Performance

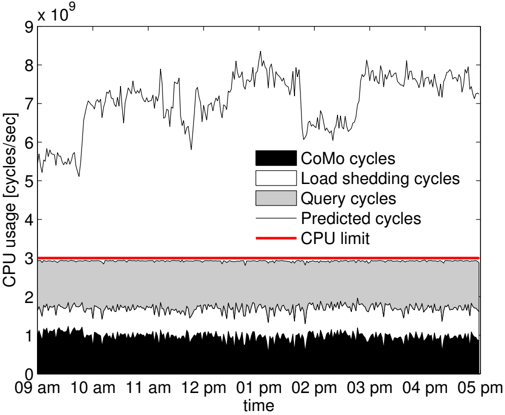



Figure: Stacked CPU usage (Predictive Load Shedding)

# Results: Load Shedding Performance

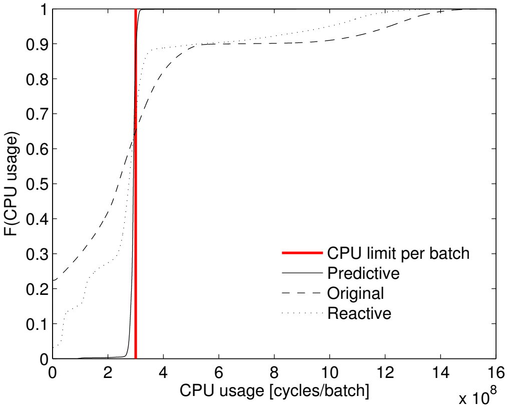
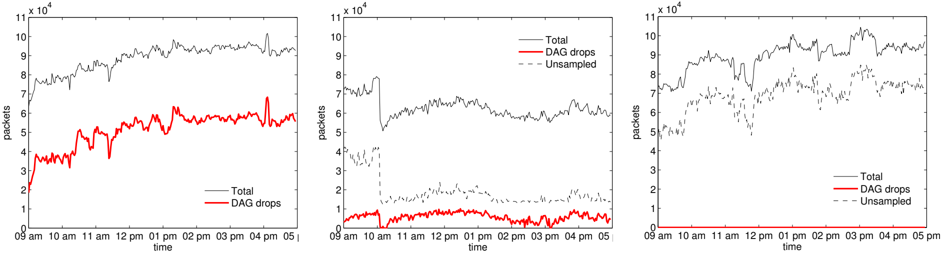



Figure: CDF of the CPU usage per batch

# Results: Packet Loss



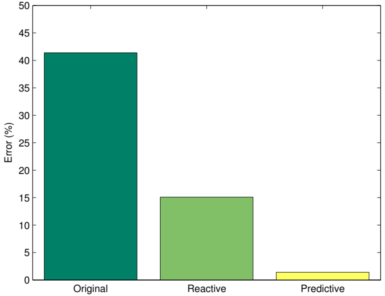

(a) Original CoMo (b) Reactive Load Shedding (c) Predictive Load Shedding

Figure: Link load and packet drops

# Results: Error of the Queries

| Query | original | reactive | predictive |
|---|---|---|---|
| *application (pkts)* | 55.38% ±11.80 | 10.61% ±7.78 | 1.03% ±0.65 |
| *application (bytes)* | 55.39% ±11.80 | 11.90% ±8.22 | 1.17% ±0.76 |
| *counter (pkts)* | 55.03% ±11.45 | 9.71% ±8.41 | 0.54% ±0.50 |
| *counter (bytes)* | 55.06% ±11.45 | 10.24% ±8.39 | 0.66% ±0.60 |
| *flows* | 38.48% ±902.13 | 12.46% ±7.28 | 2.88% ±3.34 |
| *high-watermark* | 8.68% ±8.13 | 8.94% ±9.46 | 2.19% ±2.30 |
| *top-k destinations* | 21.63 ±31.94 | 41.86 ±44.64 | 1.41 ±3.32 |

Table: Errors in the query results (*mean* ± *stdev*)



# Outline

# Conclusions

- Effective load shedding methods are now a basic requirement
  - Increasing data rates, number of users and application complexity
  - Robustness against traffic anomalies and attacks
- Predictive load shedding scheme
  - Operates without knowledge of the traffic queries
  - Does not rely on a specific model for the input traffic
  - Anticipates overload situations avoiding uncontrolled packet loss
  - Graceful performance degradation (sampling & custom methods)
  - Suitable for non-cooperative environments
- Results in two operational networks show that:
  - The system is robust against severe overload
  - The impact on the accuracy of the results is minimized

# Future Work

- Study other system resources
  - Examples: memory, disk bandwidth, storage space, . . .
  - Multi-dimensional load shedding schemes
- Extend the prediction model
  - Study queries with non-linear relationships with traffic features
  - Include payload-related and entropy-based features
- Address resource management problem in a distributed platform
  - Load balancing and distribution techniques
  - Other metrics: bandwidth between nodes, query delays, . . .

# Availability

- The source code of load shedding system is publicly available at http://loadshedding.ccaba.upc.edu
- The CoMo monitoring system is available at http://como.sourceforge.net

powered by CoMo project Intel Research Cambridge


## Acknowledgments

- This work was funded by a University Research Grant awarded by the **Intel Research Council** and the **Spanish Ministry of Education** under contracts TSI2005-07520-C03-02 (CEPOS) and TEC2005-08051-C03-01 (CATARO)
- We thank **CESCA** and **UPCnet** for allowing us to evaluate the load shedding prototype in the Catalan NREN and UPC network respectively.