KTH Electrical Engineering

In-Network Management

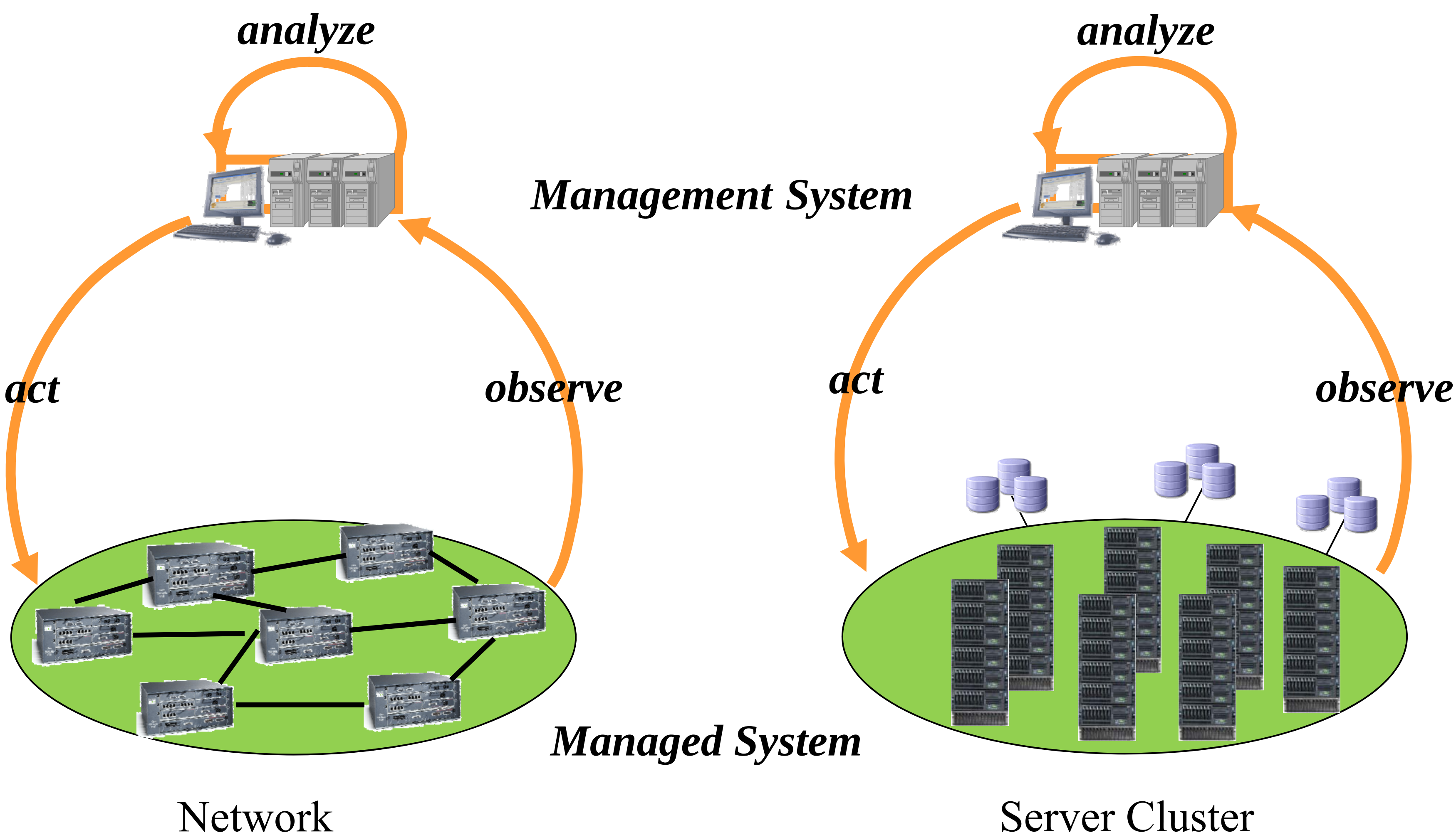
Rolf Stadler
KTH Royal Institute of Technology
Stockholm, Sweden

19th International Conference on Computer Communication and Networks (ICCCN 2010) August 2–5, 2010, Zurich, Switzerland

Outline

- Network Management
- In-Network Management
- Case Study: Real-time Monitoring
- Will it happen?

Management Systems



What is Network Management?

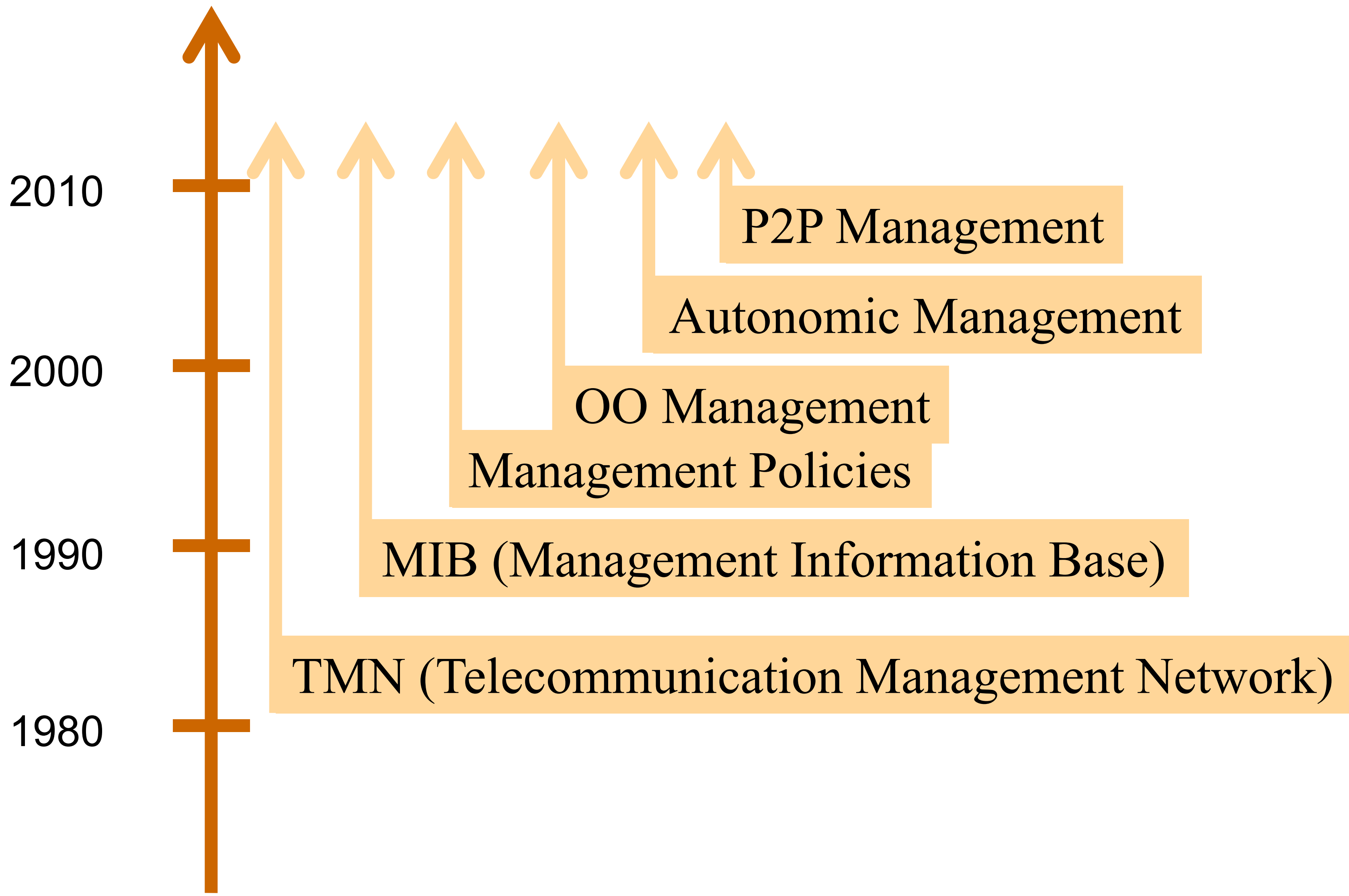
Network Management refers to the activities, methods, procedures, and tools that pertain to the operation, administration, maintenance and provisioning of networked systems *...A. Clemm, 2006.*

Management of Networks and Networked Systems involves the following five tasks (FCAPS).

- Fault Management
- Configuration Management
- Accounting Management & User Administration
- Performance Management
- Security Management

...definition from the telecom community, late 1980s.

Network Management Paradigms



Network Management Conferences

Yearly conference in spring:

- IEEE/IFIP IM (International Symposium on Integrated Network Management)
- IEEE/IFIP NOMS (Network Operations and Management Symposium)

Single-track event in fall:

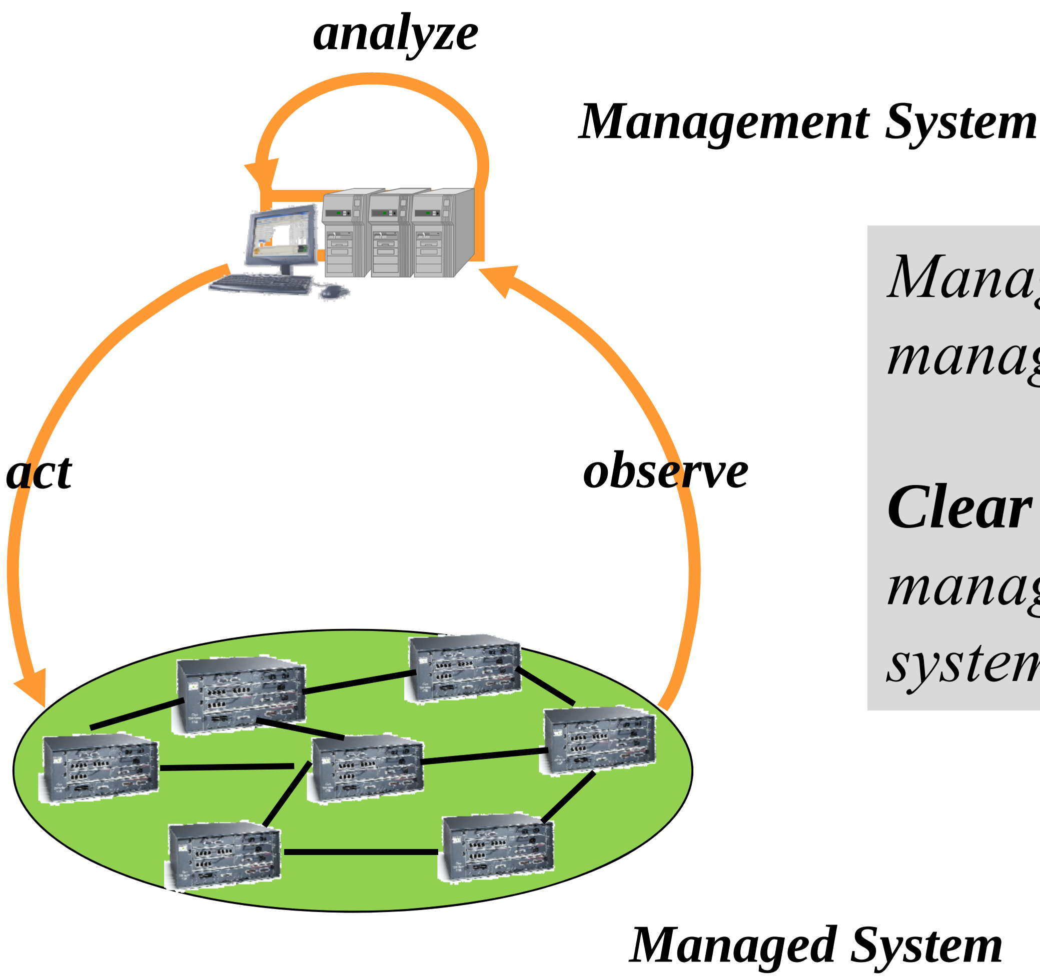
- IEEE DSOM (Distributed Systems Operation and Management)
- IEEE CNSM (Conference on Network and Service Management)

Network Management Journals

- ***IEEE Transactions on Network and Service Management (TNSM)***
  since 2007
- ***Journal of Network and Service Management (JNSM)***
  since 1993, published by Springer
- ***IEEE Communications Magazine***
  Series on Network and Service Management twice a year

- Network Management
- In-Network Management
- Case Study: Real-time Monitoring
- Will it happen?

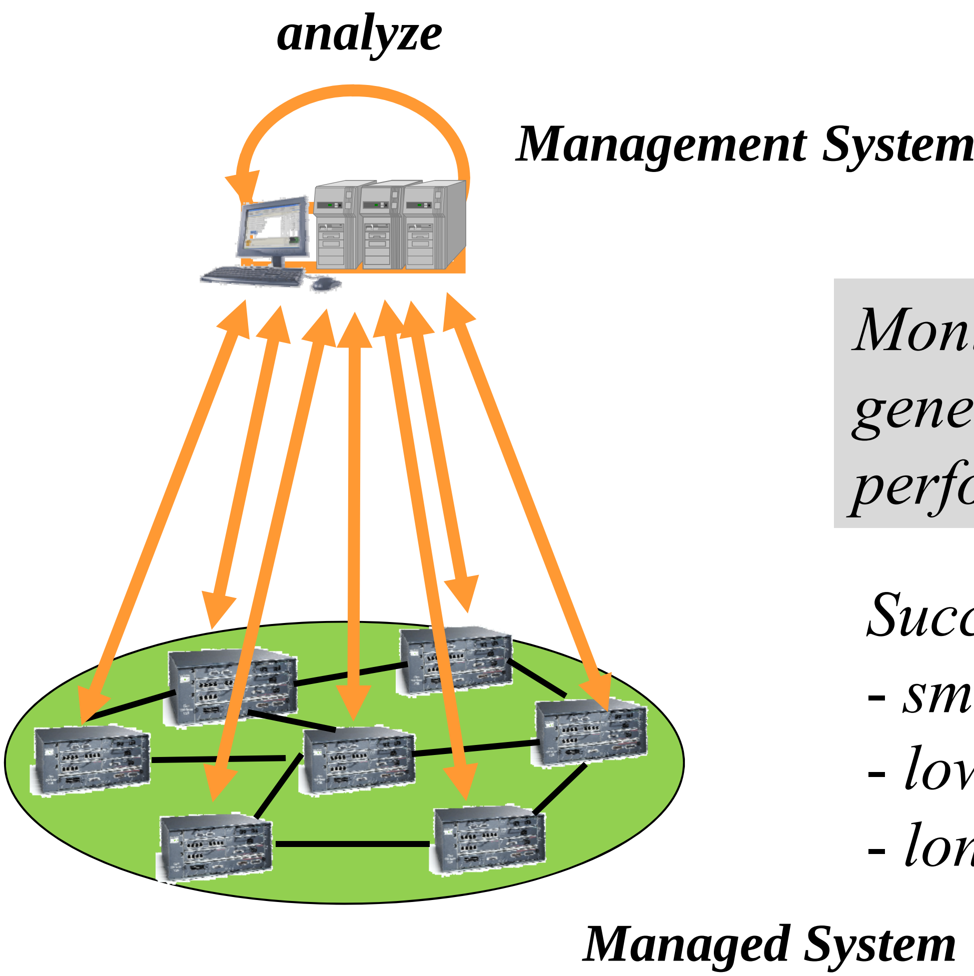
Today's Management Systems for Traditional Network Technologies



Management intelligence ***outside*** *managed system.*

Clear separation *between management system and managed system,* ***by design.***

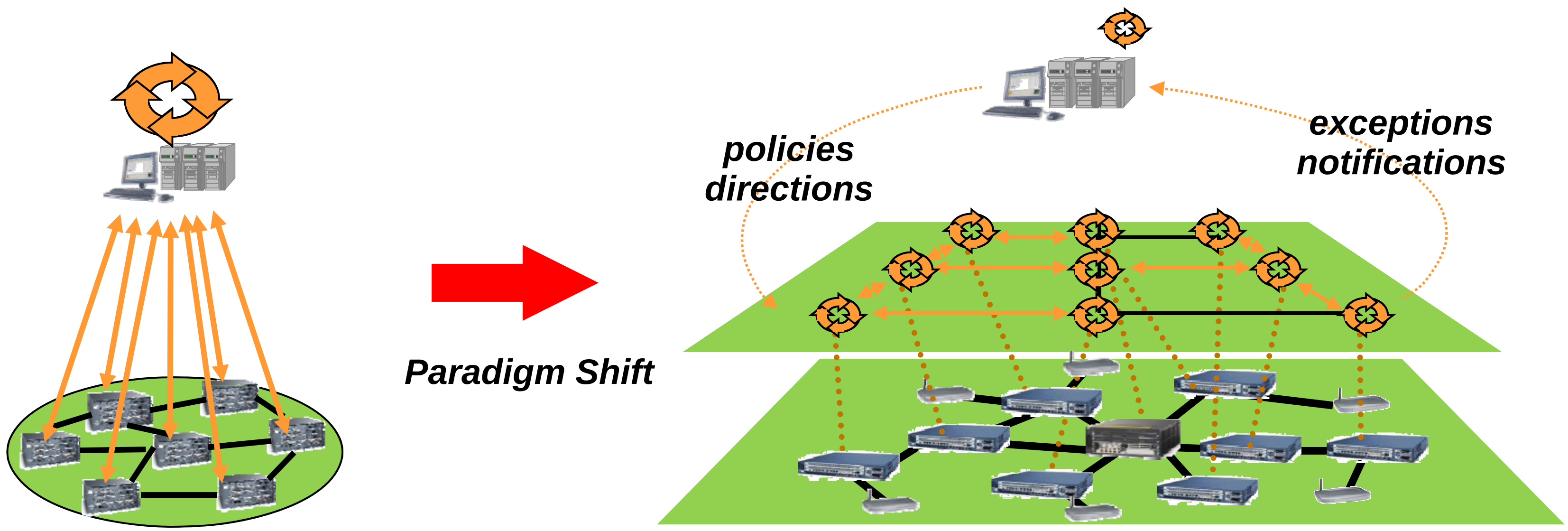
Today's Management Systems for Traditional Network Technologies (2)



Monitoring and configuration, generally FCAPS functions, performed on a ***per-device basis****.*

Successful for
- *small number of nodes (<1000)*
- *low rate of change*
- *long reaction cycles (<1 sec)*

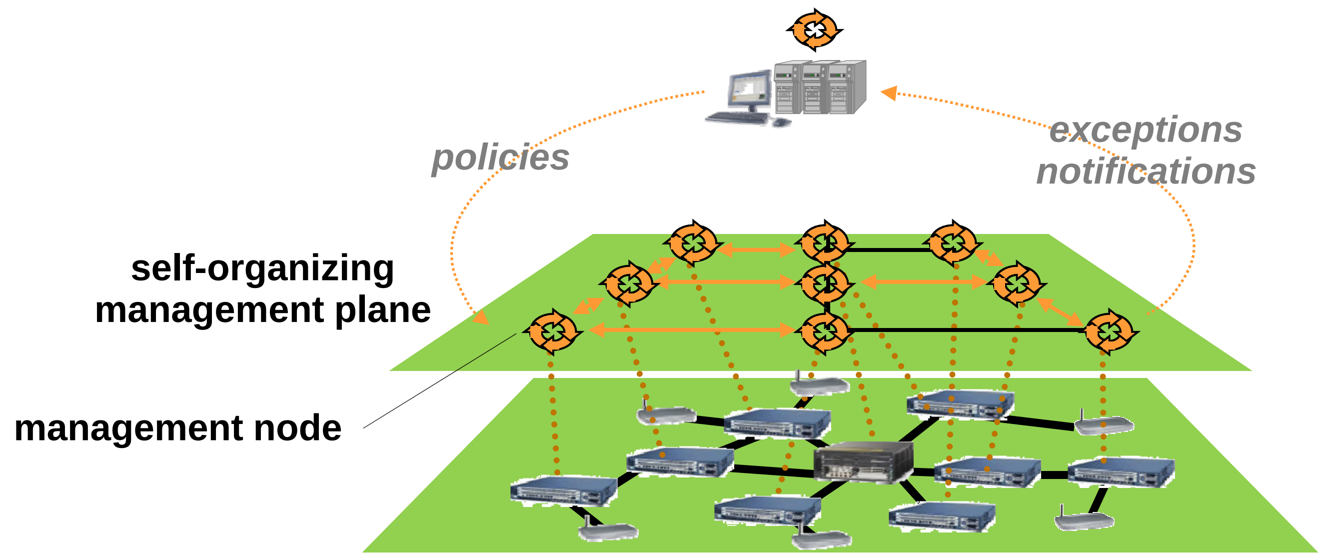
In-Network Management: Key Idea



Reduce interactions between management and managed systems

- Place management functions inside the managed systems
- Delegate tasks to a self-organizing ***management plane***
- Enabling concepts: embedding, decentralization, self-organization

In-Network Management: Engineering Aspects

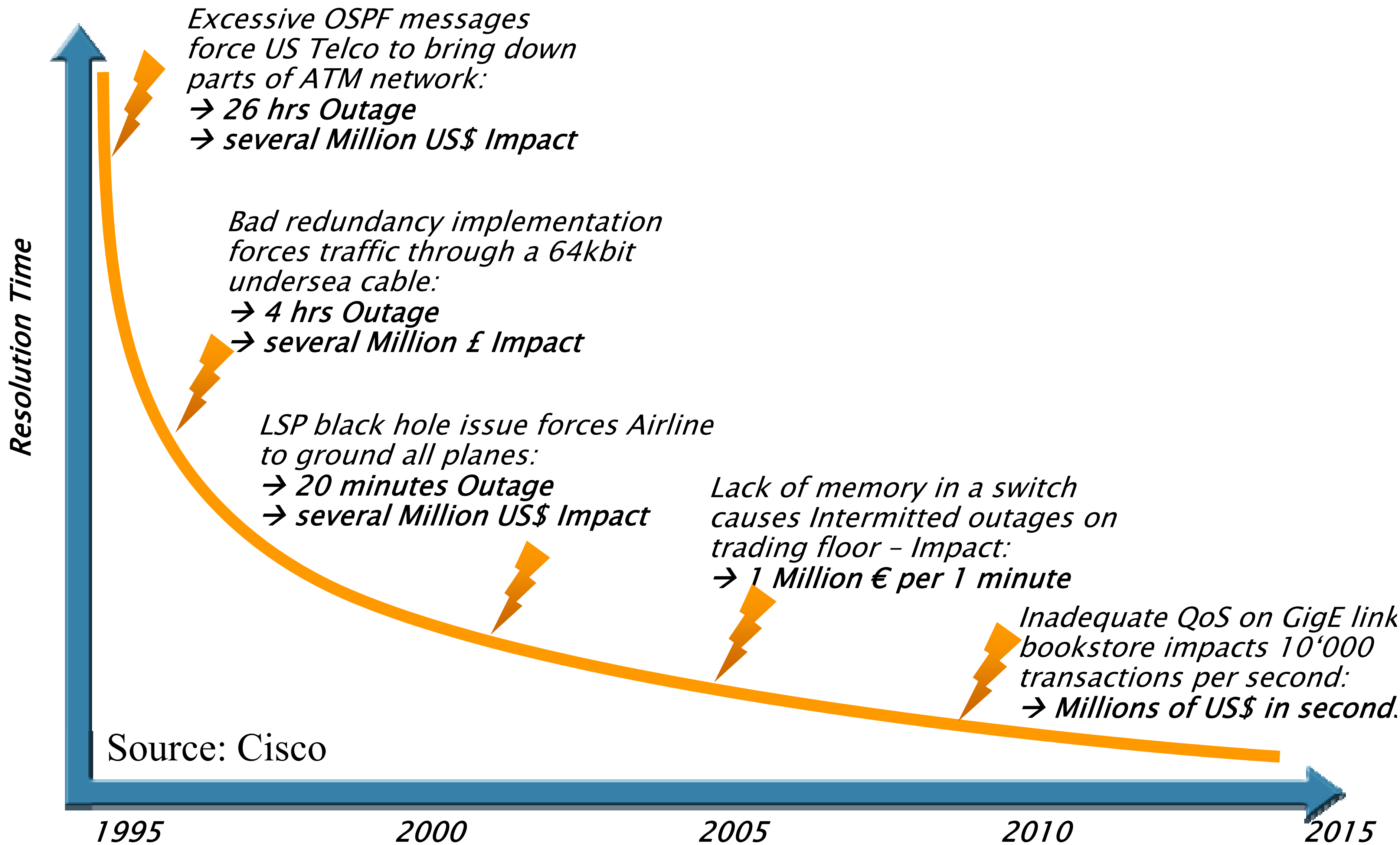


- ***Management nodes*** with processing capabilities
  —inside device, blade, appliance
- ***Peer interaction*** through neighborhood concept—overlay
- Management functions execute as
  distributed algorithms on overlay graph;
  can be invoked on each node;
  are part of a self-organizing management plane

The Drivers for In-Network Management

- Lack of management infrastructure
  energy-constraint environment
    ---sensor networks, MANETs, vehicular networks
- Avoiding bottlenecks in large-scale systems
    ---access networks, data centers, managed end-devices
- Shorten reaction time
    -dynamic environments
    -mission-critical networks
- State can be estimated and acted upon inside the network
    - Fault management
    - Routing, resource allocation

Fault Resolution Times



Side Thought: A Revival of Network Programming?

Initiatives 1995-2005:

• ***Active Networking***: active packets with state and code, customized packet processing on routers;
pursued by Internet community

• ***Programmable Networks***: focus on interfaces, e.g., for connection management, QoS;
pursued by broadband community, standardization (IEEE P1520)

Impact:

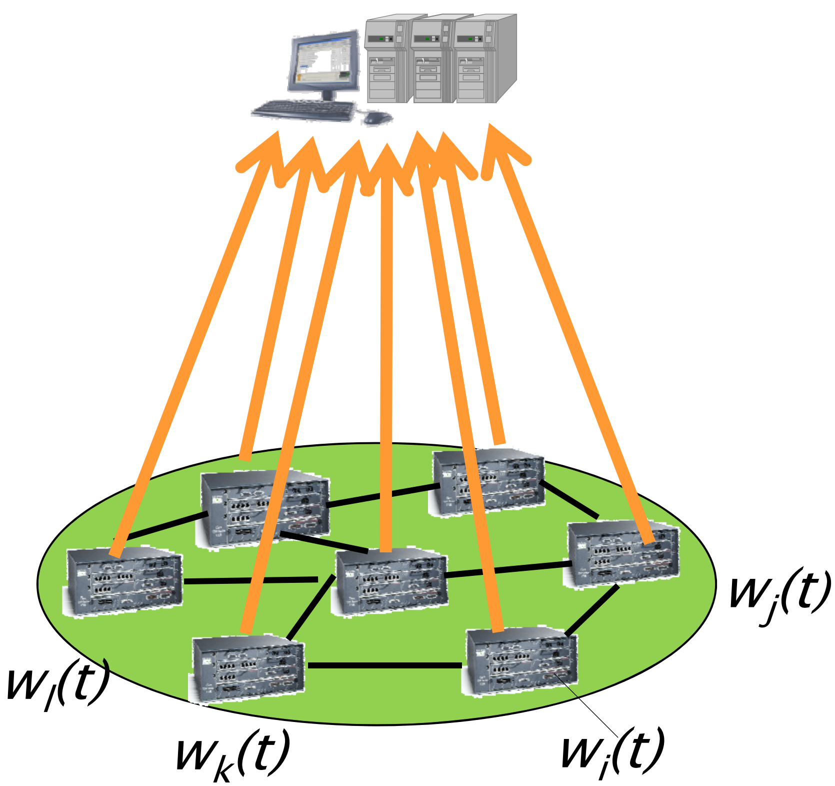
- in specialized technologies—programmable layer 4/7 switches, intelligent firewalls, …
- limited industrial impact—no adoption by major manufacturers; operators and providers valued operational safety over flexibility

- Network Managements
- In-Network Management
- Case Study: Real-time Monitoring
- Will it happen?

Monitoring Aggregates

Aggregate

$F(t)=F(w_1(t),\ldots\ldots, w_n(t))$



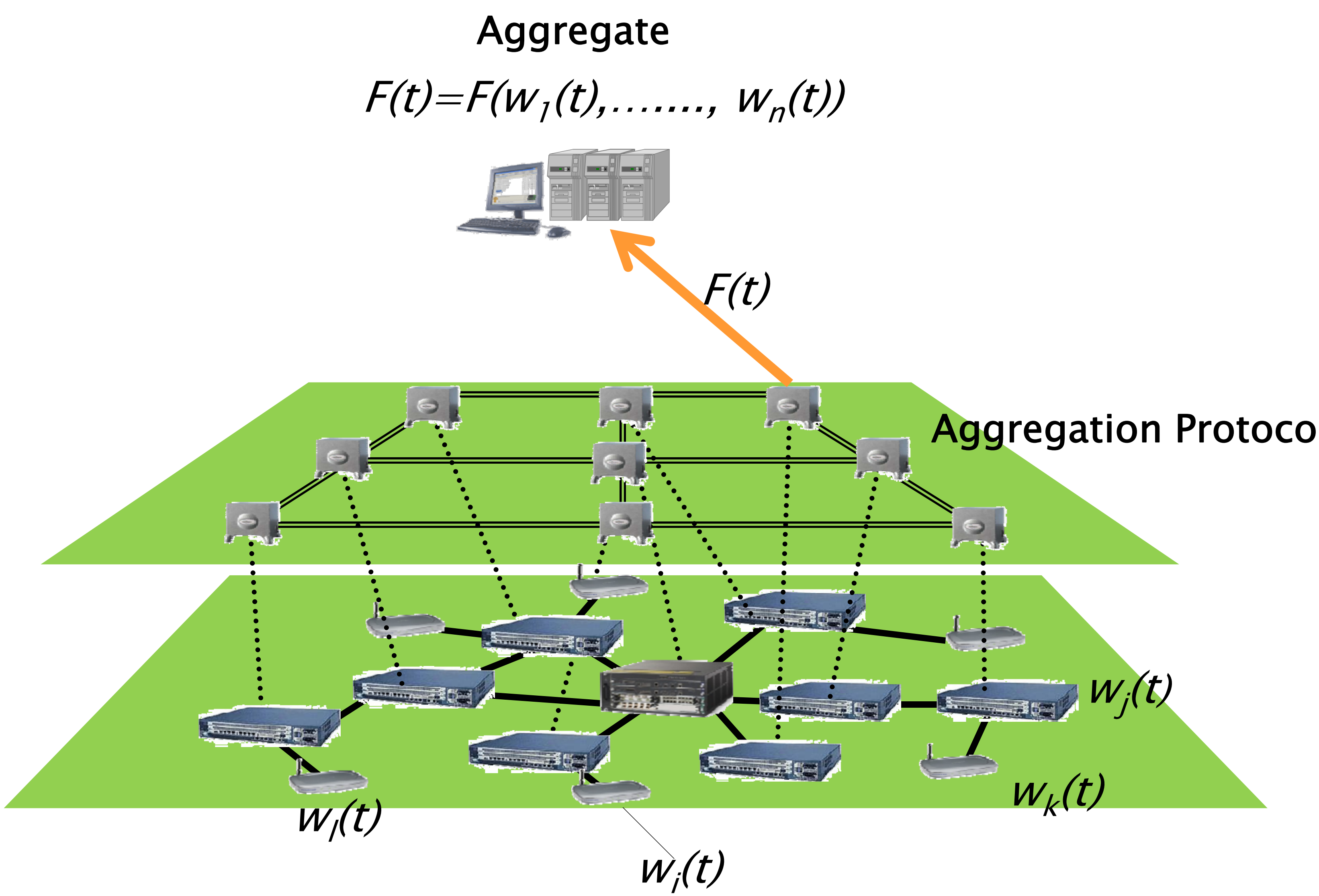
Local variables

Aggregation functions F()

$F(\ldots,w_i, \ldots., w_j, \ldots)=F(\ldots,w_j, \ldots., w_i, \ldots)$

- *Sum* $(w_1,\ldots, w_n)$, *Average(…), Max(…), Quantile(…)*
- *Distinctive Elements* $\{w_1,\ldots, w_n\}$ *Heavy hitters {… }*
- *Histogram* $\{w_1,\ldots, w_n\}$

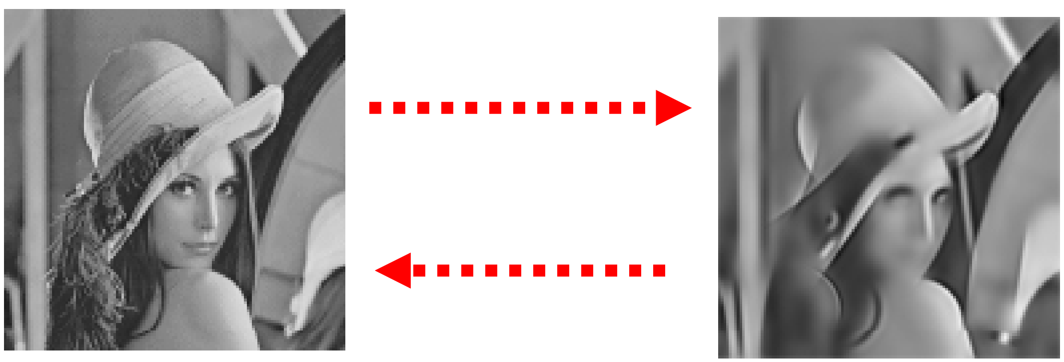
Decentralized Monitoring



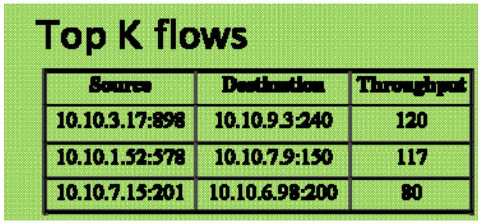
Challenges

Estimation of network states, situation awareness, threshold detection….

- ***Understanding and controlling trade-offs*** between accuracy, overhead, robustness, … dependency on the system size, dynamicity, … to build ***tunable and self-tuning systems***

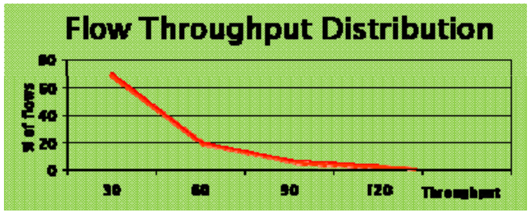


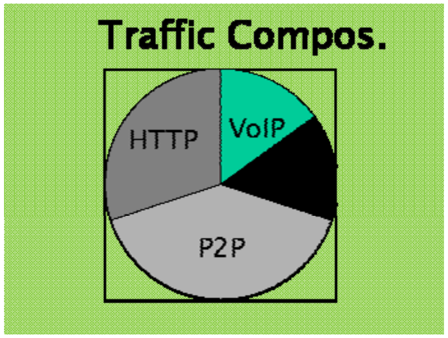
- Understanding the semantics of mgt operations on a large system under change
- Understanding the impact of estimation errors on the effect of management decisions



Top K flows

Source	Destination	Throughput
10.10.3.17:898	10.10.9.3:240	120
10.10.1.52:578	10.10.7.9:150	117
10.10.7.15:201	10.10.6.98:200	80





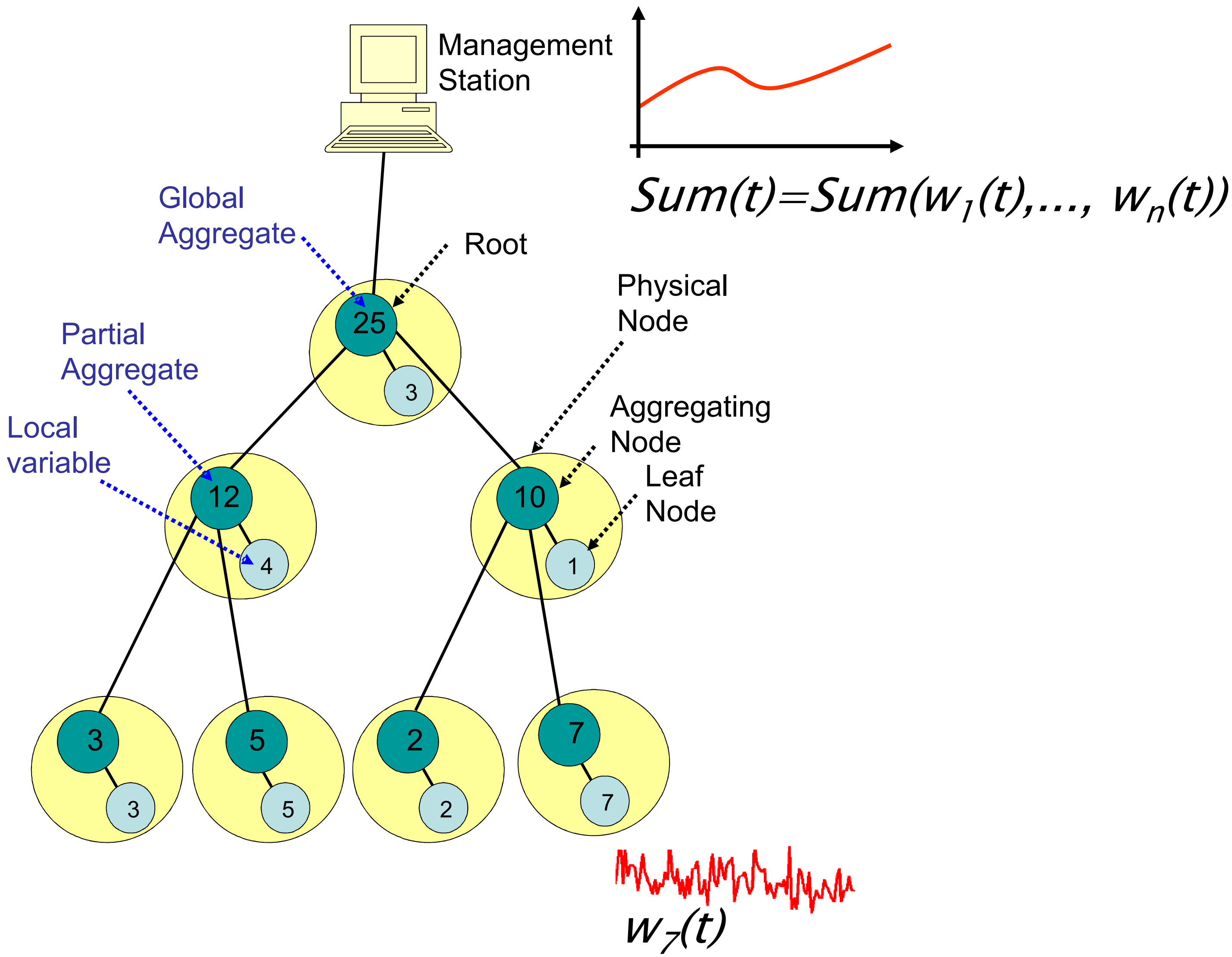
A-GAP: Protocol design goals

Provide a management application with a continuous estimate of an aggregate (sum) of local values for a given accuracy.

- Tunable trade-off: accuracy vs. overhead
  - –lowest overhead for a given accuracy objective
- Dynamic adaptation to changes
  - –changes to local values, topology, failures
- Scalability
  - –overhead increase with system size is sublinear

A. Gonzalez Prieto, R. Stadler: “A-GAP: An Adaptive Protocol for Continuous Network Monitoring with Accuracy Objectives,” IEEE Transactions on Network and Service Management (TNSM), Vol. 4, No. 1, June 2007
D. Jurca, R. Stadler, “H-GAP: Estimating Histograms of Local Variables with Accuracy Objectives for Distributed Real-Time Monitoring, “ IEEE Transactions on Network and Service Management (TNSM), Vol. 7, No. 2, June 2010.

In-Network Aggregation using Spanning Trees

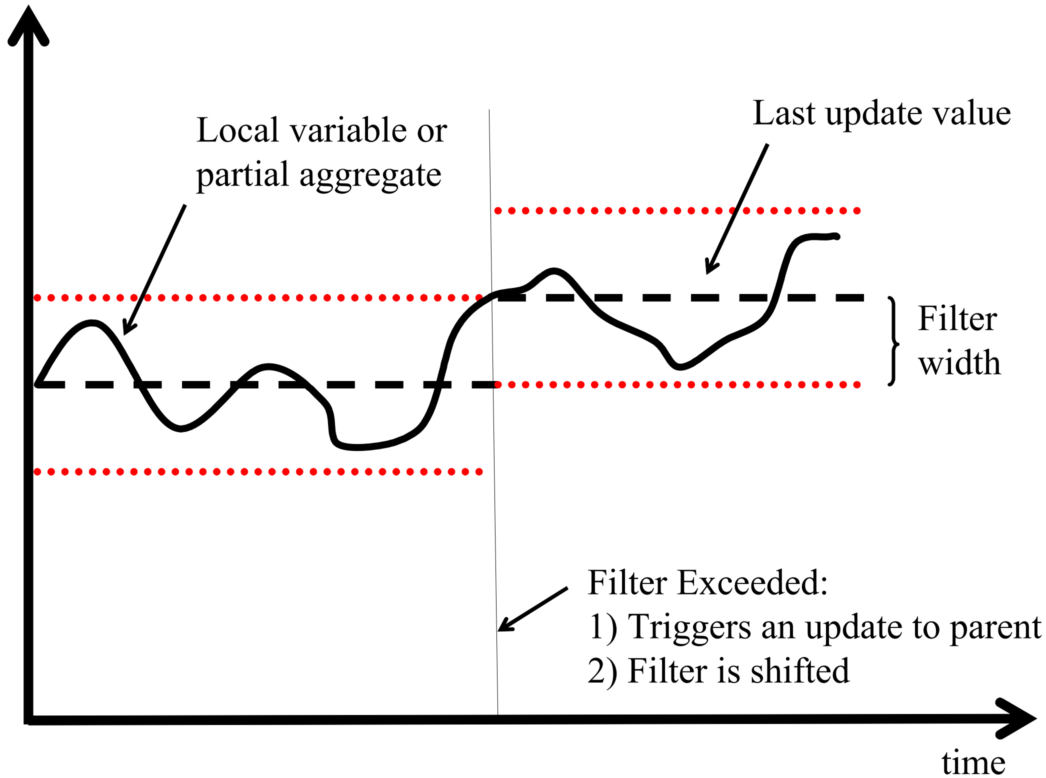


A-GAP: Protocol design principles

- Creating and maintaining spanning tree
  - Spanning tree on management overlay
  - BFS tree based on self-stabilizing protocol by Dolev, Israeli, Moran '90
- Incremental in-network aggregation on spanning tree
  - Aggregate computed bottom-up on nodes of tree
  - Result available at root node
- Filtering updates
  - Reduce protocol overhead by filtering updates while observing error objective
  - Compute filters using a distributed heuristic

S. Dolev, A. Israeli, and S. Moran, "Self-stabilization of dynamic systems assuming only read/write atomicity." ACM Symposium on Principles of Distributed Computing (PODC '90), Quebec City, Quebec, Canada, August,1990.

Local Adaptive Filters



Local filter on a node

- **Controls the management overhead** by filtering updates
- **Drops updates** with small change to partial aggregate
- **Periodically adapts** to the dynamics of network environment

Problem Formalization

Find ***filter widths*** to monitor aggregate
for a given accuracy objective, with minimal overhead

Overhead:
max processing load ω^n over all management processes

Accuracy objective:

average error $\quad \text{Minimize} \ \underset{n}{Max}\{\omega^n\} \ \ s.t. \ \ E[|E^{root}|] \leq \varepsilon$

percentile error $\quad \text{Minimize} \ \underset{n}{Max}\{\omega^n\} \ \ s.t. \ \ p(|E^{root}| > \gamma) \leq \theta$

maximum error $\quad \text{Minimize} \ \underset{n}{Max}\{\omega^n\} \ \ s.t. \ \ |E^{root}| \leq \kappa$

A Distributed Heuristic

- The global problem is mapped onto a ***local problem for each node***

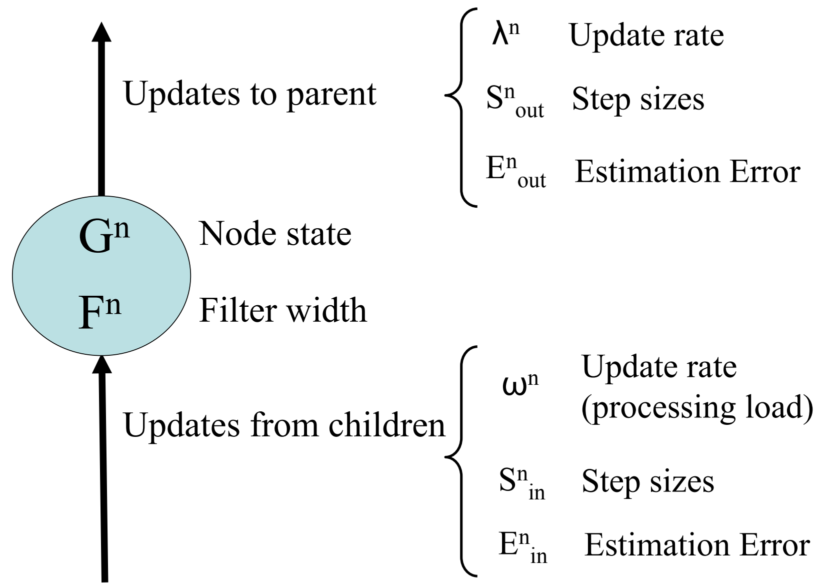
$$\text{Minimize } \underset{\pi}{Max}\left\{\omega^{\pi}\right\} \quad \text{s.t.} \quad E\left(\left|E_{out}^{n}\right|\right) \leq \varepsilon^{n}$$

- Attempts to minimize the maximum processing load over all nodes by minimizing the load within each node's neighborhood
- Filter computation: ***decentralized*** and ***asynchronous***
- Each node independently runs a control cycle:

```
every τ seconds {
    request model variables from children
    compute new filters and accuracy objectives for children
    compute model variables for local node
}
```

A Stochastic Model for the Monitoring Process

- Model based on discrete-time Markov chains
- It relates for each node *n*
  - the error of its partial aggregate
  - evolution of the partial aggregate
  - the rate of updates *n* sends
  - the width of the local filter
- It permits to compute for each node
  - the distribution of estimation error
  - the protocol overhead



Stochastic Model: leaf node

Estimating step size (MLE) X^n

Evolution of local variable

$$j^n = \begin{cases} i^n + X^n & -F^n \le i^n + X^n \le F^n \\ 0 & otherwise. \end{cases}$$

Transition Matrix

$$t_{ij}^n = \begin{cases} P(X^n = j^n - i^n) & \left|j^n\right| \le F^n, j^n \ne 0 \\ P(X^n = -i^n) + P(F^n - i^n < X^n < -F^n - i^n) & j^n = 0 \end{cases}$$

Step Size

$$P(S_{out}^n = s) = \begin{cases} \sum_{z=s-F^n}^{s+F^n} P(X^n = z)P(G^n = s - z) & \left|s\right| > F^n \\ \sum_{d=-F^n}^{F^n} \sum_{z=d-F^n}^{d+F^n} P(X^n = z)P(G^n = d - z) & s = 0 \\ 0 & otherwise. \end{cases}$$

Estimation Error $E_{out}^n = G^n$

Management Overhead $\lambda^n = (1 - P(S_{out}^n = 0))$

Stochastic Model: aggregating node

<u>Input</u> → <u>Output</u>

Step Size:

$$P\left(S_{in}^{n}=s\right)=\frac{\sum_{c\in\gamma^{n}}\left(P\left(S_{out}^{c}=s\right)\cdot\Delta^{c}\right)}{\sum_{c\in\gamma^{n}}\Delta^{c}}$$

→

$$P(S_{out}^{n}=s)=\begin{cases}\sum_{k=s-F^{n}}^{s+F^{n}}P(S_{in}^{n}=k)P(G^{n}=s-k) & |s|>F^{n}\\ \sum_{d=-F^{n}}^{F^{n}}\sum_{k=d-F^{n}}^{d+F^{n}}P(S_{in}^{n}=k)P(G^{n}=d-k) & s=0\\ 0 & otherwise\end{cases}$$

Estimation Error:

$$E_{in}^{n}=\sum_{c\in\gamma^{n}}E_{out}^{c}$$

→

$$E_{out}^{n}=E_{in}^{n}+G^{n}$$

Management Overhead:

$$\omega^{n}=\sum_{c\in\gamma^{n}}\lambda^{c}.$$

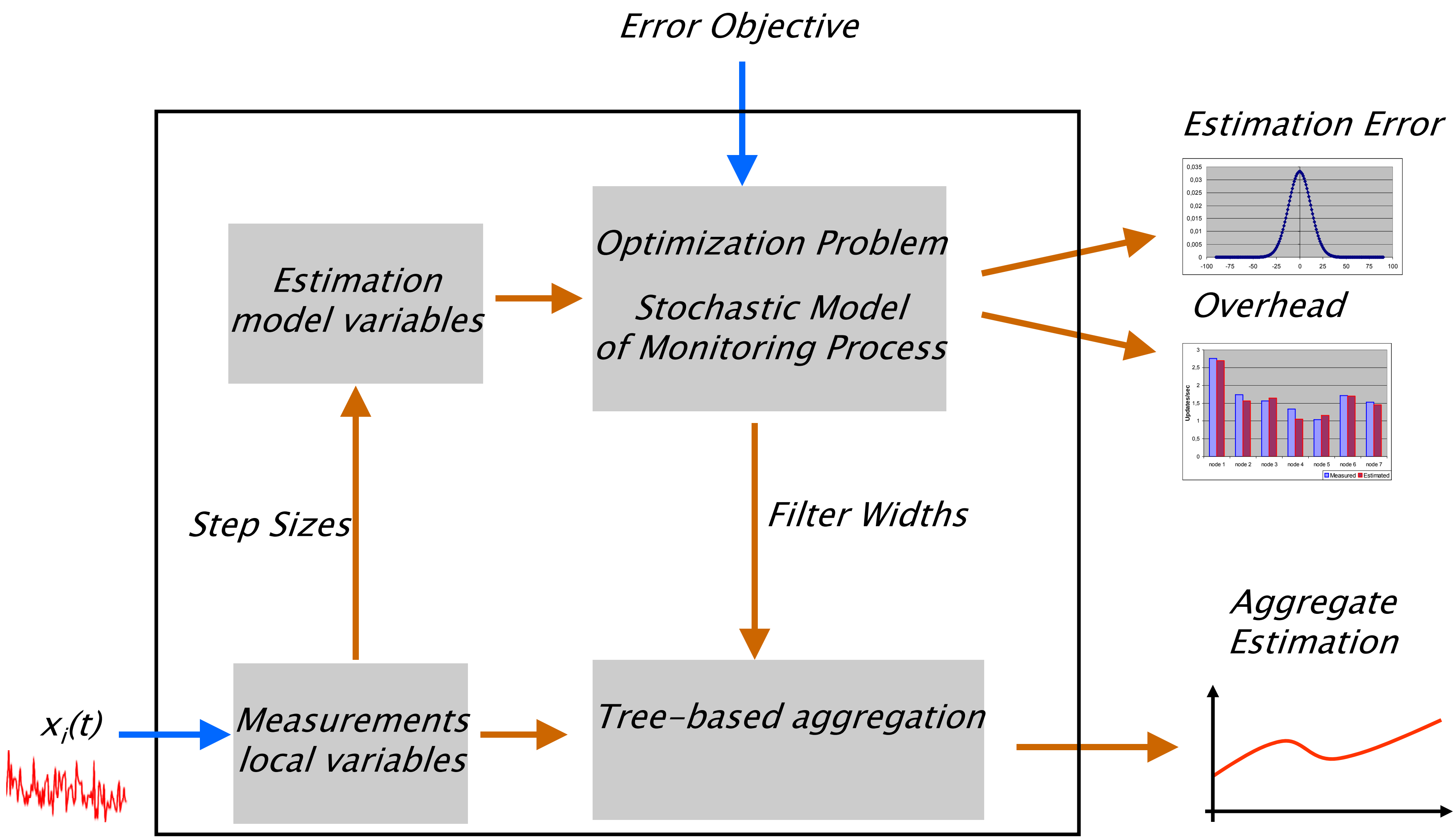
→

$$\lambda^{n}=\Delta^{n}(1-P(S_{out}^{n}=0))$$

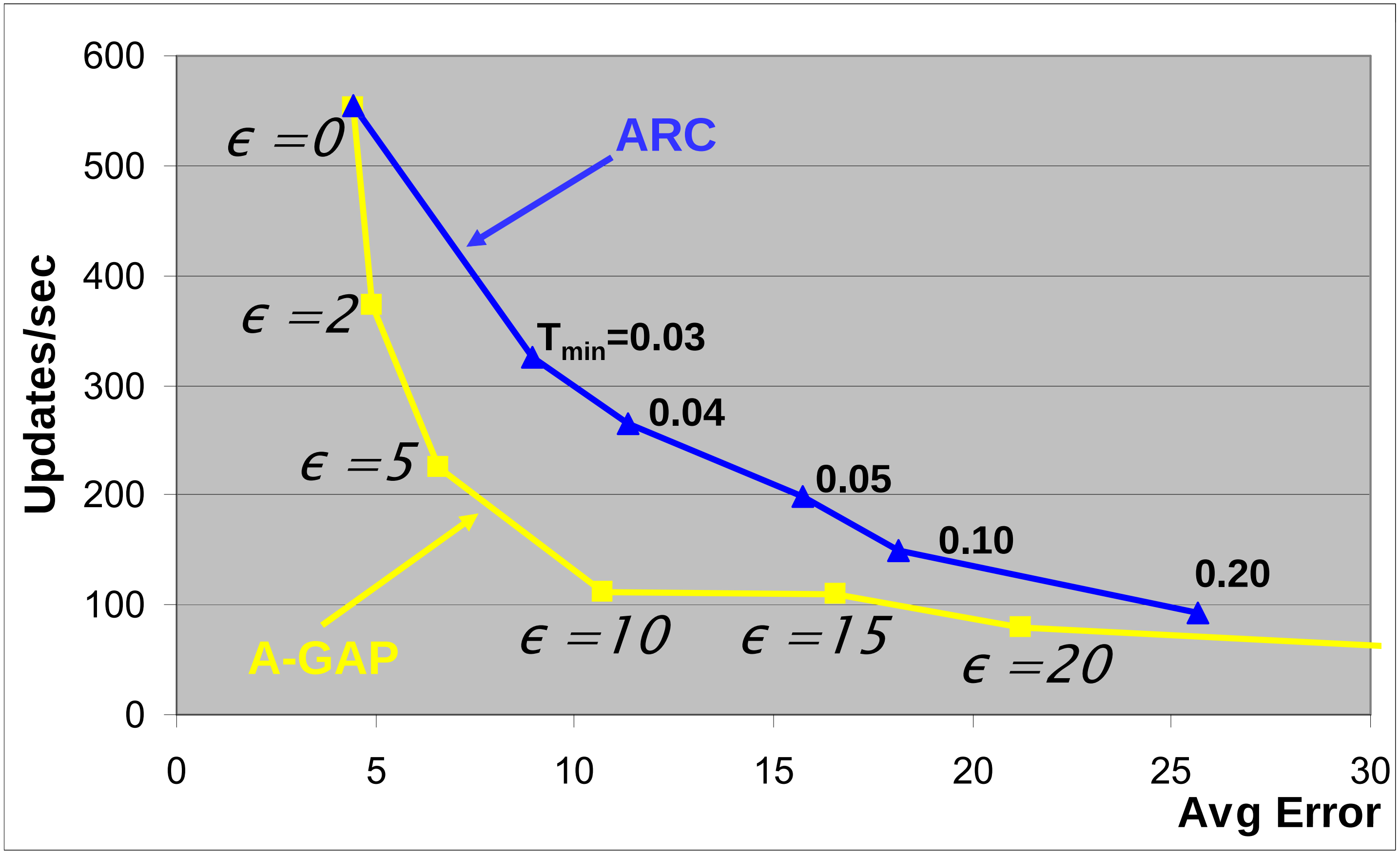
Transition Matrix:

$$t_{ij}^{n}=\begin{cases}P(S_{in}^{n}=j^{n}-i^{n}) & \left|j^{n}\right|\le F^{n},\ j^{n}\neq 0\\ P(S_{in}^{n}=-i^{n})+P(F^{n}-i^{n}<S_{in}^{n}<-F^{n}-i^{n}) & j^{n}=0\end{cases}$$

Model-based Monitoring

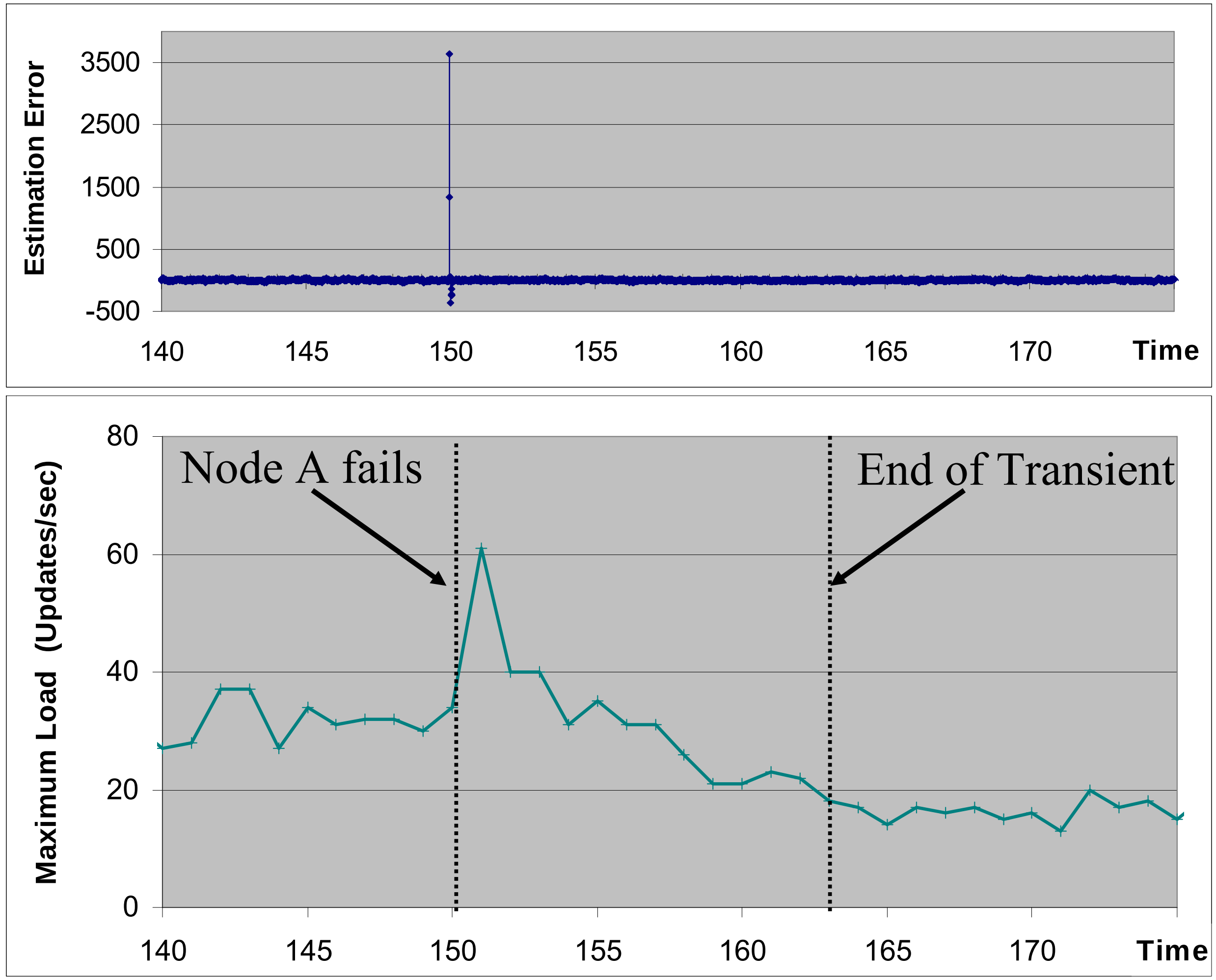


Tradeoff: Accuracy vs Overhead



- Overhead **decreases monotonically**
- Overhead depends on the **changes of the aggregate**, not on its value.
- A-GAP **outperforms** a rate-control scheme (ARC)

Robustness

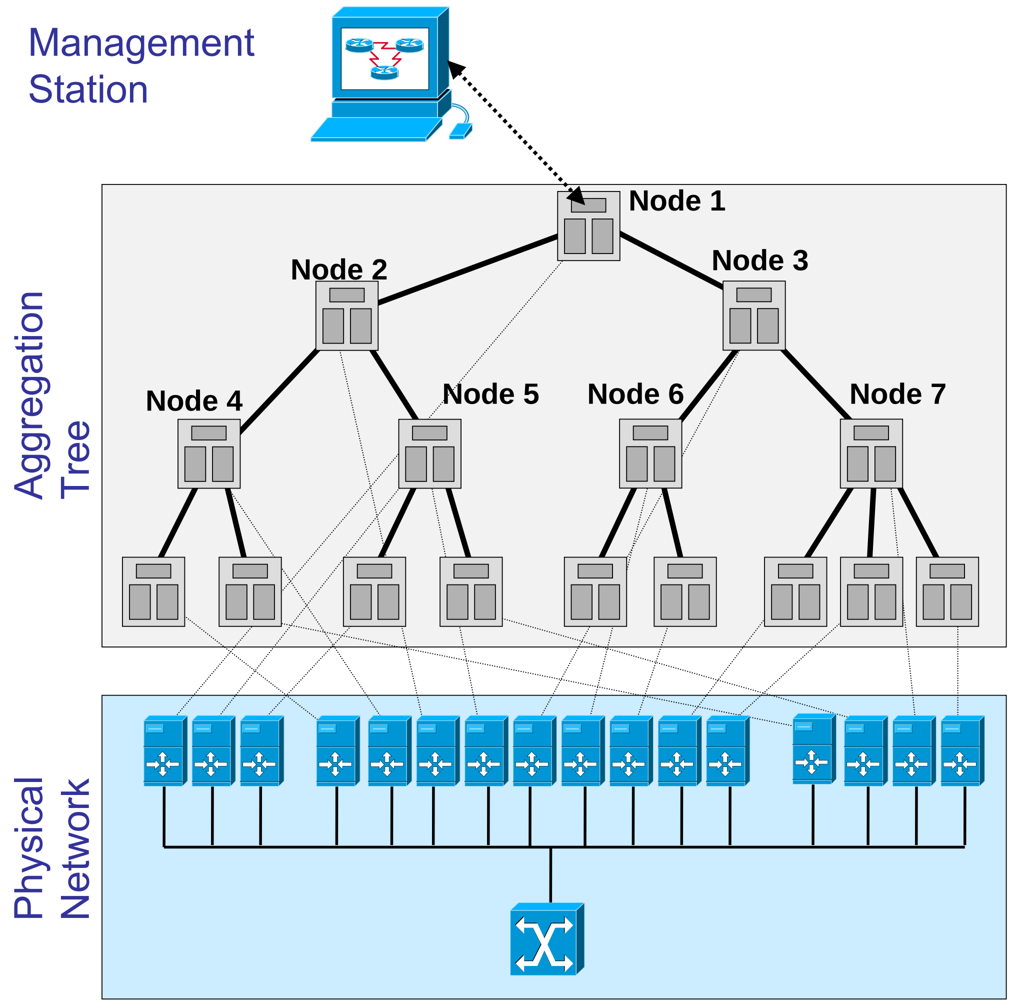


- Estimation error: several spikes during sub-second transient period
- Overhead: single peak with a long transient

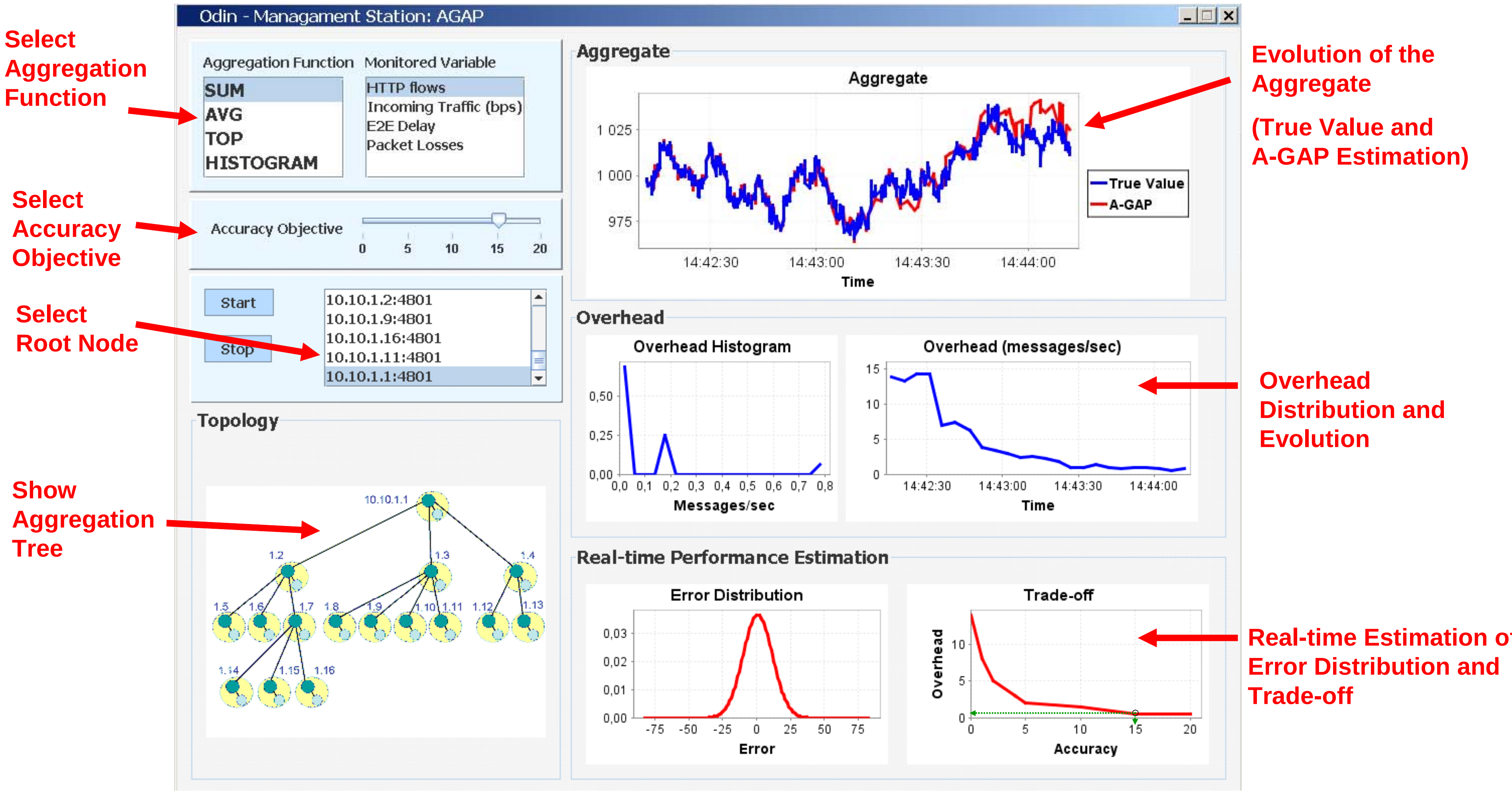
A-GAP Prototype

Lab testbed at KTH

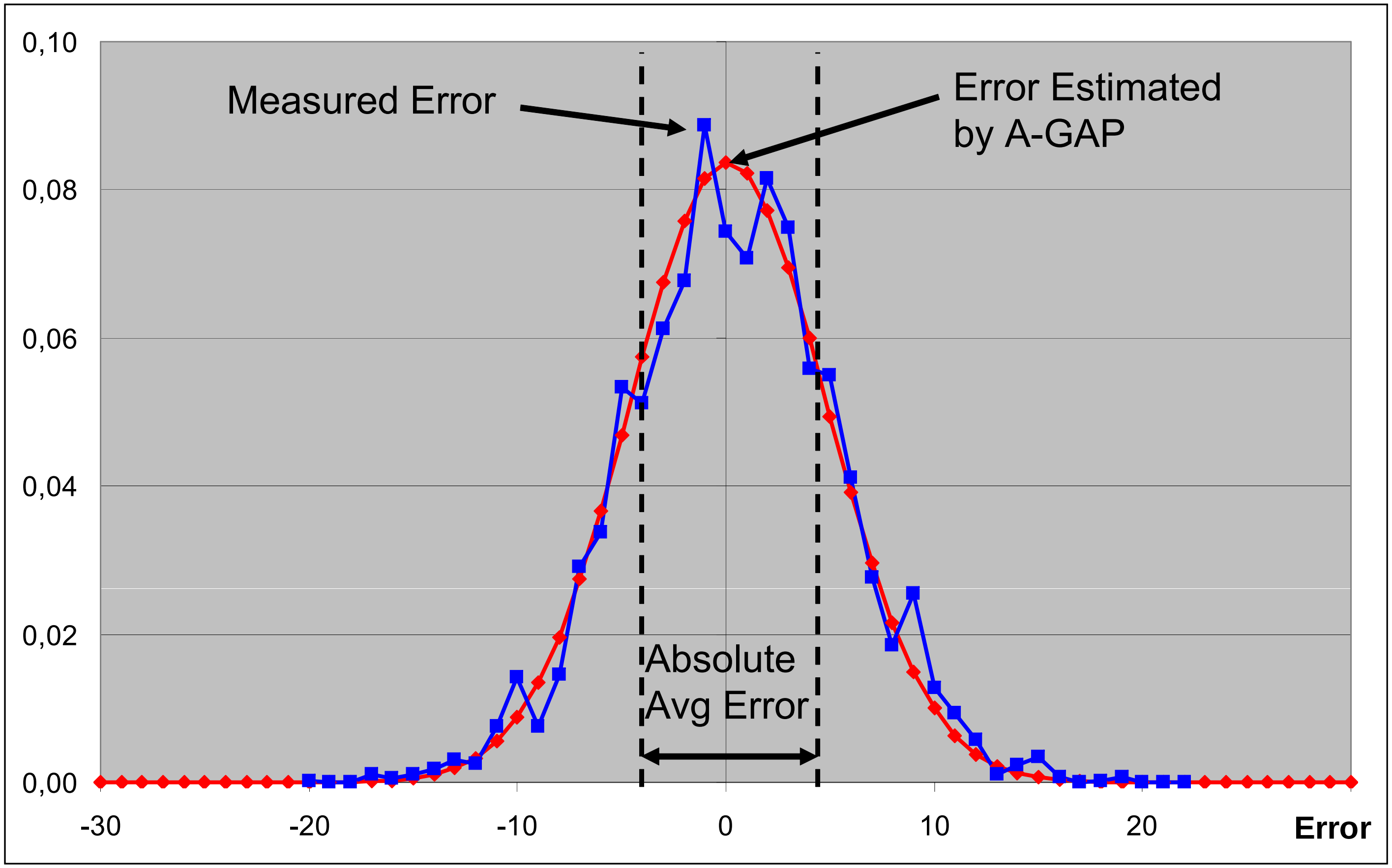
- 16 monitoring nodes
- 16 Cisco 2600 Series routers
- Smartbits 6000 traffic generator
- A-GAP implemented in Java



Prototype: Management Station Interface



Prototype: Error Estimation by A-GAP vs Actual Error



- **Accurate estimation** of the error distribution
- Maximum error >> average error (one order of magnitude)

Gossip vs. Tree-based Aggregation

Computing aggregates through gossiping

Push Synopses
[Kempe et al. '03]

- The protocol computes AVERAGE of the local variables x_i.
- After each round a new estimate of the aggregate is computed as s_i/w_i.
- ***Exponential convergence*** on connected graphs
- ***Protocol Invariants***:

$$\sum_i s_{r,i} = \sum_i x_{r,i} \,,\; \sum_i w_{r,i} = n_r$$

```
Round 0 {
  1. s_i = x_i;
  2. w_i = 1;
  3. send (s_i, w_i) to self }
Round r+1 {
  1. Let {(s*_l, w*_l)} be all pairs sent to i
     during round r
  2. s_i = Σ_l s*_l;  w_i = Σ_l w*_l
  3. choose shares α_{i,j} ≥ 0 for all nodes j
     such that Σ_j α_{i,j} = 1
  4. for all j send (α_{i,j} * s_i, α_{i,j} * w_i) to each j }
```

D. Kempe, A. Dobra, and J. Gehrke, "Gossip-based computation of aggregate information," in Proc. 44th Annual IEEE Symposium Foundations Computer Science (FOCS), Oct. 2003.

The G-GAP protocol

Round 0 {

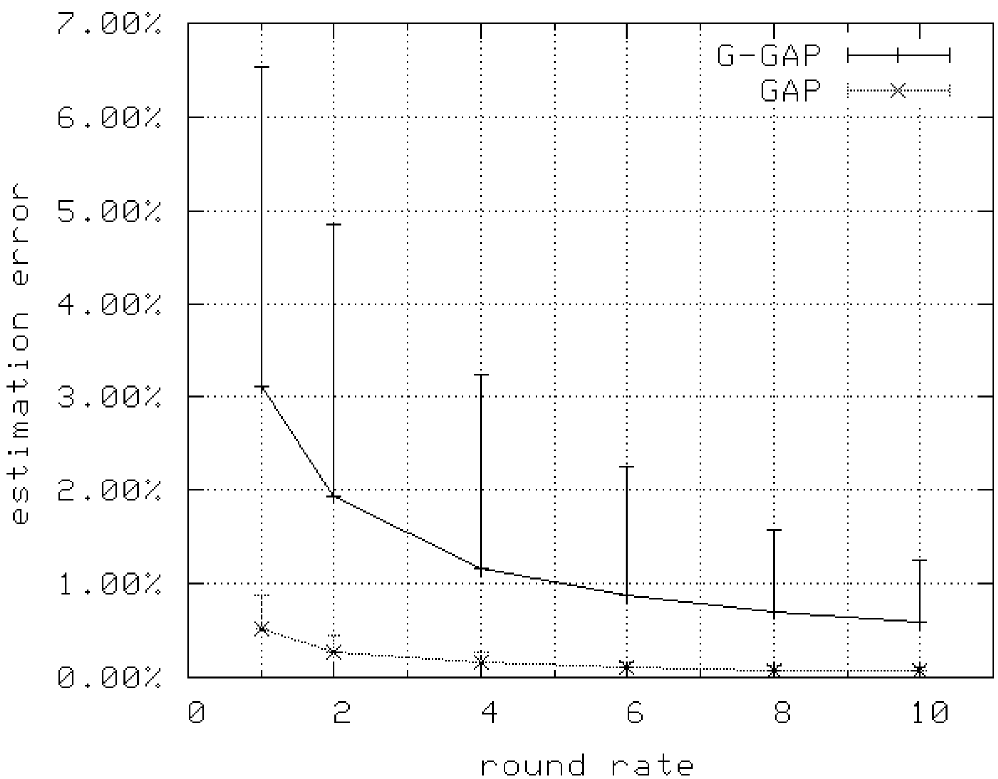
1. $s_i = x_i$;
2. $w_i = 1$;
3. $L_i = \{i\}$;
4. for each node j $(rs_{i,j}, rw_{i,j}) = (0,0)$;
5. for each node j $(srs_{i,j}, srw_{i,j}) = (0,0)$;
6. send $(s_i, w_i, 0,0,0,0)$ to self;
7. for all $j \neq i$ send $(0,0,0,0,0,0)$ to j }

Round r+1 {

1. Let M be all messages received by i during round r
2. $s_i = \sum_{m \in M} s(m) + (x_{r,i} - x_{r-1,i})$; $w_i = \sum_{m \in M} w(m)$
3. for all j $(acks_{i,j}, ackw_{i,j}) = (0,0)$
4. $L_i = L_i \bigcup orig(M)$
5. for all $j \in Neighbors$ {
   a. $(rs_{i,j}, rw_{i,j}) = (rs_{i,j}, rw_{i,j}) + \sum_{m:orig(m)=j} ((rs(m), rw(m) - acks(m), ackw(m)))$
   b. $(acks_{i,j}, ackw_{i,j}) = (srs_{i,j}, srw_{i,j}) + \sum_{m:orig(m)=j} (s(m), w(m))$
   c. if (detected_failure(j)) {
      i. $(s_i, w_i) = (s_i, w_i) + (rs_{i,j}, rw_{i,j})$
      ii. $(rs_{i,j}, rw_{i,j}) = (srs_{i,j}, srw_{i,j}) = (0,0)$
      iii. $L_i = L_i \setminus j$
   }
   }
6. for all $j \in L_i$ {
   a. choose $\alpha_{i,j} \geq 0$ such that $\sum_j \alpha_{i,j} = 1$
   b. choose $\beta_{i,j} \geq 0$ such that $\sum_j \beta_{i,j} = 1$ and $\beta_{i,i} = 0$
   c. $(srs_{i,j}, srw_{i,j}) = \beta_{i,j}(\alpha_{i,i} s_i - x_i), \beta_{i,j}(\alpha_{i,i} w_i - 1)$
   d. send $(\alpha_{i,j} s_i, \alpha_{i,j} w_i, srs_{i,j}, srw_{i,j}, acks_{i,j}, ackw_{i,j})$ to j
   e. $(rs_{i,j}, rw_{i,j}) = (rs_{i,j} + \alpha_{i,j} s_i, rw_{i,j} + \alpha_{i,j} w_i)$
   }

}

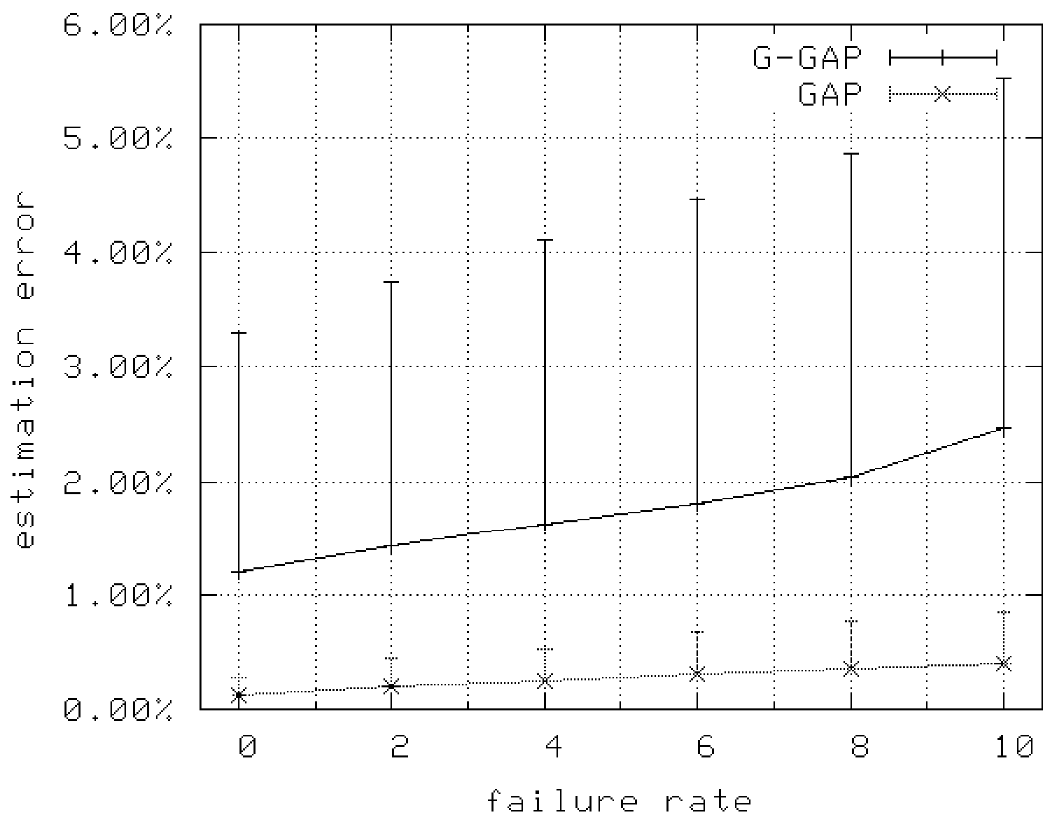
Accuracy vs. Overhead
gossip- and tree-based aggregation protocol



GAP and G-GAP
654 node network
GoCast overlay, connectivity 10
aggregation: AVERAGE
UT trace
4 rounds/sec
no failures

F. Wuhib, M. Dam, R. Stadler, A. Clemm "Robust Monitoring of Network-wide Aggregates through Gossiping," IEEE Transactions on Network and Service Management (TNSM), Vol. 6, No. 2, June 2009.

Accuracy vs. Failure Rate
gossip- and tree-based aggregation protocol



GAP and G-GAP

654 node network
GoCast overlay,
connectivity 10

aggregation: AVERAGE

UT trace

4 rounds/sec

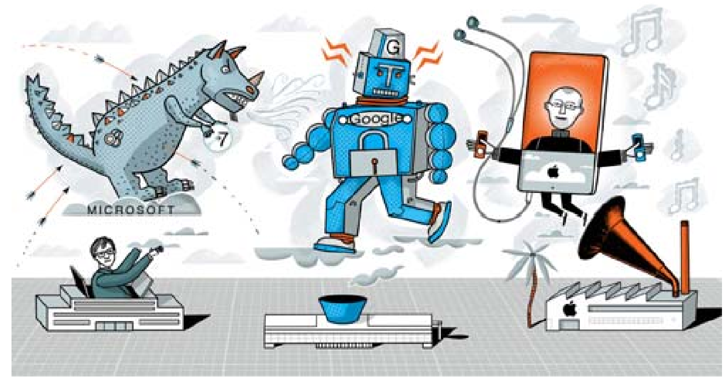
nodes fail randomly,
recover after 10 sec

Tree-based aggregation outperforms gossip-based aggregation!

- Network Managements
- In-Network Management
- Case Study: Real-time Monitoring
- Will it happen?

In-Network Management—Why it will happen

Compared to 5-10 years ago:



- New actors
  — Google, Amazon, Microsoft, Apple
- New drivers
  — data center networking, cloud computing,
- Advances in distributed computing
  — gossip protocols, algorithms for virtual topologies, understanding protocols on dynamic topologies
- Enablers of network programmability
  — manufacturers Juniper, Cisco provide open interfaces
  — OpenFlow allows for programmable control and management planes