Cooperative Congestion Control in Power Grid Communication Networks

Naveen Cherukuri
Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana, Illinois 61801
Email: cheruku3@illinois.edu

Klara Nahrstedt
Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana, Illinois 61801
Email: klara@illinois.edu

***Abstract*—Power grid digital communication networks are needed for the delivery of Phasor Measurement Unit(PMU) [1] sensor data with real time guarantees to the control centers for timely decisions. Present day network systems are unable to ensure real-time needs of the PMU data as they are not designed to support PMU devices. Hence, the need arises for new communication and control protocols. This paper describes the Cooperative Congestion Control (CCC) framework to ensure real-time guarantees for PMU data and to respond if transient deviations in real-time PMU network traffic occur. The framework utilizes a) NASPI [2], [3] aligned multiple service class queuing architecture; b) Cooperative real-time flow scheduling and bandwidth reassignment; and c) Cooperative coordination and back-pressure approaches among neighboring nodes. It then yields real-time PMU data guarantees during transient traffic pattern changes and/or overload situations. Our experiments confirm that the framework delivers real-time performance very well under different transient stress scenarios.**

I. Introduction

Infrastructure critical networks such as power grid communication networks have very strict requirements on real-time availability, reliability and security. These networks need to assure end-to-end latency and throughput guarantees to the control and data packets of many critical flows.

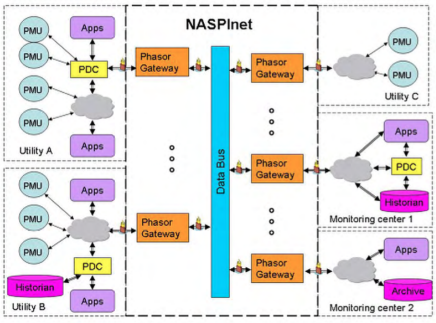

Fig. 1: NASPInet Architecture

NASPI [2], [3] standard working group is designing a wide area network infrastructure, dubbed NASPInet [2], to enable wide area sharing of PMU data. Figure 1 [2] shows a high-level conceptual architecture envisioned for NASPInet. NASPInet is composed of Phasor Gateways (PGWs) and a Data Bus (DB). The DB includes a Wide Area Network (WAN) and associated services to provide basic connectivity, QoS management, performance monitoring, cyber security, and policy enforcement over data exchanged through NASPInet. Utilities A, B and C are the power grid sensing stations (e.g., substations). Each utility contains PMU sensors, Intelligent Electronic Devices (IEDs) and other applications (Apps) aggregated at the Phasor Data Concentrators (PDCs) connected to the WAN via the PGW. There are monitoring centers which require the sensors' data for timely decisions. NASPInet has challenges which the Internet does not possess. Critical data loss (e.g., PMU control data) must be minimal and PMU traffic packets have strict end-to-end deadlines. Since power grid status monitoring is also moving towards Internet protocols and services, the existing Internet(IP) technologies must be modified with new protocols to fit into the real-time PMU requirements. One of the major problems for the power grid critical networks (NASPInet) is the congestion of the PMU data streams (flows) in case of increased traffic demands and/or other transient traffic stress situations. During the period of congestion, the real time flows may not meet their delay requirements leading the NASPInet into an unstable state. Hence, early notification of congestion and cooperative congestion control are crucial techniques to respond to unpredictable grid/traffic events and protect real-time PMU flows from violations of their deadlines. In this paper we introduce the Cooperative Congestion Control (CCC) framework for handling unpredictable traffic events and changes which cause transient congestion situations in the NASPInet.

"Cooperation" is defined in two ways. One being nodes utilizing the early congestion notification, and the other being nodes using the policy-based correlation between different flows i.e., cooperative flows. Cooperative Congestion Control assumes that all PMU sensing nodes and their flows from each power grid substation cooperate with each other during a transient network change and the CCC system ensures real-time guarantees of critical PMU flows.

The concept of Cooperative Congestion Control is explored in the wireless scenario [5] where nodes cooperate with each other to acquire access to the medium, but in the wired scenario, considering the large size of the networks and multiple domains, cooperation is explored only in a limited fashion. TCP [8] does not react fast enough to congestion and does

not protect real-time flows in terms of their delay requirements when congestion occurs. Hence, TCP is unacceptable in infrastructure critical networks, and its retransmissions and acknowledgements will even worsen the delay situation. Moreover, it doesn't consider the correlation of real-time power grid PMU measurement flows. Hence, there is a need for protocols that provide congestion control and protect real-time guarantees of critical PMU flows.

Our CCC framework does (1) react fast to unpredictable NASPInet traffic events, which cause transient congestion, (2) protect delay guarantees of real-time PMU flows, and (3) utilize the cooperative and correlated nature of PMU flows coming from the same substation.

Outline: In section II we produce related work. In section III we define the model, problem and set of assumptions considered. In section IV we describe our CCC approach and design issues and in section V validation is discussed. We conclude in section VI.

II. Related Work

Significant amount of research has been done on cooperative congestion control in Vehicular Adhoc NETworks (VANET) [5]. In the wired scenario, TCP end-to-end congestion control mechanisms [8] are sufficient for the internet flows that are mostly best effort. Congestion control/avoidance techniques in the internet scenario are based on Explicit Congestion Notification, Random Early Detection etc. [9], [11].

GridStat [4], [12], shown in Figure 3 [4], is a QoS-managed publish-subscribe framework for data delivery in the electric power grid. GridStats data plane delivers data updates through a network of middleware-level status(overlay) routers. Subscriptions are managed by GridStats hierarchical QoS management plane. The path allocation computations are typically done offline and beforehand.

Asynchronous Transfer Mode (ATM) network is a high-speed network with its link layer protocol that ensures QoS of the data packets. In case of traffic congestion, ATM networks implement back pressure algorithms [13] and send tokens towards senders to slow down the traffic generation.

III. Models and Problem Description

Power Grid Sensing Application Model: The applications

	Class A	Class B	Class C	Class D
Low Latency	Critical	Fairly Critical	Somewhat Critical	Not Critical
Reliability / Availability	Critical	Somewhat Critical	Not Critical	Fairly Critical
Data Accuracy	Critical	Somewhat Critical	Not Critical	Critical
Time Alignment	Critical	Critical	Somewhat Critical	Not Critical
Message Rate	Critical	Somewhat Critical	Somewhat Critical	Critical
Sample Application	Small Signal Stability	State Estimation	Visualization and Monitoring	Disturbance Analysis

Fig. 2: Application Classes

are broadly divided into four service classes in terms of their servicing data criticality [3], [2] as shown in Figure 2 [3]. Applications that capture PMU data belong to Class A. On the other hand, applications that capture surveillance video at substations belong to Class C. For simplicity we assume Class A serves High Real Time (HRT), Class B serves Low Real Time (LRT) and Class C serves Best Effort (BE) Apps.

Device and Data Model: PMU devices produce streams of samples with information about voltage and current signals [1]. The sampling rate is 10 to 250Hz. Previous generation of power grid sensors, the IED devices, sampled at a lower rate (e.g., 2Hz).

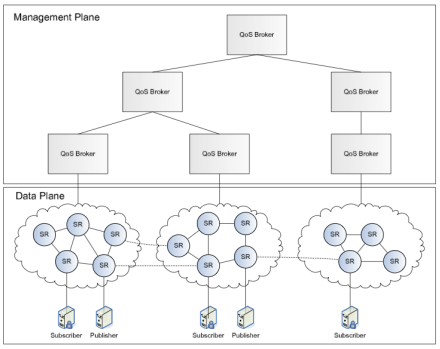

Fig. 3: Grid-Stat Status Dissemination Architecture

Network Model: One of the proposed WAN architectures for NASPInet is GridStat(Figure 3) and we assume a similar management plane model and distributed paradigm (Publish/Subscribe) for our CCC framework. It means, we assume QoS brokers which are responsible for calculating wide area paths and reservations of resources between overlay/status routers (SR). The physical path between the overlay/status routers are then leased with resources, specified by QoS brokers, to ensure bandwidth, latency and loss rate guarantees at the start of the session.

Queue Management Model: Each overlay router/PGW maintains data queues to serve PMU data streams/flows as well as data from other power grid applications. In our CCC framework we assume applications servicing data belonging to three service classes A, B, C corresponding to HRT, LRT, and BE. Hence, we will assume queuing model consisting of three queues at each PGW, servicing traffic classes A, B, C.

Traffic Model: Each flow f in the NASPInet will be characterized by parameters: *minimal sampling rate*, *maximum sampling rate*, *reserved rate*, *peak sample size*, and *average sample size*. Each flow originates at the substation, and is generated by sensing devices such as PMUs (each sensing device is called a publisher). Each flow is delivered to a control center that subscribed to receive corresponding PMU sensory data. Correlation of flows is policy dependent. We consider spatial correlation policy where flows generated from same physical space are correlated (e.g., flows from the same substation).

A. Problem Description

Real-time guarantees of PMU flows can be violated, when transient congestion occurs in the NASPInet. Transient Congestion occurs during short term unexpected traffic changes

when critical flows do not get sufficient BW, resulting in violation of end-to-end delay guarantees and packet drops .

Deviations/congestion of traffic in the power grid communication network (NASPInet) can be caused by:

1) Variable compression of the PMU data or other sensory data (e.g., surveillance video), causing variable bit rate traffic.
2) Increased sending rate of real-time (RT) PMU flows due to unexpected critical event/observation in their sensory space and causing changed traffic shape of RT flows,
3) Changing demands on RT traffic requested by control centers (subscribers) due to extended power grid state analysis, causing changes in traffic shapes of RT flows.

For example, in Figure 1, if one of the PMU sensors in utility A increases its sampling rate upon observing a critical event (e.g., voltage signal deviation), it introduces more traffic into the NASPInet than reserved for it. It is important to stress that current WAN management/data plane related techniques like GridStat do not support fast reaction to temporary, short term, and transient changes in traffic. The current QoS broker-based systems [4] yield QoS guarantees during the stable state, but when congestion happens, global adaptation takes place, i.e., the QoS brokers are contacted and overlay routers wait for QoS broker's response, i.e., until new routing tables/reservations are setup. The time interval between a congestion notification, request for action, and broker response is non-negligible and within this interval transient congestion occurs and real-time PMU flows with their delay requirements are not protected, i.e., violation of deadlines and packet drops occur.

Hence, we need an approach that will protect real-time PMU flows in NASPInet when transient congestion occurs, and the current reservations cannot handle the increased and changing traffic load.

IV. Approach

We propose a Cooperative Congestion Control Framework for the NASPInet data bus to solve the above described problem. The solution is achieved by introducing a) Multiple service class queueing b) Cooperative real-time flow scheduling and BW reassignment and c) Cooperative coordination and back pressure approaches among neighboring nodes to counter the transient congestion state. The *multiple service classes* queueing will allow us to differentiate treatment among the service classes A, B, C and use different scheduling policies to respond to congestion situation and protect traffic in real-time classes A and B. *Cooperative real-time scheduling* and *bandwidth reallocation* utilizes the fact that certain flows are cooperative and correlated, containing information about a shared physical space. Hence, the group of the correlated and cooperative real-time flows can help each other, i.e., bandwidth to the LRT and BE flows can be reassigned and lowered to protect the HRT flows at the overlay router/PGW. *Cooperative coordination* allows to expand the protection of RT flows during the congestion to a broader area, informing neighboring overlay router/PGW and/or sources about the experienced congestion and ask them to (a) slow down their traffic towards the congested router, and (b) apply cooperative RT scheduling and BW reallocation at their nodes as well.

A. Overlay Router/Phasor Gateway Design

Cooperative real-time flow scheduling, BW reassignment and multiple-service class queueing need differentiating the flows at the overlay router and scheduling real-time flows based on their packet deadlines and priority. The overlay router achieves flow differentiation with three major components as shown in Figure 4. The *Execution unit* performs cooperative real time scheduling on multi-service-class queues. The *Adapter unit* executes bandwidth allocation and the congestion notification protocol to achieve cooperative coordination among the neighbors. The *Controller unit* acts as the central component that does all setup activities.

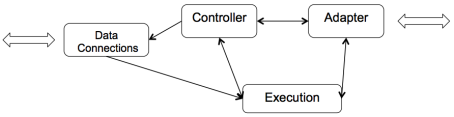

Fig. 4: Design Components

1) Controller Unit: This central component is responsible for creating adapter, execution and connections components. It also maintains the metadata, reservation information about the data flows obtained from its QoS broker during the flow set up time. Metadata includes the information about *correlated data flows*. For example, in Figure 6, flows $f1$, $f2$, $f3$ are the HRT, LRT, BE flows generated from the same substation (i.e., they are correlated). Flows $k1$, $k3$ are HRT, BE flows generated from another substation.

2) Data Connection Unit: This unit handles the connections associated with each publisher-subscriber flow (e.g., *Utility A PMU - Monitoring center 1* as shown in Figure 1) at the overlay router.

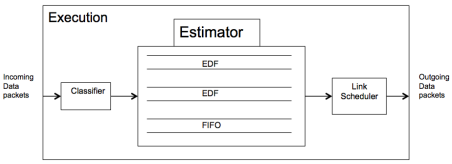

Fig. 5: Execution Component

3) Execution Unit: Execution unit (Figure 5) implements the real-time flow scheduling algorithms and the multiple service class-based queuing. Flows are prioritized depending on their service class (Real time over BE). Let us describe in detail the subcomponents of the execution unit.

Packet Classifier: We have our own classifier which inspects the incoming packet's $classid/flowid/source\ address$ and places packets into the appropriate service class queue to ensure its QoS requirements (latency, loss rate and throughput).

Queuing Strategy: As described earlier, there are three classes of flows that can exist in the system. Class A and B have queues buffering real time flows. In these classes we need to consider the deadline of packets while dequeuing those

queues. We use an $Earliest\ Deadline\ First\ (EDF)$ [6] strategy in the enqueue/deque functions for the flows of the Class A and B queues. Class C has no real-time requirements and hence it is a simple $FIFO$ queue.

Link Scheduler: This scheduler is called when the network interface hardware is ready to send the next packet into the network. The outgoing link bandwidth is shared between all classes [10]. Link Scheduler runs a $Weighted\ Round\ Robin$ scheduling policy with weights being the reserved link BW share for each class. For example in Figure 6, Class A queue may get 60% of link BW, Class B 30% and Class C 10%.

Estimator: The estimator determines packet arrival rate and bandwidth usage for each class over a period. Bandwidth usage is estimated by calculating the sent amount of data over a period. Packet arrival rate is calculated using Exponential Weighted Moving Average (EWMA) over the queue size.

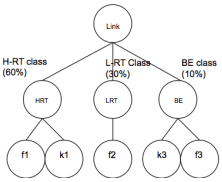

Fig. 6: Class Hierarchy at the Overlay Router

4) Adapter: Adapter is the primary component where the *BW reassignment* algorithm and *Cooperative coordination* protocol reside. Its main responsibilities include: a) Maintain the link sharing queue hierarchy and update statistics periodically, b) Adapt to any transient congestion occurrence by bandwidth reassignment, and c) Communicate with the adapters of neighboring nodes notifying them about the new rate (BW) assignments.

Class Hierarchy Structure: Adapter maintains a class hierarchy structure (Figure 6) corresponding to each outgoing link. This structure does the book-keeping of bandwidth shares, usage, drops, backlog, demand, priority per each flow and per each class (e.g., for *f1, f2, f3, k1, k3* flows and their HRT, LRT, BE classes membership).

1) Each class/flow is characterized by a) $assured\ rate\ AR$ (the reserved rate), b) $minimum\ rate\ MR$(the minimum threshold rate of the class/flow), c) $ceil\ rate\ CR$ (above which the rate cannot increase), d) $priority\ P$ (HRT>LRT>BE and the flows of the same class have the same priority), d) $actual\ rate\ R$ (demand).
2) Each class/flow will have a *state* associated with it, based on the class and flow characteristics defined by the AR, MR, CR and R parameters, as specified above. Each class/flow can be in one of the three states: (a) *congestion state* if $AR<R<=CR$, (b) *stable state* if $MR<R<=AR$, and (c) *minimal threshold state* if $R=MR$.

For the adapter of the congested overlay router to control the transient congestion, it needs to estimate the congestion locally, perform BW reassignment and coordinate with adapters (section IV-B) of other nodes.

Local Congestion Estimation and BW Reassignment: Overlay Router/PGW estimates congestion locally from the state of each flow sharing its outgoing BW. When all the flows at an overlay router/PGW are in *stable state* or *min threshold state*, the router/PGW node is in *stable state* without any congestion. When any of the flows is in *congestion state*, the router/PGW node's adapter does congestion control through BW reassignment.

Our *BW reassignment* algorithm, shown in **Algorithm 1**, considers all flows which are in congested state. The BW reassignment algorithm selects a list of victim LRT & BE flows for each HRT flow i (i.e., LRT & BE flows whose BW allocation is reduced in order to increase BW and protect real-time performance of the HRT flow i). The list of victim flows for HRT flow i is selected based on the *priority P* of flows and the *correlation* among the flows and kept in the data structure *M(i)*. The correlated flows information is obtained from the controller unit. For example, from Figure 6, *M(f1)* contains $f3$, $k3$, $f2$ i.e., when $f1$ gets congested, the flows in *M(f1)* transfer their BW share to $f1$. BE flows $k3$ and $f3$ are preferred as victim BE flows for HRT $f1$ over LRT $f2$. $f3$ is preferred over $k3$ as $f1$ and $f3$ are correlated. For each HRT congested flow i (in the order of class priority), the BW shares of LRT & BE victim flows in $M(\mathrm{i})$ (ensuring Min Threshold rate), are transferred to satisfy HRT flow i's need. Congestion Notification Protocol (section IV-B) informs

Algorithm 1 Bandwidth Reassignment Algorithm

```
for each HRT flow i in congested state (priority order) do
  for each victim flow b in M(i) do
    needed ⇐ i.R − i.AR
    oldb = b.AR
    b.AR ⇐ max(b.MR, b.AR − needed)
    needed− = (oldb − b.AR)
    i.AR+ = (oldb − b.AR)
    if needed ≤ 0 then
      break
    end if
  end for
end for
```

neighboring adapters about the changed BW shares of all flows.

B. Congestion Notification Protocol

We consider an ATM-like back pressure [13] rate control algorithm in our CCC framework. When the publisher (PMU) increases its sending rate pertaining to a critical event, it introduces more traffic into the network than reserved for its flow. Depending on the availability of excess BW in the downstream PGWs/overlay routers, some nodes can experience congestion because of the increased PMU flow rate. The congested nodes control the congestion by BW reassignment. With BW reassignment the high priority flow gets required BW share, while the low priority victim flows get congested. By this we protect the real-time guarantees of

the PMU flows. Explicit congestion notification to the low priority victim flows' senders takes the system back into stable state. This notification is done through the cooperative protocol between each nodes' adapter. Whenever, at a node congestion is estimated and BW is reassigned, the adapter of the node sends a control packet to its upstream node with the flows' new weights. The adapter of the upstream node triggers its BW reallocation according to the new weights in the control packet and notifies its upstream adapter about the flows. This proceeds on till the sender (BE App) which responds to the control packet by decreasing/increasing its sending rate. We use TCP for the reliable delivery of control packets and UDP for the data packets (as we do not need retransmissions, acknowledgements etc.).

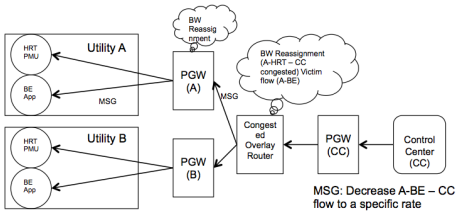

Fig. 7: Congestion Notification Protocol

Let us explain the protocol with the help of an example in Figure 7. When a critical event occurs in Utility A, the A-HRT PMU increases its sending rate. This leads to congestion at the overlay router that is shared between Utility A and Utility B flows. The overlay router through its BW reassignment algorithm when identifies congestion, selects victim flows and recalculates the link BW sharing weights. Based on the spatial correlation policy A-BE is selected as the victim flow. Adapter of the overlay router notifies the new weights to its cooperative upstream adapter (PGW-A). PGW-A responds to the notification by reassigning its outgoing BW and communicating with the cooperative upstream sender A-BE App with the control packet. The sender A-BE App reacts by decreasing its sending rate. Similarly when a subscriber (CC) wants to receive A-HRT PMU - CC flow at increased rate it notifies to its PGW that follows the above steps, with reassignment done along the path and the senders (PMU and App) notified about their new sending rates.

V. Implementation and Evaluation

We have implemented the overlay router components and the notification protocol on linux machines. $iproute2$ package's tc modules are used to interact with the kernel. These modules help in setting up the kernel network queueing parameters and classifier filters. We use a hierarchical token bucket [7] approach to limit each flow's bandwidth usage to its reserved rate. Kernel support is needed for the packet real time scheduling and BW reassignment. An image of the kernel multiple service queues is created which is used by the user module to periodically query the statistics, used for the BW reassignment.

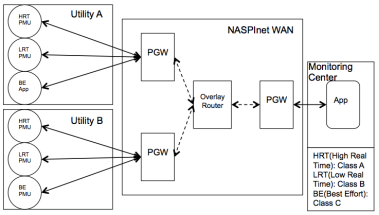

Fig. 8: Test Scenario

Our testbed constitutes of 2 substations with 3 sensors each (1 HRT PMU, 1 LRT PMU, 1 BE App), three PGWs, one overlay router and 1 control center as shown in Figure 8.

Scenario: We evaluate our CCC framework in the above test scenario, where utility A and utility B with 3 sensors each are connected to the Monitoring center via the NASPInet WAN. Congestion can occur at PGWs/overlay routers.

Parameters: For simplicity all the sensors generate sample packet of average size 260 bytes (phasor fields and digital channel fields) [1]. Service class queues are maintained at PGWs/overlay routers to hold 20 data packets of each flow. The sensors can operate at a frequency ranging from 1Hz to 60Hz. The minimum threshold bandwidth for the classes are HRT - 5200 Bytes, LRT - 2600 Bytes , BE - 600 Bytes. End-to-end deadline assumed for Class A HRT flows is 50ms and Class B LRT flows is 100 ms. The broker response/global adaptation time is taken to be 60 sec (the goal is to protect HRT flows during this period when the system is not stable).

Test: At *t=0* all flows start with an initial frequency of 20Hz. At *t=40* sec the Utility A's HRT PMU sensor increases its frequency to 40Hz. Later at *t=60* sec Utility B's HRT PMU sensor increases its frequency to 40 Hz. At *t=80* sec A-HRT PMU and B-HRT PMU increase their frequencies to 50 Hz. Finally, at *t = 90* sec Utility A-HRT PMU and B-HRT PMU increase their frequencies to 60 Hz.

Figure 9 (a), (b), (c) show the latency, throughput and drop rate when the CCC framework is not used, i.e., the case just with global adaptation waiting for decision from the broker. Figure 9 (d), (e), (f) show the same with CCC in place. TABLE I shows the percentage of undropped real-time packets that missed their deadlines to the packets generated without and with the CCC framework. Dropped, deadline missed and legally sent packets sum upto number of packets generated.

From *t=0* to *t=40* sec all the flows are in stable state. At *t=40* sec the A-HRT flow doubles its frequency. When our framework is not used, all the extra packets above reserved rate are dropped and the also the deadlines of the sent packets are missed as shown in TABLE I. When the CCC framework is used, A-BE and B-BE are selected as the victim flows and BW reassignment/congestion notification happen which dramatically decreases the deadline miss rate and the overflow drop rate. Later at *t=60* sec the B-HRT flow doubles its frequency. B-BE is selected as the victim flow and BW reassignment/notification occurs. Both the A-BE and B-BE flows are decreased to their minimum threshold 600 Bytes.

At *t=80* sec both A-HRT and B-HRT increase their fre-

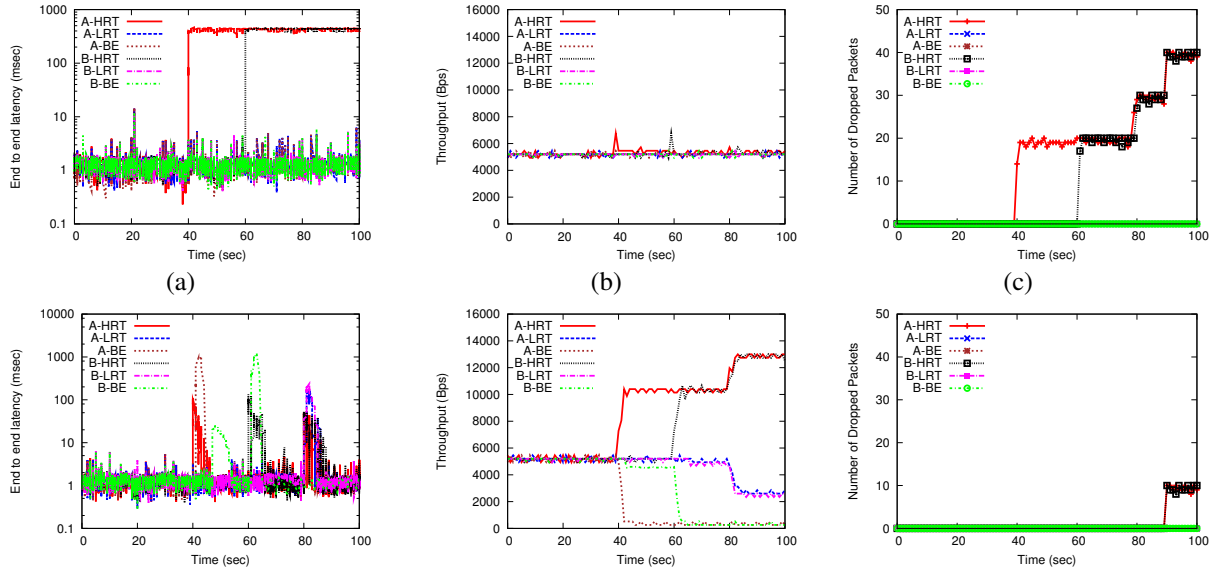

Fig. 9: *Implementation, 11-node testbed: Without CCC framework (a) End to end latency, (b) Throughput, (c) Buffer Overflow dropped packets, With CCC framework (d) End to end latency, (e) Throughput, (f) Buffer Overflow dropped packets*

Flow	$0-40$	$40-60$	$60-80$	$80-90$	$90-100$
A-HRT	0%,0%	50%,4%	50%,0%	40%,1.8%	33%,0%
A-LRT	0%,0%	0%,0%	0%,0%	0%,1%	0%,0%
B-HRT	0%,0%	0%,0%	50%,3%	40%,1%	33%,0%
B-LRT	0%,0%	0%,0%	0%,0%	0%,1%	0%,0%

TABLE I: Percentage of Deadline Missed Undropped Packets in each time interval without and with CCC framework

quency to 50 Hz. Without CCC framework usage we can see all extra packets getting dropped due to buffer overflow, whereas with framework A-LRT and B-LRT are selected as victims. Both A-LRT and B-LRT reach their min threshold 2600 Bytes. At *t=90* sec both A-HRT and B-HRT increase their frequencies to 60 Hz. As per the BW reassignment algorithm there are no more victim flows and hence all the extra packets get dropped. Finally, at *t=100* sec the global adaptation by the broker happens and the system again goes into stable state. Our CCC framework made possible to achieve the real-time guarantees of the PMU flows during the transient instability period (between 40th and 100th sec).

VI. Conclusions and Future Work

In this paper we highlight the overlay congestion problem that occurs in NASPInet. Our Cooperative Congestion Control solution for infrastructure critical real time networks is described. We evaluate our design and protocol and observe that local adaptation via cooperative scheduling, BW reallocation and cooperative nodes coordination maintains stability in the system during the transient congestion periods and results in overall better performance.

Acknowledgment

This material is based upon work supported by the Department of Energy - TCIPG project under Award Number DE-OE0000097.

References

[1] IEEE Standard for Synchrophasors for Power Systems, IEEE C37.118 2005.

[2] North American SynchroPhasor Initiative. (2009, May) Data Bus Technical Specifications for North American Synchro-Phasor Initiative Network. [Online]. Available: http://www.naspi.org/resources/dnmtt/naspinet/naspinet databus final spec 20090529.pdf

[3] Rakesh Bobba, Erich Heine, Himanshu Khurana and Tim Yardley. Exploring a Tiered Architecture for NASPInet, 1st IEEE PES Conference on Innovative Smart Grid Technologies (ISGT '10), Gaithersburg, Maryland, January 2010.

[4] Bakken, D. E., Hauser, C. H., Gjermundrd, H., and Bose, A. Towards more flexible and robust data delivery for monitoring and control of the electric power grid, Technical Report EECS-GS-009, School of Electrical Engineering and Computer Science, Washington State University, May 2007.

[5] Bouassida, M.-S., and Shawky, M. A cooperative congestion control approach within vanets: Formal verification and performance evaluation. *Eurasip Journal on Wireless Communications and Networking* – (2010).

[6] Hoang Nguyen, Raoul Rivas, Klara Nahrstedt. iDSRT: Integrated Dynamic Soft Real-Time Architecture for Critical Infrastructure Data Delivery over WLAN. QSHINE 2009: 185-202

[7] HTB for Linux, http://luxik.cdi.cz/ devik/qos/htb

[8] Floyd, S. Tcp and explicit congestion notification. *ACM Computer Communication Review 24* (1994), 10–23.

[9] Floyd, S., and Jacobson, V. Random early detection gateways for congestion avoidance, IEEE/ACM Trans. Netw.1,4(Aug.1993), 397-413.

[10] Floyd, S., and Jacobson, V. Link-sharing and resource management models for packet networks, IEEE/ACM Transactions on Networking, Vol. 3 No. 4, pp. 365-386, August 1995.

[11] Harrison, D., and Kalyanaraman, S. An edge-to-edge overlay congestion control architecture for the internet, INFOCOM 2001.

[12] Hauser, C. H., Bakken, D. E., Dionysiou, I., Gjermundrd, K. H., Irava, V. S., and Bose, A. Security, trust and qos in next-generation control and communication for large power systems. *International Journal of Critical Infrastructures 2007* (2007).

[13] Ohsaki, H., Murata, M., Suzuki, Y., Ikeda, Y., and Miyahara, H. Rate-based congestion control for atm networks. *ACM SIGCOMM Computer Communication Review 25* (1995), 60–72.