Early Congestion Notification For Cyber-Physical Smart Grid Nodes. [*]

Ashish Choudhari, Harini Ramaprasad
Department of Electrical and Computer Engineering,
Southern Illinois University Carbondale

Bruce McMillin, Sriram Chellappan
Department of Computer Science,
Missouri University of Science and Technology

ABSTRACT

Cyber-Physical System (CPS) is a computer network of computational components that monitor and control switched physical systems interconnected by physical infrastructures. To integrate various components in CPS, we follow the *invariant* based approach, where system is integrated based on the correctness of components instead of their functionality through conjunction of non-interfering logical invariants. Our distributed, adaptive power migration algorithm uses this approach to schedule power migration messages in the network to the desired nodes without compensating for stability of computer network as well as physical system. In this paper we mainly focus on network congestion and explore a well known Explicit Congestion Notification (ECN) scheme from CPS context and demonstrate the significant improvement in overall CPS efficiency and stability. Experimentation results show how the power transfers between smart grid nodes are unaffected if nodes are allowed to exploit ECN scheme while also taking necessary actions to reduce network congestion.

1. INTRODUCTION

Future Cyber-Physical Smart Power Grid [9] system should have embedded computing devices that monitor and control distributed generation, storage and power transfers while also maintaining safety, reliability and efficiency in a secure manner. Maintaining stability and correctness of all the components in CPS is a major challenge. Compensating for stability or correctness in one component is reflected in other components. For example, incorrect actions taken in physical domain can cause network to go unstable. Therefore, it is crucial to have correct scheduling of actions in domains to ensure the overall system stability.

Although there are methodologies that consider one or two components of a system and compose stability, such as switched-systems theory [6] that models stability of a plant, Hybrid automata [8], timed I/O automata [2] that represents a mix of continuous and discrete states in verification process [4, 15], our framework of composing components with stability in CPS is based on $invariants$ technique. Figure 1 shows the composition of cyber, physical, network and scheduling components through conjunction of logical invariants, where overall system stability and correctness is expressed in the form of invariants ($I_{C1} \wedge I_{C2} \wedge I_{Cn}$). This approach is not only limited to smart grid design, but can also be generalized to different cyber-physical systems with different functionalities. The scheduling domain in Figure 1 refers to scheduling of power transfer messages between smart-grid nodes over the communication network. Thus, "Network" and "Scheduling" being tightly coupled, are depicted forming one invariant in Figure 1. In this paper, we insist scheduling of actions (power messages) at *appropriate* times such that the system stability is maintained, rather than performing actions at pre-defined intervals as done in traditional real-time scheduling. In Cyber-Physical smart grid system, stability and state of a physical system depends on, state of the communication network carrying power migrate messages and power migrate acknowledgment messages between the nodes. Therefore, state and stability of communication network is also affected by number of outstanding (in transit) power migrate messages.

In recent work [18], we developed a scheduling invariant for our distributed, adaptive algorithm for scheduling power migrations between smart grid nodes and demonstrated that conjunction of such a scheduling invariant and an invariant

[*]This work was supported in part by the Future Renewable Electric Energy Delivery and Management Center, a National Science Foundation supported Engineering Research Center under grant NSF EEC-081212

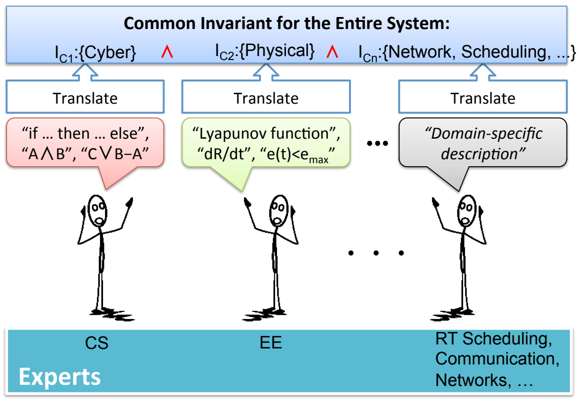
Common Invariant for the Entire System:
I_C1:{Cyber} ∧ I_C2:{Physical} ∧ I_Cn:{Network, Scheduling, ...}
Translate
Translate
Translate
"if ... then ... else", "A∧B", "C∨B–A"
"Lyapunov function", "dR/dt", "e(t)<e_max"
"Domain-specific description"
CS
EE
RT Scheduling, Communication, Networks, ...
Experts

Figure 1: Overview of invariant-based approach

for physical system state is necessary to maintain overall system stability. In order to improve efficiency along with stability, components in CPS must have certain amount of inter-component information. In the current paper, we focus on improving the efficiency while also maintaining the stability of smart grid nodes by exploiting the network congestion information obtained from Early Congestion Notification (ECN) scheme. In this scheme, packets are marked indicating impending congestion, instead of dropping them [7, 13, 14]. We allow the smart grid nodes to sense the possible upcoming network congestion and change the amount of power being transferred with every power migrate message in order to compensate for reduced rate of power transfers. As of our knowledge, this is the first work that explore ECN scheme from CPS context for the benefit of physical system efficiency as well as take necessary action to reduce network congestion.

The rest of this paper is organized as follows. Section 2 provides some background information and discusses related work. We present our system model and assumptions in Section 3. Section **??** briefly discusses our adaptive scheduling algorithm. In Section 5 we explain Early Congestion Notification technique and propose modifications required for scheduling algorithm to support ECN. Section 6 discusses an idea to vary amount of power transferred with every power transfer message. Our experimental setup and results are presented in Section 7 followed by conclusion in Section 8.

2. BACKGROUND AND RELATED WORK

Most of the Cyber-physical systems are switched, continuous time dynamic systems with changes occurring at discrete time intervals [12]. Analyzing CPS is a complicated task, as it consists of tight coupling of various components that are heterogeneous in nature, such as, cyber, physical and network. In CPS, actions performed in one component should not affect the stability in another components. Therefore, it is mandatory to schedule *correct* actions at *correct* time in each component of CPS in order to maintain overall system stability. However, only stability and correct scheduling is not enough, the system should also be adaptive in nature when uncertain components are present in the composed CPS. Typically, most of the CPS kind of applications, involve communication network as one of the important components ("smart grid" as a prime example). Standalone physical systems with computational capability, such as power plants, vehicular systems, medical devices, and robotics - just to name few - have been deeply studied in the literature, but when these kind of systems are interconnected through unreliable or uncertain communication network, like internet, physical system aspects, assumptions, control strategies and functional behavior is affected due to the unpredictability of transmission time in the communication network.

A verification technique for Cyber-Physical systems was recently proposed in [11] assuming communication network to be CAN, FlexRay. etc, wherein authors use structured properties of network infrastructures and efficiently bound network delays. In [10], authors propose an idea of scheduling power demands for optimal energy management in smart-grid, but they do not consider the network behavior. In traditional real-time systems, dynamic adaptation is normally performed at "sporadic intervals" using elastic scheduling techniques [3] and feedback scheduling [17], but CPS requires continuous adaptation. Adaptive scheduling proposed in [5], dynamically changes the task execution rate based on the observed system behavior, but requires complete abstraction of physical and network parameters. Similar approach can be taken to dynamically change the power transfer rate across the smart-grid nodes based on the observed network conditions, but abstraction of internet like network parameters (eg. RTT, Round-Trip time) is non-trivial. In our recent paper [18], we proposed an adaptive algorithm which schedules power migrate messages between CPS smart-grid nodes based on the observed round-trip times in internet like network. This algorithm forms invariant which in conjunction with physical system invariant achieves stability of the overall system. The main reason of unpredictable round-trip times (in internet) is the nature of the traffic, increase in traffic might lead to exponential increase in transmission delays and can also cause messages to drop.

[******add citations] **Protocols for CPS,** although internet protocols such as TCP are well known for reliability, TCP relies on packet drops caused due to overflow of queues at gateways as an indication of network congestion. In smart grid context, every message is responsible for a small amount of power in the grid, therefore a loss of message is directly associated to the grid stability. TCP by controlling its transmission rate can effectively reduce packet drops if transport is capable of ECN. But, congestion information obtained from network is completely hidden from the application running over TCP. In order to know ECN in application, User Datagram Protocol (UDP) is a perfect choice. It is then application's responsibility to adapt according to network conditions and take necessary actions upon detection of congestion in the network. [******end citations]

3. SYSTEM MODEL AND ASSUMPTIONS

Power Management Architecture. Figure 2 shows the architecture of a future generation smart grid (SmartGrid) [9]. The system is essentially a microgrid consisting of energy storage devices (DESD), energy resources (DRER) and LOADs. Each node is potentially owned and located in a residence or business and the basic idea is to share power among nodes in order to benefit the overall system. Intelligent flow controllers (nodes) contain Solid State Transformers (SSTs), that are physical actuators controlling power flow to and from a shared electrical bus under the direction of co-operating Distributed Grid Intelligence (DGI) processes.

The DGI processes are cyber algorithms that choose, negotiate and manage power transfers among nodes based on local information and information about the states of other nodes that is periodically exchanged among nodes. In the current work, we assume that all nodes are synchronized, for the sake of simplicity. Nodes periodically exchange state information with each other in a state collection phase. Next, a negotiation phase is performed in which nodes conduct negotiations to identify which power transfers need to be performed among nodes. The identified power transfers between pairs of nodes are then performed in a power transfer phase. One cycle is depicted in Figure3, with one or more negotiation and power transfer phase pairs after a state collection phase. This entire cycle of state collection, negotiation and power transfer phases is repeated.

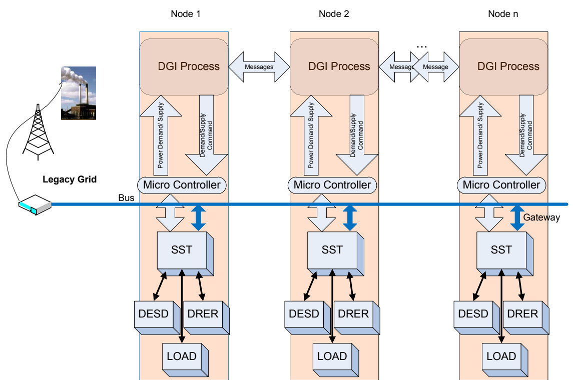

Figure 2: Smart Grid Power Management Architecture

Power Transfer Model. Power transfers within one phase are performed as a series of (periodic) power migrations, each transferring a given quantum (say δ) of power. The cyber algorithm on the source (sender) node sends appropriate control signals to the local physical actuators to add a quantum of power to the electrical bus and sends a power

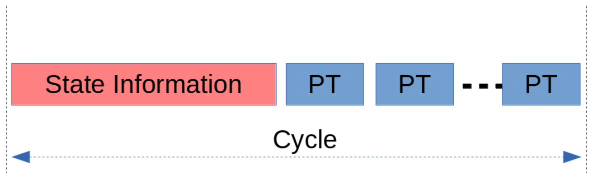

Figure 3: A single cycle of collecting state information followed by many negotiation and power transfer phases (PT)

migration message to the destination (receiver) node signaling this. Upon receiving this power migration message, the destination node sends control signals to its local physical actuators to remove a quantum of power from the electrical bus and sends an acknowledgement message to the source node.

Physical System Model. The physical system is a finite inertia microgrid, that is, a power system with a small number of generators and loads that acts independently of the grid. There is a relatively small rotating generator, whose inertia dominates the dynamics as described in the next section. There are some number of controllable nodes that participate in the power management by acting as either loads or generators. Nodes with excess generating capacity transfer power to nodes that have excess load.

Communication Model. Each node in the network runs an adaptive message scheduling algorithm that schedules power migrate messages in any given topology, based on the observed communication latencies at a given node.

4. ADAPTIVE COMMUNICATION

Physical system stability at any point of time is determined by the product of power migration messages K in transit and amount of power δ transferred with every message. Stability of network depends only on K, where K is a function of rate(s) of power migration and observed round-trip time between source and receiver nodes in the communication network. To provide safe and stable operation, an upper bound $Kmax_{global}$ for maximum outstanding messages is established prior to power transfer phase, based on physical and network restrictions. $Kmax_{global}$ is then distributed as in Figure 4 among the nodes that have excess power and needs to undergo power migrations. For example, in Equation 1, $Kmax_{global}$ is distributed among n nodes in proportion to excess power. After distribution, the message scheduling algorithm running at node s, adapts the power message migration rate r_s or, inversely, the period p_s such that the number of outstanding messages of node s at any given time, never exceeds its maximum allowed outstanding messages $Kmax_s$.

$$Kmax_s = \lfloor (\frac{P_s}{\sum_{i=1}^{n} P_i}) * Kmax_{global} \rfloor \tag{1}$$

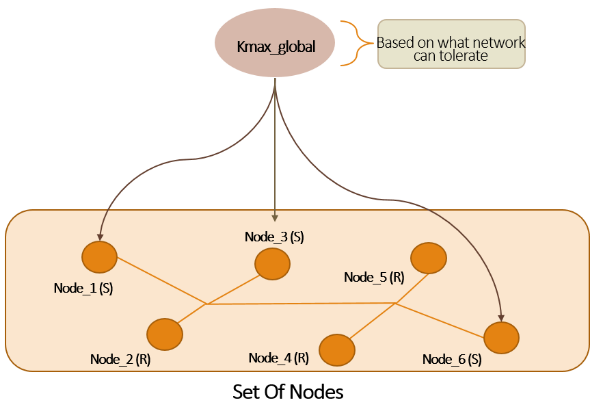

Figure 4: $Kmax_{global}$ Distribution Among Sender Nodes

4.1 Scheduling Algorithm

Every power migrate message initiated by node s is assigned a relative deadline D_s which is determined from the current expected response time (RT_s^{ex}). Where, RT_s^{ex} is the average of certain number of previously observed response times (response time also known as round-trip time). If an acknowledgment m_a of message m is not received before its deadline (*deadline miss*) indicates that there is a possibility of congestion in the network. Therefore, in an effort to reduce congestion, the algorithm increases its expected response time RT_s^{ex} by a pre-defined margin ($RTMargin$) and determines a larger transmission period p_s (1/rate) and a larger relative deadline. In the worst case, power migration will only be done if acknowledgments for all the outstanding messages are received. Whereas, if certain $CtrMax$ (response time counter limit) number of acknowledgments are received consecutively with smaller response times than expected, then the algorithm reduces its transmission period ($1/r_s$) and calculates a more tighter (smaller) relative deadline, ensuring that physical and network stability is still maintained. Note that $RTMargin$ and $CtrMax$ are configurable system parameters and are assumed to be constant for all nodes in a given power transfer phase, but in this paper we do make changes to the $CtrMax$ policy in the later section.

Following is a table of events which sets the scheduling algorithm in motion.

	Event Name	**Sub Event**	**Function**
1	*Initialize*	n/a	* Configure and set system parameters
2	*Sent Power*	n/a	* Assign deadline for message m
3	*Ack Recv*	*Better Ack*	* Increment $RTCtr$, IF ($RTCtr = CtrMax$) then update r_s, RT_s^{ex}, $deadline$ and reset $RTCtr$.
		Expected Ack	* Reset $RTCtr$.
		Ack Of Dln Miss	* IF ($RT_{s(m)} > RT_s^{ex}$) then update r_s, RT_s^{ex}, $deadline$.
4	*Dln Miss*	n/a	* Increment RT_s^{ex} by $RTMargin$ and update r_s, RT_s^{ex}, $deadline$.
5	*Recv Power*	n/a	* Send signal to local actuator

Message Scheduling Invariant (I_S).

For a given power transfer phase and based on above $Kmax_s$, p_s constraints, following scheduling invariant I_s can be formed, shown in Equations 2 - 5.

$$\{I_S = I_k \wedge I_c \wedge I_p\} \tag{2}$$

$$I_k : K_s < Kmax_s \tag{3}$$

$$I_c : RT_s^{ex} \leq PT_e - t \tag{4}$$

$$I_p : t - LT(s) \geq p_s \tag{5}$$

Here, PT_e is the end time of the power transfer phase, t is the time at which the invariant is evaluated and $LT(s)$ is the time at which the last power migration message was initiated by node s. Work in [18] shows the necessity of conjunction of physical I_p and scheduling I_s invariants in order to maintain system stability.

5. ADAPTIVE SCHEDULING WITH EARLY CONGESTION NOTIFICATION

In this section, we propose modifications required for adaptive power scheduling algorithm [18] to support Early Congestion Notification (ECN) obtained from the ECN capable transport. We explain ECN mechanism in 5.1 and then propose the integration of ECN in scheduling algorithm in 5.2.

5.1 Early Congestion Notification Mechanism

Congestion detection and avoidance for TCP through ECN (Early Congestion Notification) was proposed in [7]. Essentially, ECN mechanism is implemented by maintaining two bits in IP header making four possible combinations as in Table 1. If ECT, CE are $0, 0$ then the transport(Sender, Receiver, Network) is considered to be not ECN-capable. If ECT, CE are $0, 1$ or $1, 0$ then the transport is ECN-capable and if ECT, CE are $1, 1$ then the transport is ECN-capable but also the packet into consideration has experienced congestion.

Gateways, in the network maintain Q_{min} and Q_{max} thresholds for average queue size Q_{avg} for every outgoing port. When a packet arrives at the gateway, one among the following actions is taken,

IF $Q_{min} \geq Q_{avg} \leq Q_{max}$: Set ECN bit.
IF $Q_{avg} < Q_{min}$: Take no action.

	ECT	CE	Codepoint
	0	0	Not-ECT
RFC 3168	0	1	ECT(1)
	1	0	ECT(0)
	1	1	CE

Table 1: ECN Field In IP Header

IF $Q_{avg} > Q_{max}$: Drop packet.

Receiver, upon arrival of packet with ECN bit set (Congestion experienced), sets a flag in the acknowledgment packet header and send it back to Sender.

Sender, when receives this acknowledgment (having congestion flag set), reduces the rate of transmission in order to avoid possible upcoming congestion. For example, in TCP, sender reduces the transmission rate by reducing its congestion window size and slow start threshold. This mechanism avoids the unnecessary packet drops during mild congestion. As stated in [23], ideally, marking based network can avoid congestion through cooperative actions of responsive sources and completely eliminate packet drops.

Although there are variations of ECN, such as Backward ECN (BECN) [19, 20], Forward ECN (FECN) [21], Enhanced Forward ECN (E-FECN) [22], we experiment using the congestion notification obtained from the receiver in acknowledgment packets (power acknowledgments in smart grid context).

5.2 Integration: Scheduling With ECN

According to original scheduling algorithm, deadline miss event is triggered at a sender node s when an acknowledgment m_a of a message m is not received before its deadline d. Reason for deadline miss is longer response time RT (Round-Trip Time of m) than expected. RT is affected by factors such as, link delays, increase in traffic (sharing same path in network) leading to queue buildup at gateways and processing delay at receiver.

In ECN capable transport, as stated in section 5.1, if Q_{avg} reaches Q_{min}, message m is marked as congestion experienced (ECN bit is set in m). When sender s detects ECN bit set in acknowledgment m_a, it is not guaranteed that m_a will have a larger RT than the system is currently expecting (RT_s^{ex}). ECN is just an indication of *impending* congestion. Thus, m_a with ECN bit set may not trigger a deadline miss event, or will not cause the rate to reduce. Therefore, to adapt due to ECN, we introduce a boolean flag ECN_{Status} and a new sub event *Detected ECN* under *Ack Received Event* in the scheduling algorithm. Depending on ECN_{Status} the scheduling algorithm changes its behavior dynamically and switches between the following modes,

Normal Mode: $ECN_{Status}(0)$**:** No congestion detected. Algorithm runs as per section 4.1.

ECN Mode: $ECN_{Status}(1)$**:** ECN mode is activated when a first acknowledgment is received indicating congestion. Figure. 5 shows the flowchart of *Ack Received* event in ECN capable transport. In this mode, ECN_{Status} is set to 1, $RTCtr$ is reset to zero and RT_s^{ex} is incremented by $K_s{*}RTMargin$ instead of increasing by only $RTMargin$, as done in *Deadline Miss* event. If $K_s < Kmax_s$, node parameters are recalculated using Equations 6 - 8. Otherwise, r_s is set to $1/RT$ and other node parameters are calculated using Equations 7 and 8. As we increase RT_s^{ex} in proportion to K_s (outstanding messages of sender node), rate r_s of power transfer also reduces in proportion to K_s. ECN mode remains active for currently observed RT, after that it is deactivated and the algorithm returns back to Normal mode. The reason to do this is to avoid frequent adaptations due to acknowledgments indicating congestion. In a simple sense, ECN mode is activated only if the algorithm is currently running in Normal mode. Switch from ECN mode to Normal mode is implemented by $ECN\_Mode\_Dact\_t$ variable.

Problem. Message acknowledgments received after the scheduling algorithm has switched to ECN mode, might have similar or even better (smaller) RT to that of the first acknowledgment which caused the algorithm to switch to ECN mode (ECN being indication of *impending* congestion). In this scenario, if $CtrMax$ number of consecutive acknowledgments are received with better RT, then adaptation due to better RT will be performed, essentially, sub event *Ack received is better* in *Ack Received* will be triggered. In sub event *Ack received is better*, message transfer rate r_s is increased and tighter deadline D_s is calculated. Thus, the efforts made to avoid congestion (by reducing r_s) after switching to ECN mode are compensated. The rate r_s decreased after switching to ECN mode is again increased by triggering of *Ack received is better*, because $CtrMax$ number of messages are most likely to appear consecutively with better RT.

$$r_s = \frac{(Kmax_s - K_s)}{RT_s^{ex}} \tag{6}$$

$$p_s = \frac{1}{r_s} \tag{7}$$

$$D_s = RT_s^{ex} + RTMargin \tag{8}$$

Solution. To avoid triggering of *Ack received is better* event in ECN mode, we increase the $CtrMax$ limit to $2{*}K_s$, as shown in Figure.5, in *Detected ECN* event block. Increasing $CtrMax$ does not completely eliminate the triggering of *Ack received is better* event, but instead makes it more tough to happen. Although $CtrMax$ is a configurable parameter

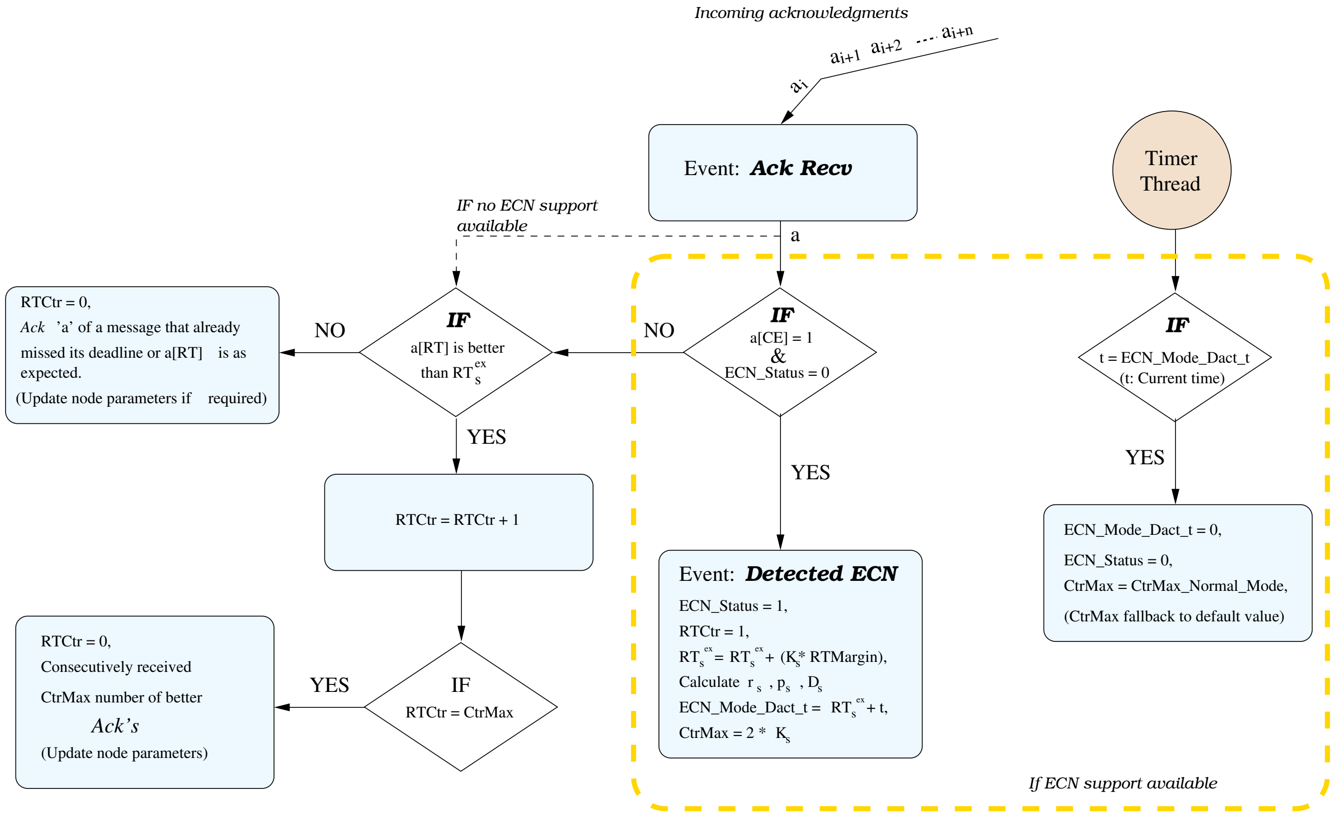
Incoming acknowledgments
a_i a_{i+1} a_{i+2} ---- a_{i+n}
Event: Ack Recv
Timer Thread
IF no ECN support available
a
NO
IF a[CE] = 1 & ECN_Status = 0
IF t = ECN_Mode_Dact_t (t: Current time)
RTCtr = 0, Ack 'a' of a message that already missed its deadline or a[RT] is as expected. (Update node parameters if required)
NO
IF a[RT] is better than RT_s^{ex}
YES
YES
YES
RTCtr = RTCtr + 1
Event: Detected ECN
ECN_Status = 1, RTCtr = 1, $RT_s^{ex} = RT_s^{ex} + (K_s * RTMargin)$, Calculate r_s, p_s, D_s, ECN_Mode_Dact_t = RT_s^{ex} + t, CtrMax = 2 * K_s
ECN_Mode_Dact_t = 0, ECN_Status = 0, CtrMax = CtrMax_Normal_Mode, (CtrMax fallback to default value)
RTCtr = 0, Consecutively received CtrMax number of better Ack's (Update node parameters)
YES
IF RTCtr = CtrMax
If ECN support available

Figure 5: Flowchart Of *Ack Recv* **Event With ECN Capable Transport**

in ECN mode, similar to as in Normal mode, we recommend setting the value of $CtrMax$ proportional to outstanding messages K_s. Note that $CtrMax$ is calculated only once per ECN mode switch.

6. VARYING QUANTUM OF POWER (δ)

In previous section, we have successfully integrated a feature of ECN in our adaptive power message scheduling algorithm. In this section we make changes to the policy of amount of δ carried by every power transfer message. The idea is to vary δ size in proportion to period p_s (or 1/rate, $1/r_s$). But when we talk about internet like network, it is non-trivial to bound the rate of transmission. So, each time a message scheduling invariant I_S is evaluated, we make an estimate on total amount of power that can be transferred in the current power transfer phase PT, provided, the current scheduling parameter r_s (rate) does not change. Based on this estimate, we calculate δ as in Equation.9.

$$\delta = \frac{P_G - P_A}{r_s * (PT_e - RT_s^{ex})} \tag{9}$$

Where, P_G is total power granted to receiver node r, P_A is total power acknowledged from node r (node r (receiver node's) stamps total power received in current PT (power transfer phase) in every ack), r_s is current rate of power transfer message, PT_e is absolute end time of current power transfer phase and RT_s^{ex} is current observed response time of message.

Problem. Message scheduling invariant I_S formed by conjunction of sub invariants I_k, I_c and I_p, will not reflect the correct scheduling system state if δ is allowed to vary. As I_k, satisfied if $K_s{<}K_{max_s}$, no more reflects the amount of outstanding power in the grid. K_s only reflects the total outstanding messages in the network, because in varying δ scenario, every power transfer message might carry different amount of δ along them.

Solution. We derive the bounds on δ as, δ_{min} and δ_{max} based on the physical system tolerance (allowance) [****input from physical system guys that δ_{min} and δ_{max} can be provided, ensuring that the system stability won't be affected] without compensating for stability. Now in the physical system analysis, in section**??**, where we say "I_{P1} ultimately imposes a limit on $\delta * K$", will not hold. Therefore, we modify the term δK in Equation 3 to $P_T - P_A$ and restate the Equation as in 10.

$$\left\{ \begin{array}{c} I_{P1} : (\omega - \omega_0)^2 \, (D\omega + m) \\ +(\omega - \omega_0)(kP^2) > (P_T - P_A)(\omega - \omega_0) \end{array} \right\} \tag{10}$$

Where, P_T is the total power transferred to node r (receiver node) and P_A is the total power acknowledged from node r. (Note that P_G is total power granted, power transfer stops due to the end of PT or P_G = P_T = P_A).

We also modify the sub invariant I_k in scheduling invariant I_S as in Equations 11-13.

$$I_k : I_{kp} \wedge I_{kn} \tag{11}$$

$$I_{kp} : (\delta_{new} + (P_T - P_A)) < (\delta_{max} * K_{max_s}) \tag{12}$$

$$I_{kn} : K_s < K_{max_s} \tag{13}$$

Where, δ_{new} is obtained from Equation 9.

7. EXPERIMENTAL SETUP

We have implemented our algorithm *MsgSchedModule* for adaptive message scheduling with ECN and varying δ support using C++ in linux environment. Boost portable C++ source libraries [24] were used for timing needs in our *MsgSchedModule*. Network was emulated using OMNet++, an open source network simulator [1, 16]. OMNet++ is a discrete event simulator that provides an extensible, modular, component-based C++ simulation library. OMNet++ also supports emulation of network in real time instead of just being a simulator [25]. Real-time scheduler class *cRealTimeScheduler()* in OMNet++ was used in our network emulation. Figure 6 shows the network setup used to emulate every power transfer phase. Nodes (denoted as *realPeer50* and *simPeer0* to *simPeer8*), communicate with each other over links and routers (*router0* to *router5*). Nodes are also capable of generating traffic in the network. Note that *MsgSchedModule* is an external module which connects to *realPeer50* through socket to *localhost* at configured *port*. In this paper, we experiment using *realPeer50* as an excess power node. Routers employ first-in-first-out (FIFO) queue with configurable service time Q_{st} for every (total 4 ports) outgoing port. Routers being ECN capable, maintain queue average Q_{avg}, check Q_{avg} against Q_{min} and Q_{max} thresholds and set congestion experienced (CE) bit in messages if required. Note that Q_{min}, Q_{max} and Q_{st} are configurable parameters. For the current paper experiments, Q_{min}, Q_{max} and Q_{st} are set to 3, 20 and 0.2s (seconds) respectively. In the current paper, we focus on studying the behavior of our *MsgSchedModule* featured with ECN support and varying δ support versus non ECN support and constant δ. For a given power transfer phase, the required input data for *realPeer50* as P_G (amount of power granted by *realPeer50* to some node, say *simPeer8*), K_{max}, RT_s^{ex}, $RTMargin$ and $CtrMax$ (for Normal mode, because in ECN mode $CtrMax$ increases in proportion to outstanding messages K) is assumed to be available from state information phase (Figure 3).

System configuration, *realPeer50* is excess power node, *simPeer8* is the node demanding 20000 $units$ of power and $K_{max_{global}}$ is set to 20. Since there is only one node *realPeer50* with excess power in the experimenting power transfer phase, K_{max} for *realPeer50* is 20, i.e *realPeer50* can have maximum of 20 outstanding power transfer messages in the network. Quantum of power δ is set to 10 for experiments with constant δ, whereas δ is allowed to vary between 10 to 30 (δ_{min}, δ_{max}) for experiments with varying δ. Route between *realPeer50* and *simPeer8* is through *router0-router2-router4-router5*. To experiment with traffic, *simPeer7* and *simPeer5* communicate with each other through route *router2-router4* and generate messages such that they occupy *50 percent* of total bandwidth. For all the experiments with traffic, traffic is generated from time $t = 5000$ to $t = 10000$.

Considering that *realPeer50* granted 20000 $units$ of excess power it has to *simPeer8*, *realPeer50* initializes its local variables as, $P_{G(8)} = 20000$, $P_{T(8)} = 0$ and $P_{A(8)} = 0$. In case of constant δ, whenever *realPeer50* wants to transfer δ amount of power, invariants I_S and I_P are evaluated, if they return $true$, or are satisfied, then only a power transfer message is dispatched to *simPeer8*. Whereas in varying δ case, before I_k in I_S is evaluated, *MsgSchedModule* calculates δ size at that instance of time by Equation 13 and if calculated δ is greater than δ_{max} then δ is set equal to δ_{max} or if calculated δ is less than δ_{min} then δ is set equal to δ_{min}.

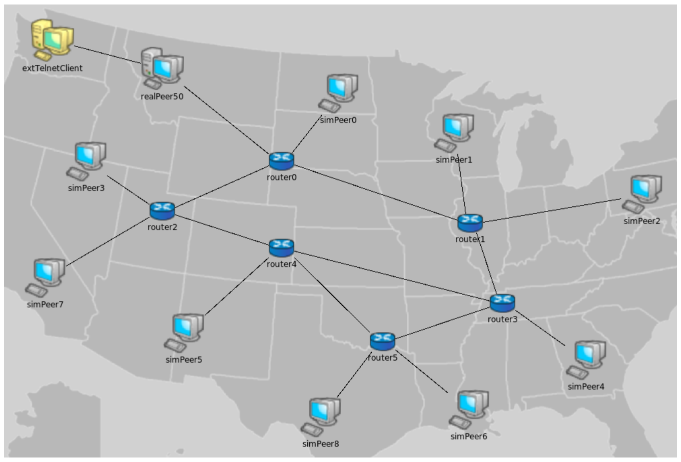

Figure 6: Emulated Network in OMNet++

7.1 Experimental Results

8. CONCLUSIONS

9. REFERENCES

[1] The OMNeT++ discrete event simulation system.
[2] R. Alur and D. L. Dill. A theory of timed automata. *Theoretical Computer Science*, 126(2):183–235, 1994.

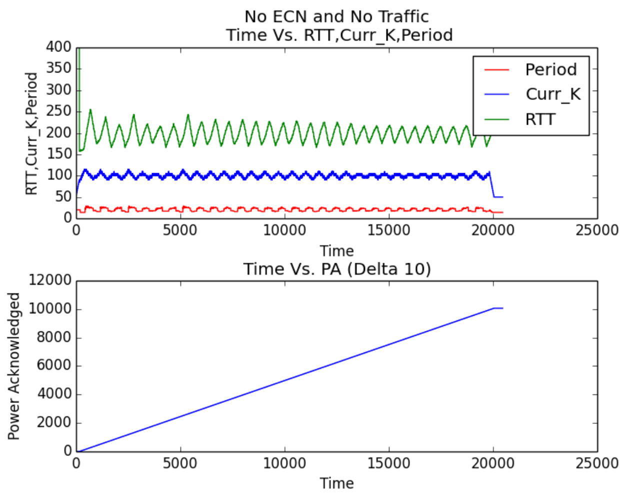

Figure 7: No ECN, No Traffic, δ = 10

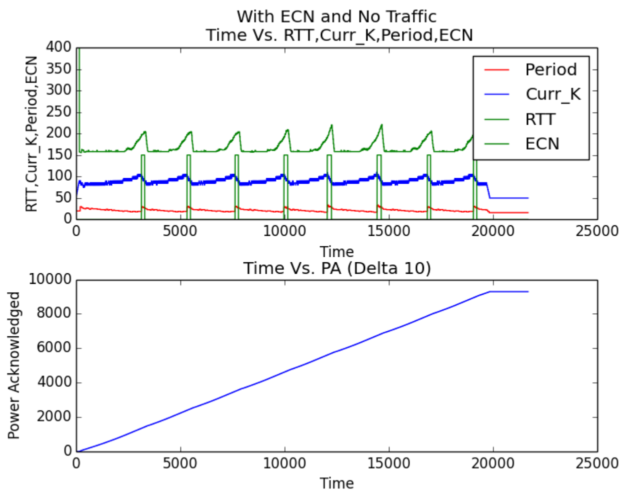

Figure 9: With ECN, No Traffic, δ = 10

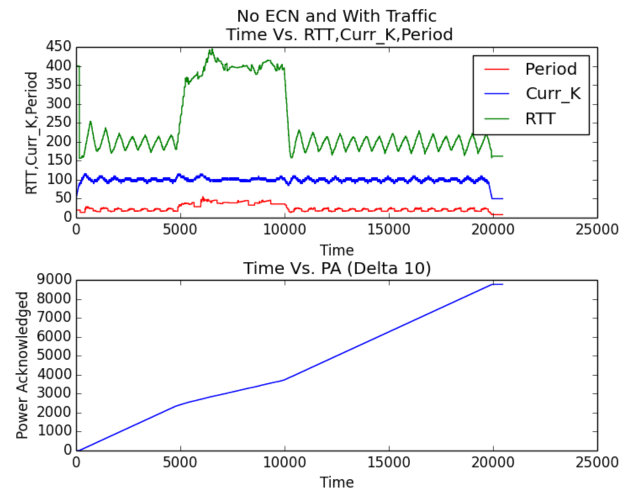

Figure 8: No ECN, With Traffic, δ = 10

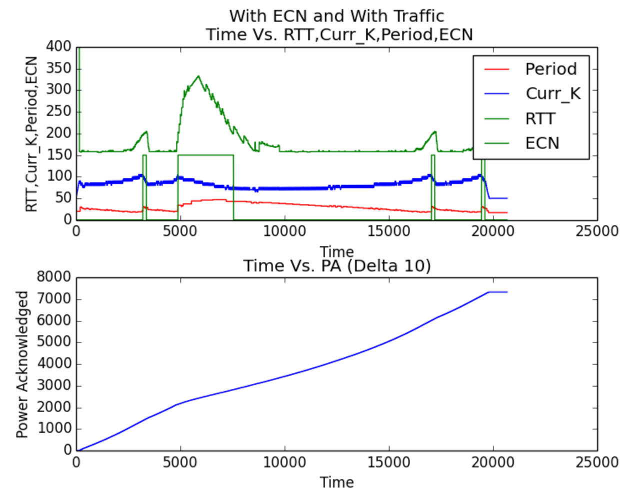

Figure 10: With ECN, With Traffic, δ = 10

[3] G. Buttazzo, G. Lipari, M. Caccamo, and L. Abeni. Elastic scheduling for flexible workload management. *Computers, IEEE Transactions on*, 51(3):289 –302, mar 2002.

[4] A. Chutinan and B. H. Krogh. Computational techniques for hybrid system verification. *IEEE Transactions on Automatic Control*, 48:64–75, January 2003.

[5] B. Doerr, T. Venturella, R. Jha, C. Gill, and D. Schmidt. Adaptive scheduling for real-time, embedded information systems. In *Digital Avionics Systems Conference, 1999. Proceedings. 18th*, volume 1/17 pp. vol.1, pages 2.D.5–1 –2.D.5–9 vol.1, nov 1999.

[6] M. C. F. Donkers, W. P. M. H. Heemels, N. van de Wouw, and L. Hetel. Stability analysis of networked control systems using a switched linear systems approach. *IEEE Transactions on Automatic Control*, 56:2101–2115, 2011.

[7] S. Floyd. Tcp and explicit congestion notification. *ACM SIGCOMM Computer Communication Review*, 24(5):8–23, 1994.

[8] T. A. Henzinger. The theory of hybrid automata. In *IEEE Symposium on Logic in Computer Science*, pages 278–292, 1996.

[9] A. Q. Huang, M. L. Crow, G. T. Heydt, J. P. Zheng, and S. J. Dale. The Future Renewable Electric Energy Delivery and Management (FREEDM) System: The energy internet. *Proceedings of the IEEE*, 99(1):133–148, Jan. 2011.

[10] I. Koutsopoulos and L. Tassiulas. Control and optimization meet the smart power grid: scheduling of power demands for optimal energy management. In *Proceedings of the 2nd International Conference on Energy-Efficient Computing and Networking*, e-Energy '11, pages 41–50, 2011.

[11] P. Kumar, D. Goswami, S. Chakraborty, A. Annaswamy, K. Lampka, and L. Thiele. A hybrid approach to cyber-physical systems verification. In *49th ACM/EDAC/IEEE Design Automation Conference (DAC)*, pages 688 –696, june 2012.

[12] D. Liberzon. *Switching in Systems and Control*. Birkhauser, Boston, 2003.

[13] K. Ramakrishnan and S. Floyd. A proposal to add explicit congestion notification (ecn) to ip. Technical report, RFC 2481, January, 1999.

[14] K. Ramakrishnan, S. Floyd, D. Black, et al. The addition of explicit congestion notification (ecn) to ip, 2001.

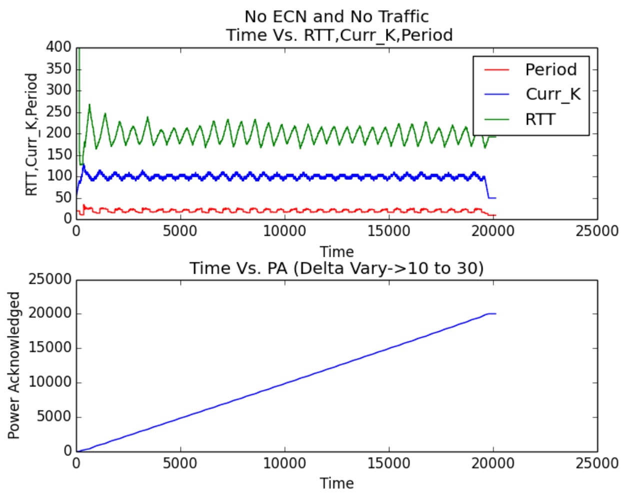

Figure 11: No ECN, No Traffic, Vary δ 10 to 30

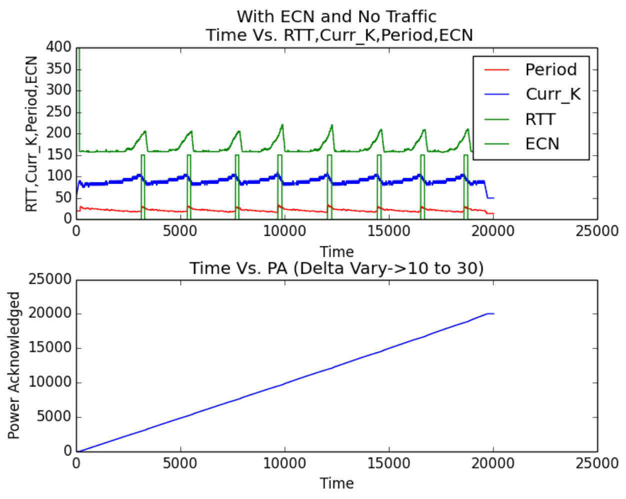

Figure 13: With ECN, No Traffic, Vary δ 10 to 30

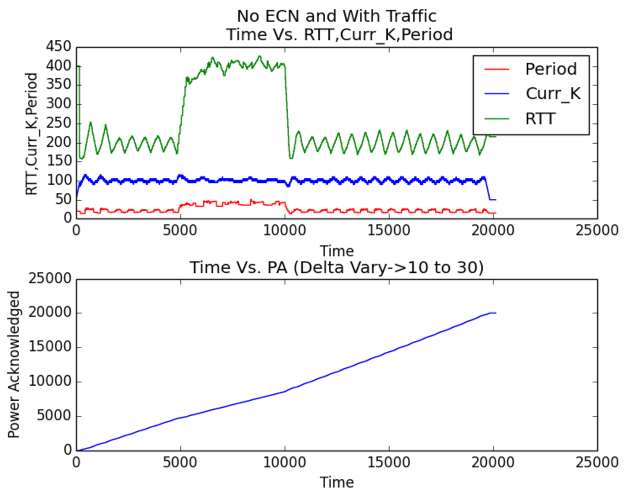

Figure 12: No ECN, With Traffic, Vary δ 10 to 30

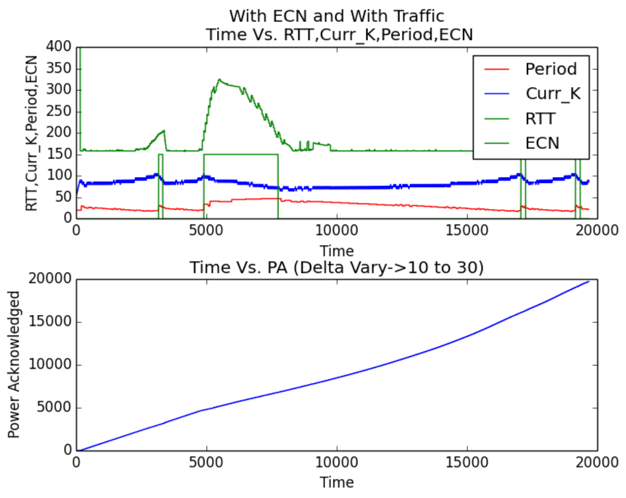

Figure 14: With ECN, With Traffic, Vary δ 10 to 30

[15] C. J. Tomlin, I. Mitchell, A. M. Bayen, and M. Oishi. Computational techniques for the verification of hybrid systems. *Proceedings of the IEEE*, 91:986–1001, July 2003.
[16] A. Varga. The omnet++ discrete event simulation system. *Proceedings of the European Simulation Multiconference (ESM'2001)*, June 2001.
[17] F. Xia, G. Tian, and Y. Sun. Feedback scheduling: an event-driven paradigm. *SIGPLAN Not.*, 42(12):7–14, Dec. 2007.
[18] A. Choudhari, H. Ramaprasad, T. Paul, J. Kimball, M. Zawodniok, B. McMillin, S. Chellappan. Stability of a Cyber-Physical Smart Grid System using Cooperating Invariants. In *Proceedings of the 37th Annual IEEE International Computer Software and Applications Conference*, COMPSAC, Kyoto, Japan, 2013.
[19] D. Bergamasco. Datacenter Ethernet Congestion Management: Backward Congestion Notification. *IEEE 802.1 Meeting, May 2005.*
[20] D. Bergamasco, R. Pan. Backward Congestion Notification Version 2.0. *IEEE 802.1 Meeting, September 2005.*
[21] J. Jiang, R. Jain, C. So-In. Congestion Management for Ethernet In Datacenter Application Using Forward Explicit Rate Notification. *WUSTL technical report, 2007.*
[22] C. So-In, R. Jain, J. Jiang. Enhanced Forward Explicit Congestion Notification (E-FECN) Scheme for Datacenter Ethernet Networks. In *Proceedings of 2008 International Symposium on Performance Evaluation of Computer and Telecommunication Systems*, SPECTS, Edinburgh, UK, June 16-18, 2008.
[23] M. Malowidzki. Simulation-Based Study Of ECN Performance In Red Networks.
[24] The boost library. *http://www.boost.org*
[25] C. P. Mayer, T. Gamer. Integrating real world applications into OMNeT++. *Telematics Technical Report, TM-2007-2*, Universität Karlsruhe (TH), Feb. 2008. ISSN 1613-849X.