A

QoS-driven Data Processing Algorithms for Smart Electric Grids

Kedar Khandeparkar, Indian Institute of Technology, Bombay
Krithi Ramamritham, Indian Institute of Technology, Bombay
Rajeev Gupta, Iqlect, India

Smart-grid applications have widely varying data needs as well as bandwidth and latency requirements. The usual approach to accumulating the available data (e.g., from Phasor Measurement Units) at a centralized site and executing all the applications there leads to large network latencies. This paper proposes techniques where data packets are prioritized and disseminated based on applications' data needs and semantics. In particular, these techniques systematically exploit in-network processing capability and filter data in the dissemination network. This filtered data is assigned higher priority compared to the raw unfiltered data– helping meet QoS requirements of various applications. Performance evaluation of our distributed techniques over a test-bed designed for the Indian Electric Grid demonstrates that processing latency for various smart grid applications reduces by at least 50% for large PMU data sizes, compared to the traditional centralized approach.

CCS Concepts: •**Computing methodologies** → **Distributed algorithms;**

General Terms: Design, Algorithms, Performance

Additional Key Words and Phrases: Smart grid; Query processing; Latency; Bandwidth; PMU; PDC

ACM Reference Format:
Kedar Khandeparkar, Krithi Ramamritham, and Rajeev Gupta, 2017. QoS-driven Data Processing Algorithms for Smart Electric Grids. *ACM Trans. Cyber-Phys. Syst.* V, N, Article A (January YYYY), 24 pages.
DOI: http://dx.doi.org/10.1145/0000000.0000000

Abbreviations

ASM	Angular Stability Monitoring
CEUT	Centralized Execution with Unfiltered data forwarding Technique
COI	Center of Inertia
DEFT	Distributed Execution with Filtered data forwarding Technique
LPDC	Lower level Phasor Data Concentrator
LSE	Linear State Estimation
MCGG	Monitoring Coherent Groups of Generators
PDC	Phasor Data Concentrator
PMU	Phasor Measurement Unit
SE	State Estimation
SPDC	Super Phasor Data Concentrator

This work is supported by the DeitY, Govt. of India, and TCS, under grant DeitY/R&D/ITEA/4(3)/2012.
Author's addresses: Kedar Khandeparkar and Krithi Ramamritham, Computer Science Department, IIT Bombay, Mumbai 400076; Rajeev Gupta, Iqlect, India.
Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
© YYYY ACM. 2378-962X/YYYY/01-ARTA $15.00
DOI: http://dx.doi.org/10.1145/0000000.0000000

1. INTRODUCTION

1.1. The Electric Grid Infrastructure

The emergence of the smart electric grid is driven by the increased need for achieving reliability, stability, distributed generation, and integration of renewable sources. A large number of monitoring and sensing devices, such as Phasor Measurement Units (PMUs), are being deployed throughout the network. Typical PMU data measured at a substation (generation or transmission) has three voltage phasors, three current phasors, one frequency value, a GPS time-stamp, and other analog and digital values [IEEE Std C37.118.2 2011]. In a smart grid, the signals are typically sampled and communicated at high rates – several 10s or 100s of times per second – to augment or even replace the conventional supervisory control and data acquisition (SCADA) system in which measurements are available 4 times per second from remote terminal units. A typical PMU data packet size is 100 bytes, but larger packet sizes are expected to be the norm soon as more and more grid parameters get included for monitoring [PGCIL. 2012; IEEE Std C37.118.2 2011]. Thus, with PMUs having 90 phasors and 45 analog values at 100 Hz of sampling frequency, data from a single PMU could exceed 700 kbps [Martin 2015]. For India's grid, a total of 1600 PMUs is envisaged, making the total data size exceed 1 Gbps. These PMU measurements are synchronized using a GPS clock, enabling a consistent snapshot of the system. These provide the data necessary to assess grid performance and enhance the ability to control system operations and management.

Various analytical tools and algorithms are used to aggregate and analyze the PMU data from different geographical locations [Phadke 1993]. Phasor Data Concentrators (PDCs) at one or more levels aggregate and integrate the PMU data. Lower level PDCs (LPDCs) aggregate data from PMUs that are located at different places, time align the data, and send the aggregated data to a higher level or super PDCs (SPDCs). Figure 1 shows a three-layer hierarchical structure of the planned electric grid [PGCIL. 2012] with GPS synchronized PMUs sending data to PDCs. Grid applications are run at SPDC over the aggregated data. These grid applications include angular stability monitoring (ASM), voltage monitoring, island detection, frequency monitoring, monitoring coherent groups of generators (MCGG), system state estimation (SE), etc. In this paper, we raise the following questions pertaining to a smart grid executing a number of such applications:

— Is it possible for centralized approaches– wherein all data is collected and processed at a central site to meet the timing needs of smart grids that are being rolled out across the world?
— What is the nature of the real-time processing as well as the data needs of typical applications, and do they demand other approaches besides the usual centralized approach?
— What is the impact of (a) the presence of concurrent applications, and (b) failures in the grid, on the above questions?

We answer these questions by specifically examining three different grid applications as use cases, as discussed next.

1.2. Grid Applications and their Requirements

Grid applications have varying quality of service (QoS) requirements, usually specified in terms of sensor data rates to be supported, data required, criticality of the application, tolerable data latency, geographic movement of data, and the deadline for bulk data transfer [Bakken et al. 2011]. In this paper we consider three smart grid applications.

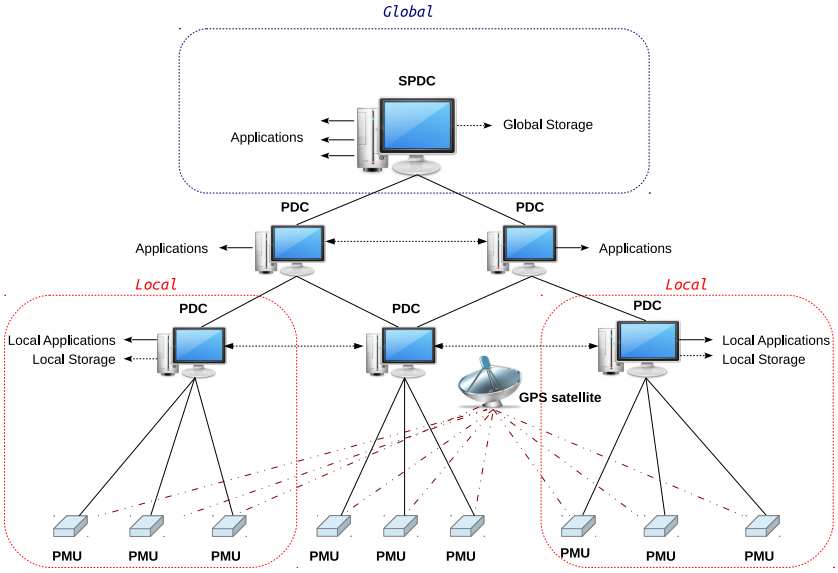


Fig. 1. The structure of the proposed Indian electric grid

Angular Stability Monitoring (ASM): Phase angle differences across different terminal nodes (also known as buses in power systems) of a transmission line in a system are a measure of static stress across the grid and its propensity to instability [Hu and Venkatasubramanian 2007]. Thus, phase angle differences are required to be monitored with respect to predetermined stability thresholds.
Monitoring Coherent Groups of Generators (MCGG): In this application, certain groups of generators having similar behavior are monitored such that the difference between the centre of inertia of the generators in the group from the global centre of inertia is within a given threshold [Alsafih and Dunn 2010]. If the difference is more than the specified threshold, the grid may experience *islanding*.
State Estimation (SE): Power system state estimation is a statistical technique to estimate the system state of a bus in the face of noisy and missing PMU data [Jones 2011].

Table I lists some of the QoS requirements for these three grid applications [Bakken et al. 2011; Phadke and Thorp 2010]. This paper explores semantics-aware distributed query processing approaches by carefully examining these applications. Semantics is of two kinds: data semantics and application semantics. Data semantics involves the physical meaning of data and its characteristics whereas application semantics involves domain knowledge about what applications are trying to achieve. For example, consider the angular stability monitoring application where phase angle difference is monitored so that its value is less than 2 degrees. For this application, it does not matter whether the actual angle difference is 0.5 or 1 degree as long as it is below the threshold. Application semantics also includes the data needs of the applications (voltage, current, and frequency), the data rate required by the applications, whether the application is a monitoring, control or protection application, etc.

Table I. QoS Requirements of Grid Applications

Application	Latency (millisec)	Data items (Voltage: V, Current: I)	Data rate (packets/sec)	Time-Criticality
ASM	50	V angles	50	High
MCGG	50	V angles	50	Medium
SE	100	V, I magnitude and angles	$25-50$	Low

1.3. The Challenge of Timely Execution of the Applications

The PMU data dissemination frameworks suggested in [Vanfretti and Chow 2011; Chenine and Nordström 2011; Armenia and Chow 2010; Adamiak et al. 2005] send all the data from PMUs to higher level PDCs without considering their QoS requirements. In different embodiments, PMUs can send data directly to highest level SPDC where applications queries are executing or PMUs can send data via lower level PDCs in a heirarchical manner. Henceforth, we refer to the PMUs sending all the data to SPDC via LPDCs as ***C**entralized **E**xecution with **U**nfiltered data forwarding **T**echnique* (CEUT). Figure 2a shows LPDCs forwarding all the data packets received from downstream PMUs to SPDC. In CEUT, the PDC waits for the arrival of data packets associated with the same time-stamp from PMUs till it times out. If a PDC does not receive all the data before the time-out, it sends whatever has been received to higher PDCs. An LPDC parses the input packet, extracting the measured values from the binary PMU data packet that are compliant to IEEE C37.118.2 format [IEEE Std C37.118.2 2011]. For time-aligning the packets from multiple nodes, it temporarily buffers the extracted data, and once all the data with the same time-stamp arrive at the LPDC, they are aggregated to create a single data packet. This data packet is then sent to the SPDC. The SPDC parses and time-aligns data packets from LPDCs and these are given to applications running at SPDC.

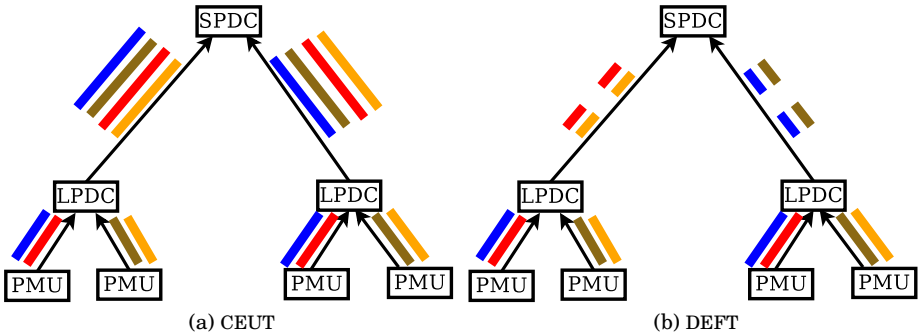
With PDCs at each level waiting for the arrival of PMU data packets to be aggregated and then sending to higher PDCs, a delay or drop of a PMU packet will cause the corresponding LPDC to wait until time-out. The overall delay from PMU to SPDC can ultimately affect the QoS of smart grid applications. Moreover, pushing all the data packets from downstream PMUs/PDCs to higher levels PDCs, irrespective of the data required for applications, also increases the bandwidth requirement.

To showcase the importance to meeting timeliness requirements of certain applications, [Zhu et al. 2010] simulate a real-world scenario with a fault of certain duration injected into a system is presented. As the fault may result in oscillations that grow with time, making the system unstable, it has to be cleared within a certain duration. The authors study the effect of different amount of delay in sending PMU data to a control application that is executed in a CEUT based fashion. It shows that beyond a certain time delay tolerated by the control application, the oscillations grow and the system becomes unstable. Hence, we need a data dissemination framework with efficient algorithms to disseminate required data to the applications while ensuring their timeliness requirements.

This motivates the primary focus on this paper: *How do we design algorithms to disseminate required data to the applications so as to ensure their timeliness requirements?*

1.4. Contributions of this paper

We present the ***D**istributed **E**xecution with **F**iltered data forwarding **T**echnique* (DEFT), and its derivatives, as an alternative to CEUT. In this technique, the data delivered to the SPDC depends on the applications running there. DEFT performs in-network processing of smart grid applications thereby minimizing application specific data transfers and delays due to centralized computations. DEFT is an application



(a) CEUT (b) DEFT

Fig. 2. CEUT vs. DEFT.

semantics-aware processing and communication solution in which the application is divided into coordinated sub-applications (sub-queries) executed in a distributed manner at LPDCs and the SPDC, and application relevant filtered and aggregated data gets forwarded from LPDCs to SPDC, as shown in Figure 2b.

Modelling Grid applications that use PMU data – as continuous threshold queries:- Grid applications over PMU data can be modelled as continuous threshold queries [Gupta and Ramamritham 2012] over multiple data streams. A threshold for example could specify the maximum allowed phase angle difference. We propose a method through which these queries are divided into a number of sub-queries. Given the frequency of a sinusoidal waveform, the phase angle is a linear function of time. This relationship is used to model the continuous sub-queries of ASM application that are disseminated at local nodes. The modelled subqueries are basically the local threshold conditions on the phase angles whose difference is monitored at higher PDC. These conditions are derived from the query threshold and only when these conditions are violated will data transfer to higher nodes occur. Threshold specified for the query is translated into thresholds for these sub-queries. Individual sub-queries involve data at a particular PDC, hence, can be executed at LPDCs. Results of individual sub-queries can be aggregated at SPDC to produce the result of the applications. Various methods exist for dividing a continuous aggregation query into a number of sub-queries. These sub-queries are executed at different nodes in the network. While deriving the sub-queries from the query, we have to make sure that query threshold violation occurs only if the threshold of at least one of its sub-queries is violated. Otherwise, we may get into a situation where the application threshold is violated (e.g., indicating a possible onset of angular instability) and the application site is not getting the required data. This is a *false negative* case in which the application improperly indicates that no threshold violation has occurred when in reality it may have. On the other hand, it is also possible that the SPDC, where the application query is executed, keeps on getting the data even when the threshold is not violated. Thus, it is non-trivial to have zero false negatives with minimum data transfer – so that the application gets all the data when needed and gets as little as possible when not needed.

In-network processing of smart-grid applications is categorized along two dimensions 1) Whether the sender node filters the data based on some condition, and 2) Whether different nodes executing the sub-queries share information about each other or not. ASM is an example where nodes executing a sub-query filter the data based on their local conditions. MCGG and SE applications are examples where the nodes executing sub-queries communicate, in this case, data from the common (boundary) buses.

Prioritizing Data Dissemination and Processing:- A smart grid will have a number of applications running simultaneously, each with its own QoS requirements. As smart-grid applications have varying timeliness requirements, we propose a priority based scheme in DEFT to assign more importance to time-critical applications compared to other less time-critical monitoring applications. If we run these applications using DEFT, unlike CEUT where all data gets sent to SPDC, we can prioritize data for applications having more stringent timeliness requirements (e.g., ASM) compared to ones having lenient QoS (e.g., SE). We propose a framework where a PDC processes data packets based on application priority by preempting processing of data needed by lower priority applications. Note that besides application specific filtered data packets, an SPDC may also require raw PMU data packets; these can be sent lazily. We propose multiple schemes to send raw data along with prioritized application specific data to SPDC.

Making all the data available at SPDC by combining Centralized and Decentralized approaches to improve generality and reliability:- Raw data from PMUs may be required at SPDC to analyse the impact of certain events such as load trips, generator trips etc., on the entire system. We propose the following two schemes that could be used in combination with priority based DEFT:

— DEFT+CEUT where, PMUs send raw packets to SPDC via LPDC. Here, DEFT offers prioritized and filtered data transfer and because CEUT is added with its data packets sent at the lowest priority, we get all the data at SPDC without interfering with the time-constrained application data.
— DEFT+CEUT-*direct* where PMUs send data directly to SPDC (i.e., bypassing LPDC), which processes them with low priority. Here, CEUT-*direct* provides an additional level of reliability by using different network paths for the raw data.

Simulation and empirical Evaluation using real grid characteristics:- We simulated a smart grid containing 662 PMUs that cover the Eastern and Western regions of India. Our algorithms are shown to incur considerably lower latency, even in cases where some of the packets get dropped by the network.

Outline: We start with a discussion of CEUT and DEFT realizations of ASM along with their performance evaluation in Section 2 and Section 3 respectively. We then discuss CEUT and DEFT realizations of MCGG and SE and their performance evaluation in Sections 4 and 5, respectively. In Section 6 we discuss DEFT with priority based approaches along with the performance of various algorithms when applications are executed concurrently in the grid. In this section, we also analyze when DEFT+CEUT is preferred over DEFT+CEUT-*direct*. Section 7 presents the related work followed by conclusion in Section 8.

2. ANGULAR STABILITY MONITORING

Power flow in transmission lines of an electrical network depends on the difference between the phase angles of the voltages at the two ends of the transmission lines. Let V_s and V_d be the bus voltage magnitudes at the two ends of a transmission line and let θ_t^s and θ_t^d be the corresponding phase angles. Then power flow in the line is proportional to $V_s V_d \sin(\theta_t^s - \theta_t^d)$. Quiescent angular separation should be low to ensure safe operations of the grid. After a disturbance, oscillations in angular difference occur but are usually damped and we get back to equilibrium. Angular instability occurs when the difference in phase angles between the ends of the transmission line increases uncontrollably [Hu and Venkatasubramanian 2007]. Then synchronization between the two areas is lost– resulting in separation of the two areas, loss of generation units in those areas, and ultimately a blackout occurs. Thus, if the fault can be identified and rectified within *fault clearing time*, angular instability can be prevented. If θ_t^s and θ_t^d

are the phase angles measured by PMUs located at buses s and d respectively and $\theta^{\mathbf{sd}}_{\mathbf{Th}}$ is the maximum phase angle difference allowed, an application correcting the fault does not need for any data as long as:

$$|\theta^s_t - \theta^d_t| \leq \theta^{\mathbf{sd}}_{\mathbf{Th}} \quad \forall\, t \tag{1}$$

ASM with CEUT comprises of monitoring (1) at SPDC by extracting the phase angle information from the aggregated data packets sent by the LPDCs. Each LPDC in turn receives data packet from the PMUs located at the two buses whose phase angles are being monitored at the SPDC.

2.1. Performance of ASM with CEUT

In this sub-section we report performance evaluation of the ASM in a variety of scenarios, using a combination of analysis as well as simulation. We first describe the experiment data as well as testbed setup. We then discuss different components of the application processing latency for LPDC and SPDC. We describe how our simulation results can be used to estimate processing latency for any application on the grid.

2.1.1. Experimental Data. We simulated the Eastern and Western regional grid of India with data aggregated at their respective LPDCs and SPDC. Unlike 3-layer architecture in [PGCIL. 2012] with PDCs located at state, regional and national level, our simulation environment is a 2-layer architecture with the PDCs at regional level (Eastern and Western) receiving data directly from PMUs and the national PDC (SPDC) receiving data from regional PDCs. The performance of all the algorithms was compared using the data generated from this simulation.

Table II shows the specifications of the simulated grid. The tie lines are the electric transmission lines connecting the two regions. Together the two regions have 662 buses with PMUs at each bus sending 50 packets per second. For simulation, 14 pairs of buses were selected that directly or indirectly connect the two regional grids. These 14 pairs were monitored at SPDC. Simulated data was generated by a transient stability program [Shubhanga and Papa Rao 2006] that enables modeling of the power system dynamics and faults. The simulation was carried out over a duration of 30 seconds. A three phase ground fault was injected at simulation time 0.1 second on a boundary bus that continued till 0.135 sec.

Table II. Simulation Configuration

Total buses	662
Total transmission lines	3006
Buses and transmission lines in Eastern region	363, 1532
Buses and transmission lines in Western region	299, 1469
Packet generation rate of each PMU	50
Angle difference pairs	14
Number of tie lines	5

2.1.2. Experimental Set-up. The hardware setup for the test bed consists of 4 Intel(R) Core(TM) i5-3230M CPU machines with 4 GB RAM and Ubuntu 12.04 installation, connected via a 100 Mbps LAN. The Eastern and Western grids are simulated on one machine each along with their respective LPDCs. SPDC is simulated on a third machine. A transient stability program written in MATLAB simulates different faults and generates PMU data for the grids at 50 samples per sec. Different processes for PMUs, LPDCs and SPDC are started on different machines. The PMU processes generate 50 packets per sec that are compliant with [IEEE Std C37.118.2 2011]. Details of PMU and PDC simulator are given in the Appendix.

The primary performance metrics used are average latency and bandwidth usage. The latency is composed of *network latency*– from PMUs to LPDC and LPDC to SPDC– and *processing latency* at LPDC and SPDC. To obtain PMU to SPDC end-to-end latency, all the machines running PMUs, LPDCs and SPDC were synchronized using precision time protocol. The observed network latency and observed processing latency at LPDC and SPDC were obtained by setting the checkpoints on the path of data packets from PMUs to LPDCs and LPDCs to SPDC. For different algorithms, it was observed that the processing latency dominates and varies much more compared to the network latency. Hence, the observed processing latency on the testbed was used for estimating the latency on the Indian Electric Grid.

2.1.3. Estimating latencies at LPDCs and SPDCs for CEUT. We conducted experiments using the testbed described in sub-section 2.1.2 with a PDC processing varying number of PMU data packets of different sizes. The observed processing latency at the PDC was used to create a model which can be used to estimate these latencies at PDCs of any arbitrary topology. The processing latency at a PDC consists of three components:

(1) time to parse incoming data packets (D^{parse}), and
(2) time to create an *aggregated* data packet using received data packets from PMUs/PDCs to send to higher level PDC (D^{agg})
(3) time for application specific processing (D^{app})

For different PDCs (LPDC/SPDC) one or more of these components can be zero, as we explain in this section. We start with giving expressions for the individual latency components first. Each PDC needs to parse the incoming packets from PMUs or lower level PDCs. Since parsing of different packets can be done independently, we use multiple threads to parse them in parallel. Specifically, if the PDC is receiving packets from N lower level PMUs/PDCs and it is implemented with C cores, then D^{parse} will be proportional to N/C. For lowest level LPDC, the number of nodes sending packets (PMUs) is very large; but for higher level PDCs (and SPDC), D^{parse} will be dominated by the lower level PDC sending the data from a large number of PMUs. In comparison, value of D^{agg} depends on the sum total of data size handled by any node.

For estimating D^{parse} and D^{agg}, we conducted two sets of experiments, E1 and E2. These experiments were done without any applications running on these nodes (i.e., $D^{app} = 0$).

(1) E1: This experiment was done for lowest level PDC receiving data from a large number of PMUs. As the number of PMUs increases, we expect larger values of D^{parse} and D^{agg}. The value of D^{parse} is expected to be linear function of the number of PMUs per core. Further, the larger the data size, more a thread is likely to spend in processing a packet and there will be more thread switching overhead. Hence we perform this experiment with different packet sizes. Table III gives values of D^{parse} based on these experiments. For an LPDC with C cores receiving data from N PMUs, one needs to multiply the number corresponding to data size (in KB) received from each PMU by N/C to get the value of D^{parse}. The value of D^{agg} can be obtained using E2, as explained below.
(2) E2: This experiment was done for intermediate PDCs and SPDCs. These PDCs work with much lower fan-in. Hence, rather than the number of threads, the processing latency is dominated by the processing of lower level PDC data packet comprising of the maximum number of PMUs. In this case, D^{parse} was found to be 59.7 micro secs for each KB of data packet-size per PMU; whereas corresponding value of D^{agg} was found to be 43.6 micro secs for each KB of data packet-size per PMU. Thus, if an intermediate PDC receives largest data packet consisting of data from N_{max} PMUs with each PMU K KB of data in its frame, then it's parsing and

aggregation times will be $59.7KN_{max}$ and $43.6KN_{max}$ respectively. Similarly, D^{agg} for lowest level LPDC is also estimated to be $43.6KN$, irrespective of the number of cores (as aggregation happens only once for a number of PMUs).

Table III. D^{parse} at LPDC for varying data sizes (K KB), per PMU per core in E1

K (KB)	0.1	0.5	1	1.5	2	2.5
D^{parse} (microsec)	124	289.6	360.48	407.2	477.6	564.4

To summarize, we have derived expressions for the estimated processing latency at LPDC and SPDC (or intermediate PDC) for CEUT. As explained earlier, one or more of D^{parse}, D^{agg}, and D^{app} can be zero for these cases. In the worst case, the data relevant for application running at SPDC may be present in the largest data packet sent by the LPDC; and the data from all the LPDCs may reach the SPDC at the same time. The worst case processing latency at LPDC and SPDC in CEUT is:

$$D^{lpdc}_{ceut} = D^{parse}(K, N) + D^{agg}(K, N) \quad (2)$$

$$D^{spdc}_{ceut} = D^{parse}(K, N_{max}) + D^{app} \quad (3)$$

where N_{max} is the maximum number of PMUs whose data is present in the lower level PDC data packets. There is no application specific processing at LPDC for CEUT, hence, D^{app} is not present in its expression. Similarly, an SPDC need not send aggregated data to any higher level PDC, hence, D^{agg} is 0 for it. Network latencies are ignored in these expressions.

2.1.4. PMU Data size for simulated Eastern and Western region grids and Validation of estimated processing latencies for PDCs. The data sizes of most of the PMUs in the experiment varied from 80 bytes to 120 bytes. The mean and variance of PMU data size were 96 and 22 respectively. For simplicity, we consider the PMU data size as 0.1 KB (K), number of PMUs $N = 363$ and $N_{max} = 363$. It implies from Table IV that the processing latency– observed and estimated– for LPDC and SPDC are very close to each other.

Table IV. Observed and Estimated Processing Latency at simulated LPDC (Western region PDC) and SPDC (National PDC) on a quad core processor ($C = 4$) and $K = 0.1$ KB

	Observed Processing Latency (millisec)	Estimated Processing Latency (millisec)
LPDC	13.3	(124*363/4)+43.6*0.1*363)/1000=12.8
SPDC	1.8	59.7*0.1*363/1000 = 2.1

2.1.5. Latency from PMU to SPDC for ASM with CEUT. The simulated end-to-end latency from PMU to SPDC for ASM with in CEUT was 18.63 millisec of which 15.1 millisec was due to D^{lpdc}_{ceut}+D^{spdc}_{ceut}. Also, the bandwidth required from LPDC to SPDC with CEUT was 6.3 Mbps.

2.2. Handling loss of packets in the Network with CEUT-*direct*

In addition to aggregated data from LPDCs, the applications running SPDC may also be provided with an additional level of reliability. Accordingly, we propose CEUT-*direct* where PMUs multicast the data packets to both the LPDCs and SPDC. Thus, with CEUT and CEUT-*direct*, the SPDC would receive redundant copies of the same data items.

The average latency for ASM with CEUT+CEUT-*direct* was 20.4 millisec and was higher than CEUT (18.63 millisec). This is because SPDC had to process the aggregated

data packets from the simulated LPDCs of Eastern and Western regions and the data packets from 662 PMUs of these regions. Replicated data also needs to be handled.

3. ASM WITH DEFT

As explained in section 2, we need to monitor multiple buses and send an alert if the difference in the phase angles is above the specified threshold as given in equation (1). For monitoring $\theta_t^s - \theta_t^d$, rather than sending all the phase values of θ_t^s and θ_t^d, we can assign thresholds to individual data values such that threshold θ_{Th}^{sd} is violated only if one or both of these thresholds are violated. But, unlike phase difference, the phase angles vary continuously with time. Hence, we need to model the phase angles and get the corresponding sub-queries with time-varying thresholds over individual phase angle values. Given the frequency of the electrical signal, the phase angle is a linear function of time:

$$\theta_t = f \times t \times 360^\circ + \theta_0 \ (mod\ 360^\circ)$$

where f is the frequency offset from nominal frequency in Hz, t is measurement time in seconds, and θ_0 is the phase angle at time $t = 0$. Using this data model, we have time varying thresholds over individual values of phase angles which bound the phase angles with an upper bound and a lower bound. Further, as per the data model, if the frequency is the same for both the angles whose difference is monitored, these two bounds should be linear with time, parallel to each other, and the distance between them should be equal to the phase angle difference threshold (θ_{Th}^{sd}) as in Figure 3.

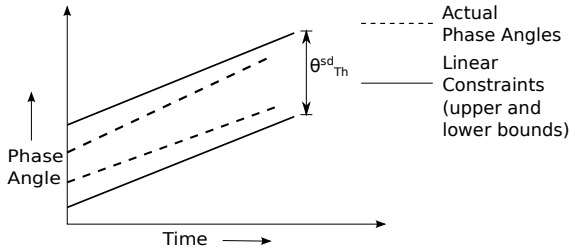


Fig. 3. Angle Difference Threshold Monitoring Technique

We can derive the individual values of thresholds using the method described below. If the frequencies for buses s and d, corresponding to θ^s and θ^d, are equal ($f_s = f = f_d$), the trajectory of individual values of the phase angles will be parallel to each other. Let us assume that initially θ^s is greater than θ^d. If β_s is the distance of the upper bound from linear equation estimating θ_t^s and β_d is the distance of the lower bound from linear equation estimating θ_t^d, we get,

$$\beta_s + \beta_d = \theta_{\mathbf{Th}}^{\mathbf{sd}} - |(\theta_0^s - \theta_0^d)| \tag{4}$$

where $\beta_s \geq 0$ and $\beta_d \geq 0$ and θ_0^s, θ_0^d are the phase angles of buses s and d respectively at time t=0. We can then set same or different values for β_s and β_d. Thus the linear constraints, for both θ^s and θ^d, would be,

$$\begin{aligned} \theta_0^d - \beta_d \leq (\theta^s - f \times t \times 360)\ (mod\ 360^\circ) \leq \theta_0^s + \beta_s \\ \theta_0^d - \beta_d \leq (\theta^d - f \times t \times 360)\ (mod\ 360^\circ) \leq \theta_0^s + \beta_s \end{aligned} \tag{5}$$

The values of β_s and β_d can be dynamically obtained using the method outlined in [Olston et al. 2003]. If the frequencies at the two buses are different, the upper bound and lower bound on the phase angles of the buses will be not be same. Hence, we

derive the equation of the linear bounding constraints based on the average frequency ($f_{avg} = (f^s + f^d)/2$), i.e., use f_{avg} in (5) while dividing the β values in proportion to the corresponding frequency values. Henceforth we refer to violation of type (1) as a *global violation* and violation of (5) as a *local violation*. We have two important results here: 1) Equation (5) does not result in false negatives, and 2) Equation (5) is a necessary, but not sufficient condition for global violation. The proof of these results is given in the Appendix.

3.1. Creating the Data Model

The messages transferred in DEFT for ASM have the following steps:

(1) The *Model Creation* at the SPDC involves calculating values for β_s, β_d, f_{avg} of (5) using the data from both the LPDCs.
(2) The SPDC then disseminates this model to both the LPDCs.
(3) If any LPDC detects a *local violation* given data at t, the corresponding data are sent to the SPDC.
(4) The SPDC then requests for data from other LPDC to check for the violation of (1).
(5) If SPDC does not detect any *global violation*, it updates the model using the new phase angle and frequency values and sends to both the LPDCs.
(6) If SPDC detects a *global violation*, the operator is alerted about the possibility of loss of synchronization in the two areas whose phase angle difference is being monitored.

3.2. Estimating latencies at LPDCs and SPDCs in DEFT

In DEFT, if the LPDC receives the data packets from all the N PMUs at the same time and the data packets relevant for the subquery of the application are then parsed, the processing latency at LPDC comprises of $D^{parse}(K, N)$. The worst case processing latency at LPDC and SPDC in DEFT is:

$$D^{lpdc}_{deft} = D^{parse}(K, N) + D^{app} \tag{6}$$

$$D^{spdc}_{deft} = D^{app} \tag{7}$$

where D^{app} is different at LPDC and SPDC for DEFT. For DEFT, any PDC need not collect and forward the raw data from lower level nodes, hence, there is no D^{agg} component in their corresponding expressions. But, processing required to send data to higher level nodes is specific to the application, hence, included in D^{app}.

3.3. Comparison of ASM with DEFT versus CEUT

In this sub-section we present latencies experienced by ASM application for CEUT and DEFT. First we give observed latencies for the simulation experiments discussed in sub-subsection 2.1.1. Then we give the estimates for the same for the whole Indian electric grid.

3.3.1. Latency for ASM from PMU to SPDC. In this section we give simulation results for the setup described in sub-subsection 2.1.2. We measure the end-to-end latency from simulated PMU data generation to application specific calculation at SPDC. Figure 4 shows the latency for CEUT and DEFT. The benefits of DEFT are evident from the figure. In DEFT, each of the 14 angle difference queries monitored at SPDC corresponded to a model comprising of local conditions that were monitored at both the LPDCs. As the PMU data was sent at 50 packets per second, the number of data points in the simulation duration for the 14 subqueries monitored at each LPDC were $50 * 30 * 14 = 21000$. It was observed that, during the *fault duration* of 35 millisec, only 12 global violations occured whereas very few local violations (1185) occured at other times. Thus, out of

42000 data points from both the LPDCs, the total number of violations reported were only 1197. Out of 12 global violations, 6 happened at time 0.1 sec and rest at 0.13 sec. The total number of global and local violations accounted for 3% of the entire simulation time whereas the rest 97% of the time all the PMU data was filtered at LPDCs. Figure 4 shows that for this 97% of the time, the latency of ASM with DEFT was just 2.24 millisec as the query execution stopped at LPDC. This latency (D^{lpdc}_{deft}) is mainly due to D^{parse} at LPDC. For the rest 3% of the time, the latency for ASM with DEFT was 8.28 millisec as SPDC pulls data from peer LPDCs for each local violation leading to 32 kb of data transfer for the entire period of simulation. Out of 8.28 millisec, D^{spdc}_{deft}+D^{lpdc}_{deft} accounted a major delay of 5.87 millisec while the rest was due a network delay. This corresponds to 87% and 55% savings in average latency over CEUT when query result was delivered at LPDC (97% of the time) and SPDC (3% of the time) respectively. Even though the query result was delivered to SPDC 3% of the time, the percentage drop is still considerable. Also, 3% of time data dissemination from LPDC to SPDC leads to 21 Kbps of bandwidth required with DEFT as opposed to 6.3 Mbps with CEUT.

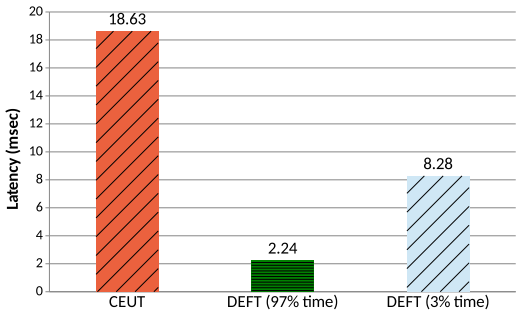


Fig. 4. Latency reduction for ASM when run alone.

3.3.2. Estimated processing latency for ASM with DEFT on the Indian Electric Grid. We consider the angle difference between the substations of Gujarat and Maharashtra states is monitored at Western region PDC. The sub-queries' conditions are then monitored at their respective state PDCs.

For DEFT algorithm, when the grid is steady, there are no local violations and the query executes in steps at LPDC. Hence, D^{spdc}_{deft}=0. At LPDC, the processing latency for the sub-query (D^{app}) is very small (a few micro secs) and is ignored ($D^{app} \approx 0$). Thus, the estimated processing latency for ASM comprises of parsing time at LPDC $D^{parse}(K, N)$.

In case of local violation at any LPDC, the SPDC pulls data from other LPDCs. Thus, the processing of application at SPDC may be delayed due to queuing of the data request at peer LPDC when it is busy parsing data packets sent by PMUs. As D^{app} for ASM at SPDC is negligible, the processing latency at SPDC is only due to parsing at peer LPDC ($D^{spdc}_{deft} = D^{parse}(K, N)$)

The number of transmission lines per substation is expected to grow in the near future. The PMUs located at these substations would be required to measure the current (current phasors) of each transmission line [Martin 2015]. Thus, the data packet size per PMU would increase with increase in transmission lines at the substation. Also,

IEEE C37.118.2-2011 standard [IEEE Std C37.118.2 2011] does not limit the number of phasors that a PMU can measure. For the Indian grid, there are substation PMUs having more than 2.5 KB data packet [PGCIL. 2012]. In this context, we evaluate the impact of large data sizes for ASM with CEUT and DEFT in the Indian grid. For PMU data of 2.5 KB, the estimated processing latency for ASM with and without local violations are 23.4 ($2*141.1*83/1000$) millisec and 11.7 ($141.1*83/1000$) millisec respectively. The query execution time for ASM at Western region PDC with CEUT is very small (a few micro secs) and hence, ignored. The estimated processing latency at PDCs of Maharashtra (D^{lpdc}_{ceut}) and Western region (D^{spdc}_{ceut}) is $(141.1 * 83 + 43.6 * 2.5 * 83)/1000 = 20.8$ millisec and $59.7 * 83 * 2.5/1000 = 12.4$ millisec respectively. Hence, the processing latency from Maharashtra to Western region PDC is 33.2 (20.8+12.4) millisec. Thus, ASM with DEFT has 30% and 65% reduction in latency for the query result delivered at SPDC and LPDC respectively.

In summary, ASM with DEFT achieved significant reductions in latency both in the simulation environment and in the Indian electric grid. Also, in the simulation, DEFT required lesser bandwidth compared to CEUT as data was sent from LPDC to SPDC for only 3% of entire simulation time.

4. MONITORING COHERENT GROUPS OF GENERATORS

Electric grids cover vast geographic areas. To study the stability of such systems, it is not practical to model the details of the entire grid. Rather, it is common practice to represent parts of the grid by equivalent models while preserving the general behavioral characteristics of the system. One step of creating these equivalent models is the identification of coherent groups of generators. When a remote disturbance occurs, coherent generators *swing* together and can, therefore, be represented by a single equivalent (generating) machine.

When a system is subjected to a sufficiently small magnitude of disturbance, it results in electromechanical oscillations (modes) in transmission lines. Engineering characteristics of a mode comprise of an oscillatory frequency, damping, and wave spectrum. The mode shape describes the amplitude and phasing of the oscillation throughout the system. Accurate estimates of the electromechanical modes are required for safe and reliable operation of the grid. Each mode is characterized by the coherent groups of generators swinging against another coherent groups of generators with an approximate phase difference of 180°. A group of generators is coherent if all the generators in the group have similar response characteristics to changes in the operating conditions of a power system. There are different types of modes (local machine-systems, intra-plant mode, local mode, and inter-area mode) with different frequency of oscillations for the coherent groups of generators [Shubhanga and Anantholla 2006]. The local machine-systems oscillations (0.7 to 2 Hz) involve one or more synchronous machines at a power station swinging together against a comparatively large power system or load center. The intra-plant mode oscillations (1.5 to 3 Hz) typically involve two or more synchronous machines at a power plant swing against each other. The local mode oscillations (0.8 to 1.8 Hz) generally involve nearby power plants in which coherent groups of machines within an area swing against each other. The inter-area mode oscillations (0.1 to 0.5 Hz) usually involve combinations of many synchronous machines in one area of a power system swinging against machines on another area of the system. The modal analysis described in [Shubhanga and Anantholla 2006] is used to determine the number of coherent groups in each mode, which coherent group swing against the other coherent group, which generators are part of each coherent group, etc. The modal analysis is a compute intensive algorithm that involves solving first order differential algebric equations using numerical or analytical methods such

as Jacobian based approaches. The implementation of this method in the simulation is beyond the scope of the paper. Hence, in the current simulation, we have performed offline modal analysis of the Eastern and Western region of Indian electric grid using the numerical methods that were implemented in MATLAB.

Of all the oscillation modes that were obtained from the modal analysis, some of the oscillations could be monitored at the local (LPDC) level. In this paper we only consider the low frequency inter-area oscillations which can be monitored only at SPDC. Thus, the monitoring was done in the simulation over the coherent groups of generators obtained from this analysis. In real life, the configuration of the coherent groups of generators for an inter-area oscillation mode changes very less frequently (typically less than 5) in 24 hours.

Each coherent group of generators is associated with a centre of inertia (COI) that needs to be computed using their rotor angles of the generators. The rotor angle of a generator is same as the phase angle measured by the PMU located at the generating substation. The COI for a coherent group of generators is the weighted average of rotor angles of the generators in the group. The weight associated with the angle of each generator is its inertia constant [Alsafih and Dunn 2010]. When a small load trips (a disturbance), there is an imbalance between cumulative generation and cumulative load and COI shifts. The system can still remain in the equilibrium if the difference between the COI of a coherent group i, δ_{COI_i} from the global centre of inertia, δ_{COI} remains within the threshold values. The monitoring query at higher level PDC is:

$$\epsilon_l \leq |\delta_{COI_i} - \delta_{COI}| \leq \epsilon_u \tag{8}$$

where ϵ_l, ϵ_u are lower and upper bounds respectively. The violation of condition (8) alerts the system operator about possible islanding. The modal analysis discussed above can be performed to determine if the configuration of coherent groups has changed due to violation of (8).

MCGG implemented using CEUT comprises of computing group COIs and global COI at SPDC to monitor (8). This requires extracting the phase angles of the generators involved in the coherent groups at SPDC from aggregated data received from LPDCs. We now discuss how to compute group COI and global COI from phase angles at SPDC. Suppose there are N groups in an inter-area swing mode with each group having $k_i, i \in [1, N]$ generators. The group i COI with k_i generators is given by [Alsafih and Dunn 2010]:

$$\delta_{COI_i} = \frac{\sum_{j=1}^{k_i} H_j * \delta_j}{G_i} \tag{9}$$

where H_j and δ_j are the inertia constant and rotor angle of generator j respectively and $G_i = \sum_{j=1}^{k_i} H_j$. G_i is called the inertia constant of group i. From the group COI computed for all the N groups, the global COI is computed as:

$$\delta_{COI} = \frac{\sum_{i=1}^{n} G_i * \delta_{COI_i}}{W} \tag{10}$$

where δ_{COI_i} is COI of group i and $W = \sum_{i=1}^{n} G_i$. The SPDC then monitors the deviation of each group COI from global COI as:

$$\epsilon_l \leq |\delta_{COI} - \delta_{COI_i}| \leq \epsilon_u, \forall i \in [1, N] \tag{11}$$

4.1. MCGG with DEFT

Unlike CEUT, where group COIs and global COI are computed at SPDC, in DEFT, each LPDC computes center of inertia of the group of generators from which it receives data. The group COI is then sent to SPDC to calculate global COI. In the context of Indian electric grid [PGCIL. 2012], inter-area oscillation modes across the states are monitored at regional PDCs and those across the regions are monitored at National PDC. The MCGG, specifically for Eastern and Western regions (that we have considered in our simulation) have modes with a coherent group of generators from one region swinging against the coherent group of generators in other region. If the computation of group COI for MCGG at one regional PDC does not depend on the PMU data of generators from other region, it can be computed directly at the regional PDC and sent to SPDC. But there can be low frequency oscillation modes wherein some coherent groups that are dominated by generators of a region may have coherency with few generators of other region. In two of the three modes that are monitored at National PDC (which is the SPDC) in our simulation, generators at the boundary of the Eastern region were found to swing with generators of the Western region and vice-versa. Hence, for the distributed query execution of MCGG, to calculate group COI at LPDC (regional PDC in the simulation), PMU data of a few generators of one region (say Eastern) that formed a coherent group with the generators of the other region (say Western) was sent to LPDC of that region (Western).

Sub-query at LPDC to compute group i COI with k_i generators is given by (9). The group COIs from the LPDC is then sent to SPDC where the global COI is computed by (10). The SPDC then executes the query given in (11).

Consider all LPDCs monitoring the m modes, with each mode having n generators. In CEUT, each LPDC sends Kmn KB to SPDC during a cycle (20 millisec), where K is the PMU data size in KB. In DEFT, $18 + 4m$ bytes are sent by each LPDC to SPDC, that includes, meta-info (16), statistics (2) and group COIs ($4m$). For K=0.1 KB, m=3 and n=20, DEFT sends only 0.5% of the application data compared to CEUT, which leads to improved processing latency.

4.2. Comparison of MCGG with DEFT versus CEUT

In this sub-section, we first give the observed processing latency for MCGG in the simulation environment. We then give the estimated processing latency for the Indian electric grid.

4.2.1. Simulation results for MCGG from PMU to SPDC. Figure 5 gives the simulation results for the setup described in Section 2.1.2. It shows MCGG with DEFT resulted in 41% reduction in average latency compared to CEUT. Out of 12 millisec PMU to SPDC end-to-end latency for MCGG, D^{lpdc}_{deft}=9.4 millisec and D^{spdc}_{deft}=1.25 millisec. The reduction in average latency compared to CEUT can be chiefly attributed to the distributed computation of application queries at LPDCs and low processing latency of the sub-query results at SPDC. The drop in bandwidth with DEFT was because, both the LPDCs sent just 30 bytes in each cycle (20 millisec) for the sub-query results executed at LPDCs unlike 15918 and 14782 bytes sent by the two respective LPDCs in CEUT.

4.2.2. Estimated processing latency for MCGG with DEFT on the Indian Electric Grid. In this section we provide an estimation of latency for MCGG with DEFT for the Indian electric grid with the topology given in [PGCIL. 2012]. For that evaluation, we consider the coherent groups of generators belonging to Maharashtra and Gujarat state being monitored at Western region PDC. With DEFT, the observed D^{app} at LPDC in the simulated Eastern and Western regional grid of India was negligible (in microsec) and is

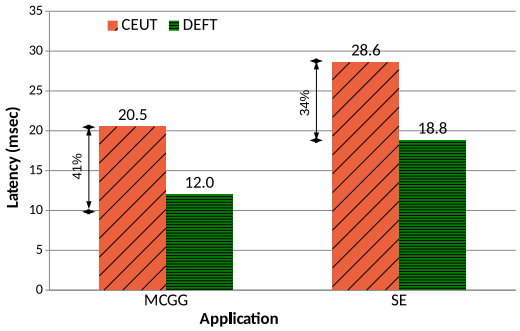


Fig. 5. Latency reduction for applications. The significant reduction in the average latency for MCGG and SE with DEFT approach is chiefly attributed to filtering and distributed computation of application queries at LPDCs and very low processing time of the sub-query results at SPDC.

ignored (D^{app}=0). However, the observed D^{app} at SPDC with DEFT was 1.25 millisec. The number of generators in the coherent groups of an inter area oscillation mode across the states are typically less than those across the regions. Hence, if we consider Western region PDC monitoring low-frequency modes, with generators of states Maharashtra and Gujarat, D^{app} at Western region PDC would be less than 1.25 millisec. Considering D^{app}=1.25 millisec, the estimated D^{lpdc}_{deft} and D^{spdc}_{deft} at Western region PDC for MCGG with PMU data size of 2.5 KB is, 11.7 millisec and 1.25 millisec respectively. Thus, the total processing latency of MCGG with DEFT (D^{lpdc}_{deft}+D^{spdc}_{deft}) is 12.95 millisec.

In comparison, CEUT has total processing latency of 35.4 millisec. It includes D^{lpdc}_{ceut} at Maharashtra PDC, $(141.1*83+43.6*2.5*83)/1000$=$20.8$ millisec and D^{spdc}_{ceut} at Western region PDC, $59.7 * 83 * 2.5/1000$+2.2=14.6 millisec respectively. The 2.2 millisec here is the observed D^{app} for simulated Eastern and Western regional grid of India with CEUT. Thus, MCGG with DEFT has 63% of reduction in the processing latency over CEUT.

MCGG with DEFT achieved significant reductions in latency both in the simulation environment and in a model for the Indian electric grid. Also, in the simulation, DEFT required lesser bandwidth compared to CEUT.

5. STATE ESTIMATION

The state of the system includes voltage magnitude and angle of buses. State estimation plays a vital role in various other applications. Though less time-critical compared to other grid applications, the operators rely heavily on results of state estimation to determine the health of the grid. Before the advent of PMUs, the measurements (voltage and current) in a SCADA system were available from remote terminal units. These measurements were not time synchronized which led to time skew errors. In addition to these errors, the measurements contain white noise. State estimation is a statistical method to obtain the state of the bus by minimizing the error in these measurements. If z is the measurement vector and x is current state vector then the model used in power system state estimation [Jones 2011] is:

$$z = h(x) + e \tag{12}$$

where h is the vector of nonlinear measurement functions between x and z, and e is the measurement white noise vector. The non-linearity here is due to time skew errors in

the measurements. In state estimation, i.e., estimating x from z, function h depends on admittance values in various buses. Performing state estimation involves solving non-convex optimization problems. But, typically, such models are iteratively linearized.

Due to the time synchronized measurements provided by the PMUs, the relation in (12) for an observable system is linear and can be solved in a single step. A system is observable if there are enough PMUs, such that the state estimator can estimate all the states of the grid. The method of estimating the states from the PMU measurements is known as linear state estimation (LSE). LSE reduces the computational complexity unlike nonlinear state estimation, which may take many iterations to converge towards a solution. The linear relationship between the measurements and the state is [Navalkar 2012]:

$$\mathbf{Z} = \mathrm{M}\mathbf{V} + \epsilon \tag{13}$$

where $\mathbf{Z}$ is the measurement vector, $\mathbf{M}$, the model matrix, which depends on admittance values, $\mathbf{V}$ is the vector of the bus voltage estimates and ϵ indicates the white noise in the measurements. The solution $\hat{\mathbf{V}}$ of the LSE problem, $\min \frac{1}{2}\|\mathbf{Z} - \mathrm{M}\hat{\mathbf{V}}\|_2$, is:

$$\hat{\mathbf{V}} = \mathrm{M}^{+}\mathbf{Z} = (\mathrm{M}^{\mathbf{H}}\mathrm{M})^{-\mathbf{1}}\mathrm{M}^{\mathbf{H}}\mathbf{Z} \tag{14}$$

where, $\mathbf{M}^{+}$ is also known as Penrose Moore inverse of $\mathbf{M}$. Operator $\mathbf{H}$ is a Hermitian operator. It should be noted that $\mathbf{M}$ is a large and sparse complex matrix. Thus, computations in the state estimation application involve large sparse matrices $\mathbf{M}$ and their complexity increases with the size of the grid.

As a smart grid consists of many interconnected and geographically distributed regions (or areas), each region sending data to its LPDC can perform LSE of the region. For example, in the context of Indian electric grid, the PDC for Maharashtra grid can perform LSE using the phasor information from the data packets of all the substation PMUs in the state. However, globally optimal control action requires knowing the global state of the system. SE with CEUT involves performing LSE at SPDC using the phasor information extracted from the aggregated PMU data packets sent by LPDCs.

5.1. SE with DEFT

Instead of performing state estimation at SPDC using all the phasors data present in the data packets sent by downstream LPDCs, distributed algorithm proposed in [A. Tajer and Poor 2012] use only the limited information sent by the LPDCs. The states estimated by the distributed algorithm are close to those estimated by LSE at SPDC. To find the state estimate of each bus in the system, these algorithms perform the weighted average of the estimates obtained from the state estimators running at each downstream LPDC. The weight associated with each estimate is also shared along with the estimate by the region LPDCs to the central node so as to perform the weighted average of the states. Besides reduction in communication traffic and latency at SPDC, another benefit of distributed SE is the privacy of the smart grid [A. Tajer and Poor 2012]. Communication channels between LPDCs and SPDC are vulnerable to cyber-attacks and hence sending only estimates of the network state will hide the data of the individual consumer and generator from out-of-network entities.

We now discuss the implementation of the distributed algorithm for state estimation with respect to DEFT that comprises of the dividing the state estimation problem into subqueries at LPDC, execution of those subqueries at LPDCs, sending the subquery results to SPDC, and the computations performed at SPDC. Consider two regions connected by a transmission line (also known as tie line) such that LPDC of each region perform LSE given by (13). The buses from both the regions that are connected by the tie line are known as boundary buses. The state specific data sent from LPDC of each region to SPDC comprises of different components such as non-boundary bus estimates

computed using LSE, estimates of the state of the boundary buses of both the regions, current of the tie line connecting the boundary bus and the voltage at the boundary bus in the region, and the weights corresponding to estimates and measurements. Hence, for the network having two regions, the data packet from each LPDC contains the estimates of both the boundary buses that are connected by the tie line. Unlike in [A. Tajer and Poor 2012] that obtains the global state estimate of each bus by performing weighted average of the estimates, the SPDC in our simulation uses weighted least squares method to obtain the global estimates of the boundary buses. The benefit of this method over weighted average is that it tries to minimize the error between the measurements and the estimations of these measurements corresponding to boundary buses. Further, as LSE at each region leads to two estimates per boundary bus that are available at SPDC, a weighted least squares over these redundant estimates would give a unique estimate per boundary bus. This would improve the accuracy of the global state of the boundary buses. For the non-boundary buses, the estimates obtained from LSE at region are unique and are retained as a global state of those buses. The solution of the weighted least square problem viz., min $\frac{1}{2}\|\mathbf{W_s^{-\frac{1}{2}}}(\mathbf{Z_s} - \mathbf{M_s}\hat{\mathbf{V}}_\mathbf{s})\|_2$ at SPDC is [Navalkar 2012]:

$$\hat{\mathbf{V}}_\mathbf{s} = (\mathbf{M_s^H W_s M_s})^{-1}\mathbf{M_s^H W_s Z_s} \tag{15}$$

where, $\hat{\mathbf{V}}_\mathbf{s}$ are the boundary bus voltage estimates, $\mathbf{M_s}$ is a model matrix, and $\mathbf{W_s}$ is weight matrix. More details of deriving $\mathbf{W_s}$ can be found in [Navalkar 2012].

As the amount of data transmitted from each LPDC to SPDC in SE with DEFT depends on the boundary buses of each region and the number of electric transmission lines connecting these regions, it varies for different grid topologies. To estimate the bandwidth used by the application with DEFT and CEUT in the two regions– r_1 and r_2 – with both regions having n buses, b of which are the boundary buses, connected by b transmission lines. Also, let both regions have l transmission lines in total. The Meta info per PMU packet comprises of 16 bytes and the size of each phasor is 8 bytes [IEEE Std C37.118.2 2011]. Neglecting the Meta info from PMU data, each LPDC in CEUT would send $8(n + l)$ bytes of data to SPDC, where 8 is the size of each phasor in bytes.

In DEFT, LSE of each region would estimate the states of the boundary buses of both the regions. This would lead to $2b$ voltage estimates at each LPDC. Each LPDC also receives PMU data from PMUs located at b boundary buses. Thus, considering phasor data of 8 bytes, the measured and estimated values lead to a sum total of $24b$ bytes of data. The tie line currents measurements (b) at each LPDC lead to $8b$ bytes of data. Hence the data sent to SPDC comprising of measurements (b voltages and b currents) and estimates ($2b$ voltages) is $32b$ bytes. Further, the weights associated with measurements and estimates lead to $16b$ bytes, where each weight value requires 4 bytes. Each regional LPDC also sends the voltage estimates of the non-boundary buses ($n - b$) to SPDC. Thus, the total data sent by each LPDC is $40b + 8n$ bytes. In comparison to CEUT, DEFT sends $\frac{40b+8n}{8(n+l)}$ fraction data to SPDC. For simplicity, let the two regions, r_1 and r_2 have the same configuration as Eastern region of the Indian electric grid given in Table II. Accordingly, n = 363, b = 5 and l = 1532. In this case, DEFT sends less than 20% data compared to CEUT leading to reduction in latency for SE.

5.2. Comparison of SE with DEFT versus CEUT

In this sub-section, we first present the simulation environment for grid state estimation application on the setup described in Section 2.1.2 and then give the observed processing latency for the application. We further give the estimations of the processing latency for the Indian electric grid.

5.2.1. Simulation environment. For the SE with DEFT, each LPDC (Eastern and Western region) after performing LSE, sends the phasor estimates, measurements, and weights of the boundary buses to SPDC. The SPDC then performs weighted least squares from the measurements and estimates received from both the LPDCs.

5.2.2. Latency for SE from PMU to SPDC. The simulation results presented in Figure 5 show that SE with DEFT had 34% reduction in average latency and had 81% drop in bandwidth usage compared to CEUT. Out of 18.8 millisec PMU to SPDC end-to-end latency for SE, $D^{lpdc}_{deft} + D^{spdc}_{deft}$ from equations (6) and (7) contributed the highest i.e., 16.4 millisec. The significant decrease in average latency of SE with DEFT is mainly attributed to the fact that the size of the model matrix at LPDCs was approximately half that of the one at SPDC in CEUT, and very low processing latency due to calculation of weighted least squares on a smaller size model matrix at SPDC. Similarly, in DEFT, two LPDCs sent just 2256 and 3068 bytes respectively compared to 15918 and 14782 bytes with CEUT, resulting in reduced bandwidth.

5.2.3. Estimated processing latency for SE on the Indian Electric Grid. We assume that the four regional PDCs (Northern, Eastern, Western and North Eastern) perform LSE and National level SPDC performs weighted least squares. This would require the state PDCs (including Maharashtra PDC) to parse and send the aggregated PMUs data to regional PDCs. Accordingly, the D^{lpdc}_{ceut} at Maharashtra PDC with PMU data of 2.5 KB is 20.8 millisec.

D^{parse} at Western region PDC (intermediate PDC) from equation (6) is 12.3 millisec. As given in [Navalkar 2012], the LSE performed on the above mentioned four regional grids of India together took 23 millisec. With the assumption that each regional PDC would take one-fourth of the processing required for LSE at National PDC, performing LSE at the regional PDC would take 5.75 millisec (D^{app}). Thus, D^{lpdc}_{deft}=$12.3 + 5.75 = 18.05$ millisec

Since the size of the matrix at SPDC is much smaller compared to that required for LSE at LPDC, 5.75 millisec would be a strict upper bound on the execution time for weighted least squares. Considering weighted least squares processing latency (D^{spdc}_{deft}=D^{app}) as 5.75 millisec, the estimated processing latency for SE with DEFT in the Indian electric grid is $20.8 + 18.3 + 5.75 = 44.85$ millisec – well within the 100 millisec latency requirement of SE. In comparison, the estimated processing latency for CEUT is 180.8 millisec. It includes the processing latency at Maharashtra PDC (D^{lpdc}_{ceut}), Western region PDC (D^{lpdc}_{ceut}), National PDC (D^{spdc}_{ceut}) of 20.8 millisec, 65 millisec and 95 millisec respectively with D^{app} of 23 millisec at National PDC to perform LSE.

SE with DEFT achieved a considerable reduction in latency in the simulation environment. However, reduction in processing latency with DEFT on Indian electric grid was quite significant. Further, in the simulation, DEFT required lesser bandwidth compared to CEUT.

6. EXTENDING DEFT

In this section, we present an extension to DEFT that takes cognizance of applications priority when multiple concurrent applications are executed on the setup described in Section 2.1.2. We also present two extensions to this priority based DEFT that sends all the raw PMU data to SPDC.

6.1. Handling concurrent applications

The priority of executing the applications at a PDC depends on the time-critical nature of the application. For example, as given in Table I, ASM has higher execution

priority over MCGG, as in ASM, control action needs to be taken quickly to prevent propagation of instability throughout the system. SE has the lowest priority. Though DEFT exploits the application semantics for data dissemination, LPDCs and SPDCs treat all the application packets equally. This means, the processing of data packets relevant to low priority applications would not be preempted even when data packet relevant to a high priority application arrives at an LPDC. In DEFT that considers applications priority, PDCs preempt processing of data packets relevant to low priority applications when packets for high priority applications arrive, thus preserving the order of execution of applications based on their time criticality. The implementation details for priority based DEFT along with the priority allocation in Linux platform are given in the Appendix.

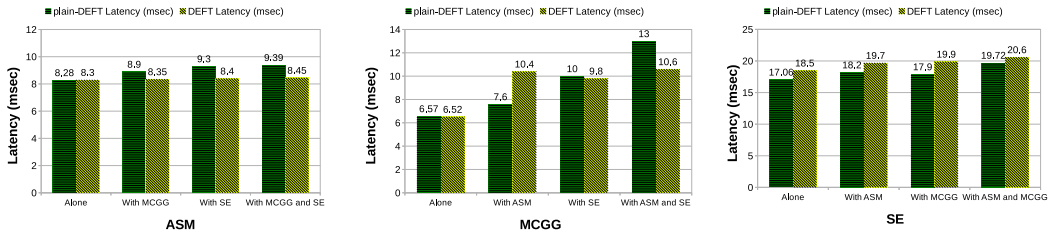
Henceforth, we refer DEFT that does not consider application priorities under concurrent execution as plain-DEFT, while the one that considers the application priorities as DEFT.

6.1.1. Latency from PMU to SPDC under concurrent execution of applications— with and without priority cognizance. Figure 6a shows the average latency with *plain*-DEFT and DEFT for ASM when run singly and concurrently with other applications. We see that unlike *plain*-DEFT, with DEFT, the average latency remained almost the same when ASM was run singly and concurrently with other applications. As DEFT executed ASM with the highest priority, having other applications executing concurrently does not have much impact. The marginal increase in latency of ASM with DEFT from 8.35 millisec to 8.45 millisec was due to thread switching time during preemption of low priority threads.

Figure 6b shows the average latency of MCGG with DEFT was higher when run with ASM compared to *plain*-DEFT. This is because, in DEFT, threads processing MCGG relevant data got preempted by threads that process ASM relevant data. The latency of MCGG with DEFT slightly reduced when run with SE unlike when run with ASM, as MCGG has higher execution priority compared to SE.

Figure 6c shows that the average latency for SE with DEFT was more than *plain*-DEFT as SE has the lowest priority among all the applications, in DEFT, the threads processing of packets relevant to SE would always be preempted by other threads.

From Figure 6, we summarize that as DEFT takes cognizance of the priority of the applications, it is more suitable for time-critical applications.



(a) ASM: singly and concurrently (b) MCGG: singly and concurrently (c) SE: singly and concurrently

Fig. 6. Average latency: running singly and concurrently with *plain*-DEFT and DEFT.

6.2. Making all data to available at SPDC

Our main work so far has been confined towards application semantics-aware PMU data dissemination. The SPDC, however, may require all PMU data from downstream

LPDCs/PMUs to perform post-analysis of certain events such as load trips, generator trips etc. This system wide data from all PMUs can reveal insights. Further sending over raw data from PMUs provides an additional level of reliability to loss of packets in the network. Hence, we extend our existing DEFT approaches to *send all PMU data to SPDC*. We propose two methods, namely, 1) DEFT+CEUT and 2) DEFT+CEUT-*direct*. In the first method, just like application specific data, all the PMU data is time-aligned at LPDC, a combined packet is created, and disseminated to SPDC. This raw PMU data is typically sent at lowest priority. CEUT-*direct* has already been discussed in sub-subsection 2.2. In this method, PMUs multicast data to both LPDC and SPDC. The first method requires an additional combined packet creation operation at LPDC whereas the second method requires additional parsing of PMU data packets at SPDC.

6.2.1. Latency from PMU to SPDC for applications with all the data dissemination techniques. Figure 7 shows the latency for individual applications with all the data dissemination techniques on the setup described in Section 2.1.2. *plain*-DEFT and DEFT have already been compared earlier. With DEFT+CEUT, the latency for all applications was almost the same as the latency with DEFT. This implies that sending all PMU data to SPDC with the lowest priority does not significantly affect the latency of individual applications. However, with DEFT+CEUT-*direct*, the latency of individual applications slightly increased in comparison to DEFT+CEUT. This is because, before the application specific data from LPDCs could reach SPDC, PMU data packets from PMUs had reached SPDC. As SPDC was busy processing a total of 662 data packets from PMUs of Eastern and Western electric grid, even though the packets with partially computed results had reached the SPDC, their processing was delayed due to the preempting threads processing PMU data packets.

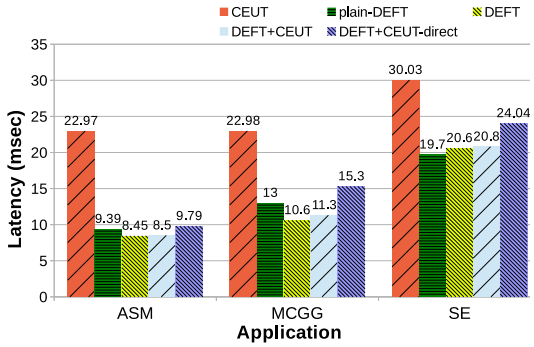


Fig. 7. A comparison of latency for all the data dissemination techniques.

6.2.2. Latency from PMU to SPDC during loss of filtered data packets for ASM from LPDC to SPDC with DEFT+CEUT. We now evaluate the performance of DEFT+CEUT when the filtered data packets sent from a LPDC for ASM application is lost. Under these circumstances, the applications (at SPDC) can continue to execute the angle difference query using the angular information extracted from the raw PMU data that is sent by both the LPDCs at lowest priority. We also consider here MCGG and SE applications concurrently running with the ASM application. Without any loss of the application specific data packets, the PMU to SPDC end-to-end observed latency for ASM as shown in Figure 7 was 8.5 millisec.

When filtered packet for ASM from any one LPDC is lost, SPDC needs to wait for the arrival of the aggregated data packet sent by each LPDC with CEUT. Before sending the aggregated data packet from an LPDC, the subqueries of MCGG and SE are executed with DEFT. Hence, processing latency at LPDC comprised of observed D^{lpdc}_{ceut} (16.3 millisec) to parse and aggregate data packets from all PMUs including thread pre-emption overheads and observed D^{app} for the subqueries of ASM, MCGG and SE. The sum total of D^{app} for ASM, MCGG and SE was 3.4 millisec.

At SPDC, as the execution of ASM whose filtered data packet was lost can be done by extracting phase angle information from the aggregated data packet, the processing latency was dominated by D^{parse} (2.0 millisec). Thus, PMU-SPDC end-to-end latency for ASM during the loss of filtered ASM packet was 24.3 millisec.

This average latency with DEFT+CEUT is slightly greater than that of ASM with CEUT (22.97 millisec) as shown in Figure 7. However, we note here that as the aggregated raw PMU data is also available at SPDC in DEFT+CEUT, it is suitable under loss of in-network filtered data packets of any application sent from LPDCs to SPDC.

6.3. Summary of the Performance Studies

Our performance studies reveal the following

— DEFT achieved a significant reduction in latency and bandwidth for all the applications compared to CEUT.
— DEFT techniques (with CEUT and CEUT-*direct*) were more reliable than *plain*-DEFT with respect to latency guarantees of critical application and ensured that applications were executed in the decreasing order of their priorities. Furthermore, sending all the raw PMU data to SPDC, along with application specific filtered data, did not impact the latencies of the grid applications.
— DEFT+CEUT is the algorithm to be recommended for a typical smart-grid. If the network links are lossy one may consider using DEFT+CEUT-*direct*.

7. RELATED WORK

In-network query processing, data filtering, and distributed execution with aggregation have been studied extensively in the literature. Authors of [Deshpande et al. 2004] use probabilistic models of sensor data for minimizing data acquisition for given user query. Model-based estimation of data values to filter data values that are not *very* different from the estimated values is presented in [Edara et al. 2008]. In comparison, our technique uses *application semantics* to perform in-network query processing for smart grid applications. Thus, our technique can be used for exact monitoring and not just probabilistic monitoring.

Data traffic reduction using filtering at the data sources is proposed in [Gupta et al. 2010; Olston et al. 2003]. These filtering conditions are derived from sufficient conditions for providing *no-false-negative* guarantees for threshold queries. Authors of [Gupta and Ramamritham 2012] use data aggregation across distributed sources. These techniques perform in-network query processing on sensor values. In comparison, we use data modeling and in-network processing to monitor the grid efficiently.

Various techniques for QoS guaranteed delivery of data have been studied in the literature. Gridstat [Bakken et al. 2011] discusses the implementation of a data dissemination network with delivery guarantees in a smart grid. Our approach optimizes data dissemination and is orthogonal to Gridstat. Thus, our solution can be used with Gridstat or any other data delivery mechanism. We also retain the existing client-server framework for PMU-PDC communication that is currently adopted by the power system domain – unlike Gridstat, which is a new middleware paradigm. The stream computing based approach for optimizing the data transmission in [Hazra et al. 2011]

makes way for an easy development of applications and allows for graceful degradation during times of overload. Our work has some similarity with [Hazra et al. 2011; Arya et al. 2011] in the use of partial computations to get the query results. However, the techniques discussed in [Hazra et al. 2011; Arya et al. 2011] do not take into consideration application semantics. We also show the implications of our techniques on the topology of the Indian electric grid.

8. CONCLUSION

High data rate and strict latency requirements of critical real-time monitoring applications in the smart grid prompted us to design in-network query processing techniques that allow for flexible bandwidth sharing among smart grid applications. We achieved significant latency reductions for critical applications. The proposed distributed approach DEFT, is semantics-aware – high data filtering when the grid is stable, and the application receiving almost all the data when the grid may move into instability. We designed novel approaches of using estimation functions for PMU data and derived efficient in-network monitoring sub-queries based on them. The techniques also take cognizance of priority of applications while concurrent applications are executing at upper-level nodes such that latency of time critical applications does not suffer. Also, sending raw data from PMUs even at the lowest priority, enhances the reliability of the techniques to loss of packets in the network.

We thus provided a general framework to devise efficient in-network query processing techniques for various applications in the smart grid. In this approach, the application queries are converted into sub-queries to be executed at different PDCs. This framework included priority based application specific data dissemination in the smart grid. Using this, one can meet the QoS requirements of various applications while ensuring that reliability of the grid does not suffer. Using performance results we showed that our algorithms lead to better bandwidth and latency of grid applications while having robust to packet losses in the network.

However, the distributed in-network filtering techniques have some disadvantages compared to a centralized technique. The distributed techniques introduce the complexity of managing multiple nodes (e.g., SPDC needs to continuously monitor various query parameters and characteristics to generate sub-queries), along with their resources, failure handling, reliability, etc.

ACKNOWLEDGMENTS

We thank DeitY, Govt. of India, and TCS for their financial support; PGCIL for sharing details of the Indian grid; Swadesh Jain and Mayur Kale for helping in our work.

REFERENCES

S. Kar A. Tajer and H.V. Poor. 2012. *Distributed state estimation: a learning based framework*. Cambridge University Press.

Mark Adamiak, Bogdan Kasztenny, and William Premerlani. 2005. Synchrophasors: definition, measurement, and application. *Proceedings of the 59th Annual Georgia Tech Protective Relaying, Atlanta, GA* (2005), 27–29.

H.A. Alsafih and R.W. Dunn. 2010. Identification of Critical Areas for Potential Wide-Area based Control in Complex Power Systems based on Coherent Clusters. *Universities Power Engineering Conference (UPEC)* (September 2010).

Andrew Armenia and Joe H Chow. 2010. A flexible phasor data concentrator design leveraging existing software technologies. *Smart Grid, IEEE Transactions on* 1, 1 (2010), 73–81.

V. Arya, J. Hazra, P. Kodeswaran, D. Seetharam, N. Banerjee, and S. Kalyanaraman. 2011. CPS-Net: In-network aggregation for synchrophasor applications. In *2011 Third International Conference on Communication Systems and Networks (COMSNETS 2011)*. 1–8. DOI:http://dx.doi.org/10.1109/COMSNETS.2011.5716510

D.E. Bakken, A. Bose, C.H. Hauser, D.E. Whitehead, and G.C. Zweigle. 2011. Smart Generation and Transmission With Coherent, Real-Time Data. *Proc. IEEE* 99, 6 (june 2011), 928 –951. DOI:http://dx.doi.org/10.1109/JPROC.2011.2116110

Moustafa Chenine and Lars Nordström. 2011. Modeling and simulation of wide-area communication for centralized PMU-based applications. *Power Delivery, IEEE Transactions on* 26, 3 (2011), 1372–1380.

Amol Deshpande, Carlos Guestrin, Samuel R Madden, Joseph M Hellerstein, and Wei Hong. 2004. Model-driven data acquisition in sensor networks. In *Proceedings of the Thirtieth international conference on Very large data bases-Volume 30*. VLDB Endowment, 588–599.

P. Edara, A. Limaye, and K. Ramamritham. 2008. Asynchronous in-network prediction: Efficient aggregation in sensor networks. *ACM Trans. on Sensor Networks (TOSN)* 4, 4 (2008), 25.

R. Gupta and K. Ramamritham. 2012. Query planning for continuous aggregation queries over a network of data aggregators. *Knowledge and Data Engineering, IEEE Trans. on* 24, 6 (2012), 1065–1079.

R. Gupta, K. Ramamritham, and M. Mohania. 2010. Ratio threshold queries over distributed data sources. In *Data Engineering (ICDE), 2010 IEEE 26th International Conference on*. IEEE, 581–584.

J. Hazra, K. Das, D.P. Seetharam, and A. Singhee. 2011. Stream computing based synchrophasor application for power grids. In *Proceedings of the first international workshop on High performance computing, networking and analytics for the power grid*. ACM, 43–50.

Dongchen Hu and Vaithianathan Venkatasubramanian. 2007. New wide-area algorithms for detection and mitigation of angle instability using synchrophasors. In *Power Engineering Society General Meeting, 2007. IEEE*. IEEE, 1–8.

2011. IEEE Standard for Synchrophasor Data Transfer for Power Systems. *IEEE Std C37.118.2-2011 (Revision of IEEE Std C37.118-2005)* (2011), 1–53. DOI:http://dx.doi.org/10.1109/IEEESTD.2011.6111222

2013. IEEE Guide for Phasor Data Concentrator Requirements for Power System Protection, Control, andMonitoring. *IEEE Std C37.244-2013* (May 2013), 1–65. DOI:http://dx.doi.org/10.1109/IEEESTD.2013.6514039

Kevin David Jones. 2011. *Three-Phase Linear State Estimation with Phasor Measurements*. Master's thesis. Blacksburg, VA.

Kedar Khandeparkar and Nitesh Pandit. May, 2012. Design and Implementation of IEEE C37.118 based Phasor Data Concentrator and PMU simulator for Wide Area Measurement System. *Technical Report, Dept. of Electrical Engg., IIT, Bombay.* (May, 2012).

Kedar Khandeparkar, Krithi Ramamritham, Rajeev Gupta, Anil Kulkarni, Gopal Gajjar, and Shreevardhan Soman. 2015. Timely Query Processing in Smart Electric Grids: Algorithms and Performance. In *Proceedings of the 2015 ACM Sixth International Conference on Future Energy Systems, e-Energy 2015, Bangalore, India, July 14-17, 2015*. 161–170. DOI:http://dx.doi.org/10.1145/2768510.2768535

Ken Martin. 2015. private communication. (Jan 2015).

Juri Lelli Michael Kerrisk, Peter Zijlstra. 2014. Linux Programmer's Manual. (Oct. 2014). http://man7.org/linux/man-pages/man7/sched.7.html.

Prashant V. Navalkar. 2012. *Phasor Measurement Unit Based Linear State Estimate Or-Diagnostics And Application to Secure Remote Backup Protection of Transmission Lines*. Ph.D. Dissertation. Dept. of Electrical Engg., IIT, Bombay, India.

C. Olston, J. Jiang, and J. Widom. 2003. Adaptive filters for continuous queries over distributed data streams. In *Proceedings of the 2003 ACM SIGMOD international conf. on Management of data*. ACM, 563–574.

PGCIL. 2012. Unified Real Time Dynamic State Measurement (URTDSM). (Feb. 2012).

AG Phadke. 1993. Synchronized phasor measurements in power systems. *Computer Applications in Power, IEEE* 6, 2 (1993), 10–15.

A. G. Phadke and J. S. Thorp. 2010. Communication needs for Wide Area Measurement applications. In *Critical Infrastructure (CRIS), 2010 5th International Conference on*. 1–7. DOI:http://dx.doi.org/10.1109/CRIS.2010.5617484

K.N. Shubhanga and Yadi Anantholla. 2006. Manual for a Multi-machine Small-signal Stability Programme. *Dept. of Electrical Engg. NITK, Surathkal* (2006).

K.N. Shubhanga and B.V. Papa Rao. 2006. Manual for a Transient Stability Programme (PART-1: Symmetrical Fault Analysis). *Dept. of Electrical Engg. NITK, Surathkal* (2006).

Luigi Vanfretti and Joe H Chow. 2011. Synchrophasor Data Applications for Wide-Area Systems. In *17th Power Systems Computation Conference (PSCC), Stockholm, Sweden, 2011*.

Kun Zhu, Ji Song, Moustafa Chenine, and Lars Nordstrom. 2010. Analysis of phasor data latency in wide area monitoring and control systems. In *Communications Workshops (ICC), 2010 IEEE International Conference on*. IEEE, 1–5.

Online Appendix to: QoS-driven Data Processing Algorithms for Smart Electric Grids

Kedar Khandeparkar, Indian Institute of Technology, Bombay
Krithi Ramamritham, Indian Institute of Technology, Bombay
Rajeev Gupta, Iqlect, India

A.1. Proof of Correctness of ASM with DEFT

To prove: 1) Equation (5) does not result in false negatives, and 2) Equation (5) is a necessary, but not sufficient condition for global violation.
Proof-1: As shown in Figure 8, at time t_0, points A_0 and D_0 are upper and lower threshold values for θ^s and θ^d respectively and points B_0 and C_0 are the actual angle values. With no global violation, points B_0 and C_0 shall remain within points A_0 and D_0 (collinearity property) i.e.,

$$|A_0D_0| = |A_0B_0| + |B_0C_0| + |C_0D_0| \tag{16}$$

At time t_1, $|\theta^s_t - \theta^d_t| > \theta^{sd}_{Th}$, which implies

$$|A_1D_1| - |B_1C_1| < 0 \tag{17}$$

Re-arranging Equation (16) with angle values at time t_1

$$|A_1D_1| - |B_1C_1| = |A_1B_1| + |C_1D_1| \tag{18}$$

From Equations' (17) and (18) we get

$$|A_1B_1| + |C_1D_1| < 0$$

This is a contradiction. Hence, we prove that violation of global threshold implies violation of local threshold conditions and thus, there are no false negatives.
Proof-2: At time t_2 in Figure 8, $B_2 > A_2$, $C_2 > D_2$ and $|B_2C_2| < \theta^{\mathbf{sd}}_{\mathbf{Th}}$. We can see the violation of local conditions for θ^s even when the condition $|\theta^s_{t_1} - \theta^d_{t_1}| \leq \theta^{\mathbf{sd}}_{\mathbf{Th}}$ holds true. This implies, a local violation is a necessary but not sufficient condition for global violation.

A.2. Details of the PMU and PDC Simulator

PMUs and PDCs were simulated using the open source grid simulator, iPDC. This simulator implements IEEE C37.118.2-2011 and IEEE C37.244 Standards for PMU and PDC respectively [Khandeparkar and Pandit 2012]. IEEE C37.118.2-2011 [IEEE Std C37.118.2 2011] standard defines four kinds of packets (frames): *command*, *data*, *configuration*, and *header*. More details of packet formats can be found in [IEEE Std C37.118.2 2011]. The PDC processes these packets from PMUs and other PDCs that are compliant to IEEE C37.244 Standards.

A.3. Data sizes for different messages of DEFT in ASM

Model Creation requires a total of 90 bytes of data transfer out of which 32 bytes are from LPDC to SPDC and 58 bytes are from SPDC to LPDC. A local violation at any time t from one LPDC leads to a message transfer of 32 bytes from LPDC to SPDC. The SPDC to other LPDC request and response for same time-stamp data is 46 bytes and 32 bytes respectively. If SPDC detects no global violation, it creates a new model

© YYYY ACM. 2378-962X/YYYY/01-ARTA $15.00
DOI: http://dx.doi.org/10.1145/0000000.0000000

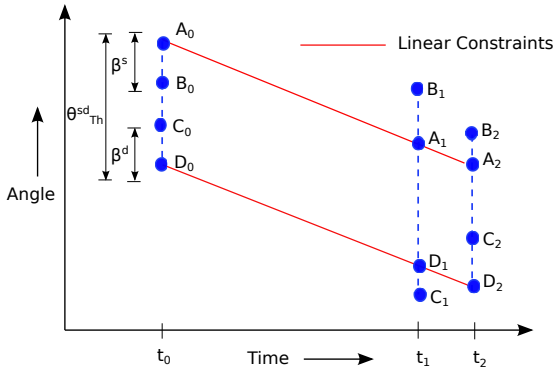


Fig. 8. Violation of Threshold Conditions

with, possibly, modified local thresholds and disseminates to LPDCs. Thus, a local violation costs 226 bytes of data transfer. If there was a global violation, there is no model recreation. In the case of global violation, 110 bytes are transmitted from LPDCs to SPDC.

A.4. Implementation oriented details of PMU Data Dissemination Techniques

In this section, we present the internal design and working of various PMU data dissemination techniques that perform data processing at different stages of PDC (such as parsing, time-aligning and partial computation of applications at PDCs from PMU data packets) based on the applications priorities.

A.4.1. CEUT. In CEUT, PDCs at intermediate layer send aggregated data to higher PDCs with the centralized execution of applications at each PDC [Khandeparkar et al. 2015]. The design and implementation of CEUT is motivated by [IEEE Std C37.244 2013], [Armenia and Chow 2010], [Chenine and Nordström 2011]. The basic functionalities of PDC are adopted from [IEEE Std C37.244 2013]. In line with [Armenia and Chow 2010], a dedicated thread is used to parse each PMU data packet at the LPDC. Further, the same thread performs the parsing and time-align operation. At LPDC, the time-align operation temporarily buffers the parsed data from multiple PMUs. Once the data packets from all the PMU with the same time-stamp are buffered, LPDC sends the aggregated data packet to the SPDC. At SPDC, the parsing and the time-align operations are similar to that of LPDC. There is a dedicated thread to parse data packet from each LPDC. The SPDC then delivers data to each of the applications.

A.4.2. DEFT without priority cognizance of applications. In DEFT without priority based execution of concurrent applications, the time complexity to execute a subquery is different for different applications. For example, ASM requires only checking local threshold violations whereas, SE requires performing matrix computations with all PMU data. Hence, during the parsing of a PMU packet at LPDC, the application relevant PMU data are processed by an independent thread and then the partially computed subquery result is packaged and sent to the SPDC. The time-aligning in DEFT is required to temporarily store data to be retrieved later when local violations occur at peer LPDC for the ASM application. At SPDC, the data packets with partially computed results of applications are parsed and dispatched to the application to get the final computed results.

A.4.3. DEFT with priority cognizance of applications. In DEFT with priority based execution of concurrent applications, there are separate threads to parse, time-align and perform application specific computations for each received PMU packet. Each of these threads is associated with pre-defined static priorities based on the priority of the applications. This differentiates the processing of more important data packets from other less important data packets when the prior ones are queued at the PDC. Further, if the same data is used by multiple applications, the threads performing application-specific computations are associated with priority values in the decreasing order of their criticality. As explained earlier, time-aligning in DEFT is required to temporarily store data, the application-specific tasks always have a higher precedence over the time-aligning of data packets. Hence, the time-align thread in priority based DEFT is associated with very low priority so that it can be pre-empted to process higher priority queued PMU data packet.

A.5. Priority assignment for DEFT in Linux

The Linux scheduler decides which runnable thread is to be executed by the CPU based on real-time scheduling policy (SCHED_FIFO, SCHED_RR etc.) and scheduling priority of the thread [Michael Kerrisk 2014]. The scheduling priorities are in the range 1-99, with 1 the lowest and 99 the highest priority. More details on scheduling policies and priority levels of a Linux kernel scheduler can be found in [Michael Kerrisk 2014].

The implementation of all data dissemination techniques is carried over Linux platform. We have used SCHED_FIFO scheduling policy for all threads with appropriate scheduling priorities based on the criticality of the task performed by the thread. As given in Table I, threads that parse PMU data packets corresponding to ASM have a high priority (95), MCGG has medium priority (90) and SE has low priority (85). The threads corresponding to the time-align operation have the lowest priority (80) among all other tasks.