Coral: Reliable and Low-latency P2P Convergecast for Critical Sensor Data Collection

Daniel Germanus*, Abdelmajid Khelil†, Johannes Schwandke*, Neeraj Suri*
*CS Department, TU Darmstadt, Hochschulstraße 10, Darmstadt, Germany
{germanus,schwandke,suri}@cs.tu-darmstadt.de
†HUAWEI European Research Center, Riesstraße 25, München, Germany
abdelmajid.khelil@huawei.com

***Abstract*—Data collection is a key ingredient for a plethora of distributed systems. Strict responsiveness, i.e., the reliable and timely data delivery is a mandatory requirement for critical applications. For a widened application focus towards large scale critical applications in heterogeneous operational environments, data dissemination infrastructures are inevitably required to provide robustness against frequent perturbations. In this work, we present Coral, a highly reliable and low latency data collection peer-to-peer protocol. Its fully decentralized design allows operation under unfavorable conditions and immediate adaptation towards such effects. Coral provides for a latency optimized and path redundant data collection along with in-network aggregation, i.e., convergecast, for applications with ultra low latency requirements that span across large geographic extents.**

***Index Terms*—Peer-to-Peer protocol, Data collection, Robustness, Responsiveness, Reliability, Wide Area Monitoring System, Smart Grid**

I. Introduction

Contemporary Critical Information Infrastructures (CII) inherently demand robust and reliable data collection to ensure safe operation in order to protect people and surroundings. In this work, we focus on service availability and integrity for large scale CII applications that require the collection of a very large number of time-critical data points generated by a plethora of data-rich sensor nodes towards a small subset of geographically far sink nodes. Typically, CII nodes are hierarchically organized and sensor measurements are aggregated on their way towards the sink following the network hierarchy. This convergecast communication is typical in Wide Area Monitoring Systems (WAMS) environments such as Supervisory Control and Data Acquisition (SCADA) or system protection and is predictably the enabler of a wide range of future applications and systems such as the Internet of Energy, Internet of Vehicles or Internet of Things in general. High data availability demands stem from CII application requirements and commonly include timeliness and reliability aspects. Robustness on the other hand addresses the data collection dependability w.r.t. to external faults which are induced by the CII's operational environment. This environment may involve heterogeneous software and hardware node types, include various communication network operators and technologies, and span across states or continents.

Research partially funded by DFG GRK 1362 (TUD GKmM).

The previously described problem space – i.e., strict reliability requirements on behalf of the CII as well as a complex operational environment – has been partially addressed beforehand in other work [1], [2], [3]. Unfortunately existing approaches lack at least one of the following criteria: (i) robustness, (ii) scalability, (iii) integration in large scale and heterogeneous environments.

Peer-to-peer (P2P) computing characteristics address the aforementioned requirements. However, existing P2P protocols mostly optimize towards unicast or multicast data dissemination [4], [5] or efficient addressing schemes [6], [7], [8]. The continuous adaptation of a fully decentralized convergecast communication P2P infrastructure to a harsh operational environment has not been addressed before to our best knowledge.

A. Contributions

We propose *Coral*, a novel P2P protocol that addresses the aforementioned criteria and fulfills the strict CII application data availability requirements with acceptable additional overhead.

Coral is a decentralized P2P protocol which only requires local knowledge about the underlay network. Coral consists of three parts: (i) periodic link latency measurements, (ii) latency minimal and disjoint path search, and (iii) robust and reliable convergecast routing. Coral's robustness addresses communication or node failures in the underlay network that result in permanent or transient message loss. The efficiency of the three protocol parts allows for global scale deployments and addresses heterogeneous environments that involve different communication infrastructures.

Coral addresses CII data dissemination quality of service (QoS) requirements. It attempts to minimize end-to-end data dissemination latencies and provides redundant communication paths.

Coral has been implemented and validated using an established simulator. The evaluation was performed using a contemporary case study, i.e., system protection workload on a realistic power transmission network topology.

Performance measurements confirm that Coral is a viable solution to increase wide area services' availability and integrity.

B. Paper Structure

We present related approaches and technologies in Section II. A comprehensive system model is provided in Section III and Coral is presented in Section IV. Section V discusses the evaluation of Coral's simulator implementation and provides a result discussion. Conclusion and future work can be found in Section VI.

II. Related Work

A typical CII convergecast application can be found in SCADA systems. Contrary to our considerations, SCADA usually implies bidirectional communication to address the actuator functionality. SCADA is usually employed in field area networks located on the premises of a critical infrastructure like a power utility. Despite a very high amount of measurements and commands, SCADA is usually deployed in a smaller geographic extent and runs either on a dedicated communication network or in the domain of a single operator in contrast to a WAMS.

SCADA systems unfortunately impose deterministic data collection tree topologies which makes them highly vulnerable to perturbations. Therefore, we believe a self-organized/self-healing architecture is compulsory for data collection techniques in CIIs.

The flourish of convergecast usage has been proven in the area of Wireless Sensor Networks (WSN), e.g., [9]. There, convergecast supports the in-network suppression of non-useful data in order to save the batteries of sensor nodes and the precious bandwidth of short range wireless links. WSNs are self-organized/self-healing networks, however, they are tailored to battery-powered nodes which are deployed in geograpically restricted areas.

In order to enhance the responsiveness of SCADA systems, we recently proposed to augment these systems using established P2P techniques to address perturbations in the operational environment [2].

Likewise, microgrid architectures have been proposed that employ overlay networks as a robust communication medium [10], [3]. Microgrids demand robustness, yet their operational environment has small scale characteristics.

Unfortunately, P2P-enabled SCADA or microgrid systems still fail to provide ultra low latency as they use the existing P2P techniques that do not explicitely design convergecast functionality, nor provide for ultra-low latency, which is our objective to develop Coral.

WAMS address large geographic extents and may be part of a CII that has high robustness and reliability demands. WAMS applications usually require m to n communication with $n \ll m$. This fits best to the previously mentioned convergecast requirement in many ultra low latency applications like system protection (SP) in the power grid [11].

There exists a wide range of approaches addressing the mentioned requirements for a large geographic extent of the operational environment (e.g., [1], [12]). Nevertheless, none of these methods addresses decentralization, robustness, or adaptability to perturbations as a whole in multipurpose networks.

Various P2P protocols have been proposed as underlying framework in order to increase the resilience of CII applications. However, most P2P protocols optimize either towards (i) multimedia data dissemination or (ii) efficient addressing of overlay resources. Timeliness is important for multimedia data dissemination, but reliability is not a mission-critical goal, as occasional omissions can be tolerated. Efficient addressing schemes focus on reliability but timeliness is of second rank.

III. System Model

This section provides models for the underlay and overlay network as well as for CII applications and considered perturbations.

A. Underlay Model

We model the underlay network as a directed graph $D_u = (V_u, E_u)$ with V_u as the digraph's vertex set and E_u as its edge set. We assume there exists a routing layer for D_u, so that vertices are able to transmit data among them. Vertices refer either to data-rich sources, e.g., phasor measurement units (PMUs), sinks on various hierarchy levels reflected by the application, or forwarding nodes like routers, firewalls or the like. Edges represent network connections between two vertices. Edges are directed to address differing link latencies for up- and download directions. The link latency is provided as edge weight and can be evaluated using the weight function $w_u(e_u)$, for $e_u \in E_u$.

B. Overlay Model

We define an overlay model as a digraph $D_o = (V_o, E_o)$ with $V_o \subseteq V_u$. Furthermore, $E_o = \{(v,u)|$ there exists a path from v to u in $E_u \wedge (v,u)$ maintain a neighbor relationship $\}, \forall u, v \in V_o$. The weight of an overlay edge $e_o \in E_o$ is represented by the summed up weights of e_o's constituting underlay edges.

C. Application Model

The envisaged application employs data-rich sensors with time-critical data points. Sensor measurements require low latency transmission to geographically far sinks. Measurements may occur with very high sampling rates, which we therefore consider a continuous data stream. An example for the aforementioned application scenario is a deployment of synchronized PMUs. Sinks are usually hierachically organized, i.e., 1^{st} level sinks aggregate sensor measurements before sending it to sinks on higher levels. This hierarchical aggregation refers to the convergecast network functionality. An example for such sinks are phasor data cocentrators (PDC).

D. Perturbation Model

A robust data collection system should overcome specific failures as the timely and reliable message exchange is vital to the class of considered applications. The perturbation model focusses on application data message loss and application data integrity violations. The root causes of these two failures can be considered in the following two failure modes which we also consider for robustness evaluations:

1) Communication Failures: Communication failures induce message loss or message corruption and directly impact the reliability of the application. Possible reasons for communication failures can be localized power outages, physical damage, or electromagnetic interference effects.

2) Node Failures: Unavailability of nodes leads to message loss in the lack of an alternate path. The list of possible reasons is similar to the one of communication failures and can be extended by software failures and maintenance tasks.

Both, communication failures and node failures can be transient or permanent. The data collection system addresses these failures using various robustness features. They are described in the next Section.

IV. Our Approach: Coral

Coral is a P2P protocol designed for extremely low latency and reliable convergecast for sensor data collection with potentially very high sampling rates.

Coral is fully decentralized. Its four core mechanisms, i.e., neighbor selection, periodic heartbeats, source-sink path search and robust & reliable convergecast depend on local knowledge only. Consequently, no single point of failure exists for the protocol itself. Coral peers maintain neighbor relationships with overlay edge adjacent peers only. No hierarchy among peers exist, nevertheless Coral is suitable for hierarchical applications. Coral peers can be assigned to three roles, namely sources, sinks and forwarders. Furthermore, a Coral peer can be simultaneously assigned to source and sink roles, e.g., to reflect hierarchies in the CII application. Source/sink peers are usually colocated on the CII application's source/sink underlay nodes. Peers get unique IDs assigned which helps searching and describing overlay paths.

To join an overlay network, a peer instance p has to perform the boostrap process by choosing its initial neighbor peer from a list of potential neighbors. This list is generated out-of-band and not part of the Coral protocol. Once p has joined the network after the bootstrap process, it starts the four core protocol mechanisms which are described next. Parameters discussed in this Section are listed in Table I.

A. Neighbor Selection

The neighbor selection mechanism generates a connected overlay graph. Each peer maintains at least n_{min} and at most n_{max} connections to neighboring peers. Therefore, the closest neighbors are selected according to a distance notion $\delta : p_1 \times p_2 \rightarrow \mathbb{R}^+$ with $p_1, p_2 \in V_o$ and $p_1 \neq p_2$. Throughout this work, we use the peers' geographic coordinates and the euclidean distance for δ as we assume that geographic distance correlates with latency. Other choices for δ could be the network latency between two peers, the overlay or underlay hop count, or the numerical distance of an overlay address space as it is common for DHTs.

Once p joined the overlay, the bootstrap peer provides its neighbor set to p. The neighbor set consists of peer IDs and the peers' coordinates. If p determines other peers within acceptable distance, then p sends them a neighbor request message. In case the message is accepted, an overlay link is established.

In case p maintains less than n_{min} neighbors, Coral tries in short time intervals ($s_{aggressive}$) to find more neighbors. If p maintains more than n_{min} neighbors, the time intervals become longer (s_{normal}) and for n_{max} neighbors maintained by p, Coral stops looking for new neighbors.

B. Source to Sink Path Search

A connected overlay graph between source and sink peers is required for efficient application message transport. In order to address the responsiveness and fault tolerance requirements, source peers determine multiple shortest – i.e., low latency – paths towards their corresponding sink nodes. Coral's network model considers asymmetric network connection links and latencies. The path selection process for the actual data transmission is described in Section IV-C1.

1) Search Messages: Source peers send SEARCH messages to their neighbor peers. Each message contains the key of the corresponding sink peer, a time-to-live (TTL) value, a variable length list to describe the overlay path, and latency measurements. SEARCH messages get dropped if either the TTL counter turns zero or if the current peer is already on the SEARCH message's path list. Otherwise, the TTL is decreased, the current peer appended to the path list, and the SEARCH message is forwarded to the neighbor set except to those neighbor peers that are already on the path.

If the key matching sink receives the SEARCH message, then it prepares a RESPONSE message that is sent in reverse order along the search path. The source peer stores after reception of the RESPONSE message the path together with the latency measurement in a set of possible paths. Paths can be assigned a timeout after which their validity expires and the latency measurement is re-evaluated.

2) TTL Optimization using Expanding Ring Algorithm: In order to reduce the SEARCH message amount in the system, we use a variant of an expanding ring algorithm to adjust SEARCH message TTL values. For too small TTL values no search paths would be found and in case for too large TTL values, suboptimal searches (e.g., those in the wrong geographical direction) are unnecessarily long running. Therefore, a minimum and maximum TTL value (TTL_{min}, TTL_{max}) is specified as well as the minimum and maximum amount of paths that should be evaluated for a single search. The approach also addresses changes in the overlay topology, e.g., as a consequence on peer failures.

C. Reliable Data Collection

Coral's data collection is based on two mechanisms: (i) path selection and (ii) hop wise data transport from source to sink peer. The CII application provides data, e.g., sensor measurements, to the source peer along with an underlay sink descriptor. The latter one allows to determine the corresponding sink peer's key. We assume that the source to sink path search (cf. Section IV-B) has already finished. Then, the source peer is able to start the path selection process which is described in the next subsection.

TABLE I
SELECTED CORAL PARAMETERS

Parameter	Description
d	Path duplication: DATA messages are sent along d different paths to the sink
w_1, w_2	Penalty weights: Path selection process to prioritize either latency or disjointness criteria
$i_{feedback}$	Feedback interval: Sink peers send each $i_{feedback}$ data messages a FEEDBACK message back to the respective source peer
n_{max}	Maximum neighbors
n_{min}	Minimum neighbors
$s_{aggressive}$	Interval for aggressive neighbor searches
s_{normal}	Interval for normal neighbor searches
$timeout$	Timeout for neighbor peers
i_{HB}	Interval for HEARTBEAT messages
i_{Route}	Interval for searching routes
TTL_{init}	Initial SEARCH message TTL
TTL_{min}	Minimum SEARCH message TTL
TTL_{max}	Maximum SEARCH message TTL
TTL_{step}	TTL adaptation step size

1) Path Selection: The source peer selects d paths, i.e., according to the path duplication parameter, from the list of all known paths to the sink. Two path criteria are considered, i.e., latency and disjointness. The selection process is based upon a weighted sum approach that assigns to the latency and disjointness criteria a penalty weight value $w_1, w_2 \in \mathbb{R}^+$ to prioritize one criterion over the other.

2) Data Transfer: Once a source peer has finished the path selection process, CII application data is organized in datasets that consist of d DATA messages with identical payloads. The dataset's DATA messages are transported towards the sink peer using d paths. DATA messages contain a path ID, path information, and a path index that each forwarding peer increments in order to refer to the current hop. Sink peers either wait a limited time so the dataset of d messages can be re-constituted in order to assess the payload integrity (cf. Section IV-D3 or the sink peer accepts the first DATA message and ignores subsequent ones from the same dataset. DATA messages are encapsulated in UDP/IP messages instead of TCP/IP to reduce the network overhead and negative latency impact. Source peers receive in regular intervals of $i_{feedback}$ received DATA messages a feedback message. These allow to re-assess end-to-end latencies of paths.

D. Robustness Features

Coral provides three protocol intrinsic robustness features, namely periodic heartbeats, broken path handling, and sink-side majority voting. This subsection discusses them.

1) Periodic Heartbeats: Peers send HEARTBEAT messages each i_{HB} time units to all their neighbors in order to perform (i) a liveness check of the neighbors and (ii) latency measurements. If a liveness check fails after $timeout$ time units, the specific neighbor peer is evicted from the HEARTBEAT issuing peer's neighbor set. The latency measurement occurs in two steps. During the first step, the HEARTBEAT receiving peer r evaluates the message latency for the sending process from the HEARTBEAT issuing peer p and updates the neighbor set entry. The second step occurs once r sends a HEARTBEAT message to p, as r tells p about the latency measurement performed during the first step. Clearly, when no initial measurements exist, this is a three-step process.

2) Broken Path Handling: If a peer p has evicted peer q from its neighbor list (due to a HEARTBEAT timeout) and receives a DATA message with q being the next hop, then p returns a PATH ERROR back to the source peer s such that s removes paths that include q. Coral has a so called *jump forward* mechanism that instructs p to evaluate the DATA message's path information beyond the current path index regarding potential neighbor peers. If such a peer can be found, the DATA message gets forwarded and otherwise it gets dropped.

3) Sink-side Majority Voter: A majority voter located on sink peers can be used to perform data integrity checks. Therefore, related payloads of d DATA messages are checked for equality on the sink. In case $\lceil \frac{d}{2} \rceil$ payloads are found equal, it passes the majority voting process.

V. EVALUATION

We implemented Coral using the OverSim [13] P2P protocol framework which runs in the established OMNeT++ simulator [14]. This section presents the application model, the concrete underlay model and how we derived it, simulation settings that reveal Coral parameters used throughout the evaluation, and metrics definitions. Finally, this section presents simulation experiments for both, performance and robustness evaluations, and highlights the results.

A. Simulated CII Application

We select a system protection CII application model with PMUs as data sources, PDCs as sinks. We consider a source sampling rate of 50Hz and assume that measurements are encapsulated using IEEE C37.118 of 104 bytes length within UDP/IP datagrams in order to measure the network bandwidth consumption of CII application traffic. The plain CII application traffic generated on a single source peer per second is therefore $50 \cdot 104$ Bytes ≈ 5.05 KByte/s.

B. Underlay Network Topology

We considered the power transmission network topology of New Zealand (NZ) to build a realistic underlay network model. Therefore, we used publicly available information from Transpower, the company that operates NZ's power transmission network. They provide a listing of substations and power plants as well as power lines and line voltages. Substations and plants can be regarded as underlay vertices V_u and the lines that connect them are considered as underlay edges E_u. As no information about employed communication link technologies and link latencies is publicly accessible, we created a line length distribution. Then, we grouped transmission line lengths in multiples of 20km which results in 5 groups. We assume that optical fiber communication links are provided next to high voltage transmission lines and assigned link latencies as

TABLE II
COMMUNICATION LINK LATENCIES

Line length (km)	Latency (ms)
$l \leq 20$	0
$20 < l \leq 40$	1
$40 < l \leq 60$	2
$60 < l \leq 80$	4
$80 < l \leq 100$	8

TABLE III
CORAL SIMULATED EVENTS

t (s)	Event
0	Initialisation, neighbor search, heartbeats
120	Peers start searching paths
180	Data message transmission starts
230	Start of failure scenarios

TABLE IV
CORAL PARAMETER VALUES

Parameter	Value	Parameter	Value
d	3	i_{HB}	100ms
w_1; w_2	1; 3	$timeout$	1s
$i_{feedback}$	20	i_{Route}	20s
n_{max}	8	TTL_{init}	10
n_{min}	4	TTL_{min}	4
$s_{aggressive}$	2s	TTL_{max}	16
s_{normal}	8s	TTL_{step}	2

described in Table II. These are the values returned by the weight function $w_u(e_u)$ for $e_u \in E_u$.

Furthermore, $|V_u| = 162$ and we determined 14 out of these 162 vertices to act as 1^{st} level PDCs corresponding to the 14 loading zones as indicated by Transpower.

C. Simulation Settings

Specific Coral parameter values used throughout the simulation experiments are provided in Table IV.

Furthermore, Coral is scheduled in the simulator according to the data provided in Table III. This data is crucial to interpret subsequent time series plots in this section.

D. Metrics

Coral is assessed regarding its network overhead and responsiveness. Network overhead is measured in bytes per second and grouped in three categories, i.e., (i) data bytes which refers to the CII application workload, (ii) protocol bytes which includes neighbor searches, route searches, and feedback messages, (iii) heartbeat messages. Responsiveness is considered as a combination of latency and reliability. Latency is measured in terms of source-to-sink (or end-to-end) delay in miliseconds. Reliability is defined as the amount of messages arrived at the sink divided by the amount of messages sent at the source.

E. Simulation Experiments

Figure 1 depicts the average outgoing traffic and latencies for all peers with a measurement interval of 1 second and 95% confidence intervals in a failure free scenario. Data transport (blue plot) is in the range of 90 KB/s and latency (gray plot) in the range of up to 10ms. The amount of initial protocol bytes/s (red plot) decreases and stabilizies as Coral repeatedly performs the expanding ring algorithm. Heartbeat network overhead is in the range of 3 KB/s. For all simulation experiments, we assume that 148 PMUs emit PMU measurement messages at 50 Hz (i.e., once every 20ms) along $d = 3$ different paths, which refers to approximately 22,000 messages per second. Although not depicted for space limitation reasons, we ran experiments with PMUs sending data at 100 Hz which intuitively resulted in a higher amount of Coral data transport bytes, but the higher frequency does not alter the network overhead for heartbeats and searching.

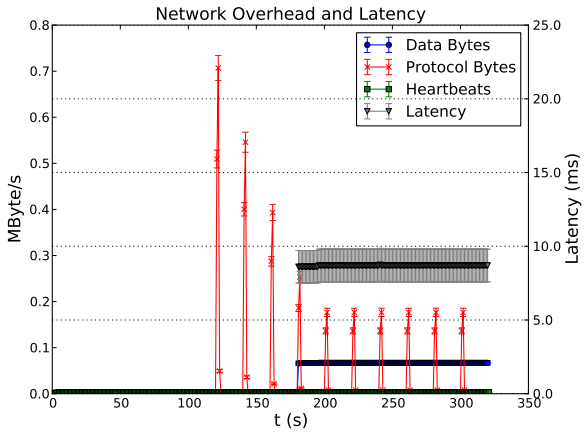

Fig. 1. Traffic and Latency Evaluation – Failure Free Scenario

Furthermore, we present two perturbation scenarios, namely (i) random peer failures and (ii) targeted peer failures on $[0, d]$ out of d disjoint paths. For both scenarios, failing peers do not recover.

1) Random Peer Failure: Figure 2 depicts the average outgoing traffic and latencies with 95% confidence intervals and 30% random peer failures at $t = 230s$. The temporary data bytes decrease after the failure corresponds to message loss on failing peers. Likewise, the latency increase is a consequence of failing peers that are part of preferred routes. As failed peers do not recover, the latency does not decrease again to the initial range.

Figure 3 depicts the CII reliability and percentages indicate how many peers fail at $t = 230s$. Intuitively, higher percentages of failing peers have a higher impact on the reliability. As an exmaple, 30% failing peers entail a reliability of approximately 0.42. For higher percentages of peer failures it takes longer for the reliability measurement to fit to 1.0 again. This is because of the additional neighbor search overhead in order to provide d substitute paths for those paths affected by the failures. After all, even with 50% randomly failing peers, Coral achieves a reliability higher than 0.16.

2) Targeted Peer Failures: Figure 4 shows the reliability for targeted peer failures for $f = 1, \ldots, 3$ failing peers at $t = 230s$ on $d = 3$ disjoint paths with 10 hops path length.

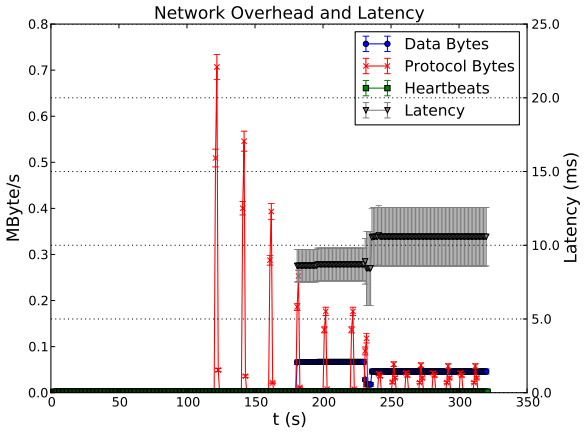

Fig. 2. Traffic and Latency Evaluation for 30% Randomly Failing Peers

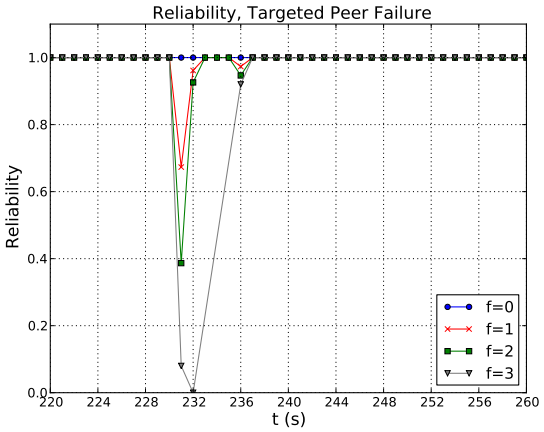

Fig. 4. Reliability of a single PMU-to-PDC Communication with $f = 1, 2, 3$ Peer Failures on $d = 3$ Different Paths

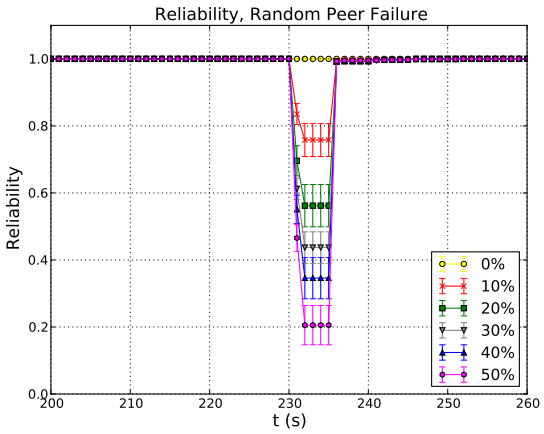

Fig. 3. Reliability with Varying Amount of Randomly Failing Peer

Clearly, three failing peers which cut off all d paths towards the sink temporarily results in 0 reliability. However, Coral establishes new paths and adapts within few seconds to the perturbation. For 2 and 3 failing paths a slight increase in latency can be denoted subsequent to the peer failures.

VI. Conclusion

Coral provides a scalable, decentralized and highly reliable technique for designing robust future CII communication middlewares. Simulations experiments using a realistic network topology confirm Coral's high responsiveness. Coral adapts within few seconds to perturbation scenarios of up to 50% of failing nodes to quickly fulfill strict CII requirements again. Also, Coral continuously assesses end-to-end latencies and searches new message routing paths if needed in order to address CII application timeliness requirements.

Coral's path search uses a flood approach and we tend to compare it with existing shortest path heuristics regarding the network overhead and search result quality. To reduce network usage peaks, we plan to address indeterministic back off timings. Furthermore, we plan to implement Coral in a distributed testbed environment.

References

[1] H. Gjermundrod et al., "GridStat: A flexible QoS-managed data dissemination framework for the power grid," *IEEE Transactions on Power Delivery*, vol. 24, no. 1, pp. 136–143, 2009.

[2] Germanus, D. et al., "Increasing the Resilience of Critical SCADA Systems Using Peer-to-Peer Overlays," in *1st International Symposium on Architecting Critical Systems (ISARCS)*, 2010.

[3] K. Eger et al., "Towards P2P technologies for the control of electrical power systems," in *Eighth International Conference on Peer-to-Peer Computing (P2P'08)*. IEEE, 2008.

[4] L. Guo et al., "Measurements, analysis, and modeling of BitTorrent-like systems," in *Proceedings of the 5th ACM SIGCOMM conference on Internet Measurement*. USENIX Association, 2005.

[5] S. Seyyedi et al., "Hybrid CDN-P2P architectures for live video streaming: Comparative study of connected and unconnected meshes," in *International Symposium on Computer Networks and Distributed Systems (CNDS)*, 2011, pp. 175–180.

[6] I. Stoica et al., "Chord: A Scalable Peer-to-Peer Lookup Service for Internet Applications," in *Proc. SIGCOMM*, 2001, pp. 149 – 160.

[7] P. Maymounkov and D. Mazières, "Kademlia: A Peer-to-Peer Information System Based on the XOR Metric," in *Proc. IPTPS*, 2002, pp. 53 – 65.

[8] M.S. Artigas et al., "Cyclone: a novel design schema for hierarchical DHTs," in *Fifth IEEE International Conference on Peer-to-Peer Computing (P2P 2005)*, 2005, pp. 49–56.

[9] F. Karim et al., "AReIT: Adaptable Reliable Information Transport for Service Availability in Wireless Sensor Networks," in *International Conference on Wireless Networks (ICWN)*, 2009, pp. 75–81.

[10] G. Deconinck et al., "Robust overlay networks for microgrid control systems," in *Proc. Workshop on Architecting Dependable Systems (WADS 2007), co-located with 37th Ann. IEEE/IFIP Int. Conf. on Dependable Systems and Networks (DSN 2007), Edinburgh, Scotland (UK)*, 2007, pp. 148–153.

[11] D. E. Bakken et al., "Smart Generation and Transmission With Coherent, Real-Time Data," *Proceedings of the IEEE*, vol. 99, no. 6, pp. 928–951, 2011.

[12] K. Young-Jin et al., "SeDAX: A Scalable, Resilient, and Secure Platform for Smart Grid Communications," *IEEE Journal on Selected Areas in Communications*, vol. 30, no. 6, pp. 1119–1136, 2012.

[13] I. Baumgart et al., "OverSim: A Flexible Overlay Network Simulation Framework," in *Proc. GI at INFOCOM*, 2007, pp. 79 – 84.

[14] G. Pongor, "OMNeT: Objective Modular Network Testbed," in *Proceedings of the International Workshop on Modeling, Analysis, and Simulation On Computer and Telecommunication Systems (MASCOTS)*, 1993.