Needle in a cross-layer sensor stack

Vasanth Iyer*, S. Sitharama Iyengar†, Garmiela Rama Murthy*,
Kannan Srinathan*, Regeti Govindarajulu* Mandalika B. Srinivas‡ and Dhananjay Singh§

*International Institute of Information Technology, Hyderabad, India - 500 032
†Louisiana State University, Baton Rouge, LA 70803, USA
‡Brila Institute of Technology & Science, Hyderabad Campus, Hyderabad-500078, India
§National Institute for Mathematical Sciences (NIMS), South Korea.

vasanth@research.iiit.ac.in,iyengar@csc.lsu.edu,rammurthy@iiit.ac.in,
srinathan@iiit.ac.in,srinivas@bits-hyderabad.ac.in,gregeti@iiit.ac.in,dhananjayiiit@gmail.com

Abstract—**In this problem, the process start with real-valued inputs and are supposed to decide eventually on real-valued outputs. The process is permitted to send real-valued data in messages. Instead of having to agree exactly, as in the ordinary agreement problem, this time the requirement is just that they agree with within a small positive real-valued tolerance ϵ. In the standard implementation agreement problem converges to a consistent value for each sensor measurement, which is nonfaculty using Byzantine algorithm with at-least $n > 3$ faulty sensors. We define a pre-processing *BestBasis* cost function which allows to find a coherent range, which can be measured using intra-sensor in an ensemble of real-values. The overlap of the range is calculated by using a clique, with a runtime of $O(nk)$ in the worst-case, where the size of clique, k, is a variable. The computation of the pre-processing takes $O\log(D)$, where D is the number of levels in a sparse signal basis. Which are some times analogously compared to needle in a hay stack definition.**

Index Terms—**Compressed Sensing (DCS); Sensor Fusion; Pre- and Post processing of Sensors.**

I. Byzantine Nonfaulty processes

A. Introduction

- Agreement: The decision values of any pair of nonfaulty processes are within ϵ of one another.
- Validity: Any decision value for a nonfaulty process is within the range of the initial values of the nonfaulty processes.

B. Definitions

An interval representation of a graph is a family of intervals assigned to the vertices so that vertices are adjacent if and only if the corresponding intervals intersect. This introduces a class of graphs, such that it has a degree sequence of, $d_1 \geq ... \geq d_n \subseteq \Re^2$ where such an ordering is easy to find.

Definition 1 *(Clique) Is the maximum size of pairwise adjacent vertices in Graph G. The click number of a graph G is written as $\omega(G)$. A clique in an undirected graph $G = (V, E)$ is a subset of the vertex set $C \subseteq V$, such that for every two vertices in C, there exists an edge connecting the two.*

Definition 2 *(Byzantine [5] Nonfaulty processes detection) Is the collection of vertices forming a Clique, as shown in Figure 1(a).*

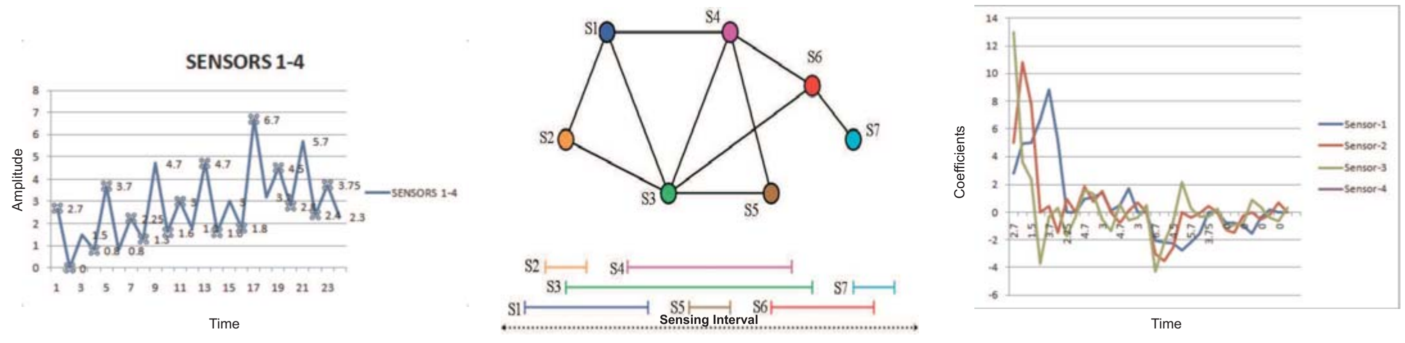
Property 1 *When these algorithms terminate, all the nonfaulty processes have the same decision values for all processes. Each chooses the $\frac{n}{2}th$ largest value in the multiset of decision values as its own final decision value.*

Lemma 1 *Since $n > 3f$, it follows that the middle value in the multiset must be among the initial values of the nonfaulty processes.*

Theorem 1 *Byzantine approximate agreement solves the approximate agreement problem for a n-node complete graph as shown in Figure 1(a), if $n > 3f$.*

Extending a multiset convergence-algorithm, not using Byzantine agreement.

Definition 3 *A multi-set has several properties of interest. The range of a multi-set, represented by ρ, is the range of values between the smallest and largest values in the multi-set. Mathematically this is written,$\rho(V) = [v_1, v_n]$. To measure how big this range is, the diameter is defined to be $\delta(V) = [v_n - v_1]$ First, if U is a finite multiset of reals with at least $2f$ elements, and $u_1, ..., u_k$ is an ordering of the elements of U in nondecreasing order, then let $reduce(U)$ denote the result of removing the f smallest and f largest elements from U, that is, the multiset consisting of $u_{f+1}, ..., u_{k-f}$. Also, if U is a nonempty finite multiset of reals, and $u_1, ...u_k$ is again an ordering of the elements of U in nondecreasing order, then let $select(U)$ be the multiset consisting of $u_1, u_{f+1}..., u_{2f+k}, ...$, that is, the smallest element of U and every fth element thereafter. We also say that the $range$ of a nonempty finite multiset of reals is the smallest interval containing all the*



(a) Possible consistent nonfaulty sensors **(b) Clique showing overlap coherent values** **(c) Wavelet Packet and possible representations**

Fig. 1. (a) Showing sparse sampling of non-faculty sensors (b) Showing network topology of faulty and non-faulty sensors (c) Showing best estimate of the corresponding calibrated sensor interval.

elements, and the width of such a multiset is the size of the range interval.

Process i maintains a variable val containing its latest estimate. Initially, val_i contains $i's$ initial value. At each round, process i does the following. First, it broadcasts its val to all processes, including itself. Then it collects all the values it has received at that round into a multiset W; if i does not receive a value from some other process, it simply picks some arbitrary default value to assign to that process in the multiset, thus ensuring that $W = n$. Then, process i sets val to $mean(select(reduce(W)))$. This is, process i throws out the f smallest and f largest elements of W. From what is left, i selects only the smallest element and every fth element thereafter. Finally val is set to the average (mean) of the selected elements. We claim that at any round, all the nonfaulty processes' vals are among the nonfaulty processes' $vals$ just prior to the round. Moreover, at each round, the width of the multiset of the nonfaulty processes' $vals$ is reduced by a factor of t at least $[\frac{n-2f-1}{f}]+1$. if $n > 3f$, this is greater than 1.

Lemma 2 *Suppose that $val_i = v$ just after round r of an execution of the Convergence-Approx-Agreement. Then v is among the nonfaulty processes' vals just before round r.*

Proof: If W_i is the multiset collected by process i at round r, then there are at most f elements of W_i that are not values sent by nonfaculty processes. Then all the elements of $reduce(W_i)$ are among nonfaulty processes' vals just prior to round r. It follows that the same is true for $mean(select(reduce(W_i))$,which is the new value of val_i. ■

Lemma 3 *The convergence of the algorithm is based on multiset reduction $[\frac{n-2f-1}{f}]+1$. if $n > 3f$, this is greater than* 1.

II. Computation of the Range the Nonfaulty Sensors

A. Introduction

Consider a real-valued signal $x \in R^N$ indexed as x(n), $n \in 1, 2, ..., N$. Suppose that the basis $\Psi = [\Psi_1, ..., \Psi_N]$ provides a K-sparse representation of x; that is, where x is a linear combination of K vectors chosen from, Ψ, n_k are the indices of those vectors, and $\vartheta(n)$ are the coefficients; $x = \sum \vartheta(n)\Psi_n = \sum \vartheta(n_k)\Psi_{n_k}$ the concept is extendable to multi-processors which need to have sequential consistency [3].

Lemma 4 *Maintains $i.i.ds$ sparse model among values of a single sensor. The collection $x = P\theta$ of all possible basis representation is called the sparsity model.*

Lemma 5 *Maintains a single operation using non-overlapping subsets, which are present in all signals and among all cost level representation $X = P\Theta$ of the processed signal. The non-overlapping coefficients represents the basis which is lossless representation of the signal ensemble.*

Proof: Given n, f, ϵ we find the joint sparsity level D, which is from a collection of the signal $Basis$ representation. A single measured signal of finite length, which can be represented in its sparse representation. This is sequential for all sensors by providing sequential consistency from Lemma 4. The Basis representation is transformed into all its possible basis representations. The number of basis for each level j can be calculated from the level definition, which preserves the coherency from Lemma 5, are also illustrated in Figure 1(b). Then all the coefficients of $Basis$ are in the range of nonfaulty processes' vals just prior to round r. It follows that the same is true for $BestBasis$ $k[coff_0, coff_1, ...coff_{n-1}]$, which is the new compressed value of val_i, as shown in Figure 1(c). Combining both the Lemmas of the shared memory [3] and the Byzantines agreement proof seen earlier [5], we predict new approximate value at-most differing by ϵ. ■

Pre-process-Computation: For a given ensemble X, we let $P_F(X) \subseteq P$ denote the set of feasible location matrices $P \in P$ for which a factorization $X = P\Theta$ exits. We define the joint sparsity levels of the signal ensemble as follows. The joint sparsity level D of the signal ensemble X is the number of columns of the smallest matrix $P \in P$. In these models each signal x_j is generated as a combination of two components: (i) a common component z_C, which is present in all signals, and (ii) an innovation component z_j, which is unique to each signal. These combine additively, giving $x_j = z_C + z_j, j \in \forall$. $X = P\Theta$. A further optimization can be performed to reduce the number of measurement made by each sensor, the number of measurement is now proportional to the maximal overlap of the inter sensor ranges and not a constant. This is calculated by the common coefficients K_c and K_j, if there are common coefficients in K_j then one of the K_c coefficient is removed and the common Z_c is added, these change does not affect the reconstruction of the original measurement signal x.

Definition 4 *The levels as shown in Figure 2(d) of single measured signal of finite length, which can be represented in its sparse representation, by transforming into all its possible basis representations. The number of basis for each level j can be calculated from the equation as $A_{j+1} = A_j^2 + 1$. So staring at $j = 0$, $A_0 = 1$ and similarly, $A_1 = 1^2 + 1 = 2$, $A_2 = 2^2 + 1 = 5$ and $A_3 = 5^2 + 1 = 26$ different basis representations.*

B. Coherency cost function of Sparse representation

A single measured signal of finite length, which can be represented in its sparse representation, by transforming into all its possible basis representations. The number of basis for the for each level j can be calculated from the equation as

$$A_{j+1} = A_j^2 + 1 \tag{1}$$

So staring at $j = 0$, $A_0 = 1$ and similarly, $A_1 = 1^2+1 = 2$, $A_2 = 2^2 + 1 = 5$ and $A_3 = 5^2 + 1 = 26$ different basis representations.

Let us define a framework to quantify the sparsity of ensembles of correlated signals $x_1, x2, ..., xj$ and to quantify the measurement requirements. These correlated signals can be represented by its basis from equation (5). The collection of all possible basis representation is called the sparsity model.

$$x = P\theta \tag{2}$$

Where P is the sparsity model of K vectors ($K << N$) and θ is the non zero coefficients of the sparse representation of the signal. The sparsity of a signal is defined by this model P, as there are many factored possibilities of $x = P\theta$. Among the factorization the unique representation of the smallest dimensionality of θ is the sparsity level of the signal x under this model.

Pre-processing	Algo 5	Algo 6	Algo 7
Implementation	flooding	flooding	wavelet
Complexity	$O(N^2)$	$O(N^2)$	$O(K)$
Calibration	data centric	coverage centric	sensor centric
Accuracy	consistent	coherent	coherent

TABLE I
CATEGORY OF PRE-PROCESSING ALGORITHMS.

C. Algorithm Definition

Theorem 2 *Each Process Element (PES) maintains a variable min-val, originally set to its own initial value. For each of $\frac{f}{k} + 1$ rounds, the PES all broadcast their minvals, then each process resets its min-val to the minimum of its old min-val and all the values in its incoming messages. At then end, the decision value is min-val.*

Algorithm 1 FloodMinVal

```
 1: $states_i$ :
 2: rounds ∈ ℵ, initially 0
 3: decision ∈ V ∪ unknown, initially unknown
 4: min-val ∈ V, initially i's value

 5: $msgs_i$ :
 6: **if** rounds ≤ $\frac{f}{k}$ **then**
 7:     send min-val to all other processes
 8: **end if**

 9: $trans_i$ :
10: rounds := rounds+ 1
11: let $m_j$ be the message from j, for each j from which a message arrives
12: min-val := $min(min - val \cup m_j : j \neq i)$
13: **if** rounds= $\frac{f}{k} + 1$ **then**
14:     decision := min-val
15: **end if**
```

Algorithm 2 FloodMinRange

```
 1: $states_i$ :
 2: rounds ∈ ℵ, initially 0
 3: decision ∈ V ∩ unknown, initially unknown
 4: min-range ∈ V, initially i's value

 5: $msgs_i$ :
 6: **if** rounds $N - \tau$ **then**
 7:     send min-range to all other processes
 8: **end if**

 9: $trans_i$ :
10: rounds := rounds + 1
11: let $m_j$ be the message from j, for each j from which a message arrives
12: min-range := $min(min - range \cap m_j : j \neq i)$
13: **if** rounds= $N - \tau$ **then**
14:     decision := min-range
15: **end if**
```

D. Sensor Centric Algorithm

DCS allows to enable distributed coding algorithms to exploit both intra-and inter-signal correlation structures. In a sensor network deployment, a number of sensors measure signals that are each individually sparse in

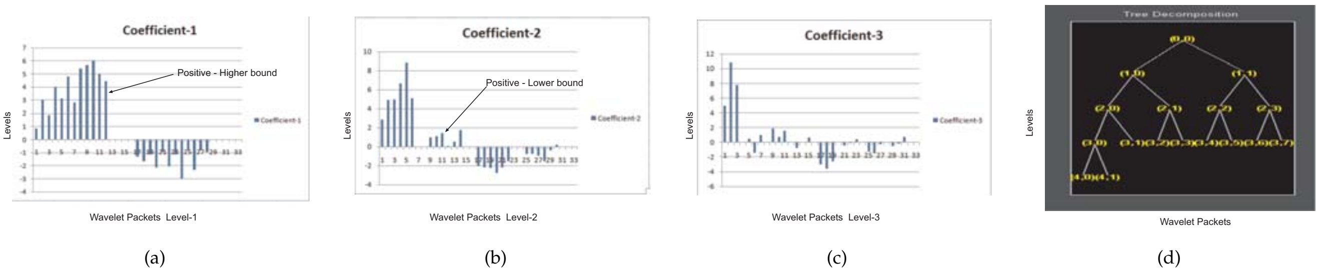


Fig. 2. Sensor-centric Data Fusion during the aggregation step using sparse wavelet model.

Algorithm 3 Wavelet Decomposition

```
1: function D = wpd(S,h,g,J)
2: N=length(S)
3: if J > $floor(\log_2(N))$ then
4:     error ('Too many levels')
5: else
6:     if $rem(N, 2^(J-1))$ then
7:         error ('Signal length must be a multiple of $2^{\%i}$)'
8:     end if
9: end if
10: $D = zeros(J, N)$
11: $D(1, :) = S$;
12: For each level in the decomposition
13: (starting with the second level)
14: for $k = 1$ :J-1 do
15:     $width = \frac{N}{2^{j-1}}$
16:     for $k = 1 : 2^{j-1}$ do
17:         $Interval = [1 + (k-1) * width : k * width]$;
18:         $D(j+1, Interval) = dwt(D(j, Interval), h, g)$;
19:     end for
20: end for
```

Algorithm 4 Generating a cost function

```
1: $J = size(D, 1)$
2: $SignalLength = size(D, 2)$
3: $N = \log_2(SignalLength)$
4: Apply cost function for each element in the decomposition
5: for $k = 1 : 2^{J-1}$ do
6:     $j = floor(\log_2(k)) + 1$
7:     $2^{(N-j+1)}$
8:     Go through all elements on the j'th level
9:     for $m = 1 : 2^{J-1}$ do
10:        $E = D(j, 1 + (m - 1(*L : m * L)$;
11:        $CostValues(k) = CostFunc(E)$
12:    end for
13: end for
```

the some basis and also correlated [6] from sensor to sensor. If the separate sparse basis are projected onto the scaling and wavelet [6] functions of the correlated sensors(common coefficients), then all the information is already stored to individually recover each of the signal at the joint decoder. This does not require any pre-initialization between sensors. The expanded wavelet optimization and its cost-functions are shown in Figure 2(a) and 2(b).

1) Joint Sparsity representation: For a given ensemble X, we let $P_F(X) \subseteq P$ denote the set of feasible location matrices $P \in P$ for which a factorization $X = P\Theta$ exits. We define the joint sparsity levels of the signal ensemble as follows. The joint sparsity level D of the signal ensemble X is the number of columns of the

Algorithm 5 Best basis search

```
1: Bottom-up search for the best basis.
2: for $J = J - 1 : -1 : 1$ do
3:     Bottom-up search for the best basis.
4:     for $k = 2^{(J-1)} : 2^{j-1}$ do
5:         $v2 = CostValues(2 * k) + CostValues(2 * k + 1)$
6:         if $v1 \geq v2$ then
7:             $Basis(k) = 1$
8:         else
9:             $CostValues(k) = v2$
10:        end if
11:    end for
12: end for
13: Fill with 2's below the chosen basis
14: for $k = 1 : (length(Basis) - 1)/2$ do
15:     if $Basis(k) == 1 \| Basis(k) == 2$ then
16:         $Basis(2 * k) = 2$
17:         $Basis(2 * k + 1) = 2$
18:     end if
19: end for
20: Convert all the 2's to 0's
21: $Basis = Basis. * (Basis == 1)$;
```

smallest matrix $P \in P$. In these models each signal x_j is generated as a combination of two components: (i) a common component z_C, which is present in all signals, and (ii) an innovation component z_j, which is unique to each signal. These combine additively, giving

$$x_j = z_C + z_j, j \in \forall \tag{3}$$

$$X = P\Theta \tag{4}$$

We now introduce a clique in a graph $G = (V_V, V_M, E)$, as shown in Figure 1 (b), that represent the relationships between the entries of the value vector and its measurements. The common and innovation components K_C and K_j, $(1 < j < J)$, as well as the joint sparsity $D = K_C + \sum K_J$.

The set of edges E is defined as follows:

- The edge E is connected for all K_c if the coefficients are not in common with K_j.
- The edge E is connected for all K_j if the coefficients are in common with K_j.

A further optimization can be performed to reduce the number of measurement made by each sensor, the number of measurement is now proportional to the maximal overlap of the inter sensor ranges and not a constant as shown in equation (4). This is calculated by the common coefficients K_c and K_j, if there are common coefficients in K_j then one of the K_c coefficient is removed and the common Z_c is added, these change does not effecting the reconstruction of the original measurement signal x.

III. Model Validation

A. Lower Bound Validation using Covariance

The Figure 2(a), 2(b), 2(c) and 2(d) shows lower bound of the overlapped sensor i.i.d. of $S_1 - S_4$, as shown it is seen that the lower bound is unique to the temporal variations of S_2. To have a coherent range prediction we need to select on-overlapping parts of the signal as discussed before. From coefficient values which are positive and have lower bound values we select the coefficient as shown in Figure 2(a). The other sparse values have zero or negative values which are selected from Figures 2 (b) and 2(c) to reproduce the signal without any loss. In our analysis we will use a general model which allows to detect sensor faults. The binary model can result from placing a threshold on the real-valued readings of sensors. Let m_n be the mean normal reading and m_f the mean event reading for a sensor. A reasonable threshold for distinguishing between the two possibilities would be $0.5(\frac{m_n+m_f}{2})$. If the errors due to sensor faults and the fluctuations in the environment can be modeled by Gaussian distributions with mean 0 and a standard deviation σ, the fault probability p would indeed be symmetric. It can be evaluated using the tail probability of a Gaussian [4], the Q-function [4], as follows:

$$p = Q\frac{\left((0.5(\frac{m_n+m_f}{2}) - m_n)\right)}{\sigma} = Q\left(\frac{m_f - m_n}{2\sigma}\right) \quad (5)$$

From the measured i.i.d. value sets we need to determine if they have any faulty sensors. This can be shown from equation (9) that if the correlated sets can be distinguished from the mean values then it has a low probability of error due to sensor faults, as sensor faults are not correlated. Using the statistical analysis package R, we determine the correlated matrix of the sparse sensor outputs as shown This can be written in a compact matrix form if we observe that for this case the co-variance matrix is diagonal, this is,

$$\Sigma = \begin{pmatrix} \rho_1 & 0 & .. & 0 \\ 0 & \rho_2 & .. & 0 \\ : & : & \searrow & : \\ 0 & 0 & .. & \rho_d \end{pmatrix} \quad (6)$$

The correlated co-efficient are shown matrix (11) the corresponding diagonal elements are highlighted. Due to overlapping reading we see the resulting matrix shows that S_1 and S_2 have higher index. The result sets is within the desired bounds of the previous analysis using DWT, which are shown in Figure 2(a) 2 (b)and 2(c). Here we not only prove that the sensor are not faulty but also report a lower bound of the optimal correlated result

Sensors	S_1	S_2	S_3	S_4
$i.i.d._1$	2.7	0	1.5	0.8
$i.i.d._2$	4.7	1.6	3	1.8
$i.i.d._3$	6.7	3.2	4.5	2.8

TABLE II
Sparse representation of sensor values.

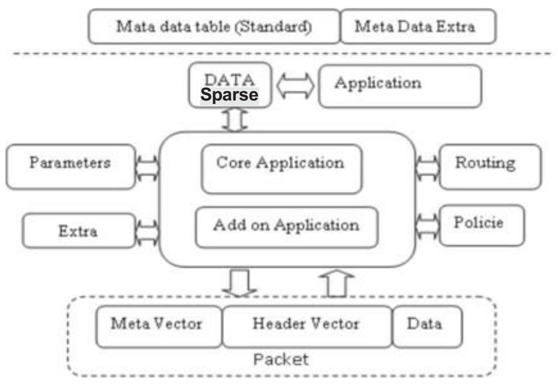


Fig. 3. TestBed deployment for large sensor network using cross-layer STACK.

sets, that is we use S_2 as it is the lower bound of the overlapping ranges.

$$\Sigma = \begin{pmatrix} \overrightarrow{4.0} & 3.20 & 3.00 & 2.00 & 2.00 & 1.60 & 1.5 & 1.0 \\ 3.2 & \overrightarrow{2.56} & 2.40 & 1.60 & 1.60 & 1.28 & 1.20 & 0.80 \\ 3.0 & 2.40 & \overrightarrow{2.250} & 1.50 & 1.50 & 1.20 & 1.125 & 0.75 \\ 2.0 & 1.60 & 1.50 & \overrightarrow{1.00} & 1.00 & 0.80 & 0.75 & 0.5 \\ 2.0 & 1.60 & 1.50 & 1.00 & \overrightarrow{1.00} & 0.80 & 0.75 & 0.5 \\ 1.6 & 1.28 & 1.20 & 0.80 & 0.80 & \overrightarrow{0.64} & 0.60 & 0.4 \\ 1.5 & 1.20 & 1.125 & 0.75 & 0.75 & 0.60 & \overrightarrow{0.5625} & 0.375 \\ 1.0 & 0.80 & 0.750 & 0.50 & 0.50 & 0.40 & 0.375 & \overrightarrow{0.250} \end{pmatrix} \quad (7)$$

The sensor data are correlated due to variations in deployment they are difficult to calibrate. The pre-processing of data will needs to correct the coefficients before applying the fusion function.

IV. TestBed Protocol STACK Design

The majority of the sensor data analysis for Table 2 uses MATLAB 7.8 version running on VISTA and the system component for routing protocols use a network protocol stack which is highly customized. The model allows to have some realistic hardware and also pass real-time messages from external events to virtual nodes for testing scalability of the network and wireless channels.

A. System Components

Sensor networks due to its constrained resources such as energy, memory, and range uses a cross layer model for efficient communications. The cross layered that model as shown in Figure 3, which comprises of the system components includes pre- processing, and routing, to accomplish sensor measurements and communications with sensor nodes. Cross layered based routing protocols use different OSI layers to do multi-hop communications. These carried out using 802.15.4

MAC standards and simulated using an event driven system. The cross-layer measurement metrics include protocol overheads, data payload overheads and MAC duty-cycling. Simulation kernel allows emulating accurate battery models for power-aware analysis and the wireless fading model realistically calculates the ideal model without signal losses with signal losses due to collision at the MAC layers. The modules can be independently designed and implemented to realize multi-hop protocols and power-aware MAC optimization with higher link quality for WSN needs.

B. Application Architecture

Application modules contain implementing distributed algorithms, which allow further power-aware optimization and help scale the number of nodes. The data collected by these applications and post processing of sensed data. Typically these functionalities can be categorized as real-time database and efficient query for specific resource limitations. The design of in-network processing and carried out using architecture call INSPIRE-DB [1]. The application models can use rich graphic capabilities to display sensor events and allow higher level distributed optimizations, which further provides higher bounds on the measured power-aware limitations of the routing algorithms.

C. Layer free architecture for cross-layer (Intra-layer) design

One of the major problems with cross layer approach is management of it and interoperabitly. Intra-layer cross layer design provides effective performance, but it causes loss of layered architecture modularity. If the device wants to communicate with other device or gateways, it should have proposed Intra-layer stack. Further, it is difficult to develop different stack for different application. So architecture or model should be such that only add on application require to control application to control the behavior of communication. In [8] author introduce CLAMP, can be used for various application with only requirement of add on application that control the behavior of parameter in the packet. But with intra layer cross layer designing it can fail, and require different stack designing for various application. In layered architecture all the parameter and information remain in header, only varies in term of number of parameter and its position in different stack. So all different stack required different management, which causes inter-operability issue and have to design from scratch. If we look at different stacks, only variation is presence or absence of parameters and its position. If we design an architecture such a way that handles parameter presence and its position, we can solve the interoperability, management and different stack designing issues. We proposed such layerless architecture, which is abstracted from layer architecture shown in Figure 3.

1) Parameter: These are the parameters that required by different layers and application for tuning and get optimum parameter for given condition. In [8] author provide the parameter list that can be used in this module.

2) Data: Original payload (Application layer payload).

3) Policies: Define the different policies & parameter used by system to generate the packet.

4) Meta vector: It is a binary vector that suggests that the header parameter list $\frac{0for(absent)}{1for(present)}$, that present in Header vector. It is in sequence according to Meta data table.

5) Header Vector: Contain parameter value that is present in Meta vector.

6) Routing: Routing modules provide forward destination, maintenance and generation of routing table.

7) Meta data: Contain the all possible parameter list present in packet header & its position in Meta vector. It should be standard for all devices.

8) Meta data extra: If any extra parameter used other than standard, it will be kept into this module.

9) Packet: Meta vector, header vector and data combine to make a packet.

10) Core Application Module: This module generates the packet, receive the packet and forward the packet using above defined modules and parameter.

11) Add on Application module: This module optimized the parameter values according to application and changes the parameter value used by core application to the packet. Processing of data is application dependent, such as event driven or polled data processing. The application layer is programmable at the top level keeping energy efficient crosslayer balanced and efficient.

V. Acknowledgements

This work was supported by NAP of Korea Research Council of Fundamental Science & Technology.

References

[1] INSPIRE-DB: Intelligent Networks Sensor Processing of Information using Resilient Encoded-Hash DataBase. 2010 Fourth International Conference on Sensor Technologies and Applications.

[2] Dror Baron, Marco F. Duarte, Michael B.Wakin, Shriram Sarvotham, and Richard G. Baraniuk. Distributed Compressive Sensing. In Proc: Pre-print, Rice University, Texas, USA, 2005.

[3] Sarita V. Adve, Kourosh Gharachorloo. Shared Memory Consistency Models:A Tutorial

[4] Richard R. Brook, and S. S. Sitharama Iyengar. Robust Distributed Computing and Sensing Algorithm, ACM, 1996.

[5] Nancy A. Lynch. Distributed Algorithms. Publisher Morgan Kaufmann 1996.

[6] Arne Jensen and Anders la Cour-Harbo. Ripples in Mathematics, Springer Verlag, 2001. 246 pp. Softcover ISBN 3-540-41662-5.

[7] Digital Signal Processing with Field Programmable Gate Array, UWe Meyer-Baese, Springer, May 2001.

[8] Sajjad Ahmad Madani, Stefan Mahlknecht, Johann Glaser, CLAMP: Cross LAyer Management Plane for low power wireless sensor networks. Sajjad Ahmad Madani, Stefan Mahlknecht and Johann Glaser. CLAMP: Cross Layer Management plane for low power wireless sensor networks. In Fifth international workshop on frontiers of information technology, FIT2007. 17.-18. December 2007.