Adaptive In-Network Processing for Bandwidth and Energy Constrained Mission-Oriented Wireless Sensor Networks

Sharanya Eswaran
Pennsylvania State University

Matthew P. Johnson
City University of New York

Archan Misra
IBM T.J. Watson Research Center

Thomas La Porta
Pennsylvania State University

Abstract

In-network processing, involving operations such as filtering, compression and fusion, is a technique widely used in wireless sensor networks (WSN) for reducing the communication overhead. In many tactical stream-oriented WSN applications, especially in military scenarios, both link bandwidth and node energy are critically constrained resources. For such applications, in-network processing itself imposes non-negligible computing cost. In this work, we have developed a unified closed-loop control framework that permits distributed convergence to both a) the optimal level of compression performed by a forwarding node on sensor streams, and b) the best set of nodes where the operators of the stream processing graph should be deployed. In particular, we enhance the Network Utility Maximization (NUM) paradigm, where resource sharing among competing WSN applications is modeled as a form of distributed utility maximization. We also show how the generalized model can be adapted to more realistic cases, where the in-network operator may be varied only in discrete steps, and where a fusion operation cannot be fractionally distributed across multiple WSN nodes.

1. Introduction

Many uses of wireless sensor networks (WSN) focus on long-running applications, operating over relatively low rates of discrete-event data. For such scenarios, the WSN is principally energy-constrained. Given that communication costs dominate computing costs (13) for relatively simple event-processing operations (such as averaging or finding the maximum of periodic temperature readings), in-network processing (e.g., (9)) has been proposed as a means to increase the network operational lifetime by reducing the volume of data transmitted to the sink. In this approach, an application is modeled as a graph of stream operators, overlaid on the physical WSN topology.

Our focus is on stream-oriented WSN applications, where several of the implicit assumptions above do not hold. In particular, many military scenarios utilize WSNs for relatively shorter-duration tactical missions, with the sensor data coming from a set of relatively high-data rate streaming sources, such as video sensors, acoustic arrays and short-range radar feeds. For such environments, bandwidth is a critical shared resource, and congestion control algorithms (e.g., (18)) must be employed to effectively share the wireless link bandwidth among the competing missions. Moreover, the in-network operators for such stream-oriented data typically comprise more sophisticated DSP-based operations (e.g., MPEG compression or wavelet coefficient computation); it has been demonstrated (11) that the computational energy footprint for such stream-oriented operations cannot be ignored. Accordingly, the application of in-network processing to such WSN applications must consider that the WSN is both bandwidth and energy constrained and that the energy cost consists of both communication and computing overheads.

While limited past work has considered the joint optimization of computing and communication costs (both bandwidth and energy) (15; 11), they do not consider the possibility of variable quality or lossy in-network processing and the resultant impact on the fidelity or accuracy of the WSN application. In our generalized model, in-network processing may be viewed as a tuning knob, with higher levels of in-network processing (e.g., higher compression or coarser quantization) resulting in higher information loss for (or lower utility to) the application, but providing the benefit of reduced network bandwidth consumption. This introduces a non-linear tradeoff in the energy costs – in general, higher levels of processing (e.g., more sophisticated compression techniques) lead to reduced transmission energy overheads, but a not-necessarily proportional increase ((14)) in the computational energy.

In this paper, we first introduce and develop a distributed, closed-loop control framework that computes the optimal level of compression performed by a forwarding node on sensor streams, taking into account both energy and bandwidth constraints. In particular, we apply the Network Utility Maximization (NUM) paradigm, pioneered in (2; 3) for congestion control, which models resource sharing among competing WSN applications as a form of distributed utility maximization. Initially, the physical location of the stream...
operators is assumed to be pre-specified. The variable energy costs of communication and computation are represented as additional constraints in the NUM optimization framework, which is then used to find the joint optimal allocation of sensor rates and the level of in-network compression that maximizes the overall system utility. Subsequently, we generalize the problem further to make the physical location of the operator graph components as another decision variable, i.e., we use NUM-based optimization to additionally determine the WSN nodes where various in-network operations are performed. An important aspect of our work is that we not only develop the distributed optimization algorithm, but also work out the intricate details of the corresponding distributed protocol and demonstrate its ability to efficiently improve system utility in practical wireless environments.

For mathematical tractability, the distributed optimization problem is initially developed using an idealized model, where a) the quality level of in-network compression is assumed to be a continuously valued-variable and b) an individual operator is allowed to be fractionally instantiated at multiple WSN nodes. The idealized model can be viewed as a continuous approximation of real-life scenarios, which are characterized by mixed-integral constraints. We then show how the above two idealized assumptions can be relaxed, generalizing our NUM technique to scenarios where the quality of in-network processing may be varied only in discrete steps, and where an operator may be instantiated only on a single node. Simulation-based studies, using a packet level protocol implementation of the generalized NUM framework, are then used to demonstrate how “adaptive operator placement” and “variable-quality in-network compression” can together result in a significant improvement (as much as 30%) in overall mission utilities.

The rest of this paper is organized as follows. In Section 2, we use an illustration of a stream-based WSN operator processing graph to illustrate the unique aspects of our problem. Section 3 then briefly summarizes other related work on this topic. Section 4 presents the mathematical models and corresponding distributed solution and protocol details for the case where the location of the operator graph components are specified a-priori. Subsequently, Section 5 extends the solution to consider the problem of optimal placement of operators of the processing graph, while Section 6 describes how the base algorithms are modified to incorporate the real-life integral constraints. In Section 7, we present our simulation results. Finally, Section 8 concludes the paper.

2. The General Framework of Variable-Quality In-Network Processing and Dynamic Operator Placement

To understand the various conflicting objectives and issues with in-network processing of sensor data streams, consider the scenario depicted in Fig. (1), which involves the use of three sensors (two video and a radar) and three different missions. The surveillance mission has high utility and is quite intolerant of loss in the raw sensor data; the detection mission requires only correlation information from the various sensor feeds to detect an event; the feature extraction mission requires the video images as input, however it can successfully extract features even if they are of relatively low resolution. In this scenario, each mission may be associated with a different operator graph, with the data rates to the ‘sink’ of each operator graph ultimately affecting the mission’s utility. The sink data rate depends on two separate variables—a) the source data rate from its relevant sensors, and b) the fractional reduction in data due to the application of individual in-network operators. To generalize this concept, we classify in-network operators into two logical types:

Compression: The downstream transmission rate of most stream-oriented data can be reduced by the application of appropriate compression algorithms, both lossless and lossy. For example, an MPEG-4 (or higher standards, such as MPEG-21) video stream can be compressed to varying data rates. From a logical standpoint, compression may be viewed as a special class of in-network operators that can be performed independently at every forwarding node; moreover, compression changes the quality (rate) of the output data, but not the data type.

Fusion: In contrast to compression, fusion may be viewed as a process of either combining or correlating data from multiple separate streams and/or altering the ‘type’ of a single data stream. An example of ‘type’ alteration involves the processing of a single audio stream to extract only the ‘talk spurs’ from the signal. We note here that even a mission (sink) can perform fusion of streams.

We thus define an operator graph as a set of fusion operators. An operator placement algorithm essentially maps each of the nodes of the operator graph to a subset of the WSN nodes in the network; compression may then be viewed as an implicit data reduction operator permitted even between
two consecutive components of the operator graph. For technical reasons (20), we limit our analysis to operator graphs that are “tree-structured”, rather than the more generalized directed acyclic graph (DAG), i.e., we do not allow recombination of data traversing multiple parallel paths at a common downstream operator. Otherwise, the analysis becomes difficult, as one needs to track not just the cumulative rate of a particular flow, but also each path taken by the flow.

Initial approaches for in-network stream processing focused on the intelligent placement of operators largely from the standpoint of energy efficiency, typically placing more selective operators closer to the data sources (e.g., (9)). Such approaches do not directly factor in the energy constraints on individual nodes, and the fact that in-network processing may consume significant computational energy.

In (20), resource-aware in-network processing has been studied, where the behavior of each individual operator is assumed to be immutable, i.e., each operator is characterized by a fixed ratio between its output data rate and the data rates of its input streams, and the location of the operator graph elements is pre-defined as well. There is no explicit consideration of the possibility of variable-quality compression at intermediate nodes; accordingly, the only ‘free variables’ are the source sensor rates, since they implicitly impact the bandwidth constraints at each individual node. Separately, (11) considered the communication+computing cost constraint, but without consideration of in-network processing. Under these models of fixed operator location and processing behavior, the problem of utility-based rate control reduces to the ‘traditional’ problem of source rate control with capacity constraints on individual links/nodes, with the added complexity due to the non-conservation of flow-rates at the nodes performing in-network processing.

An obvious extension of this framework is to allow the placement of the stream processing operators to be a decision variable as well. Prior work on fusion operator placement ((20; 5; 6; 8; 9; 10; 12)) treats it as a stand-alone problem, and does not consider the interaction with variable data compression performed at intermediate nodes. For example, in (20), the operators are placed on a minimum-cost path determined by a distributed process similar to Dijkstra’s algorithm. However, using only local neighbor-level information is not sufficient when variable quality compression is introduced, because a mission’s utility ultimately depends on the data rate that it receives, and this is obviously affected by any rate modification performed at intermediate nodes. As we shall see, the incorporation of variable quality compression at intermediate nodes complicates the process of fusion operator placement, as the placement and compression decisions become coupled. Our work is the first to study the consequences of such variable quality compression. In particular, compression allows a degree of variability that is usually absent in a fusion operator; based on the energy constraints of individual nodes, it may be better to perform fusion at a downstream node and only perform “moderate” compression at an upstream node (to satisfy the upstream node’s energy or capacity constraint).

Prior work (such as (11)) also assumes a relatively simple tradeoff between computational and communication energy overheads, with both being essentially a scalar multiple of the incoming stream data rate. Variable compression, however, introduces a more complex, non-linear tradeoff between the computational and communication costs, that is a function of the incoming data rate and the “quality” of compression that is performed. In particular, many compression algorithms (both lossy and lossless) are characterized by a non-linear energy-vs-compressibility curve, with the energy required for compression increasingly dramatically when the ratio of output to input data rates falls below a certain threshold (19). Accordingly, our optimization problem must not only consider the impact of variable quality compression on the operator placement problem, but also on the energy constraints at each node.

Based on the above discussion, the key new aspects of our problem formulation can be summarized as follows:

1. We consider the impact of variable quality compression of sensor streams, potentially performed by all forwarding nodes, on the capacity constraints and factor in the non-linear relationship between computational and communication energy overheads.
2. We also explicitly factor in the effect of such variable quality compression on the operator placement problem, and develop a solution that jointly selects both the location of fusion operators and the degree of compression that maximize cumulative system utility.

To solve this problem, we shall develop a NUM-based optimization framework and a fully-distributed protocol that seeks to jointly optimize the following free variables: i) Source Rate, x: the rate at which each sensor source transmits data, ii) Compression Factor, l: the level of compression, i.e., ratio of output rate to incoming rate, taking place at each forwarding node, and iii) Operator placement: the optimal node locations at which fusion operations take place.

### 3. Other Related Work

The classical NUM framework ((2; 3)) was recently extended in (18) to a more general WSN environment, where individual missions derive their utility from a composite set of sensors and intermediate nodes use link-layer multicast to forward sensor data downstream to multiple subscribing missions. In this WSN-centric model, the optimization problem is formulated as: $SENSOR(U, L):$

$$\text{maximize } \sum_{m \in M} U_m(X_m) \text{ subject to } \sum_{\forall(k,s) \in q} \frac{x_{k,s}}{c_{k,s}} \leq 1,$$

∀q ∈ Q, where Q is the set of all cliques in the conflict graph of the WSN; $U_m(X_m)$ represents the utility function.
of mission \( m \) (\( M \) being the set of all missions) as a function of the \( S \)-dimensional vector of rates associated with the set of sensors \( S \) and \( c_{k,s} \) is the transmission rate used by node \( k \) during the link-layer broadcast of the data from sensor \( s \); please see (18) for details. Based on this new model, a sensor (source) \( s_1 \) adapts its rate as:

\[
\frac{\partial}{\partial t} x_1(t) = \kappa \left( \sum_{m \in Miss(s_1)} w_{ms_1}(t) - x_1(t) \right) \sum_{\forall q \in f low(s_1)} \sum_{\forall (k,s) \in q} \frac{\mu_q(t)}{c_{k,s}}
\]

where \( \mu_q(t) \) (the ‘cost’ per bit charged) by each forwarding clique, is given by

\[
\mu_q(t) = \left( \sum_{(k,s) \in q} x_k(t) \right) \sum_{\forall (k,s) \in q} \frac{1}{c_{k,s}} - 1 + \varepsilon / \varepsilon^2
\]

Each mission (sink) adapts its ‘willingness to pay’ terms \( w_{ms} \) (these terms represent pure Lagrangian variables; there is no actual pricing or monetary exchange) for sensor \( s \) based on the source rates and its own utility function \( U_m(.) \), according to

\[
w_{ms}(t) = x_s(t) \frac{\partial U_m}{\partial x_s}.
\]

The notion of utility-based adaptation under in-network stream processing was first explored in (15) for wired network, where each sensor flow is assumed to pass through an arbitrary processing graph, with each operator on the graph performing a fixed fractional reduction (or increase) in the output rate. A utility-driven gradient-based algorithm was presented to adjust both the source data rate and the fraction of traffic routed over multiple paths in the query graph, so as to maximize the cumulative utility.

In the absence of any constraints on the total power consumption at a node, the problem of optimal in-network processing and rate adaptation decomposes into the multi-rate multicasting problem. This problem was studied for multi-hop wireless networks in (16), where a back-pressure based solution was developed.

In (19), Yu et al. develop a data gathering algorithm with tunable compression for communication and computation efficiency. While we share the same motivation, unlike (19), we develop a utility-based rate control model with joint congestion control for a WSN consisting of multiple competing missions that have different utilities.

4. The Network Model and The Optimization Problem

We first explain the process by which nodes select the optimal level of stream compression, assuming that the positions of the components of the operator graphs are pre-specified.

4.1 Assumptions

Our NUM-based formulation and solution makes the following assumptions:

- The routes of the various sensor flows are fixed and provided a priori. Each sensor’s data flows over a multicast tree to its set of subscribing missions.
- A fused stream cannot be subsequently disaggregated; accordingly fusion of two streams at a node is possible only if all downstream subscribers (for each of the two sensors) require the same fused information. Except at the nodes where fusion is performed, no other nodes can distinguish between a ‘raw’ or fused data stream.
- Each sensor’s flow is completely elastic, i.e., each node can adjust its transmission rate \( x_s \) by any arbitrary amount, as long as \( x_s > 0 \).
- The computational power required for compression increases with an increase in the compression factor (i.e., a decrease in the \( \text{Outputrate} / \text{Inp rate} \) ratio).
- A fusion or compression operation performed by an intermediate node is applied identically to the flow on each of the outgoing links, i.e., all the next-hop receivers in the multicast tree.

4.2 The Model

Our generic NUM model assumes a mission-based WSN environment, where each mission’s utility may be a joint function of the rate that it actually receives from multiple sensors. Let the \( i \)th mission be denoted as \( m_i \), let \( M \) be the total set of missions, and \( S \) the total set of sensors. Furthermore, for any mission \( m \), let \( set(m) \) be the set of sensors that are sources for \( m \) (i.e., contribute to the utility \( U_m(.) \)); conversely, for any sensor \( s \), let \( Miss(s) \) denote the set of missions subscribing to this sensor’s data. Let \( path(s,m) \) be the set of nodes along the path from source \( s \) to a mission \( m \). The utility of the \( m \)th mission is a function of the rate at which it receives data, denoted as \( U_m(\{x_{s,k}^{\text{rate}}\}_{s \in set(m)}) \). As in (2; 18), \( U(.) \) is assumed to be a jointly-concave function of the rates of all incoming flows. Table 1 lists the common mathematical symbols used in this paper.

The key feature of our WSN operational model is to permit each intermediate node to perform a ‘variable level of compression’, denoted as \( l_{k,s} \) (where \( 0 < l_{k,s} \leq 1 \)), that effectively alters the rate of a flow that is transmitted at node \( k \) and originated at source \( s \). \( l_{k,s} \) determines the ratio of the outgoing flow rate to the incoming flow rate for sensor \( s \) at node \( k \), i.e., \( l_{k,s} = \frac{\text{Outputrate}_{s,k}}{\text{Inp rate}_{s,k}} \). Although we focus here on stream compression, the model can be easily generalized to include decompression, by allowing \( l_{k,s} \) to lie between \( (0, \infty) \). The variable compression level, \( l \), effectively acts as a ‘tuning knob’, allowing an intermediate forwarding node to modify the outgoing data rate in a manner that balances its competing computational and communication energies, and satisfies the capacity constraints. Intuitively, a congested network (with many high rate flows) benefits from more aggressive compression, as the reduction in the link utilization permits other competing flows to transport a larger volume of sensor traffic. Conversely, a network operating at low link utilization should have little need for compression (unless its transmission energy cost is too high), as higher in-network compression always increases the computational expense at
an intermediate node and also reduces the flow’s utility to the receiving mission. The centralized model for this problem of utility maximization with adaptive in-network processing can be written as: NUM-IP(U,C,P):

\[
\text{maximize } \sum_{m \in M} U_m(\{x_{s}^{rec}\}_{s \in \text{set}(m)}) - \delta \sum_{\text{nodes},k} P_{k_{\text{tot}}}. \\
\text{subject to}
\]

1) Capacity Constraint:
\[
\sum_{\forall(k,i) \in q} \frac{x_{out}(i,k)}{c_{ki}} \leq 1, \forall q \in \text{set of cliques}, Q(2)
\]

2) Energy Constraint: \( P_{k_{\text{tot}}} \leq P_{k_{\text{max}}}, \forall \text{nodes}, k (3) \)
where \( P_{k_{\text{tot}}} = P_{k_{\text{rec}}} + P_{k_{\text{trans}}} + P_{k_{\text{comp}}} \),
\( 0 \leq \delta \leq 1 \) and \( x_{i}, x_{out} \geq 0 \forall i \)

The objective is to maximize the total utility of all missions, subject to an “energy” penalty function \( \delta \sum_{\text{nodes},k} P_{k_{\text{tot}}} \), the additional power term ensures a unique solution (e.g., when the same compression can be performed by any one of the nodes in a forwarding chain, the penalty function biases the solution to having the compression performed at the most upstream node) by creating a convex optimization objective. \( \delta \) (between 0 and 1) determines the weighting given to power consumption (vs. utility); in general, the penalty function can be the sum of any convex functions of \( P_{k_{\text{tot}}} \). The capacity and energy constraints are explained as follows:

- **Capacity Constraint**: As in (18), the capacity constraint can be expressed through the formation of a transmission-based conflict graph (CG) (similar to the approach in (17)), where each vertex of the CG refers to a \((i,k)\) transmission (the transmission of the \(i^{th}\) flow by node \(k\)) and an edge between two vertices in the CG imply that the two transmissions interfere with each other. A sufficient capacity condition is that, in any maximal clique, the sum of the air-time fractions of all the transmissions must not exceed unity, as expressed by Eq. (2).

- **Energy Constraint**: The energy constraint in Eq. (3) states that the total power consumed at a node \(k\) due to data reception \( P_{k_{\text{rec}}} \), transmission \( P_{k_{\text{trans}}} \), and computation including both compression and fusion (if a fusion node \(k\) is a fusion point, there is an additional computational cost incurred by the fusion process. Without loss of generality, we assume that this cost is proportional to the rate of the fused flow, and we also assume that the cost per unit rate is the same for compression and fusion.

### 4.3 Distributed Solution to the Optimization Problem

In order to solve this optimization problem in a distributed manner, we derive an iterative, gradient-based, distributed solution for the model shown in Eq. (1)-(3), similar to the derivation in (18; 2). We first make the problem unconstrained.

### Table 1: Most Common Mathematical Symbols

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( M )</td>
<td>Total number of missions in the network</td>
</tr>
<tr>
<td>( S )</td>
<td>Total number of sources (sensors)</td>
</tr>
<tr>
<td>( \text{set}(m) )</td>
<td>Set of sources that are used by mission ( m )</td>
</tr>
<tr>
<td>( M_{\text{iss}}(s) )</td>
<td>Set of missions subscribing to data from sensor ( s )</td>
</tr>
<tr>
<td>( U_m(.) )</td>
<td>Joint utility function of mission ( m )</td>
</tr>
<tr>
<td>( c_{ks} )</td>
<td>Transmission rate used by node ( k ) during the link-layer broadcast of the flow from source ( s )</td>
</tr>
<tr>
<td>( x_{s}^{rec} )</td>
<td>The rate at which a mission receives flow ( s )</td>
</tr>
<tr>
<td>( (k, s) )</td>
<td>Transmission of the flow that originated from ( s ) by a node ( k )</td>
</tr>
<tr>
<td>( x_{in}(s, k) )</td>
<td>Incoming data rate at node ( k ) for flow ( s )</td>
</tr>
<tr>
<td>( x_{out}(s, k) )</td>
<td>Outgoing (i.e., forwarded) data rate for flow ( s ) from node ( k )</td>
</tr>
<tr>
<td>( \ell_{k,s} )</td>
<td>Quality of compression ((0, 1)) applied by node ( k ) on flow ( s )</td>
</tr>
</tbody>
</table>

The maximum permitted power consumption at node \( k \), total power consumed at node \( k \) due to data reception, total power consumed at node \( k \) due to data transmission, total power consumed at node \( k \) due to data processing (compression and/or transformation), power consumed per bit of transmitted data at node \( k \), power consumed per bit of transmitted data at node \( k \) during compression at node \( k \), set of nodes from \( s \) to \( k \), and the multicast flow from \( s \) to all its missions.
strained by using Lagrangian multipliers as: 
\[
\begin{align*}
\text{maximize} & \quad \sum_{m \in M} U_m(x_{s}^{rec})_{s \in \text{set}(m)} - \delta \sum_{\text{nodes,}k} P_{k}^{\text{tot}} - \\
& \sum_{\forall \text{cliques},q} \mu_q \left( \sum_{\forall (k,s) \in q} \frac{x_{\text{out}}(s,k)}{c_{ks}} - 1 \right) - \sum_{\text{nodes,}k} \eta_k \left( P_{k}^{\text{tot}} - P_{k}^{\text{max}} \right)
\end{align*}
\]

where \( \mu_q \) and \( \eta_k \) are the Lagrangian multipliers. Using the first-order necessary conditions for the gradients with respect to \( x_s \) and \( l_{k,s} \), we get the following equations:

\[
\begin{align*}
\frac{d}{dt} x_s(t) &= \kappa * x_s(t) \sum_{m \in \text{Miss}(s)} \frac{\partial U_m}{\partial x_s} - \delta \sum_{\text{nodes,}k} \frac{\partial P_{k}^{\text{tot}}}{\partial x_s} - \\
& \sum_{\forall (k,s) \in q} \mu_q \sum_{\forall (k,s) \in q} \frac{\partial x_{\text{out}}(s,k)}{\partial x_s} C_{ks} - \sum_{\forall \text{Path}(s)} \eta_k \frac{\partial P_{k}^{\text{tot}}}{\partial x_s} \quad (4)
\end{align*}
\]

\[
\begin{align*}
\frac{d}{dt} l_{k,i}(t) &= \kappa_1 * l_{k,i}(t) \sum_{m \in \text{Miss}(i)} \frac{\partial U_m}{\partial l_{k,i}} - \delta \sum_{\text{nodes,}s} \frac{\partial P_{s}^{\text{tot}}}{\partial l_{k,i}} - \\
& \sum_{\forall \text{Flow}(i)} \sum_{\forall (v,i) \in q} \frac{\partial x_{\text{out}}(i,v)}{\partial l_{k,i}} C_{vi} - \sum_{\forall \text{Flow}(i)} \eta_k \frac{\partial P_{s}^{\text{tot}}}{\partial l_{k,i}} \quad (5)
\end{align*}
\]

where, \( \mu_q \) is defined as the shadow cost of congestion charged at each clique \( q \) and is given by 
\[
\mu_q(t) = p_{1,q}(\sum_{\forall (k,s) \in q} \frac{x_{s}(t)}{c_{ks}}) = (\sum_{\forall (k,s) \in q} \frac{x_{s}(t)}{c_{ks}} - 1 + \epsilon)^+ / \epsilon \delta_1
\]

Similarly, \( \eta_k \) is the shadow cost of energy charged at each node \( k \) and is given by
\[
\eta_k(t) = p_{2,k}(\frac{P_{k}^{\text{tot}}(t)}{P_{k}^{\text{max}}(t)}) = (\frac{P_{k}^{\text{tot}}(t)}{P_{k}^{\text{max}}(t)} - 1 + \epsilon')^+ / \epsilon \delta_2
\]

where \( \delta_1 \) and \( \delta_2 \) are constants greater than 0 and \( 0 \leq \epsilon, \epsilon' \leq 1 \).

Eq. (4) provides the algorithm by which the source sensors adjust their rates at each iteration; Eq. (5) shows how at each node, the degree of compression for each flow that the node forwards is varied, in each iteration. We observe the following:

- **Source rate** \( x_s \) depends on the rates at which the downstream nodes forward either this source’s flow directly (when there is no fusion), or any flow derived from this source’s flow (when there is fusion). Similarly, it also depends on the power consumed at all downstream nodes that forward either the source’s direct flow or a flow derived (via fusion) from this source.

- **The compression levels at the forwarding nodes depend on the forwarding rates and power consumption at all downstream nodes that receive this flow** (either raw or fused).

We can show that when the source and forwarding rates are independently adjusted according to Eq. 4 and 5, the network converges at the optimal global utility, with penalties paid for congestion and power consumption.

**Theorem 1.** The strictly concave function given by
\[
\nu(x_s, \{l_{k,s}\}_{k \in \text{set}(m)}) = \sum_{m \in M} U_m(x_{s}^{rec})_{s \in \text{set}(m)} - \delta \sum_{\text{nodes,}k} P_{k}^{\text{tot}} - \\
\sum_{\forall \text{cliques},q} \mu_q \left( \sum_{\forall (k,s) \in q} \frac{x_{\text{out}}(s,k)}{c_{ks}} - 1 \right) - \sum_{\text{nodes,}k} \eta_k \left( P_{k}^{\text{tot}} - P_{k}^{\text{max}} \right)
\]
is a Lyapunov function for the system of differential equations in Eq. (4), (5). The unique values \( x_s \) and \( l_{k,s} \) maximizing \( \nu(x_s, l_{k,s}) \) is a stable point of the system.

**(Proof Sketch).** With a proof similar to (2), we can easily verify that
\[
\frac{d}{dt} \nu(x_s, \{l_{k,s}\}_{k \in \text{set}(m)}) = \sum_{m \in \text{Miss}(i)} \frac{\partial \nu}{\partial x_s} \frac{d}{dt} x_s(t) + \\
\sum_{\text{nodes,}k \forall \text{flows,} s \in \text{set}(m)} \sum_{\forall \text{flows,} s \in \text{set}(m)} \frac{\partial \nu}{\partial l_{k,s}} \frac{d}{dt} l_{k,i}(t)
\]
is strictly positive when \( x_s \) and \( l_{k,s} \) are not the unique maximizing points of \( \nu \) and zero otherwise. This proves that the function \( \nu \) is a Lyapunov function for the system given by Eq. (4), (5).

**4.4 Protocol Level Implementation of the NUM Algorithm**

The biggest challenge in building a fully-distributed and localized protocol for this model arises from the presence of fusion operators at specific intermediate WSN nodes. The stream that a mission receives is now obtained by fusing one or more flows from \( \text{set}(m) \) according to a series of operators, as defined by the operator graph. An individual operator \( f \) can be viewed as a function that takes as input the rates of the flows to be fused, and gives as output the rate of the resulting fused flow. Hence, the utility of a mission \( m \) is a joint function of all \( x_i^{rec} \), where \( i \in \text{set of flows received at } m \), with some of these flows being ‘raw’ flows (potentially compressed) from the corresponding sensor, with other flows being ‘derived’ at intermediate nodes (which act as the ‘source’ for the derived flow) due to the application of a fusion operator. While Eq. (4) referred only to rate adjustment at the ‘raw’ sources (i.e., sensors), the flow \( i \) in Eq. (5) may refer to either a raw or derived flow. Accordingly, Miss \((i)\) now refers to the set of missions that received data flow of type \( i \) (raw or derived); Flow \((i)\) similarly is the multicast forwarding path of data from node \( i \) (which may be a source or a fusion point). With these modifications, the distributed formulation in Eq. (4), (5) is sufficient for deriving the optimal ‘raw’ and ‘derived’ data rates.

From a protocol perspective, however, the end-to-end feedback mechanism used in (2; 18), whereby the sinks simply convey their willingness to pay to the source sensors, is
no longer sufficient. The difficulty lies in the inability of a sink to compute its ‘willingness to pay’, due to the presence of fusion points on the data path from the source sensor to itself. For example, if a stream from source \( s \) is transformed twice by operators \( f \) and \( g \) before reaching a mission \( m \), the mission is unable to compute the term \( \frac{\partial U}{\partial x_s} \), because all it knows is the rate of the stream of type \( \langle g \cdot f \rangle \), which contributes to its utility; it is unaware of the precise source sensor rate. Here \( g \cdot f \) refers to the composition function of the form \( g(f(x_s, \ldots)) \). The solution in this case is to use the “chain rule” for partial derivatives and compute \( \frac{\partial U}{\partial x_s} \) as

\[
\frac{\partial U}{\partial x_s} = \frac{\partial U}{\partial x_m} \cdot \frac{\partial x_m}{\partial x_s} \cdot \frac{\partial f}{\partial x_s} \cdot \frac{\partial g}{\partial x_s} \cdot \frac{\partial U}{\partial x_s},
\]

where the fusion point for \( g \) and \( f \) provide the second and third terms, respectively.

Accordingly, in our modified NUM-INP protocol, the forward path carries only the data, but no other meta-data. Nodes propagate the congestion and energy costs as meta-data in signaling messages carried on the reverse forwarding path; upstream nodes use these messages to compute the compression levels, fusion fractions and the “source” rates (raw or derived) for the next iteration.

For each stream \( r \) that a mission \( m \) receives, it sends a feedback (periodically), to the node that forwarded this stream; this process is performed iteratively as the feedback travels to upstream nodes. If an intermediate node was a branching point on the multicast forwarding tree, it collects the feedback from all its child nodes and combines them into a single feedback message, consisting of: i) A marginal utility \( MU \) field, where the mission enters its marginal utility with respect to the current flow rate, \( (\frac{\partial U}{\partial x_s}) \); this is used for computing the ‘willingness-to-pay’ according to the chain rule. ii) A 4-tuple, consisting of the fields flow name (the ID of the ‘flow’), rate information (RI) (the rate at which the mission receives the flow, power information (PI) (the energy cost attributed to this flow) and congestion information (CI) (the congestion cost at all the cliques that this node belongs to, normalized by this flow’s transmissions). The 4-tuple is part of a table called the Feedback Information Table, maintained by each node.

When a forwarding node \( A \) receives a feedback message for flow \( f \) from a downstream node, it adds its own energy cost for \( f \) to the PI field (i.e., \( PI = PI + (\eta_A + \delta)P_{TA}(f) \)) and its own congestion cost for \( f \) to the CI field (i.e., \( CI = CI + \sum_{\forall q \in (A,i \not\in C_A)} \frac{\mu_q}{z_{qi}(f_{A,i})} \)) before passing the feedback message to its upstream neighbor. The RI field contains the flow rate at the mission; it is filled by the node that transmits directly to the sink and is not altered by other forwarding nodes. However, if \( A \) is also a fusion point, that creates the ‘virtual’ flow \( f \), all the fields in the table are further multiplied by the term \( \frac{\partial f}{\partial x_m}(f_{A,i}) \cdot x_{in}(s, A) \cdot x_{out}(f_{A,i}) \) before propagating the feedback upstream. The source nodes and the forwarding nodes compute the values of \( x \) and \( l \) for the next iteration according to Algorithms (1) and (2), respectively. Fig. (2)-(4) illustrate the propagation of feedback and computation of compression level for a simple example.

5. Adaptive Operator Placement

In the previous section, we assumed that the locations of the fusion operators are fixed and given a priori. In this section, we describe how the NUM framework can be enhanced for determining the optimal placement, with a goal to improve the cumulative utility of the missions. Ideally, the communication cost is lowest if a fusion operation takes place close to the sources as possible. However, due to energy constraints, nodes closer to the source may not be able to perform the fusion operation; in such situations, higher utility may be obtained by pushing the operator to a node downstream. Our approach is to integrate operator placement into the NUM framework (in parallel to source rate adaptation and adaptive compression quality), albeit as an “outer” optimization loop that occurs at a slower time-scale.

With the help of an operator graph, the forwarding trees and the mission subscription information, the nodes in a network can determine if they are candidate-locations for a fusion operator. For example, for the simplistic network shown in Fig. (5), where mission \( M \) requires the fused flow, \( f(x_1, x_2) \), the fusion can take place at node A or B or C. We assume that each node runs a preliminary protocol (details of which are not relevant to this work) to determine which fusion operations can be performed at that node. We also assume that the fusion operations can be expressed as functions of the rates of their input flows.

Our approach for joint optimization is to allow all candidate locations to perform fusion on a fraction of the input streams, and transmit the rest as raw streams. This fraction is variable and can be adjusted iteratively. When the network converges, the optimal values of fractions will be reached.

Let \( k \) be a representative candidate node for the fusion operation \( f(x_1, x_2, x_3, \ldots, x_n) \) that fuses flows \( F = \{1, 2, 3, \ldots, n\} \). Let \( \theta_{f,s}^i \) (where \( i \in F \)) represent the fraction (lying between 0 and 1) of the input flow from \( s_i \) that must be fused at node \( k \). The rest of the input flow is passed downstream, where the next candidate node can fuse all or a fraction of it (the mission sink is always a candidate for all fusion operators, and can absorb any residual “unfused” stream data). For the example shown in Fig. (5), node A fuses according to \( f(\theta_{A,s}^1, x_1, \theta_{f,s}^2, x_2) \) and forwards the two input flows and the fused flow, \( f_{low}^A \) at rates \( l_{A,s}^1 (1 - \theta_{f,s}^1) x_1 \) and \( l_{A,s}^2 (1 - \theta_{f,s}^2) x_2 \) and \( f_{low}^A = \sum_{\forall q \in (A,F)} \frac{\mu_q}{z_{qi}(f_{A,q})} \) before passing the feedback message to its upstream neighbor. The RI field contains the flow rate at the mission; it is filled by the node that transmits directly to the sink and is not altered by other forwarding nodes. However, if \( A \) is also a fusion point, that creates the ‘virtual’ flow \( f \), all the fields in the table are further multiplied by the term \( \frac{\partial f}{\partial x_m}(f_{A,i}) \cdot x_{in}(s, A) \cdot x_{out}(f_{A,i}) \) before propagating the feedback upstream. The source nodes and the forwarding nodes compute the values of \( x \) and \( l \) for the next iteration according to Algorithms (1) and (2), respectively. Fig. (2)-(4) illustrate the propagation of feedback and computation of compression level for a simple example.
5.1 Protocol-Level Modifications for Operator Placement

The introduction of adaptive operator placement requires some modifications to the signaling propagated along the reverse forwarding path. We use the relatively simple example network of Fig. (5) to explain the corresponding protocol-level modifications. Mission $m$ receives a fused flow, $f(s_1, s_2)$, and nodes $A$, $B$ and $C$ are all candidates for the fusion. This means that $m$ may receive flows fused at $A$ ($f_A$), at $B$ ($f_B$) and $C$ ($f_C$), and also the raw streams $s_1$ and $s_2$ (if the fusion points do not fuse all the data). Hence, $m$ sends feedback to $C$ with marginal utility as $\frac{\partial U_m}{\partial P_{f_A}}$, where $x_{f}$ refers to the rate of fused flow of type $f$ that is fused at node $k$. $C$, $B$ and $A$ update this message with the 4-tuple feedback information table as shown in Tables (2)-(4). In the tables, $P_{tot}(s)$ denotes the total power consumed from receiving, transmitting and processing a flow $s$; $RI_k(r)$ refers to the RI field in the message from node $k$ corresponding to flow (row) $r$. The same definition holds for $PI$ and $CI$ as well. The entry made at each partial fusion point is referred to as a cumulative entry (cum). As the example makes clear, the presence of multiple candidate fusion points leads to an increase in the number of distinct flows (a larger number of distinct rows in the feedback information table), thereby increasing the signaling overhead. However, as demonstrated in Section 7, the total signaling overhead of NUM can still be kept very low, by performing the $\theta$ adaptation at much slower time scales (e.g., per minute).

At branch points of the forwarding tree, a node simply adds the corresponding values from its child entries and propagates a single feedback message to its upstream parent. The parent node then uses the feedback message to compute both its ‘optimal’ $\theta$ value (as well as its local power and congestion cost); the expressions for computing these are detailed in Appendix A. Here, $FB^k$ refers to the feedback at $k$; if $k$ is a branch-point for a flow, $FB^k$ refers to the combined feedback from all its child nodes; $FB^k(r, c)$ refers to row $r$ and column $c$ in the information table of the feedback message at $k$. We can see from Tables (2)-(4) that only minimal amount of information is signaled; Algorithms (1)-(3) have been devised such that Eq. (4, 5, 6) can be computed precisely from this minimal meta-data and locally available information.

6. NUM Modifications to Address Practical Constraints

For mathematical tractability, the NUM-based technique for “optimal” variable in-network compression and operator placement requires both these processes to be represented as...
Table 2: Feedback Information Table from C to B (Comments are written as "//...")

<table>
<thead>
<tr>
<th>RI</th>
<th>PI</th>
<th>CI</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s_1$</td>
<td>$x_{out}(s_1, C)$ //Rate at which C transmits $s_1$</td>
<td>$\sum q(C, s_1) \in q \mu_q \frac{x_{out}(s_1, C)}{C_{C, s_1}}$</td>
</tr>
<tr>
<td>$s_2$</td>
<td>$x_{out}(s_2, C)$</td>
<td>$\sum q(C, s_2) \in q \mu_q \frac{x_{out}(f_A, f_C)}{C_{C, s_2}}$</td>
</tr>
<tr>
<td>$f^A$</td>
<td>$x_{out}(f_A, C)$</td>
<td>$\sum q(C, f_A) \in q \mu_q \frac{x_{out}(f_A, f_C)}{C_{C, f_A}}$</td>
</tr>
<tr>
<td>$f^B$</td>
<td>$x_{out}(f_B, C)$</td>
<td>$\sum q(C, f_B) \in q \mu_q \frac{x_{out}(f_B, f_C)}{C_{C, f_B}}$</td>
</tr>
<tr>
<td>cum$^1$</td>
<td>$x_{out}(f_C, C) \frac{x_{in}(s_1, C) + \partial f}{x_{in}(f_C, C) + \partial f}$</td>
<td>$\frac{x_{in}(s_1, C) + \partial f}{x_{in}(f_C, C) + \partial f}$</td>
</tr>
<tr>
<td></td>
<td>$\frac{x_{out}(f_C, C) + \partial f}{x_{out}(f_C, C) + \partial f}$</td>
<td>$\frac{x_{out}(f_C, C) + \partial f}{x_{out}(f_C, C) + \partial f}$</td>
</tr>
</tbody>
</table>

Table 3: Feedback Information Table from B to A

<table>
<thead>
<tr>
<th>RI</th>
<th>PI</th>
<th>CI</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s_1$</td>
<td>$R_lC(s_1)$</td>
<td>$C_{lC}(s_1) + \sum q(B, s_1) \in q \mu_q \frac{x_{out}(s_1, B)}{C_{B, s_1}}$</td>
</tr>
<tr>
<td>$s_2$</td>
<td>$R_lC(s_2)$</td>
<td>$C_{lC}(s_2) + \sum q(B, s_2) \in q \mu_q \frac{x_{out}(f_A, f_B)}{C_{B, s_2}}$</td>
</tr>
<tr>
<td>$f^A$</td>
<td>$R_lC(f_A)$</td>
<td>$C_{lC}(f_A) + \sum q(B, f_A) \in q \mu_q \frac{x_{out}(f_A, f_B)}{C_{B, f_A}}$</td>
</tr>
<tr>
<td>cum$^1$</td>
<td>$R_lC(f_B) \frac{x_{in}(s_1, B) + \partial f}{x_{in}(s_2, B) + \partial f}$</td>
<td>$\frac{x_{in}(s_1, B) + \partial f}{x_{in}(s_2, B) + \partial f}$</td>
</tr>
<tr>
<td></td>
<td>$R_lC(f_B) \frac{x_{in}(s_2, B) + \partial f}{x_{in}(s_2, B) + \partial f}$</td>
<td>$\frac{x_{in}(s_2, B) + \partial f}{x_{in}(s_2, B) + \partial f}$</td>
</tr>
</tbody>
</table>

Table 4: Feedback Information Table from A to $s_n$ ($n = 1, 2$) (A sends a feedback each to $s_1$ and $s_2$)

<table>
<thead>
<tr>
<th>RI</th>
<th>PI</th>
<th>CI</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s_n$</td>
<td>$R_lB(s_n)$</td>
<td>$C_{lB}(s_n) + \sum q(A, s_n) \in q \mu_q \frac{x_{out}(s_n, A)}{C_{A, s_n}}$</td>
</tr>
<tr>
<td>cum$^n$</td>
<td>$R_lB(f_A) \frac{x_{in}(s_n, A) + \partial f}{x_{in}(s_n, A) + \partial f}$</td>
<td>$\frac{x_{in}(s_n, A) + \partial f}{x_{in}(s_n, A) + \partial f}$</td>
</tr>
<tr>
<td></td>
<td>$R_lB(f_B) \frac{x_{in}(f_A, A) + \partial f}{x_{in}(f_B, A) + \partial f}$</td>
<td>$\frac{x_{in}(f_A, A) + \partial f}{x_{in}(f_B, A) + \partial f}$</td>
</tr>
</tbody>
</table>
continuous variables. These assumptions are likely to be violated in practice. We now describe how the NUM algorithm can be modified to address both these practical limitations.

Discrete Compression Levels: Most of the commonly used compression techniques provide for multiple, but discrete, compression levels (with higher compression factors requiring higher computational complexity). For instance, gzip provides 9 levels of compression, JPEG allows a range of 0 to 100 levels, and MP3 allows compression ratios ranging from 12:1 to 10:1. The discontinuity arising from such integral choices prevents the direct application of NUM’s gradient search techniques; in fact, the problem of determining the optimal choice of compression levels from a discrete set (for multiple competing missions) is shown to be NP-hard in Theorem (2). Our NUM-based heuristic is to run the protocols using a continuous compression model, but simply map the computed \( k_s \) value to the nearest valid discrete compression level at each iteration.

**Theorem 2.** The discrete-compression factor problem is NP-hard to solve optimally.

**Proof.** We reduce from EXACTLY ONE 3-SAT (1). In that problem, we are given a boolean formula of the form \( A_1 \land A_2 \land \ldots \land A_n \), where each clause \( A_i \) is of the form \( \ell_{i,1} \lor \ell_{i,2} \lor \ell_{i,3} \), where each \( \ell_{i,j} \) is a variable or a negated variable. The task is to determine whether the formula is satisfiable, with exactly one true literal in each clause.

Given a 3-SAT instance, we produce a problem instance with \( 3n \) sensors and \( 4n \) missions. The sensors are grouped into triples. There are two classes of missions, \( n \) big missions and \( 3n \) little missions. Each little mission corresponds to (has a flow from) one sensor; each big mission corresponds to a triple of sensors. (Call these big flows and little flows.)

The edges have unit capacity. The utility function for a mission is a linear function of utility received, varying over the range \([0, 1]\), to \([0, C_2]\) for big missions and \([0, C_1]\) for little missions, with \( C_2 >> C_1 \) (i.e., \( C_2 > 4nC_1 \)). (If desired, the utility functions can be made strictly concave without affecting the proof.) Call a mission satisfied if its total incoming flow is one. We set \( \delta = 0 \) and \( P_{max} = \infty \), and so set aside energy considerations. We set \( L = \{0, 1\} \), so that the only allowable compression options for an edge are no compression at all (forward all information) or complete compression (forward no information).

We construct the network with no congestion conflicts at all between the big flows or between big and little flows. The congestion conflicts between the little flows will depend on the nature of the underlying 3-SAT instance. We specify that there be a conflict between or any two little flows for sensors \((i_1, j_{i_1,j_1})\) and \((i_2, j_{i_2,j_2})\) if \( \ell_{i_1,j_1} \) is the negation of \( \ell_{i_2,j_2} \) or vice versa. A conflict means two paths share an edge, which implies that the sum of their flows cannot be greater than one. Note that the total number of conflicts (and hence the number of additional edges inserted) is bounded by \( O(n^2) \).

Due to their relatively large importance, therefore, any optimal solution will fully satisfy every big mission (contributing a total of \( 3nC_2 \) to the solution value). Hence we will have \( x_s = 1 \) for every \( s \). The real decisions, therefore, will concern which little flows will be “compressed” and how many little missions will be satisfied. Due to the conflicts among little flow triplets, the fact that \( x_s = 1 \), and the allowable compression values, all little mission flow values in an optimal solution will be \( 0/1 \).

We now claim that an optimal solution will satisfy all missions (for a total value of \( 3nC_2 + nC_1 \)) iff the EXACTLY ONE 3-SAT instance is satisfiable. If there is an optimal solution satisfying every little mission, then at least exactly little flow in each triple is not compressed in this solution. By making corresponding literals true and all other literals false, we obtain a positive 3-SAT solution. Conversely, from such a 3-SAT solution, compression decisions can be read off satisfying all little missions.

**Solitary Operator Location:** Our theoretical model assumes that a particular fusion operator may be “split” (in different fractions) across multiple nodes. In practice, many operators may not be conducive to such fractional splitting over infinitesimal time-scales; e.g., intelligent mixing of audio signals may require each of the audio streams to be mixed at a common node. However, interpreting “fractional fusion” as a process of “time-sharing” the responsibility of fusion across multiple nodes, on a longer-time scale, makes it applicable to a very large set of operators. For example if the optimal solution indicates 80% data fusion at node 1 and 20% at node 2, a logically equivalent workload partitioning may be achieved by having nodes 1 and 2 perform fusion for 80% and 20% of the time, respectively.

If fractional fusion is, nevertheless prohibited (for whatever reason), our heuristic solution is to assign the responsibility for fusion to the node with the “largest \( \theta \)”. A heuristic based approach is required because the problem of determining the best single location for a fusion operator is an NP-hard combinatorial problem as well, as shown in Theorem (3).

**Theorem 3.** The Solitary Operator Location problem is NP-hard to solve optimally.

**Proof.** Let us consider the simple network topology shown in Fig. (5), where the single mission receives data flow fused from two streams from sensors \( s_1 \) and \( s_2 \). The number of candidate nodes for placement of the fusion operation \( f \) (i.e., the number of nodes in the chain) is \( n \) (let these candidate nodes be labeled \( 1, 2, ..., n \) where node 1 is closest to the sources and \( n \) is closest to the mission). Let the \( m^{th} \) node in the chain be the optimal fusion location. Determining this node \( m \) such that the utility is maximized is equivalent to solving the following mixed-integer non-linear (MINLP)
optimization problem, which is well-known to be NP-hard:

\[
\max \quad U_m((\prod_{i=m+1}^{n} l_{i,f})f((\prod_{i=1}^{m} l_{i,x1})x_{s1},(\prod_{i=1}^{m} l_{i,x2})x_{s2}) - \delta \sum_{\forall nodes,k} P_{tot,k})
\]

subject to \quad \text{Energy and Capacity constraints}

The selection of this single fusion point may be performed at each iteration of the NUM-θ-loop (Equation 6). To achieve this, the highest θ value of downstream nodes is also propagated up the reverse forwarding path; the most upstream node among the fusion candidates can then designate the node with the largest θ as the fusion point. However, to ensure rapid convergence, the other terms (in the Feedback Information Table) carried in the signaling messages are based on the use of the ‘virtual’ continuous-θ values.

7. Evaluation

In this section we evaluate the performance of the NUM-INP protocol. We emphasize that our results are based on a packet-level protocol implementation of our NUM algorithm, and not simply based on numerical computations (such as via Matlab). The simulations were performed using Qualnet simulator on an 802.11-based multi-hop wireless network. The values of α^k_{recv}, α^k_{trans} and α^k_{comp} are taken as 0.75μJ/bit, 0.6μJ/bit and 0.54μJ/bit, based on the data from (14).

Utility Gain Due to In-network Processing: Fig. (9) compares the utilities of a network under three cases: a) with only source rate adaptation (according to WSN-NUM) but no in-network compression, b) optimal variable quality compression (according to NUM-INP) with pre-specified fusion locations and c) with joint optimization of compression and operator placement. The simulated network consists of 100 nodes of random topology in a 1500m x 1500m field. There are 25 missions and 25 sources and 15 fusion operations, whose initial locations are picked randomly from the sets of candidate locations (given by operator task graph). We can see that with NUM-INP, the global utility of the network is higher (by about 30%); the joint optimization of the operator locations results in a further 15% gain in system utility.

Fig. (8) uses a sample simulated topology to illustrate the utility gains obtained from adaptive in-network compression. The compression factor and transmission rate for each node are shown in the figure. The utility of a mission is of the form γln(1 + x_{rec}^θ). For missions A and B, γ = 100; for missions C and D γ = 20; for missions E and F γ = 1; for missions G and H γ = 0.25. The rates at which each mission receives data is also shown in the figure. As illustrated, in our model, missions that have higher utility receive higher data rate. On the contrary, if there is no in-network compression, then all the missions receive at a uniform rate of 11.57 kbps. The values shown within parentheses are the compression factors and rates when only four discrete compression levels (0.25, 0.5, 0.75 and 1.0) are allowed—we observe that the rates achieved, while lower than the continuous-case optimal, are still higher than that achieved without in-network adaptive compression.

Protocol Overhead: Figures (10) and (11) show the signaling overhead involved in the protocol, for the case of joint optimization of compression and operator placement. Fig. (10) shows the contour of the signaling overhead per node per iteration, for different numbers (levels) of cascaded fusion operators (i.e., the diameter of the operator graph), and for different sizes of sub-network (number of intermediate forwarding nodes) within each fusion level. Each ‘sub-network’ consists of segments similar to Fig. (5). We can see that the signaling overhead increases linearly with increase in chain length, i.e., increase in the number of potential candidates for that fusion operation, but it is fairly constant for increase in cascading levels. This is due to the need to have separate entries for each potential fusion point in the Feedback Information Table; however, across multiple fusion operations, the data is cumulatively accumulated in the cum field, and hence it does not incur much overhead. Fig. (11) shows the contour for signaling overhead as bytes per node per second. We can see that the signaling cost is amortized over time, since longer chains of nodes also implies longer
Performance Scalability: Fig. (12) shows the percentage gain in utility achieved by NUM-INP protocol, compared to simple source rate adaptation (WSN-NUM), when the number of missions and sources in the network are varied. We see that the gain increases with an increase in the number of competing missions and sensor sources. We experimented with different topologies (such as tree, random, etc) and observed similar results in all cases. The relative gain with in-network processing is higher when the number of missions is larger; adaptive in-network compression and fusion helps to alleviate congestion bottlenecks, while adhering to the energy consumption constraints.

NUM-INP under “Realistic Constraints”: To study the impact of discrete compression levels, we compare the utility degradation that results as a function of the number of discrete compression levels permitted. We map a compression factor value to a particular level, depending on how many levels are available. For example, when 10 levels of compression are allowed, we let $level1 = 0.1$, $level2 = 0.2$, and so on. Fig. (13) plots the system utility (normalized over the optimal utility with continuous compressibility). We see that the utility is close to and above 95% of the optimal for 10 or more number of discrete levels, but drops rapidly when the number of distinct compression levels is very small.

Fig. (14) shows the normalized utility, as a function of the number of fusion operators, when partial fusion is prohibited and fusion is performed at the candidate node with the highest $\theta$ value. (For each fusion operator, the number of candidate nodes was randomly chosen to be between 2 and 10.) We see that the utility remains close to the optimal utility even as the number of fusion operations in the network is increased, with only at most 5% loss in system utility. By comparing this result to Figure 9, where adaptive operator placement offers an additional 20% gain in utility, we see that joint optimization of compression and operator placement is beneficial, even when fractional operator placement is not permitted.

8. Conclusion

In this work, we have developed a utility-based framework for adaptive in-network processing in WSNs which maximizes the sum of mission utilities by jointly optimizing the source data rate, the degree of stream compression and the location of fusion operators. Experiments demonstrate our protocol’s robustness and show that it can achieve up to 40% higher utility than pure source-rate adaptation, with only modest signaling overhead. Moreover, simple heuristics to the NUM algorithm enable it to provide close-to-optimal utility, even under more discrete constraints on compression levels or operator placement. In ongoing work, we are extending this framework to dynamically modify the level of in-network processing, taking network lifetime into account.

A. Appendix

Algorithm 1 Computing source rate, $x_s$ for each iteration

```plaintext
COMPUTE $\frac{dx_s}{dt}$:
if isEmpty($FB^s.InfoTable$) then
    WillToPay = $x_{out}(s,s) \times FB^s.MU$
else
    WillToPay = $(FB^s(s,RI) + FB^s(cum^s,RI)) \times FB^s.MU$
end if
powCost = $FB^s(cum^s,PI) + FB^s(s,PI) + (\eta_s + \delta)x_s \frac{\partial p_{out}}{\partial x_s}$
congCost = $FB^s(cum^s,CI) + FB^s(s,CI) + \sum_{q:(s,s) \in q} (\frac{\eta_s}{c_{ss}}x_s \frac{\partial x_{out}(s,s)}{\partial x_s})$
$\frac{dx_s}{dt} = \kappa(WillToPay - powCost - congCost)$
```
Algorithm 2 Computing compression factor, $l_{k,s}$ for each iteration

```plaintext
COMPUTE $\frac{dl_{k,s}}{dt}$:
if isEmpty($FB^k, InfoTable$) then
    WillToPay = $x_{out}(s,k) \ast FB^k.MU$
else
    if $s$ is not an input for the fusion operation at $k$ OR
    isEmpty($FB^k, cum^s$) then
        WillToPay = $FB^k(s, RI) \ast FB^k.MU$
    else
        WillToPay = $FB^k(cum^s, RI) \ast FB^k.MU$
end if
end if
powCost = $FB^k(cum^s, PI) + FB^k(s, PI) + (\eta_k + \delta)l_{k,s}$
congCost = $FB^k(cum^s, CI) + FB^k(s, CI) + \sum_{q \in Q(k,i)} (\mu_{k,i}l_{k,s} \frac{\partial x_{in}(s,k)}{\partial x_{in}(s,k)})$
\[
\frac{dl_{k,s}}{dt} = \kappa_1 (WillToPay - powCost - congCost)
\]
```

Algorithm 3 Computing fusion fraction, $\theta$ for each iteration

```plaintext
COMPUTE $\frac{d\theta_{k,f}}{dt}$:
$\{^*\theta_{k,f}$ is the fraction $\theta$ at fusion point $k$ for fused flow $f$ and fusion input $^*\theta\}$
if isEmpty($FB^k, InfoTable$) then
    WillToPay = $l_{k,f} \ast x_{in}(i,k) \frac{\partial f}{\partial x_{in}(i,k)}) \ast FB^k.MU$
else
    WillToPay = $-1 \ast (FB^k(cum^f, RI) + FB^k(f, RI) \frac{\partial f}{\partial x_{in}(i,k)}) \ast \frac{\theta_{k,f}'}{(1-\theta_{k,f}')} \ast FB^k.MU$
end if
powCost = $-1 \ast (FB^k(cum^f, PI) + FB^k(i, PI) + FB^k(f, PI) \frac{\partial f}{\partial x_{in}(i,k)} (\eta_k + \delta) \frac{\partial \theta_{k,f}'}{(1-\theta_{k,f}')})$
congCost = $-1 \ast (FB^k(cum^f, CI) + FB^k(i, CI) + FB^k(f, CI) \frac{\partial \theta_{k,f}'}{(1-\theta_{k,f}')})$
\[
\frac{d\theta_{k,f}}{dt} = \kappa_2 (WillToPay - powCost - congCost)
\]
```

References