Distributed Adaptive Sampling, Forwarding, and Routing Algorithms for Wireless Visual Sensor Networks

Johnsen Kho Long Tran-Thanh Alex Rogers Nicholas R. Jennings
School of Electronics and Computer Science,
University of Southampton,
Southampton, SO17 1BJ, UK.
{jk05r,ltt08r,acr,nrj}@ecs.soton.ac.uk

ABSTRACT

The efficient management of the limited energy resources of a wireless visual sensor network is central to its successful operation. Within this context, this paper focuses on the adaptive sampling, forwarding, and routing actions of each node in order to maximise the information value of the data collected. These actions are inter-related in this setting because each node's energy consumption must be optimally allocated between sampling and transmitting its own data, receiving and forwarding the data of other nodes, and routing any data. Thus, we develop two optimal decentralised algorithms to solve this distributed constraint optimization problem. The first assumes that the route by which data is forwarded to the base station is fixed, and then calculates the optimal sampling, transmitting, and forwarding actions that each node should perform. The second assumes flexible routing, and makes optimal decisions regarding both the integration of actions that each node should choose, and also the route by which the data should be forwarded to the base station. The two algorithms represent a trade-off in optimality, communication cost, and processing time. In an empirical evaluation on sensor networks (whose underlying communication networks exhibit loops), we show that the algorithm with flexible routing is able to deliver approximately twice the quantity of information to the base station compared to the algorithm using fixed routing (where an arbitrary choice of route is made). However, this gain comes at a considerable communication and computational cost (increasing both by a factor of 100 times). Thus, while the algorithm with flexible routing is suitable for networks with a small numbers of nodes, it scales poorly, and as the size of the network increases, the algorithm with fixed routing is favoured.

Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*Intelligent agents, Multiagent systems*

General Terms

Algorithms, Experimentation, Management, Performance

Keywords

Decentralised mechanism, distributed constraint optimization, information metric, inter-related adaptive sampling and routing

1. INTRODUCTION

Due to their flexibility and ease of deployment, wireless sensor networks, composed of battery powered sensor nodes that wirelessly communicate and route information sampled from the environment through the network to a base station, are becoming increasingly prevalent in a wide variety of applications, including environmental monitoring [8], area surveillance [1, 10], and object tracking in battlefields [4]. In particular, the rapidly increasing computational power of the nodes deployed within such networks has allowed them to perform ever more sophisticated tasks, and recently, *wireless visual sensor networks* (WVSN), whose nodes consist of spatially distributed smart camera devices, which are capable of performing basic capturing and processing of visual data, before forwarding it to the base station to be fused and analysed, have received increasing attention within the research community [13].

Such networks are intended for distributed image acquisition over large, and possibly hostile environments, and as such, are required to operate for extended periods of time with minimal human intervention. However, the increased computational power of the nodes within a WVSN (compared to those typically deployed within a conventional wireless sensor network), the large amounts of visual information that they collect, and the high energy cost of wirelessly communicating this information through the network, mean that efficient energy management is a key challenge in these networks.

To date, this challenge has been addressed through two complementary approaches: namely (i) hardware and (ii) software solutions. Within the former, advances in chip manufacture have successfully reduced the power consumption of nodes, helping to improve their longevity, and, in turn, the network's lifetime [3]. From the latter perspective, work has addressed the two main actions that such sensor nodes can vary in order to make their energy management more efficient: (i) their *sampling rate* (i.e. *how much* visual data they acquire) and (ii) their *communication* of data capabilities (those include selecting the most energy efficient path between the node and the base station given that the nodes may have different energy constraints and communication costs).

In particular, recent work has explored decentralised coordination algorithms that enable the nodes to autonomously adapt and adjust their sampling and communication behaviours. This coordination is computationally expensive since the sampling and communication decisions are inter-dependent. This is because each node's energy consumption must be optimally allocated between *sampling* and *transmitting* its own data, *receiving* and *forwarding* the data of other nodes, and *routing* any data. In such a setting, the choices of one node can potentially affect all other nodes in the network. However, much of this work has specifically addressed nodes that are assumed to be extremely low power, computationally constrained devices. As such, it considers simple heuristic algorithms that allow the nodes to make local decisions to improve the overall performance of the network (see Sect. 5 for more details).

While such approaches have proved valuable in the context in which they were developed, when applied to WVSN they do not fully exploit the computational power available to the nodes. Furthermore, the large amounts of visual data that the nodes within the WVSN collect and communicate means that the communication resources available to the decentralised coordination mechanism are much greater than those of many conventional wireless sensor networks. When taken together, the move to WVSN means that there is now the possibility of deploying algorithms that can optimally maximise the overall effectiveness of the network through distributed computation, rather than local heuristics. It is this challenge that we address in this paper, and to this end, we adopt an agent-based approach in which each node is represented as an autonomous agent (with private information regarding its own state), and the complex, inter-connected, and rapidly changing network, as a multi-agent system. The individual agents must then cooperatively coordinate their activities to achieve system-wide goals.

Against this background, in this paper, we develop a novel optimal decentralised algorithm that varies each node's sampling, transmitting, and forwarding rates to ensure all nodes in a network focus their limited resources on maximising the information content of the data collected at the base station. This algorithm assumes that the route by which data is forwarded to the base station is *fixed* (either because the underlying communication network is a tree, or because an arbitrary choice of route has been made), and uses a distributed dynamic programming approach to extensively truncate the search space of potential allocation decisions. We then extend this approach to deal with *flexible* routing, in which each node not only makes optimal decisions regarding the sampling, transmitting, and forwarding actions, but also determines the optimal route by which this data should be forwarded. To ground and evaluate these algorithms, we empirically evaluate them and show that they represent a trade-off in optimality, communication cost, and processing time. In more detail, we show that when deployed on sensor networks with loopy topology (i.e. where data can follow multiple paths to the base station), the algorithm with flexible routing is able to deliver approximately twice the quantity of information to the base station compared to that of the algorithm using fixed routing. However, this gain comes at considerable communication and computational cost (increasing both by a factor of 100 times).

The remainder of this paper is organized as follows. In Sect. 2 we state the formal model of our distributed constraint optimization problem. In Sect. 3 we detail our two novel algorithms and show how we find the optimal local allocation decisions. Our experimental results are presented in Sect. 4. We present previous work in this area in Sect. 5 and we conclude in Sect. 6.

2. PROBLEM DESCRIPTION

Here, we formalise the generic adaptive sampling, transmitting, forwarding, and routing problem that we face. To this end, let n be the number of nodes within a WVSN system and the set of all nodes (or agents) be $I = \{1, \ldots, n\}$. Each node $i \in I$ can sample at s_i different rates over a period of time. Its set of possible sampling (or frame) rates is denoted by $C_i = \{c_i^1, \ldots, c_i^{s_i}\}$. Each element of this set, c_i^j, is a positive integer that describes the number of samples that the node takes during any specific time interval.

Each node has private information regarding the information content of the samples that it acquires, and this is represented by an array of 2-tuples $\mathfrak{F}_i = \left[(0,0), \left(c_i^1, v_i^{c_i^1}\right), \ldots, \left(c_i^{s_i}, v_i^{c_i^{s_i}}\right)\right]$, where the first value of each tuple is the number of samples that the node may take and $v_i^{c_i^j}$ is the corresponding information content for those c_i^j samples. Given the fact that more samples will generally generate visual data with a higher information content, we define $v_i^{c_i^j} = \alpha_i \cdot c_i^j$, where α_i is a weighting factor (with support $[0, 1]$) that models the typical situation that the sensors within the network are heterogeneous, having different capabilities (i.e. the resolution of their cameras, the quality of their optics, or the sophistication of their image processing algorithms) and fields of view, and thus, samples from some sensors will contribute more to the total amount of information collected at the base station than others [11]. However, we note that our algorithms are not restricted to this linear information valuation function and, in some domains, it may be more valid to model the information as a strictly concave function where continuing to increase the sampling rate generates decreasing gains in information content [2]. We assume that should the node choose to acquire no samples, it will yield zero information value. Furthermore, we assume that the visual data captured by a node needs to be processed at the base station with that of other nodes, and therefore the information content of the data will only be accounted for if it arrives successfully at the base station.

We further assume that each node has an energy budget, B_i (also a private value initially known only to the node), such that its total energy consumption can not exceed this budget. We consider three specific kinds of energy consumption for each node in the network; namely the energy required to (i) acquire, e_i^{s}, (ii) transmit, e_i^{Tx}, and (iii) receive, e_i^{Rx}, a single sample. We disregard the energy required for other types of processing since it is negligible in comparison. Now, since the node has to transmit its own data towards the base station, the total energy required for this activity is thus $E_i^{\mathrm{S}} = e_i^{\mathrm{s}} + e_i^{\mathrm{Tx}}$ per sample (we will later on refer to the combination of these processes as *sensing*). Similarly, the node could potentially spend a portion of its energy to help its neighbourhood nodes to *forward* their own samples (and/or data that this node is forwarding for another node). This incoming data forwarding process requires a total energy of $E_i^{\mathrm{F}} = e_i^{\mathrm{Rx}} + e_i^{\mathrm{Tx}}$ per sample.

Each node initially stores its collected samples into its local memory buffer in order to be transmitted at a later stage. The transmission period and interval are predetermined. During each transmission phase, the transmitter module of each node is turned on for the purpose of transmitting data or message packets to the base station in a multi-hop fashion. Battery-powered visual sensor nodes typically offer reasonably small on-board memory and, hence, at the end of the transmission phase, each node's memory buffer is flushed, reinitialized, and ready to store new sampled data [6].

We describe the route through which the samples, c_i^j, will be transmitted to the base station by the vector $R(c_i^j) = (r_i^1, \ldots, r_i^b)$, where $r_i^l \in I$. The first element of this vector is the origin node that actually takes the samples. Each subsequent element of this vector is unique and r_i^l will forward the data to r_i^{l+1}. Thus, for the data value of c_i^j samples to be taken into account, its routing set must contain the base station node as its last node.

Given the formal description of the problem above, we now wish to maximise the value of the collected data that arrives at the base station. That is, we wish to solve:

$$\mathcal{D}_i^* = \arg \max_{\{\mathcal{D}_i, \mathfrak{F}_i\}} \sum_{i=1}^{n} \sum_{c_i^j \in C_i} d_i^{R(c_i^j)} v_i^{c_i^j} \qquad (1)$$

In this expression, $d_i^{R(c_i^j)} \in \mathcal{D}_i \in \{0, 1\}$ is a decision variable where "1" represents a state where the node carries out the corresponding c_i^j sampling action and the samples follow the $R(c_i^j)$ route to arrive at the base station, and "0" represents the state where the node does not carry out the corresponding c_i^j sampling action. This objective function is maximised subject to the energy budget constraint on each node $i \in I$, such that:

$$B_i \geq c_i^j E_i^{\mathrm{S}} + f_i E_i^{\mathrm{F}} \tag{2}$$

where f_i represents the total incoming data (or forwarded samples from its set of neighbourhood nodes D_i) and is given by:

$$f_i = \sum_{d \in D_i} c_d^j + f_d \tag{3}$$

where $i \in R(c_d^j)$. Note that the total outgoing number of samples from node i is thus $c_i^j + f_i$. We must also constrain the node to chose one and only one sampling rate, such that:

$$\sum_{c_i^j \in C_i} d_i^{R(c_i^j)} = 1 \tag{4}$$

for all different possible routes in the network.

The problem, as formulated above, is similar to *multiple-choice knapsack* problems[1] (i.e. NP-complete resource allocation or distributed constraint optimization problems) [12], that exhibit the *optimal substructure* property[2]. Given this insight, we propose algorithms based on the sort of computationally efficient dynamic programming technique that are often used on such knapsack problems for solving multi-agent distributed coordination problems.

3. THE ALGORITHMS

We now focus on the algorithms used by the nodes to make optimal use of their energy resources in order to cooperatively sense, forward, and route data to the base station. Our approach places higher priority on those samples that have a higher information content, and this is achieved by exchanging coordination messages between connected nodes. To this end, we distinguish three types of messages being exchanged by the nodes; namely (i) *actual data messages* containing visual data sampled by the nodes, and two types of *coordination* messages composed of (ii) *meta-data messages* describing the information content of the visual data together with the number of samples taken to produce that data (i.e. $v_i^{c_i^j}$ and c_i^j respectively), and (iii) *control messages* containing the allocation decisions. In WVSNs, the size of the actual data messages overwhelms that of the coordinations messages and, hence, exchanging these before sending the actual data can significantly increase the information collected at the base station by making more efficient use of each node's constrained energy.

The goal of the algorithms that we derive is to calculate the optimal sampling and routing actions of each node. This is given by:

$$\begin{aligned} Cmax_I = & \{(i, c_i^j, R(c_i^j)) | d_i^{R(c_i^j)} = 1, \\ & \forall i \in I, \forall c_i^j \in C_i, \forall d_i^{R(c_i^j)} \in \mathcal{D}_i^*\} \end{aligned} \tag{5}$$

[1]There are m items and the set of all items $T = \{1, \ldots, m\}$. Each item $t \in T$ has a value v_t and a weight w_t. The items are subdivided into o categories and exactly one item must be taken from each category. The maximum weight that can be carried in a bag is G. Given these, we need to determine which items to include in the bag such that the total weight does not exceed its given limit, while the total value is maximised.

[2]This property means that the optimal solution can be constructed efficiently from optimal solutions to its subproblems.

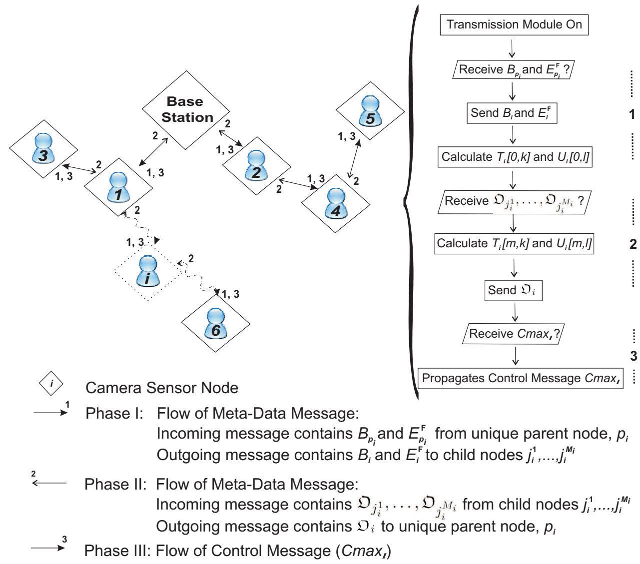


Figure 1: The flow of the algorithm in a connected and undirected tree-structured WVSN. We assume that communication is bi-directional and multiple nodes within range can thus establish a connection. Dotted node i could represent any subtrees in the network.

and represents a set of 3-tuples indicating for each node in the network, the sensing and forwarding rates that it should adopt, and the route that it should use to transmit its own and its forwarded data to the base station, in order to maximise the objective function in (1), subject to the constraints in (2) and (4). We now present our two novel adaptive sensing, forwarding, and routing algorithms. Both of them are efficient as they satisfy the data flow conservation of the network where no energy is wasted by transmitting data that later will not be forwarded to the final destination.

3.1 Algorithm With Fixed Routing

In this case, each node $i \in I$ can only forward its data to exactly one other node (which will later be referred as its *parent*). This may be because the underlying communication network of the WVSN is tree structured, or because it actually exhibits loops but an arbitrary choice of route has been made (effectively turning the loopy communication network into a tree). An example of a WVSN whose underlying network structure is a tree structure is shown in Fig. 1. Note that in such tree-structured networks, there is only one unique route between each node and the base station (e.g. $R(c_4^j) = (4, 2, \text{base station})$ and $R(c_3^j) = (3, 1, \text{base station})$).

In general, the nodes within a network will deplete their energy resources at different rates since they will have different sampling rates, and will be transmitting different quantities of visual data. Each node $i \in I$ thus needs to compute the highest information value it can transmit by using at most $b_i^k \leq B_i$ of its energy. As described earlier, the energy consumption of node i only includes E_i^{S} and E_i^{F} (i.e. the energy to sense and forward a sample respectively). It is therefore sufficient that b_i^k satisfies:

$$\begin{aligned} & b_i^k = c_i^j E_i^{\mathrm{S}} + f_i E_i^{\mathrm{F}} \quad \text{where } c_i^j, f_i \geq 0 \\ & b_i^k \leq B_i \end{aligned} \tag{6}$$

where c_i^j is its own number of samples and f_i is the number of forwarded incoming samples.

Now, let $\mathfrak{D}_i = \Big[\big(b_i^1, Vmax_i^1, Cmax_i^1\big), \ldots, \Big(b_i^{K_i}, Vmax_i^{K_i},$

Algorithm 1 Optimal adaptive sensing and forwarding with fixed routing.

1: **loop**
2: **if** $sTime = NOW$ **then** ▷ Time to sample.
3: $readings \leftarrow$ PERFORMSAMPLING($sTime$) ▷ Sampling action, c_i^j.
4: SETSTIME($sTime + sRate$)
5: **end if**
6: **if** $tTime = NOW$ **then** ▷ Time to transmit, transmission module is turned on.
7: $[B_{p_i}, E_{p_i}^{\mathrm{F}}] \leftarrow$ WAITMETADATA(p_i) ▷ Receives B_{p_i} and $E_{p_i}^{\mathrm{F}}$ from its unique parent node, p_i.
8: **for** each $j_i^m \in J_i$ **do** ▷ Iterates each child node in $J_i = \left\{ j_i^1, \ldots, j_i^{M_i} \right\}$.
9: SENDMETADATA($j_i^m, [B_i, E_i^{\mathrm{F}}]$) ▷ Sends B_i and E_i^{F} to child node j_i^m.
10: **end for**
11: CALCFIRSTROWTABLES($readings$) ▷ Calculates the 1^{st} rows of T_i and U_i using (7) and (10) respectively.
12: **if** $\neg leafNode$ **then**
13: **for** each $j_i^m \in J$ **do**
14: $\mathfrak{O}_{j_i^m} \leftarrow$ WAITMETADATA(j_i^m) ▷ Receives $\mathfrak{O}_{j_i^m}$ from child node j_i^m.
15: CALCTHERESTTABLES($\mathfrak{O}_{j_i^m}$) ▷ Calculates the other rows of T_i and U_i using (8) and (11) respectively.
16: **end for**
17: **end if**
18: $\mathfrak{O}_i \leftarrow$ CALCMETADATAARRAY() ▷ Determines $\mathfrak{O}_i$ which is basically the last row of U_i.
19: SENDMETADATA($p_i, \mathfrak{O}_i$) ▷ Sends $\mathfrak{O}_i$ to unique parent node, p_i.
20: $Cmax_I \leftarrow$ WAITCONTROLMESSAGE(p_i) ▷ Receives control message from unique parent node, p_i.
21: PROPAGATECONTROLMESSAGE($j_i^m, Cmax_I$) ▷ Sends control message to each child node, $j_i^m \in J_i$.
22: PERFORMTRANSMIT($readings, Cmax_I$)
23: SETNODEOPTIMALBEHAVIOUR($Cmax_I$) ▷ Sets node's optimal sensing and forwarding actions.
24: SETTTIME($tTime + tRate$) ▷ Node sets its next transmitting time.
25: $readings \leftarrow \{\}$
26: **end if**
27: **end loop**

$Cmax_i^{K_i}\big)\big]$ denote an array of 3-tuples sorted incrementally by b_i^k where $k = 1, \ldots, K_i$, and b_i^k is the energy limit that satisfies (6) and will later on be referred to as the labels of $\mathfrak{O}_i$. $Vmax_i^k$ is the maximum information value that node i can transmit to its parent by using at most b_i^k, and $Cmax_i^k$ is the set of sensing and forwarding actions that will produce data with the value of $Vmax_i^k$.

The algorithm installed on each node runs in three phases (see Fig. 1 and Algorithm 1). In the first, meta-data message propagation is initiated by the base station. To this end, messages containing the value of each node's energy budget, B_i, and energy consumption for forwarding, E_i^{F}, are propagated down the tree (i.e. as soon as any node receives this information from its unique parent node, p_i (see state 1 or line 7), it sends its own information to its set of children, $J_i = \left\{ j_i^1, \ldots, j_i^{M_i} \right\}$ (line 9)). Having sent this information each node i then enters an idle mode in which it waits for the $\mathfrak{O}$ meta-data arrays from its child nodes.

In the second phase, after all the $\mathfrak{O}$ meta-data arrays have arrived from its children (denoted by $\mathfrak{O}_{j_i^1}, \ldots, \mathfrak{O}_{j_i^{M_i}}$, see state 2 or lines 14-15), node i then calculates its own $\mathfrak{O}_i$ (line 18). To do so, it constructs a table, T_i, which has $M_i + 1$ rows numbered from 0 to M_i, and K_i columns, where K_i is the number of all the b_i^k values that satisfy (6). See Table 1 in which each column k has label b_i^k. Let $T_i\,[x, y]$ denote the element of the table that is in the x^{th} row and the column with label b_i^y. As every node could choose not to sample (yielding zero value), then $\mathfrak{O}_{j_i^m}\,[0] = T_i\,[m, 0] = 0$ for all $0 \leq m \leq M_i$, where $\mathfrak{O}_{j_i^m}\,[x]$ is the x^{th} element of $\mathfrak{O}_{j_i^m}$. Moreover, we can assume that the set of labels in each $\mathfrak{O}_{j_i^m}$ that node i has received is the same as the set of labels in its table T_i. We will show how we can guarantee this later on. Hence, T_i's other elements are filled as follows:

$$T_i\,[0, k] = \max\{v_i^{c_i^j}\} \qquad (7)$$

$$T_i\,[m, k] = \max_{0 \leq r \leq k} \left\{ T_i\,[m-1, r] + Vmax_{j_i^m}^{k-r} \right\} \qquad (8)$$

for all $1 \leq k \leq K_i$ and $1 \leq m \leq M_i$, where $(c_i^j, v_i^{c_i^j}) \in \mathfrak{F}_i$, and $\mathfrak{F}_i$ is the array of 2-tuples defined in the previous section.

According to (7), we can see that $T_i\,[0, k]$ stores the maximum information value of data that can be delivered to node i's parent by sensing only (with the energy consumption not exceeding the energy limit b_i^k). Due to the fact that each of the sets of labels in $\mathfrak{O}_{j_i^m}$ is equivalent to the set of labels of table T_i, (8) gives the maximum value of data that node i can deliver to its parent (noting that this data does not only include its own sensed data but also its children's data that will potentially be forwarded through it). Hence, $T_i\,[1, k]$ is the maximum value that can be sent by taking into account the sensed data and the data from j_i^1, with respect to the b_i^k energy limit. $T_i\,[2, k]$ stores the maximum value when the data from child node j_i^2 is also included. In general, $T_i\,[m, k]$ is the maximum information value that node i can transmit to its parent, given the b_i^k energy limit. The data considered is the potential forwarded data from child nodes $j_i^1, \ldots, j_i^m$ and node i's own sensed data.

Note that while it is necessary to construct the entire table, as in conventional dynamic programming solutions to the multiple-choice knapsack problem, it is only the last row that provides useful meta-data regarding the maximum information values of data that can be transmitted given different feasible values of b_i^k. Indeed, it is only the last element of this row that represents the maximal information value that node i can transmit to the parent node.

To illustrate how the elements of the table are calculated in a clearer way, consider Tables 1 and 2 in which the information values of node i's sensed data and the $Vmax_{j_i^m}^k$ values of $\mathfrak{O}_{j_i^m}$ arriving from its child nodes j_i^m respectively are chosen arbitrarily for illustrative purposes. The rows of Table 1 represent the set of nodes whose data has been taken into account. For instance where $row = i$, if node i has b_i^0, b_i^1, b_i^2, b_i^3, or b_i^4 amount of energy limit, in return it will be able to sense it own data with the maximum value of 0, 12.34, 14.56, 28.25, or 50.98 correspondingly. These values are calculated using (7). $\mathfrak{O}_{j_i^1}$ then arrives (see Table 2 where $row = \mathfrak{O}_{j_i^1}$) from its child node j_i^1 containing the maximum values that this node could potentially forward to node i.

The elements of T_i's second row (i.e. $row = \{i \cup j_i^1\}$) can thus be calculated using (8). These elements represent the maximum information that node i could send by taking into account not only its own sensed data, but also the data that could be potentially forwarded from its child node j_i^1. For instance where $column = b_i^1$, node i could allocate all its b_i^1 energy resources to either sense and transmit its own data or to forward data from its child node j_i^1. In this case, the node chooses to sense and transmit its own data since it has a higher value. Where $column = b_i^2$, however, the node again allocate all its b_i^2 energy resources to either sense its own data or to forward its child node j_i^1's data. Alternatively it could as well divide its b_i^2 energy resources by allocating a portion of b_i^1 energy resources to its own and another b_i^1 to its child node. In this case, it turns out that the latter alternative yields the highest information value of 19.32. T_i's other elements are calculated in a similar way.

Now, the next step of the algorithm is to calculate $\mathfrak{O}_i$. To do so, let U_i denote a table similar to T_i. However, its labels b_i^l, now, satisfy the following:

$$\begin{aligned} & b_i^l = (c_i^j + f_i)E_{p_i}^{\mathrm{F}} \quad \text{where } c_i^j, f_i \geq 0 \\ & b_i^l \leq B_{p_i} \end{aligned} \qquad (9)$$

Table 1: The T_i table of node i. Its $\mathfrak{D}_i$ meta-data array is represented by the dotted rectangle.

T_i	b_i^0	b_i^1	b_i^2	b_i^3	b_i^4	b_i^k
$\{i\}$	0	12.34	14.56	28.95	50.98	⋱
$\{i \cup j_i^1\}$	0	12.24	19.32	45.89	58.23	
⋮	⋮	⋮	⋮	⋮	⋮	
$\{i \cup j_i^1 \cup \ldots \cup j_i^{M_i}\}$	0	12.34	28.78	45.89	58.23	

Table 2: $\mathfrak{D}_{j_i^m}$ meta-data arrays that have arrived from each child nodes $j_i^1, \ldots, j_i^{M_i}$.

$\mathfrak{D}_{j_i^m}$-s	b_i^0	b_i^1	b_i^2	b_i^3	b_i^4	b_i^k
$\mathfrak{D}_{j_i^1}$	0	6.98	15.67	45.89	48.99	⋱
⋮	⋮	⋮	⋮	⋮	⋮	
$\mathfrak{D}_{j_i^{M_i}}$	0	6.79	28.78	35.89	51.88	

where B_{p_i} is the energy budget of i's unique parent node, p_i, and $E^{\mathrm{F}}_{p_i}$ is the value of energy consumption of the parent for forwarding a sample. Recall that these values were delivered to node i in the first stage. Let L_i denote the number of all b_i^l that satisfy (9). Similarly, we can calculate table U_i's elements in a similar fashion to those of T_i as described earlier, but with the new labels:

$$U_i\left[0, l\right] = \min\left(\max\{v_i^{c_i^j}\}, T_i\left[0, K_i\right]\right) \tag{10}$$

$$U_i\left[m, l\right] = \min\left(\max_{0 \le r \le l}\left\{U_i\left[m-1, r\right] + Vmax_{j_i^m}^{l-r}\right\}, T_i\left[m, K_i\right]\right) \tag{11}$$

for all $1 \le l \le L_i$ and $1 \le m \le M_i$, where $(c_i^j, v_i^{c_i^j}) \in \mathfrak{F}_i$.

We can now construct the meta-data array of node i such that $\mathfrak{D}_i = \left[\left(b_i^1, U_i\left[M_i, 1\right], Cmax_i^1\right), \ldots, \left(b_i^{L_i}, U_i\left[M_i, L_i\right], Cmax_i^{L_i}\right)\right]$, where $U_i\left[M_i, l\right]$ is the maximum information value that node i can transmit to its parent node (by using at most b_i^k energy) which can subsequently forward the received i's data by using at most b_i^l energy. $Cmax_i^l$ is the set of sensing actions that will produce data with the value of $U_i\left[M_i, l\right]$. Hence, once $\mathfrak{D}_i$ is sent to the parent node, its labels will be the same as those in table T_{p_i} of the parent node. This second phase meta-data message containing $\mathfrak{D}_i$ propagates up the network arriving back at the base station (line 19).

In the third phase of the algorithm, each parent node will have received meta-data arrays from all of its children. The base station will be able to calculate the highest information value it can potentially receive from all the nodes beneath it in the network. Based on the structure of $\mathfrak{D}_i$, each node i can easily determine what amount of data it should receive from each child node and, hence, how many samples it should acquire and transmit itself. A control message containing this set is then propagated down the network (see state 3 or lines 20-21), and this control message informs each node of its optimal decisions (lines 22-23). In this way, there is a guarantee that all of the data transmitted by each node will reach the base station. The control message eventually reaches the leaf nodes which then start to acquire and transmit visual data as planned.

3.2 Algorithm with Flexible Routing

Next, we consider the algorithm which assumes flexible routing, and makes optimal decisions regarding both the sensing and forwarding actions that each node should perform, and also the route by which data should be forwarded to the base station (see Fig. 2 for an illustration of this case). In order to make the routing de-

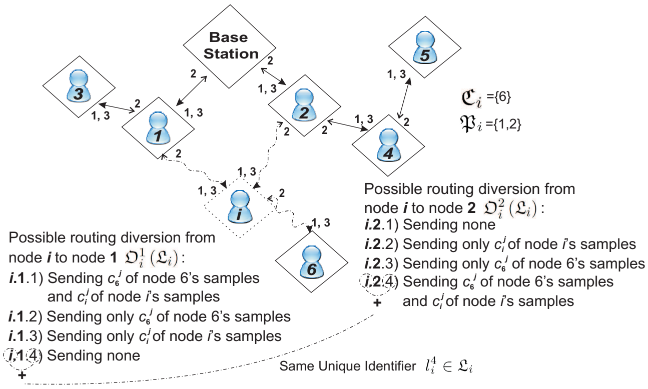


Figure 2: The flow of the algorithm that assumes flexible routing and makes optimal decisions regarding both sensing, forwarding, and next-hop (or routing) decisions. The phases involved in this algorithm are similar to those in the algorithm for fixed routing.

cision tractable, we place one minor restriction on the routes that our algorithm can consider. Specifically, we assume that the nodes always forward their data toward the base station; that is, they will not forward data to a node that is further from the base station (in terms of hop count) than themselves. We believe this is a reasonable assumption. There may be cases where several nodes are better off taking longer paths. However, in general such paths will deplete the energy resources of a greater number of nodes, and are thus unlikely to be optimal solutions. Furthermore, we assume that the data samples of a node can only be sent as a bundle (i.e. they are indivisible). The data readings of different nodes can, however, be sent through different routes to the base station.

With these assumptions, we now need to organize the nodes into different *levels*. The first consists of all the nodes that have a 1-hop shortest path to the base station (nodes 1 and 2 in Fig. 2). Nodes that belong to the second level have a 2-hop shortest path to the base station (nodes i, 3, and 4). Nodes 5 and 6 have a 3-hop shortest path. Now, according to this hierarchy, each node can only forward its data to higher level nodes within its transmission range. In Fig. 2, for example, node i has two potential shortest routes to choose from; namely (i) node 1 which results in route $R(c_i^j) = (i, 1,$base station) and (ii) node 2 which results in route $R(c_i^j) = (i, 2,$base station). Node i does not consider routing through node 6 since 6 is a greater hop count away from the base station than it is. Furthermore, as we can see from the figure, node i has potentially two bundles of data to consider (its own and data that it is forwarding for node 6). In addition, it has two possible shortest paths to choose between (either through node 1 or 2 for each of the bundled data). Thus, a number of routing options exist for this node. It could send both bundles of data through node 1, such that both $R(c_i^j)$ and $R(c_6^j)$ contain $(i, 1, \ldots)$, or it could send them through node 2. Other alternatives are to send each of them separately through each possible route, such that $R(c_i^j)$ contains $(i, 1, \ldots)$ and $R(c_6^j)$ contains $(i, 2, \ldots)$, or the other way around.

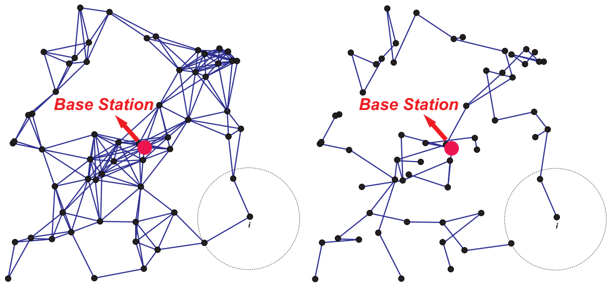
Now, let $\mathfrak{P}_i$ denote the set of parent nodes (which are nodes with a one hop shorter shortest path to the base station) of node i and $\mathfrak{C}_i$ denote the set of its descendants. Thus, at each node $i \in I$, there are at most $|\mathfrak{P}_i|^{|\mathfrak{C}_i|+1}$ possibilities to forward the bundled data, where $|\mathfrak{P}_i|$ and $|\mathfrak{C}_i|$ are the sizes of $\mathfrak{P}_i$ and $\mathfrak{C}_i$ respectively. This is because each node has to forward $|\mathfrak{C}_i| + 1$ bundles through $|\mathfrak{P}_i|$ different shortest paths. Next, let $\mathfrak{L}_i$ denote the set of these possibil-

Algorithm 2 Optimal adaptive sensing and forwarding with flexible routing.

1: **loop**
2: **if** $sTime = NOW$ **then** ▷ Time to sample.
3: $readings \leftarrow$ PERFORMSAMPLING$(sTime)$ ▷ Sampling action, c_i^j.
4: SETSTIME$(sTime + sRate)$
5: **end if**
6: **if** $tTime = NOW$ **then** ▷ Time to transmit, transmission module is turned on.
7: **for each** $p_i^{\mathfrak{n}} \in \mathfrak{P}_i$ **do** ▷ Iterates each parent node, $p_i^{\mathfrak{n}} \in \mathfrak{P}_i$.
8: $[B_{p_i^{\mathfrak{n}}}, E_{p_i^{\mathfrak{n}}}^{\mathrm{F}}] \leftarrow$ WAITMETADATA$(p_i^{\mathfrak{n}})$ ▷ Receives $B_{p_i^{\mathfrak{n}}}$ and $E_{p_i^{\mathfrak{n}}}^{\mathrm{F}}$ from parent node $p_i^{\mathfrak{n}}$.
9: **end for**
10: **for each** $j_i^m \in J_i$ **do** ▷ Iterates each child node in $J_i = \{j_i^1, \ldots, j_i^{M_i}\}$.
11: SENDMETADATA$(j_i^m, [B_i, E_i^{\mathrm{F}}])$ ▷ Sends B_i and E_i^{F} to child node j_i^m.
12: **end for**
13: CALCFIRSTROWTABLES$(readings)$ ▷ Calculates the 1^{st} rows of T_i and $U_i^{p_i^{\mathfrak{n}}}$ (for each parent node, $p_i^{\mathfrak{n}}$ in $\mathfrak{P}_i$) using (7) and (10) respectively.
14: $\mathfrak{C}_i \leftarrow \{i\}$ ▷ Adds this node to the set of descendants $\mathfrak{C}_i$.
15: **if** $\neg leafNode$ **then**
16: **for each** $j_i^m \in J_i$ **do**
17: $\mathfrak{D}_{j_i^m}^i \leftarrow$ WAITMETADATA(j_i^m) ▷ Receives $\mathfrak{D}_{j_i^m}^i$ from child node j_i^m.
18: CALCTABLESWITHIDENTIFIER$(\mathfrak{D}_{j_i^m}^i)$ ▷ Calculates the other rows of T_i using (8) by identifying the same forwarding partition with the same unique identifier.
19: $\mathfrak{C}_i \leftarrow \mathfrak{C}_i \cup j_i^m$ ▷ Adds child node j_i^m to the set of descendants $\mathfrak{C}_i$.
20: **end for**
21: **end if**
22: **for each** $p_i^{\mathfrak{n}} \in \mathfrak{P}_i$ **do**
23: $\mathfrak{L}_i \leftarrow$ PARTITIONPOSSIBLEFORWARDING$(\mathfrak{C}_i)$ ▷ Partitions the possible forwardings using a mapping function that decides the direction of each bundle, u_i^j, from one of its descendants in $\mathfrak{C}_i$.
24: $\mathfrak{D}_i^{p_i^{\mathfrak{n}}} \leftarrow$ CALCMETADATAARRAY$(\mathfrak{L}_i)$ ▷ Calculates the other rows of $U_i^{p_i^{\mathfrak{n}}}$ using (11) to forms its own $\mathfrak{D}_i^{p_i^{\mathfrak{n}}}$ meta-data for parent node $p_i^{\mathfrak{n}}$.
25: SENDMETADATA$(p_i^{\mathfrak{n}}, \mathfrak{D}_i^{p_i^{\mathfrak{n}}})$ ▷ Sends $\mathfrak{D}_i^{p_i^{\mathfrak{n}}}$ to parent node $p_i^{\mathfrak{n}}$.
26: **end for**
27: $Cmax_I \leftarrow$ WAITCONTROLMESSAGE$(p_i^{\mathfrak{n}})$ ▷ Receives control message from parent node $p_i^{\mathfrak{n}}$ in $\mathfrak{P}_i$.
28: PROPAGATECONTROLMESSAGE$(j_i^m, Cmax_I)$ ▷ Sends control message to each child node, $j_i^m \in J_i$.
29: PERFORMTRANSMITINCROUTING$(readings, Cmax_I)$
30: SETNODEOPTIMALBEHAVIOURINCROUTING$(Cmax_I)$ ▷ Sets node's optimal sensing, forwarding, and next-hop decisions.
31: SETTTIME$(tTime + tRate)$ ▷ Node sets its next transmitting time.
32: $readings \leftarrow \{\}$
33: **end if**
34: **end loop**

ities (with $|\mathfrak{L}_i| = |\mathfrak{P}_i|^{|\mathfrak{C}_i|+1}$) and each $l_i^t \in \mathfrak{L}_i$, a possible partition of forwarding at node i. That is, $l_i^t = \left[F\left(u_i^1\right), \ldots, F\left(u_i^{|\mathfrak{C}_i|+1}\right)\right]$ where u_i^j is the j^{th} bundle that might arrive at node i from one of its descendants, $F\left(u_i^j\right)$ is a mapping function that decides the forwarding direction (or path) for this particular bundle, and $u_i^{|\mathfrak{C}_i|+1}$ is node i's own bundle of samples.

Given this, our algorithm with flexible routing is similar to that with fixed routing, and as before, it runs in three phases (see Algorithm 2). The first, in which the parent nodes send their information regarding $B_{p_i^{\mathfrak{n}}}$ and $E_{p_i^{\mathfrak{n}}}^{\mathrm{F}}$ to their child nodes (where $p_i^{\mathfrak{n}} \in \mathfrak{P}_i$), is identical (see lines 7-13). There are, however, slight modifications in the next phase. These modifications are needed to keep track of all the possible partitions of forwarding for nodes which have more than one shortest path routes to the base station. In more detail, in the second phase, instead of sending one $\mathfrak{D}_i$ to a unique parent (as in the case of tree-structured networks), here, each node i has to calculate all the $\mathfrak{D}_i^{p_i^{\mathfrak{n}}}\left(l_i^t\right)$ meta-data arrays for each $l_i^t \in \mathfrak{L}_i$ partition of forwarding for each $p_i^{\mathfrak{n}} \in \mathfrak{P}_i$ (see lines 23-25). Specifically, this is achieved by first calculating the T_i table as we did for the first algorithm (line 17). In this case, however, we join each of



(a) (b)

Figure 3: (a) A randomly created and connected WVSN (of 60 nodes) whose underlying communication network exhibits loops. (b) The resulting tree-structured network formed when each node makes an arbitrary choice of the route that its data will take toward the base station. The dotted circle in each graph represents the wireless range of node i. In both these networks, all nodes are set with the same transmission range.

the arriving $\mathfrak{D}_{j_i^m}^i\left(l_{j_i^m}^t\right)$ from its children $j_i^1, \ldots, j_i^{M_i}$ with those that belong to the same forwarding partition with the same unique identifier (line 18). The unique identifier is formed and attached to a particular partition of forwarding when there are more than one possible routes to forward to (line 23). As in Fig. 2, a feasible unique identifier could be the index of $l_{j_i^m}^t$. Next, we calculate $U_i^{p_i^{\mathfrak{n}}}$ tables for each $p_i^{\mathfrak{n}} \in \mathfrak{P}_i$ as in the first algorithm (line 24). The rest of the second phase and third phase remain the same as that of the algorithm with fixed routing described previously (see lines 27-30).

4. EMPIRICAL EVALUATION

We now seek to evaluate their performance and effectiveness when applied to typical WVSN whose communication networks exhibit loops. Our goal in this empirical evaluation is to quantify the performance of the algorithms in terms of the quantity of information that they deliver to the base station, and the communication and computational costs of the coordination. We expect the algorithm with flexible routing to deliver more information, but make greater demands of computation and communication resources (because of the large number of alternative routes for the data that it must evaluate). However, given that the algorithm with fixed routing can always be applied in this setting by ignoring the fact that there exist alternative routing options, and just making an arbitrary choice, we are interested in the trade-off between the loss in information and the saving in resources that results. We first describe the experimental setup and then go onto the actual evaluation.

In our experiments, we create instances of a WVSN by randomly deploying the nodes within a unit square, and connecting them according to a randomly determined radio transmission range (extending this range as necessary to ensure that there are no unconnected nodes). Each resulting WVSN exhibits a loopy communication network such that for each node there are multiple alternative routes to the base station. We consider twenty different sampling actions for each node such that the possible sampling rates, C_i, of each node are initialized to $C_i = \{1, \ldots, 20\}$. The corresponding information content $v_i^{c_i^j}$ for each $c_i^j \in C_i$ sample is generated using the generic information metric (defined in Sect. 2), with the factor, α_i, randomly drawn from a uniform distribution with support $[0, 1]$. The energy budget of each node is randomly generated with a predetermined maximum value that ensures the network as a whole is energy constrained. We scale this predetermined max-

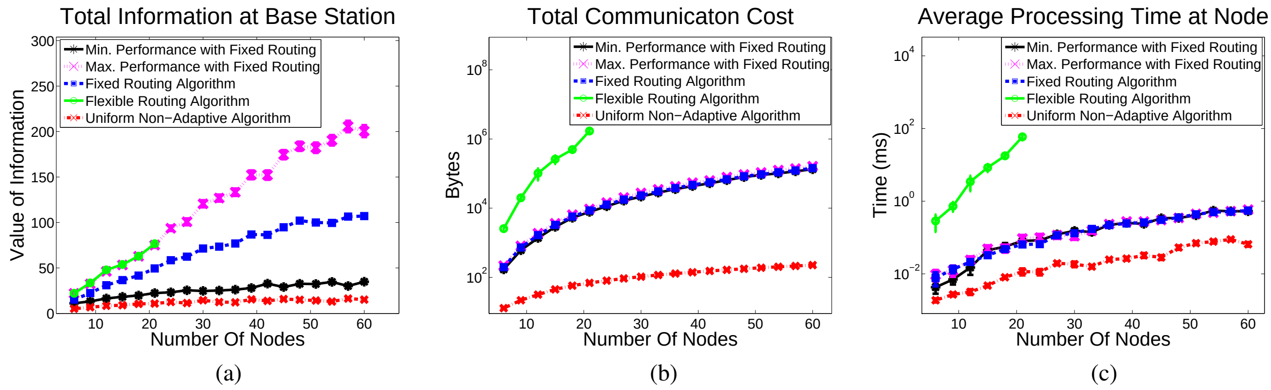


Figure 4: Simulation results showing the performance of the algorithms with flexible, fixed (with maximum and minimum performance), and uniform non-adaptive routing against (a) total information collected at the base station, (b) total communication cost for coordination, and (c) average computation time at each node.

imum value with the number of nodes in the network since larger networks require sensors to forward data for a larger number of nodes. We assume that each real valued number inside a coordination message (e.g. the value of B_i or c_i^j) occupies 4 bytes of communication cost, and the energy consumption for sensing and forwarding a sample is fixed throughout the entire experiment[3].

We apply our algorithm with flexible routing just once, directly on the loopy communication network of the WVSN (see Fig. 3(a) for an exemplar scenario), such that it determines both the optimal sensing and forwarding actions, as well as the routes. Prior to applying our algorithm with fixed routing, we allow each node to make an arbitrary choice of the route that its data (and any data that it forwards for other nodes) will take toward the base station. This effectively turns the loopy communication network into a tree-structured one, with each node effectively selecting their parent in the tree (see Fig. 3(b)). We then apply our algorithm with fixed routing to calculate the optimal sensing and forwarding decisions of each node. For each instance of the WVSN, we repeat this process 100 times, averaging over the unique instances of trees that result. We perform repeated experiments by creating 100 instances of the WVSN with 6, 9, . . . , 60 nodes for the algorithm with fixed routing, and only up to 21 nodes for the algorithm with flexible routing (due to its increased computational cost).

We also benchmark our two algorithms against a uniform non-adaptive algorithm with fixed routing. This algorithm dictates that each sensor $i \in I$ in the network should simply choose to allocate its energy budget, B_i, equally to itself and each of its descendants such that it will naïvely sample and transmit the minimum of $\left(\frac{B_i}{|\mathfrak{C}_i| \cdot E_i^{\mathrm{S}}}, \frac{B_{p_i}}{|\mathfrak{C}_{p_i}| \cdot E_{p_i}^{\mathrm{S}}}\right)$ times regardless of whether the samples will eventually be forwarded towards the base station. $|\mathfrak{C}_i|$ and $|\mathfrak{C}_{p_i}|$ are the numbers of descendants of node i and node i's parent, p_i, respectively, and B_{p_i} is the energy budget of node p_i. E_i^{S} and $E_{p_i}^{\mathrm{S}}$ are the energy required by node i and p_i correspondingly in order to sense a sample.

We present the results of the simulation process described above in Fig. 4. The error bars shown represent the standard error in the mean, and we note that in some cases, the error bars are smaller that the plotted symbol size. Considering first Fig. 4(a), we observe that the algorithm with flexible routing delivers close to twice the quantity of information to the base station as does the fixed routing algorithm. This is as expected since in loopy communication networks, there are typically many alternative routes available for routing data, and the flexible algorithm is able to exploit them[4]. The uniform non-adaptive algorithm, however, performs poorly as it has no intelligence of adapting the nodes' actions. In the same figure, we also show the mean maximum and minimum performance of the algorithm with fixed routing (averaged over different trees for the same loopy network). Note that by making an appropriate choice of parent, we can derive performance close to that of the algorithm with flexible routing (without incurring any additional computation or communication cost as will be explained shortly).

However, the increased information delivered by the algorithm with flexible routing comes at considerable communication and computational cost. Figures 4(b) and 4(c) show the total size of coordination messages exchanged by the nodes and the average computation time of each node (both are presented on a logarithmic scale). Specifically, Fig. 4(b) shows that typically only a few tens of kilobytes of coordination message packets are required by the algorithm with fixed routing, while the algorithm with flexible routing exhibits approximately two orders of magnitude more; with a few megabytes of coordination message packets being exchanged. Likewise, Fig. 4(c) shows that the average computation time of a node required by the algorithm with fixed routing is typically less than 1 millisecond, while that of the algorithm with flexible routing approaches 100 milliseconds (a two orders of magnitude increase)[5]. The increase in terms of computation time is due to the additional time which the flexible routing algorithm requires in order to enumerate each possible partitions of forwarding.

More generally, these results indicate that the algorithm with flexible routing is able to deliver significantly more information to the base station, but incurs considerable additional computation and communication costs in doing so. The choice of algorithm thus depends on the application domain. If the network is small, and the size of the actual data messages is large, then the algorithm

[3]Note that we do not consider the failure, addition, or removal of nodes. Also, we do not consider the dropping or corruption of meta-data or control message packets, and hence assume that message packets are always transferred successfully to the destination.

[4]We remark that the quantity of information delivered does not increase monotonically. This is an artifact of the experimental setup since the scaling of the nodes' energy budget does not fully account for the necessary increase in sample forwarding.

[5]Measurements were performed on a 3GHz desktop PC. Typically, the nodes within a WVSN will use much lower powered processors and, thus, while we would expect the ratio between the algorithms to be the same, the overall computation time is likely to be longer.

with flexible routing is most appropriate. However, this algorithm scales poorly as the size or connectivity of the network increases (due to the exponential growth in the number of possible combinations of routing options that it must evaluate). In such cases, the size of the coordination messages may rapidly approach that of the actual data messages and, hence, coordination may not actually yield any energy saving. To address this, the algorithm with fixed routing may be run on the original loopy network by having each node make an arbitrary choice of route. While the quantity of information delivered to the base station will be reduced (by up to 50% in our experiments), this solution will scale well and use minimal communication and computational resources.

5. RELATED WORK

The work that is most closely related to ours is that of Padhy *et al.* who developed a decentralised adaptive sampling and routing protocol named *Utility-based Sensing and Communication Protocol* [8]. Within this mechanism, each node adjusts its sampling rate depending on a valuation function that assigns a value to newly sampled data. This protocol is intended for low power, computationally constrained devices, and as such, relies on a heuristic approach to estimate the opportunity energy cost used by each sensor for sampling, forwarding, and routing data. The protocol is not efficient and the integration of the node's actions is very limited since there is no guarantee that the transmitted data will actually be forwarded to the base station. For instance, there might be cases where nodes with data of a high value are unable to send their data to the base station because intermediate nodes have depleted their energy. The protocol could thus result in no data collection.

In a somewhat similar setting, Mainland *et al.* present a market-based approach for determining efficient node resource allocations [5]. Rather than manually tuning node resource usage, or providing specific algorithms as we do here, their approach defines a virtual market in which nodes sell goods (e.g. data sampling, listening, or forwarding) in response to global price information that is established by the end user. However, this approach involves an external coordinator to set prices in order to induce any particular global behaviour, and it is not clear how this price determination should be performed in order to elicit desirable system-wide properties.

Within the multi-agent systems literature, another useful technique that has emerged for solving distributed coordination problems is that of *distributed constraint optimization* (DCOP). A number of algorithms in the area of DCOP have been developed; including *asynchronous distributed optimization* (ADOPT) [7] and *distributed pseudotree optimization procedure* (DPOP) [9]. Both are guaranteed to converge to the optimal solution while using only localized communication and computation. However, they are not specifically tailored to the specific problem that we address here, and since these algorithms are complete, they require an exponential increase in the total message size being exchanged (unlike the case of our algorithm with fixed routing). This is unrealistic for WVSNs in which the nodes are typically installed with limited computational, storage, and memory resources.

6. CONCLUSIONS

In this paper, we have considered the problem of adaptive sampling, forwarding, and routing within WVSNs in order to manage the limited energy resources of nodes in an effective and efficient way. We have developed two novel optimal decentralised algorithms: one which assumes fixed routing and calculates the optimal sensing and forwarding actions that each node should perform, and one which assumes flexible routing, and makes optimal decisions regarding both the integration of actions that each node should choose, and also the route by which this data should be forwarded to the base station. In an empirical evaluation, we showed that the algorithm with flexible routing delivered approximately twice the quantity of information to the base station, but at considerably higher communication and computational cost. Thus, while the algorithm with flexible routing is suitable for networks with a small numbers of nodes, it scales poorly, and as the size of the network increases, the algorithm with fixed routing is favoured.

Our ongoing work in this area includes relaxing the restriction that the nodes may only forward data to nodes that are closer to the base station (in terms of hop count) than themselves and, in particular, we would like to characterise the circumstances in which this may yield some benefit. More significantly, we would also like to develop a principled algorithm for making the choice of route when applying the algorithm with fixed routing to loopy WVSNs (rather than having the nodes make an arbitrary choice of parent in order to convert the loopy network into a tree-structured network as we have done here). Our empirical results indicate that the performance of the algorithm with fixed routing is very close to that of the algorithm with flexible routing if the appropriate fixed route is selected (see Fig. 4)[6], and thus, there is great potential in doing so.

7. REFERENCES

[1] M. Bramberger, A. Doblander, A. Maier, B. Rinner, and H. Schwabach. Distributed embedded smart cameras for surveillance applications. *Journal of IEEE Computer*, 39(2):68–75, 2006.

[2] J. Kho, A. Rogers, and N. R. Jennings. Decentralised control of adaptive sampling in wireless sensor networks. *ACM Transactions on Sensor Networks (In Press)*, 5(3), 2009.

[3] R. Kleihorst, B. Schueler, A. Danilin, and M. Heijligers. Smart camera mote with high performance vision system. In *Proceedings of the ACM SenSys Workshop on Distributed Smart Cameras*, October 2006.

[4] A. Ledeczi, A. Nadas, P. Volgyesi, G. Balogh, B. Kusy, J. Sallai, G. Pap, S. Dora, K. Molnar, M. Maroti, and G. Simon. Countersniper system for urban warfare. *ACM Transactions on Sensor Networks*, 1(2):153–77, 2005.

[5] G. Mainland, D. C. Parkes, and M. Welsh. Decentralised, adaptive resource allocation for sensor networks. In *Proceedings of the 2nd USENIX/ACM Symposium on Networked Systems Design and Implementation*, pages 315–28, 2005.

[6] G. Mathur, P. Desnoyers, D. Ganesan, and P. Shenoy. Ultra-low power data storage for sensor networks. In *Proceedings of the 5th International Conference on Information Processing in Sensor Networks*, pages 374–381, 2006.

[7] P. J. Modi, W. M. Shen, M. Tambe, and M. Yokoo. ADOPT: Asynchronous distributed constraint optimization with quality guarantees. *Artificial Intelligence*, 161(1-2):149–80, January 2005.

[8] P. Padhy, R. K. Dash, K. Martinez, and N. R. Jennings. A utility-based sensing and communication model for a glacial sensor network. In *Proceedings of the 5th International Conference on Autonomous Agents and Multiagent Systems*, pages 1353–1360, 2006.

[9] A. Petcu and B. Faltings. DPOP: A scalable method for multiagent constraint optimization. In *Proceedings of the International Joint Conferences on Artificial Intelligence*, pages 266–271, August 2005.

[10] F. Qureshi and D. Terzopoulos. Distributed coalition formation in visual sensor networks: A virtual vision approach. In *Proceedings of the 3rd IEEE International Conference on Distributed Computing in Sensor Systems*, pages 1–20, 2007.

[11] B. Rinner, T. Winkler, W. Schriebl, M. Quaritscha, and W. Wolf. The evolution from single to pervasive smart cameras. In *Proceedings of the 2nd ACM/IEEE International Conference on Distributed Smart Cameras*, pages 1–10, September 2008.

[12] P. Sinha and A. A. Zoltners. The multiple-choice Knapsack problem. *Operations Research*, 27(3):503–15, May-June 1979.

[13] T. Zahariadis and S. Voliotis. Open issues in wireless visual sensor networking. In *Proceedings of the 6th EURASIP Conference Focused on Speech and Image Processing, Multimedia Communications and Services*, pages 335–338, June 2007.

[6]Note the algorithms are not necessarily identical in this case, since the algorithm with flexible routing allows individual nodes to forward data through multiple routes.