Optimal In-Network Packet Aggregation Policy for Maximum Information Freshness

Alper Sinan Akyurek*, Tajana Simunic Rosing†
Electrical and Computer Engineering,
University of California, San Diego
*aakyurek@ucsd.edu, †tajana@ucsd.edu

Abstract—The Internet of Things is emerging as more wireless sensor networks get deployed every day, pushing decision making and processing towards the edge onto more constrained devices. In these constrained networks, using a single packet to transmit small sensed data is an inefficient use of both bandwidth and energy. Aggregating multiple measurements and packets into a single packet increases efficiency and resource utilization, but it also introduces the problem of deciding when to transmit aggregated data. In this work, we show theoretically that the classical performance metrics of energy consumption and expiration rate create a balanced tradeoff, where their combined metric of energy consumption per non-expired measurement is a constant, independent of policy selection. We introduce a new metric, information freshness, and derive an optimal policy to maximize it under both single hop and multi-hop cases. We provide a distributed algorithm to implement our optimal policy. Our case studies show that our algorithm outperforms state-of-the-art policies by more than 3.3x for information freshness and reduces energy consumption by more than 62%.

I. INTRODUCTION AND RELATED WORK

Time division multiple access (TDMA) is one of the most energy efficient ways for resource allocation [1], used in wireless sensor network (WSN) protocols such as IEEE 802.15.4e [2]. Although time division enables energy efficiency by turning the radio off during unused periods, having a fixed time slot can be inefficient, especially for small packets. In WSNs, where nodes transmit sensor readings to a common sink, it is possible to design the optimal transmission slot length. However, if the WSN has different types of sensors with different types of measurements, this may not be possible and the slot length must be over-provisioned. In such cases, packet aggregation within the network is a more efficient solution. Multiple measurements from different applications or even different nodes are aggregated into a single packet to maximize the efficiency. Furthermore, transmitting a single packet instead of segregated ones, saves energy, decreases end-to-end delay and increases the freshness of information.

Various algorithms in the literature consider in network packet aggregation. A disadvantage of small packets is that they may experience a wide range of delays, causing jitter. In [3], the authors propose using aggregation to create larger and fewer packets to mitigate jitter. They provide a heuristic for real-time applications, yet consider only a single application and optimality is not guaranteed. Another application of aggregation is used in Voice over Internet Protocol (VoIP) [4], where delay and jitter are important for quality of service. The authors propose a timeout based aggregation policy for the single application of VoIP. The proposed policy maximizes the number of concurrent calls in the network while reducing the expiration rate and jitter. Interesting work considered the effect of aggregation on Transmission Control Protocol (TCP) performance [5]. The results show that aggregation increases the total capacity, throughput, and TCP efficiency while decreasing the end-to-end delay and wireless access times.

In [6], a timeout based distributed aggregation protocol is proposed. Timeouts are based on either a fixed interval or a cascading interval based on the hop distance. However, the policies do not consider the different requirements for multiple applications. For a broader look at the literature, see [7]. If a node chooses to wait for a long time before transmitting its queue, it can aggregate more measurements into a single packet, increasing its efficiency and reducing its energy consumption, but it also increases the delay of all queued measurements, causing problems for delay sensitive applications, creating a tradeoff between energy and delay.

In this work, we first present theoretical analysis in terms of energy consumption and measurement expiration rate, and show that the energy consumption per successful measurement is a constant, independent of the policy selection. This means that any improvement in energy consumption results in an equal degradation on expiration rate and vice versa. To mitigate this tradeoff, we define a new metric: information freshness or freshness in short. We define freshness as the time a measurement has left until its expiration deadline. Freshness measures how new the sensed information is and it affects the total performance negatively once it expires. Freshness of each measurement can be maximized by transmitting them right after generation, but this would be very costly from energy consumption and resource utilization perspectives. We provide the optimal transmission policy that maximizes the total freshness per transmission, thus per energy consumption. We extend our solution to a multi-hop solution without the loss of optimality and provide a distributed algorithm to implement our policy with low overhead. Our case studies show that we can increase the information freshness per transmission by a factor of more than 3.3x while reducing the energy consumption by 62% compared to the state-of-the-art.

II. OPTIMAL POLICY FOR INFORMATION FRESHNESS

We consider a multi-hop network, where each node decides when to transmit the measurements in its queue. We define
multiple applications that generate their own measurements. Each application has its own generation and deadline requirements. As an example, in a network with environmental sensor boards, temperature can be sensed rapidly and has loose deadlines, whereas gas sensors such as carbon dioxide, need a longer time to sense, but are more time critical. A measurement is expired, when the reception time at the sink exceeds its deadline requirement. We assume that each transmission has a fixed length, fixed capacity and consumes fixed amount of energy. The radio is turned off during idle times to save energy. Freshness is the time a measurement has left until its deadline, meaning that an expired measurement still contributes negatively to the freshness metric.

A policy decides when to transmit a packet that contains all measurements in its queue, up to the capacity of the packet. We first prove that it is theoretically impossible to improve energy consumption and expiration rate at the same time, by showing that the energy per number of non-expired measurements is independent of the policy. Then we provide an optimal policy that maximizes the expected information freshness per packet for both single hop and multi-hop cases.

A. Theoretical Analysis of Transmission Intervals

Let \( \pi \) be a policy that determines, when a packet is to be transmitted. For a \( T \) number of packet transmissions, where any transmission interval \( j \) is of length \( \Delta T_j \), the total time of transmission analysis is \( T_{\text{total}} = \sum_{j=1}^{T} \Delta T_j \). We assume that the transmission happens at the beginning of the interval and any measurement that arrives during the transmission cannot be added to the ongoing transmission. If each packet transmission consumes energy \( E \), the total energy consumption is \( E_{\text{total}} = E \times T \) and the average energy consumption is: \( E_{\text{avg}} = E \times T / \sum_{j=1}^{T} \Delta T_j \)

Let \( D_i \) be the deadline for application \( i \). Within an inactive interval of length \( \Delta T_j \), the measurements that arrive within the last part of \( D_i \) do not expire, whereas the rest do. The success ratio (SR) of packets can be written as follows, where \( N \) is the number of applications and \( p_{i,j}(t, \Delta t, k) \) is the probability of generating \( k \) packets within the last \( t \) seconds of the \( j^{th} \) interval of length \( \Delta t \) for application \( i \):

\[
SR = \frac{\sum_{i=1}^{N} \sum_{j=1}^{T-1} \sum_{k=1}^{\infty} p_{i,j}(\min(D_i, \Delta T_j), \Delta T_j, k), k}{\sum_{i=1}^{N} \sum_{j=1}^{T-1} \sum_{k=1}^{\infty} p_{i,j}(\Delta T_j, \Delta T_j, k), k}
\]

Finally, the energy consumption per successful measurement is obtained as:

\[
E_{\text{success}} = \frac{E}{\sum_{i=1}^{N} \sum_{j=1}^{T-1} \sum_{k=1}^{\infty} p_{i,j}(\min(D_i, \Delta T_j), \Delta T_j, k), k}
\]

Assuming that the average packet generation rate for application \( i \) is \( \lambda_i \), the expression can be rewritten as:

\[
E_{\text{success}} \approx \frac{E}{\sum_{i=1}^{N} \sum_{j=1}^{T-1} \min(D_i, \Delta T_j)}
\]

These three metrics tell us that: 1) Energy is minimized by waiting as much as possible, 2) success ratio is maximized when the transmission interval is equal to the maximum application deadline value, transmitting as fast as possible, 3) energy per successful measurement is minimized when \( \Delta T_j < D_i \) for all applications and the expression is independent of the \( T_j \) and policy selection.

This means that there is no meaning in claiming optimality for energy consumption or expiration rate, as any improvement in one metric results in an equal degradation in the other. Thus, in this paper, we focus on a new metric of importance: information freshness or freshness in short. Freshness of a measurement is defined as the amount of time it has left until its deadline: \( \text{Freshness}(t) = D_i - (t - t_{\text{gen}}) \).

B. Single Hop Optimal Policy

We start with a single hop scenario, where the source of measurements directly transmit to the sink. We consider random deadlines for each application, where the probability distribution is given as \( q_i(d) \), which gives the probability of having \( d \) as the deadline of an individual measurement. We assume that a single packet requires a fixed amount of energy to transmit and is big enough to hold all measurements in the queue. Starting from an arbitrary point of time, we consider the problem of deciding in a predictive and proactive way, how long to wait before transmitting all measurements in the queue, such that the total freshness (TF) is maximized.

\[
TF = \max_x x \sum_{i=1}^{N} \sum_{k=0}^{\infty} \int \int (l - x + t) q_i(l) p_i(k, t) k dt dl
\]

Taking the expectations and using \( \lambda_{i,t} \triangleq \sum_{k=0}^{\infty} p_i(k, t) k \):

\[
TF = \max_x x \sum_{i=1}^{N} \int (E(D_i) - x + t) \lambda_{i,t} dt
\]

We consider two ways of generating measurements: 1) with a generation process that doesn’t change within the interval that we are considering, so that \( \lambda_{i,t} = \lambda_i \), 2) with a periodic deterministic process.

For the first case, where the rate of measurement generation is time independent within an interval, (5) reduces to a quadratic function:

\[
TF = \max_x x \sum_{i=1}^{N} E(D_i) \lambda_i - \frac{x^2}{2} \sum_{i=1}^{N} \lambda_i
\]

\( TF \) is concave and the maximum is the mean of its two roots:

\[
x^* = \frac{1}{\sum_{i=1}^{N} E(D_i) \lambda_i} \sum_{i=1}^{N} E(D_i) w_i
\]

The weight of application \( i, w_i \), is the ratio of \( \lambda_i \) to the sum of all arrival rates. The resulting optimal transmission time that maximizes the total freshness per transmission is a weighted average of application deadlines. Furthermore, the
transmission time doesn’t change if the generation process stays the same, resulting in a periodically transmitting policy implementable using TDMA schemes.

For the second case, the rate of generation is deterministic, where we calculate the Dirac-Delta function: \( \lambda_{t,t} = \sum_{n=0}^{\infty} \delta(t - nT_l) \). Using this definition, (5) becomes:

\[
\max_{x} \left[ \sum_{i=1}^{N} N_i \left( E(D_i) - x + T_i \left( N_i + 1 \right) \right) \right] \tag{8}
\]

\( N_i \) is defined as \( x/T_i \). Denoting the remainder of the floor operation as \( M_i \), such that \( x = N_i T_i + M_i \):

\[
\max_{x} \left[ x \sum_{i=1}^{N} \left( \frac{E(D_i)}{T_i} + 0.5 \right) - x^2 \sum_{i=1}^{N} \frac{1}{2T_i} \right] \tag{9}
\]

We have a quadratic concave function as in the previous case, and denoting \( \lambda_i = 1/T_i \), the optimal solution is:

\[
x^* = \left[ \frac{N}{2} + \sum_{i=1}^{N} E(D_i) \lambda_i \right] \left[ \sum_{i=1}^{N} \lambda_i \right] \triangleq T_H/2 + \sum_{i=1}^{N} E(D_i) w_i
\]

The result is similar to the continuous case, except an extra term of \( T_H \), which is the harmonic mean of all \( T_i \).

C. Multi Hop Optimal Policy

We extend our single hop policy into a multi-hop one, where each node has its own applications with their own deadlines and generation processes. We consider a linear path to the sink, which represents a branch of the routing tree. We assume that aggregated packets are further re-aggregated with new information at each hop to maximize the packet usage efficiency. We update our previous definitions to generalize into their multi-hop counterparts by adding an additional index of \( h \), representing the hop distance to the sink node. As an example, \( N_h \) represents the number of applications of the node \( h \) stamp inside, periodically. The receiving node at \( h \), denoting the floor operation as \( M \), such that \( x = N_i T_i + M_i \):

\[
\max_{x} \left[ x \sum_{i=1}^{N} \left( \frac{E(D_i)}{T_i} + 0.5 \right) - x^2 \sum_{i=1}^{N} \frac{1}{2T_i} \right] \tag{9}
\]

We have a quadratic concave function as in the previous case, and denoting \( \lambda_i = 1/T_i \), the optimal solution is:

\[
x^* = \left[ \frac{N}{2} + \sum_{i=1}^{N} E(D_i) \lambda_i \right] \left[ \sum_{i=1}^{N} \lambda_i \right] \triangleq T_H/2 + \sum_{i=1}^{N} E(D_i) w_i
\]

The result is similar to the continuous case, except an extra term of \( T_H \), which is the harmonic mean of all \( T_i \).

For the second case, the rate of generation is deterministic, where we calculate the Dirac-Delta function: \( \lambda_{t,t} = \sum_{n=0}^{\infty} \delta(t - nT_l) \). Using this definition, (5) becomes:

\[
\max_{x} \left[ \sum_{i=1}^{N} N_i \left( E(D_i) - x + T_i \left( N_i + 1 \right) \right) \right] \tag{8}
\]

\( N_i \) is defined as \( x/T_i \). Denoting the remainder of the floor operation as \( M_i \), such that \( x = N_i T_i + M_i \):

\[
\max_{x} \left[ x \sum_{i=1}^{N} \left( \frac{E(D_i)}{T_i} + 0.5 \right) - x^2 \sum_{i=1}^{N} \frac{1}{2T_i} \right] \tag{9}
\]

We have a quadratic concave function as in the previous case, and denoting \( \lambda_i = 1/T_i \), the optimal solution is:

\[
x^* = \left[ \frac{N}{2} + \sum_{i=1}^{N} E(D_i) \lambda_i \right] \left[ \sum_{i=1}^{N} \lambda_i \right] \triangleq T_H/2 + \sum_{i=1}^{N} E(D_i) w_i
\]

The result is similar to the continuous case, except an extra term of \( T_H \), which is the harmonic mean of all \( T_i \).

\[
C. Multi Hop Optimal Policy
\]

We extend our single hop policy into a multi-hop one, where each node has its own applications with their own deadlines and generation processes. We consider a linear path to the sink, which represents a branch of the routing tree. We assume that aggregated packets are further re-aggregated with new information at each hop to maximize the packet usage efficiency. We update our previous definitions to generalize into their multi-hop counterparts by adding an additional index of \( h \), representing the hop distance to the sink node. As an example, \( N_h \) represents the number of applications of the node \( h \) hops away from the sink. \( H \) is the total number of hops.

We start with the global optimization problem of maximizing the total freshness per a series of transmission to the sink, where the system has a common period of \( x \) and individual offsets at each hop of \( x_h \), illustrated in Figure 1.

Our optimization variable for multi-hop becomes:

\[
\max_{x} \sum_{h=1}^{H} \sum_{i=1}^{N_h} \int_{0}^{x} \left( E(D_{i,h}) + t - \sum_{h'=1}^{h} x_{h'} \right) \lambda_{i,t,h} dt
\]

\( x_i \) is a quadratic concave function, where the maximum can be found through the partial derivatives:

\[
\frac{\partial T_F}{\partial x} = \sum_{h=1}^{H} \sum_{i=1}^{N_h} \left( E(D_{i,h}) - x - \sum_{h'=1}^{h-1} x_{h'} \right) \lambda_{i,h} = 0 \tag{11}
\]

\[
\frac{\partial T_F}{\partial x_k} = -x \sum_{h=1}^{H} \sum_{i=1}^{N_h} \lambda_{i,h} (H - k) = 0 \tag{12}
\]

Note that (13) has no direct solution. The only feasible solution would be to assign \( x_h = 0 \) to remove (13) completely. Since a zero offset is not physically possible due to transmission and processing delays, we need to modify our optimization. We impose the constraints of \( x_h \geq S_h \), where \( S_h \) is the required time for transmission and processing delay. Since the minimization version of the optimization problem is convex, we can easily solve for the optimal point by using the Lagrangian dual, where \( \mu \) is the K.K.T. multiplier:

\[
L(x, \mu) = \sum_{h=1}^{H} \sum_{i=1}^{N_h} \left( E(D_{i,h}) - x - \sum_{h'=1}^{h-1} x_{h'} \right) \lambda_{i,h} + \sum_{h=1}^{H} \mu_h (x_h - S_h)
\]

\( \frac{\partial L}{\partial x} = \sum_{h=1}^{H} \sum_{i=1}^{N_h} \left( E(D_{i,h}) - x - \sum_{h'=1}^{h-1} x_{h'} \right) \lambda_{i,h} = 0 \tag{14}
\]

\( \frac{\partial L}{\partial x_k} = \mu_h - x \sum_{h=1}^{H} \sum_{i=1}^{N_h} \lambda_{i,h} (H - k) = 0, \forall k \tag{15}
\]

\( \mu_h > 0 \) and \( \mu_h \neq 0 \) for all \( \mu_h \). Due to complementary slackness, if \( \mu_h > 0 \), then the constraint must be at its border, such that \( x_h - S_h \leq 0 \).

Using this result in the first partial derivative expression (15), we obtain the main transmission period for the whole system:

\[
x^* = \frac{\sum_{h=1}^{H} \sum_{i=1}^{N_h} \left( E(D_{i,h}) - \sum_{h'=1}^{h} S_{h'} \right) \lambda_{i,h}}{\sum_{h=1}^{H} \sum_{i=1}^{N_h} \lambda_{i,h}}
\]

D. Distributed Solution for Multi Hop Optimal Policy

The optimal solution in (17) requires synchronization and consensus between the nodes. The best efficiency is obtained if all nodes wake up with a period of \( x^* \) even though it is not mandatory for the policy to work. The only node that requires the \( x^* \) value is the farthest node from the sink. We need information flow from sink to the farthest node to calculate \( x^* \). To achieve this, sink starts an update message with a time stamp inside, periodically. The receiving node at \( h = 1 \) uses the time stamp to calculate \( S_1 \) and adds the following values:

\[
P_i \triangleq \sum_{i=1}^{N_h} E(D_{i,1}) \lambda_{i,1} \quad R_1 \triangleq \sum_{i=1}^{N_h} \lambda_{i,1} \quad \tau_i \triangleq \sum_{i=1}^{N_h} \lambda_{i,1} S_1
\]
The receiving node at \( h = 2 \) uses the initial time stamp to calculate \( S_1 + S_2 \). It adds its own \( P_2, R_2 \) and \( \tau_2 \) values to the received ones and sends the message to the next hop. This iteration is repeated until the final node, which calculates the optimal transmission interval as:

\[
x^* = \frac{\sum_{h=1}^{H} (P_h - \tau_h)}{\sum_{h=1}^{H} R_h}
\]

To maximize efficiency and minimize \( x_h \) values, the calculated \( x^* \) value can be included in the packets towards sink, so that all nodes have the same period.

### III. Results

We performed multiple case studies, where we compared our algorithm against two state of the art algorithms: 1) Fixed [6]: transmits periodically, where the period is the average sampling interval of the applications. We also added two more periods of \( 2x \) and \( 4x \) of the original period for more comparison, 2) Cascading [6]: each node transmits periodically, depending on the hop distance to the sink.

We consider three cases: 1) single hop deterministic generation, 2) single hop Poisson generation, 3) multi-hop Poisson generation. The simulation is implemented in MATLAB using an event based scheme to capture the timing details of the scenarios. For all scenarios, we provide the results for total freshness per transmission, total energy consumption and expiration ratio. The energy consumption is calculated by scaling the total number of transmissions with the energy required to transmit and receive by the radios for the duration of the packet. For a CC-2650, the scale corresponds to 0.3mJ approximately [8]. The radio is then turned off to save energy until the next transmission. The simulation runtime is 1 hour and the arrival rate is 2 measurements/sec, corresponding to an average of 7200 measurements per application per node.

We use various deadlines, explained in each case, separately.

#### A. Single Hop - Deterministic Generation

For the deterministic single hop case, we consider two scenarios: 1) a single application, where the measurement deadlines are determined by a Gaussian random variable; 2) two applications with constant deadlines.

The results for the Gaussian case is shown in Figure 2. We set the mean at 5 seconds and increase the standard deviation to observe the effect of randomness on performance. Since the expected deadline is fixed, the optimal transmission period is also fixed, resulting in same energy consumption and freshness across all deviations. Our policy is 1.6x better in terms of freshness and reduces the energy consumption by 62% compared to the closest policy (Fixed 2s). The expiration rate increases with increased randomness, due to higher probability of having deadline values within the expiration range. 10% of measurements expire on average using our policy.

The results for the constant deadline case are shown in Figure 3. We set different deadlines for the two applications to observe how a non-uniform deadline distribution affects the performance. Since the mean deadline is the same across all cases, the optimal transmission period is also the same, leading to same energy consumption and freshness across different deadline pairs. Our policy outperforms in terms of freshness by a factor of 1.5x and consumes 62% less energy than the closest policy (Fixed 2s). However, expiration rate increases rapidly with increasing deadline diversity.

#### B. Single Hop - Poisson Generation

We consider a single hop case, where the source node generates measurements using a Poisson process. The deadlines are determined by a Gaussian random variable: \( N(5,1) \). In this scenario, we increase the number of applications running on the source node, to observe the effect of number of applications on the performance. The results are shown in Figure 4. The expected deadline for all applications is the same, resulting in the same optimal transmission period for all cases, resulting in fixed energy consumption. Freshness increases with increasing number of applications since more measurements are generated and can be transmitted. Our policy is 1.52x better in terms of freshness and reduces energy consumption by 60% compared to the closest policy (Fixed 2s). 8% of measurements expire under our policy.

#### C. Multi-Hop - Poisson Generation

For the multi-hop scenarios, we set the propagation and processing delay between each node using a uniform random variable between \([0,500]\) ms to maximize randomness. We consider a scenario, where we vary the number of hops from 1 to 9 for 5 different mean deadline values, with 2 applications running on each node. We use a uniform distribution to
determine the deadline of each measurement. The freshness results are shown in Figure 5. The values on the x-axis represents the upper limit of the uniform deadline random variable, whereas the lower limit is 1 seconds. The results show that our policy has the highest freshness value. The freshness increases with number of hops, because the number of applications also increases with network size. Furthermore, freshness also increases with increasing deadline as expected, as freshness is a measurement of remaining lifetime until deadline. For the $D = 5$ cases, it can be observed that there is a decline in freshness after 6 hops. This is due to the fact that at 6 hops, the average distance to the sink node is 1.5 seconds. Considering that the average deadline is 3 seconds, there is a probability of never being able to reach the sink node and this probability increases with the number of hops. Our policy is optimal in terms of freshness as expected and outperforms the closest policy (Fixed 2s) by a factor of 3.3x on average.

The expiration rate results are given in Figure 6. Expiration rate decreases with increasing deadline upper limits due to increased flexibility and increases with increasing number of hops due to increased propagation and processing delays. Although the minimum delay to the sink increases with increasing network size, the effect is least observable with our policy since it considers these distances in the calculation of the optimal transmission period. 22% of measurements expire on average under our policy. The energy consumption results are given in Figure 7. All other policies have fixed energy consumption since their transmission period is independent of the deadline values. For our policy, the energy consumption decreases with increasing delay as the transmission period is directly proportional to it. Our algorithm has the lowest energy consumption and reduces the consumption on average by 62% compared to the closest policy (Fixed 2s).

IV. CONCLUSION

Among network aggregation techniques, packet aggregation is one of the low-cost techniques to improve resource utilization efficiency. In this paper, we first show theoretically that classical metrics of energy consumption and expiration rate have a direct tradeoff and their combined metric of energy consumption per successful measurement is constant and independent from policy selection. We then present an optimal policy that maximizes the information freshness of packets by aggregating multiple measurements from multiple hops into a single packet. We provide a distributed algorithm to implement our optimal policy with a very low communication overhead of $O(N)$. In case studies, we show that we achieve the maximum freshness with an improvement factor of more than 3.3x over the state of the art policies while reducing the energy consumption by more than 62%.

ACKNOWLEDGMENT

This work was supported in part by TerraSwarm, one of six centers of STARnet, a Semiconductor Research Corporation program sponsored by MARCO and DARPA; and in part by NSF grant #1344153.

REFERENCES