Traffic Decorrelation Techniques for Countering Colluding Eavesdroppers in WSNs

Alejandro Proaño, Loukas Lazos, and Marwan Krunz
Dept. of Electrical and Computer Engineering, University of Arizona, Tucson, AZ, USA
E-mail:{aaproano, llazos, krunz}@ece.arizona.edu

APPENDIX

A. Proof of Proposition 1

Proposition 1: When sensors transmit one packet at a random time within an interval of T, the average number of hops that a packet can traverse per T is 1.72.

Proof: Consider a source v with a hop-count $\eta = |p(v,u)|$ to the destination u. Nodes in $p(v,u)$ randomly select their transmission times within an interval T. Let the randomly selected transmission times be denoted by $t_1, t_2, \ldots, t_\eta$. We compute the average number of hops (H) traversed over T by a packet m originating at v_1, using a combinatorial approach. The value of H depends on the arrangement order of $t_1, t_2, \ldots, t_\eta$. All possible arrangements $n_\eta!$ occur with equal probability, as the transmission times of each sensor are randomly and independently selected within T.. A packet traverses η hops during T, only if $t_1 < t_2 < \ldots < t_\eta$. This events occurs with probability,

$$\Pr(H = \eta) = \frac{1}{\eta!}.$$

Similarly, to find the probability of traversing $\eta - 1$ hops, we consider the number of arrangements that satisfy $t_1 < t_2 < \ldots < t_{\eta-1}$, but not $t_{\eta-1} < t_\eta$. Thus, we obtain,

$$\Pr(H = \eta - 1) = \frac{1}{\eta!}\left(\frac{\eta!}{(\eta-1)!} - 1\right).$$

For an arbitrary number of hops $x \leq \eta - 1$ we get,

$$\begin{aligned}\Pr(H = x) &= \frac{1}{\eta!}\left(\frac{\eta!}{x!} - \frac{\eta!}{(x+1)!}\right)\\ &= \frac{x}{(x+1)!}.\end{aligned}$$

Computing the expectation of H

$$\begin{aligned}E(H) &= \frac{\eta}{\eta!} + \sum_{x=1}^{\eta-1}\frac{x^2}{(x+1)!}\\ &= \frac{1}{(\eta-1)!} + \sum_{x=1}^{\eta-1}\frac{x^2}{(x+1)!}.\end{aligned}$$

By applying the ratio test of series on the second term of $E(H)$, we obtain

$$\begin{aligned}\lim_{x\to\infty}\frac{\frac{(x+1)^2}{(x+2)!}}{\frac{x^2}{(x+1)!}} &= \lim_{x\to\infty}\frac{(x+1)^2(x+1)!}{x^2(x+2)!}\\ &= \lim_{x\to\infty}\frac{x^2+2x+1}{x^3+2}\\ &\leq 1,\end{aligned}$$

which proves that $E(H)$ is a converging series. Finally, we compute the convergence value of $E(H)$ via numerical analysis for all $\eta \geq 1$,

$$E(H) = 1.72.$$

□

B. Proof of Proposition 2

Proposition 2: Stage 3 terminates in less than $\delta_{\max} + 1$ iterations, where $\delta_{\max}$ is the maximum degree of any node in the network.

Proof: By contradiction. Assume that Algorithm 2 does not terminate. That is, there exist a $v \in \mathcal{V}$ that is *gray* for each iteration of Algorithm 2. Based on Step 3 of Stage 1, a node u becomes *black* if

$$v = \arg\max_{u \in [\mathcal{N}_v^2]}\left\{\frac{\delta^*(u)}{\delta^*_{\max}(v)} \times \frac{1}{f(u)+1}\right\}.$$

In the worst-case scenario, v is a leaf node that cannot become *black* during Stage 2. Each time Stage 1 is executed, v becomes *gray* and is dominated by a node $u \in \mathcal{N}_v$ if,

$$\frac{\delta^*(v)}{\delta^*_{\max}(v)} \leq \frac{\delta^*(u)}{\delta^*_{\max}(u)} \times \frac{1}{f(u)+1},$$

or,

$$\delta^*(v) \leq \frac{\delta^*(u)}{\delta^*_{\max}(u)} \times \frac{\delta^*_{\max}(v)}{f(u)+1},$$

where $\delta^*_{\max}(v)$ is the maximum degree in $\mathcal{N}_v^2$ and $\delta^*_{\max}(u)$ is the maximum degree in $\mathcal{N}_u^2$. Note that because $\mathcal{G}$ is connected, $\delta^*(v) \geq 1$, and

$$1 \leq \frac{\delta^*(u)}{\delta^*_{\max}(u)}.$$

Hence,

$$1 \leq \frac{\delta^*_{\max}(v)}{f(u)+1}.$$

Because we assumed Algorithm 2 does not terminate and $\delta^*_{\max}(v)$ is finite, at some iteration n' of Stage 3, $f(u) = \delta^*_{\max}(v)$ for all $u \in \mathcal{N}_v$. Therefore, during iteration $n'+1$ we have $\frac{\delta^*_{\max}(v)}{f(u)+1} < 1$ and,

$$\delta^*(v) \geq \frac{\delta^*(u)}{\delta^*_{\max}(u)} \times \frac{\delta^*_{\max}(v)}{f(u)+1}.$$

In this case, v becomes *black* during Stage 1, contradicting the assumption that v is always *gray*. Moreover, $n' = \delta^*(v) \times \delta^*_{\max}(v)$. Also, note that in the worst case $\delta^*(v) = 1$ and $\delta^*_{\max}(v) = \delta_{\max}$, where $\delta_{\max}$ is the maximum degree in the network. Thus, $n' = \delta_{\max}$ and the maximum number of iterations of Stage 3 is $\delta_{\max} + 1$.

□

C. Proof of Proposition 3

Proposition 3: The problem of finding an SS-MCDS in arbitrary graphs is NP-complete.

Proof: We first show that the problem of computing a single SS-MCDS is NP-complete. To do so, we prove that SS-MCDS is both in NP and it is at least as hard as the Minimum Shortest-path Steiner arborescence (MSPSA) problem, that is known to be NP-complete [1]. We first define the MSPSA.

Definition 1 (Minimum Shortest-path Steiner arborescence): Given a graph $\mathcal{G}(\mathcal{V}, \mathcal{E}(\mathcal{V}))$ with positive edge weights, a target set $\mathcal{T} \subseteq \mathcal{V}$ and a unique root node $\mu \in \mathcal{T}$, a *shortest path Steiner arborescence* S is a Steiner tree rooted at μ, spanning all vertices in $\mathcal{T}$ such that each path $p(u, \mu)$ is a shortest path in $\mathcal{G}(S, \mathcal{E}(S))$, for all $u \in \mathcal{T}$. The arborescence of smallest cardinality is called minimum shortest-path Steiner arborescence (MSPSA) [2].

The following verifier for the SS-MCDS problem runs in polynomial time in the size of the input SS-MCDS ($\mathcal{D}$). In the verifier, $p(u, \mu)$ denotes the shortest path between u and the root node μ in $\mathcal{G}(\mathcal{V}, \mathcal{E}(\mathcal{V}))$, while $p_{\mathcal{D}}(u, \mu)$ denotes the shortest path between u and μ in $\mathcal{G}(\mathcal{D}, \mathcal{E}(\mathcal{D}))$.

SS-MCDS Verifier

Input: $\mathcal{G}(\mathcal{V}, \mathcal{E}(\mathcal{V})), \mathcal{D}$, and $p(v, \mu)$ for all $v \in \mathcal{V}$.

Execution: If the following are true accept, else reject:

1) For all $v \in \mathcal{V}$, either v is in $\mathcal{D}$, or there exist a $u \in \mathcal{D}$ for which $u \in \mathcal{N}(v)$
2) Nodes in $\mathcal{D}$ form a connected subgraph, if so obtain $p_{\mathcal{D}}(u, \mu)$ for all u in $\mathcal{D}$
3) For all $u \in \mathcal{D}$, $|p(u, \mu)| = |p_{\mathcal{D}}(u, \mu)|$

In Step 1, the SS-MCDS verifier checks whether $\mathcal{D}$ satisfies the DS property. If v belongs to $\mathcal{D}$, the check requires only one computation. However, in the worst case, v is not in $\mathcal{D}$ and the algorithm goes through v's 1-hop neighborhood looking for a $u \in \mathcal{D}$. Since the maximum size of $\mathcal{N}(v)$ is $|\mathcal{V}|-1$, the maximum cost of this operation is $O(|\mathcal{V}|)$. Therefore, the cost of Step 1 is $O(|\mathcal{V}|^2)$ or $O(|\mathcal{D}|^2)$ (since $|V| = O(|\mathcal{D}|)$). In Step 2, connectivity is tested by the Floyd and Warshall's Algorithm, which is of complexity $O(|\mathcal{D}|^3)$ [2]. This algorithm also outputs the shortest paths, $p_{\mathcal{D}}(u, \mu)$, between $u \in \mathcal{D}$ and μ, used in Step 3. Finally, in Step 3, each node v compares the lengths of $p(v, \mu)$ and $p_{\mathcal{D}}(v, \mu)$. The cost of this operation is $O(|\mathcal{D}|)$. Thus, the total cost of the the SS-MCDS Verifier is $O(|\mathcal{D}|^3)$. As the SS-MCDS Verifier runs in polynomial time, the SS-MCDS problem is in NP.

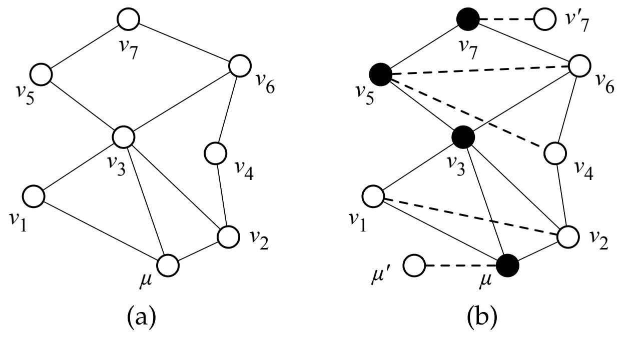

Fig. 1: (a) $\mathcal{G}(\mathcal{V}, \mathcal{E}(\mathcal{V}))$, (b) $\mathcal{G}'(\mathcal{V}, \mathcal{E}(\mathcal{V}'))$ and (S, K) formed by black nodes.

We now show that the SS-MCDS problem is NP-Hard. We first prove that MSPSA $\leq_P$ SS-MCDS. For the MSPSA problem, we define the target set as,

$$\mathcal{T} = \{v : |p(v, \mu)| \geq |p(u, \mu)|, \forall u \in \mathcal{N}_v\} \cup \{\mu\}.$$

Set $\mathcal{T}$ contains all leaf vertices of $\mathcal{G}$ (i.e., the set of nodes for which all their neighbors are closer to the origin μ) plus μ. Let function f take input graph $\mathcal{G}(\mathcal{V}, \mathcal{E}(\mathcal{V}))$, with an edge cost $w(u, v) = 1, \forall (u, v) \in \mathcal{E}(\mathcal{V})$, and an origin vertex μ. Function f constructs $\mathcal{G}'(\mathcal{V}', \mathcal{E}(V'))$ as follows:

- For each $v \in \mathcal{T}$, define a virtual node v'. Assign all nodes v' to set $\mathcal{X}$, and make
$$\mathcal{V}' = \mathcal{V} \cup \mathcal{X},$$
- Connect each v to the corresponding v' and set the edge cost $c(v', v) = 0$. Edges $(v, v'), \forall v \in \mathcal{V}$ form edge set $\mathcal{E}_1$.
- Connect each u and v in $\mathcal{V}$ with $|p(u, \mu)| = |p(v, \mu)|$ and $(u, v) \notin \mathcal{E}(\mathcal{V})$, and set the edge cost to $c(u, v) = 1$. These edges form set $\mathcal{E}_2$.
- Define $\mathcal{E}(\mathcal{V}')$ as,
$$\mathcal{E}(\mathcal{V}') = \mathcal{E}(\mathcal{V}) \cup \mathcal{E}_1 \cup \mathcal{E}_2.$$

To demonstrate the construction of $\mathcal{G}'$, consider the graph $\mathcal{G}(\mathcal{V}, \mathcal{E}(\mathcal{V}))$ presented in Fig. 1(a). Note that, the only leaf node is v_7, which makes $\mathcal{T} = \{\mu, v_7\}$. To generate $\mathcal{G}'(\mathcal{V}', \mathcal{E}(V'))$, we first define the set of virtual nodes $\mathcal{X} = \{\mu', v'_7\}$, and their respective 0-cost links $\mathcal{E}_1 = \{(\mu, \mu'), (v_7, v'_7)\}$. Finally, we add links $\mathcal{E}_2 = \{(v_1, v_2), (v_4, v_5), (v_5, v_6)\}$ of unitary cost, to connect nodes with no links in $\mathcal{E}(\mathcal{V})$ that have the same hop-count to μ. The resulting graph $\mathcal{G}(\mathcal{V}', \mathcal{E}(\mathcal{V}'))$ is shown in Fig. 1(b).

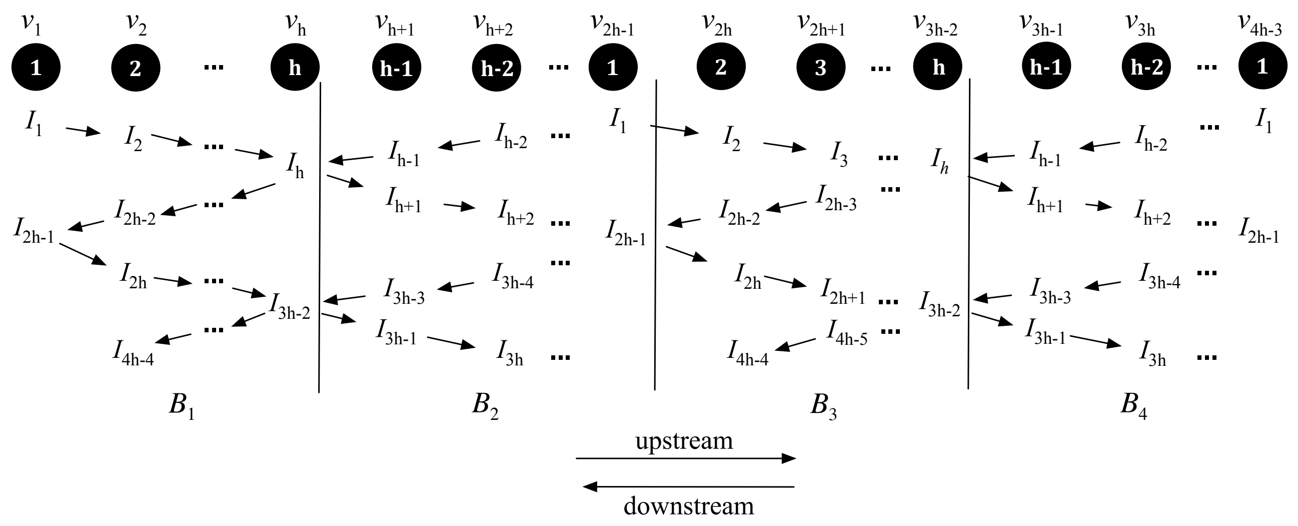
v_1 v_2 v_h v_{h+1} v_{h+2} v_{2h-1} v_{2h} v_{2h+1} v_{3h-2} v_{3h-1} v_{3h} v_{4h-3}
1 2 h h-1 h-2 1 2 3 h h-1 h-2 1
B_1
B_2
B_3
B_4
upstream
downstream

Fig. 2: Transmission schedule for a path of length $4h-3$ according to the DFAS algorithm.

We first prove that if $(\mathcal{D}, K)$ is an instance of the SS-MCDS in $\mathcal{G}'$, where $K = |\mathcal{D}|$, it is also an instance of the MSPSA in $\mathcal{G}$.

1) Each vertex $v' \in \mathcal{X}$ is dominated by its respective vertex $v \in \mathcal{T}$.
2) The shortest path property of the SS-MCDS guarantees that nodes in $\mathcal{T}$ have a shortest path with respect to $\mu \in \mathcal{D}$. Therefore, it follows that $(\mathcal{D}, K)$ is an instance of the MSPSA problem.

Conversely, if $(\mathcal{D}, K)$ is an instance of the MSPSA problem in G, we show that $(\mathcal{D}, K)$ is an instance of the SS-MCDS problem in $\mathcal{G}'$. First, note that since $\mathcal{E} \subset \mathcal{E}'$, set $\mathcal{D}$ also induces a tree rooted at μ in $\mathcal{G}'$ that includes the shortest between μ and vertices in $\mathcal{T}$. The rest of the proof is as follows,

1) As $\mathcal{D}$ is rooted at μ, it follows that the shortest path property (with respect to μ) is satisfied for all vertices in $\mathcal{D}$.
2) By the Steiner arborescence definition $\mathcal{T} \subseteq \mathcal{D}$. Thus, vertices in $\mathcal{X}$ are dominated by their respective vertices in $\mathcal{T}$.
3) Note that set $\mathcal{T}$ includes the node with the highest hop count ($h_{\max}$) to μ. It follows that $\mathcal{D}$ contains vertices with hop counts of $1, 2, \ldots, h_{\max}$, due to the shortest path property. Since edges in $\mathcal{E}_2$ connect all vertices with the same distance to μ, all vertices in $\mathcal{V}'\backslash\mathcal{X}$ are dominated as well.
4) Therefore, $\mathcal{D}$ of size K is a CDS that contains the shortest paths of all vertices in $\mathcal{D}$ with respect to μ. This makes $(\mathcal{D}, K)$ an instance of the SS-MCDS problem, which concludes both proof directions.

□

D. Proof of Proposition 4

Proposition 4: The message complexity for partitioning the WSN to MCDSs using Algorithm 5 is $O(\delta^3_{\max}|\mathcal{V}|)$. Partitioning the WSN to SS-MCDSs (Algorithm 6) yields the same complexity.

Proof: Algorithm 5 is executed in three stages: the DS generation stage, the MCDS Approximation stage, and the MCDS update stage. The overhead of each stage is analyzed as follows.

Stage 1 – DS generation: In Stage 1, a node $v \in \mathcal{V}$ broadcasts packets in three occasions: (a) in Step 1 when all nodes share their initial values of $m(v)$, δ^*, and $r(v)$, (b) in Steps 3 or 5, when a node changes colors from *white* to *black* or *gray*, and (c) in Step 4 when a *white* neighbor of v has become *gray*. Cases (a) and (b) occur only one time per node, yielding the transmission of 2 packets. Case (c) can occur up to $\delta(v)$ times, as v may become *black* after all its neighbors have become *gray*. Thus, during Stage 1, a node v may transmit up to $\delta(v) + 2$ packets. For a maximum node degree δ_{max}, this stage has a message complexity of $\mathcal{O}(\delta_{max}|\mathcal{V}|)$.

Stage 2 – MCDS approximation: In Stage 2, a node $v \in \mathcal{V}$ transmits in the following occasions: (a) in Step 1 when nodes broadcast the value of b, (b) in Step 2, when a *gray* node becomes *black*, (c) in Step 3 when a *black* node is dominated by a newly-converted *black* node, and (d) in Step 4 when a gray node overhearing the change of color from *gray* to *black*. Cases (a), (b), and (c) can occur only once per node. However, (d) can occur multiple times as all *black* nodes in the neighborhood of u can become dominated by other nodes, making v transmit $b(v)$ packets. Thus, the total number of packets that a node v can transmit during Stage 2 is $b(v) + 3$. Moreover, all transmissions must be relayed to the two-hop neighborhood of v, Therefore, the total number of transmissions generated in Stage 2 is,

$$(\delta(v)(b(v)+3))|\mathcal{V}| \leq (\delta(v)^2+3\delta(v))|\mathcal{V}| \leq (\delta^2_{max}+3\delta_{max})|\mathcal{V}|.$$

This stage has a message complexity of $\mathcal{O}(\delta^2_{max}|\mathcal{V}|)$.

Stage 3 – MCDS update: In Stage 3, stages 2 and 3 are repeated until all nodes belong to at least one MCDS.

According to Proposition 2 of the manuscript, it may take up to $\delta_{\max}$ iterations for Stage 3 to terminate. Therefore, the combined overhead of stages 1 and 2 is multiplied by $\delta_{\max}$.

Combining the overhead of all three stages, we obtain the message complexity for Algorithm 5 as $\mathcal{O}(\delta_{max}^3|\mathcal{V}|)$.

Algorithm 6 has the same complexity as Algorithm 5 because the same stages are executed. The two algorithms differ only in the CDS node selection criteria of Stage 2. In case of Algorithm 5, the gray node that interconnects the highest number of black nodes is changed to black. In case of Algorithm 6, the selection is restricted to nodes that induce shortest paths. Because nodes can change color only once, (except in the pruning stage), the two steps in the respective algorithms have the same message overhead. Therefore, the two algorithms have the same message complexity.

□

E. Proof of Proposition 5

Proposition 5: In DFAS, a packet is guaranteed to be forwarded $2h$ hops per $(4h-4)$ sub-epochs, irrespective of the flow direction and the origin sensor.

Proof: To prove Proposition 5, we apply DFAS on a path of length $4h-3$ nodes (Fig. 2). The path length is selected to accommodate four subpaths B_1, B_2, B_3 and B_4, so that the proposition can be proved, irrespective of the sensor that originates a packet transmission and the traffic direction. Without loss of generality, we select v_1 to be the pseudo-sink and label each node in the path according to Step 3 of Algorithm 8. We then allocate the transmission times for each node according to Step 4 of Algorithm 8. The sub-epoch assignments for each node are shown in Fig. 2. The arrows between sub-epochs show the possible flow of traffic in two consecutive sub-epochs, in either direction.

First, consider nodes with ids 1 or h. From Fig. 2, it is straightforward to verify that a packet m originating from a node with id 1 or h can be forwarded in the upstream or the downstream direction over

$$H_1 = 4h-4 \text{ hops}$$

in $4h-4$ sub-epochs. This is because the neighbors of nodes with id 1 or h are assigned to transmit in successive sub-epochs in either direction. Consider now a node from B_1 or B_3 with id other than 1 or h, say j, which is assigned to transmit during sub-epochs I_j, I_{2h-j}, I_{2h+j-2}, and I_{4h-j-2}. For the upstream direction, the worst-case delay for node with id j is incurred when it initiates the packet forwarding in sub-epochs I_{2h-j} or I_{4h-j-2}. This is because the upstream neighboring node of j is scheduled to transmit $2j-2$ sub-epochs after j (as opposed to being scheduled in the successive sub-epoch when j initiates the upstream forwarding in I_j or I_{2h+j-2}). Once the packet reaches the upstream neighboring node of j, it can be relayed $4h-4-(2j-2) = 4h-2j-2$ hops in the remaining sub-epochs of the $4h-4$ period. Thus, the total number of hop relays for node with id j is equal to

$$H_2 = 4h-2j-2 \text{ hops},$$

which obtains the minimum value of $H_2 = 2h$ hops for $id = h-1$. In a similar way, for the downstream direction, the worst-case delay for node with id j occurs when it starts forwarding m in sub-epochs I_j or I_{2h+j-2}. The downstream node of j is now scheduled to transmit $2h-2j$ sub-epochs after j. After the packet reaches the downstream neighboring node of j, it can be relayed $4h-4-(2h-2j) = 2h+2j-4$ hops in the rest of the $4h-4$ period. The total number of hops for node with id j is

$$H_3 = 2h+2j-4 \text{ hops},$$

which obtains its minimum value of $H_3 = 2h$ hops for $id = 2$. Note that, if the node was chosen from B_2 or B_4, the same analysis applies with the difference that H_2 would correspond to the downstream case, and H_3 to the upstream one.

Finally, the minimum number of hops irrespective of the flow direction, in a period of $(4h-4)$ sub-epochs is equal to

$$H_{\min} = \min\{H_1, H_2, H_3\} = 2h \text{ hops},$$

for $h \geq 2$. □

F. Proof of Proposition 6

Proposition 6: Let a sensor v belong to $f(v) \geq 1$ CDSs. Suppose that an event Ψ is detected by v at time $t(\Psi)$, where $t(\Psi)$ is uniformly distributed over z epochs. The delay until a CDS containing v becomes active is:

1) $d_{\min} = 0$ epochs.
2) $d_{\max} = z - f(v)$ epochs.
3) $d_{ave} = \sum_{k=1}^{z-f(v)} k \times \frac{C(z-k-1, f(v)-1)}{C(z, f(v))}$ epochs.

Proof: Without loss of generality, assume that event Ψ observed by v occurs during the first epoch. If $v \in \mathcal{D}_1$, the event is reported during this epoch and the buffering delay equals zero[1]. This defines the minimum possible rotation delay $d_{\min}$. If $v \in \mathcal{D}_2$ and $v \notin \mathcal{D}_1$, then v must wait for one epoch before it can transmit the event report to the sink. This occurs with probability,

$$\Pr[d=1] = \frac{C(z-2, f(v)-1)}{C(z, f(v))}, \quad (1)$$

where $C(n,k)$ denotes the binomial coefficient. The nominator in (1) denotes the number of ways that $f(v)$ elements (appearance frequency of v in the z CDSs) can be chosen from z elements (number of CDSs), such that one specific element (in our case, $\mathcal{D}_2$) is always selected, while another (i.e., $\mathcal{D}_1$) is always not selected. Alternatively, this can be seen as eliminating $\mathcal{D}_1$ from

1. For simplicity, we have ignored the case in which sensor v has already transmitted all its dummy packets during the first epoch, before Ψ has occurred.

$\mathcal{D}_1, \ldots, \mathcal{D}_z$, fixing $\mathcal{D}_2$ and selecting $f(v)-1$ subsets from the remaining $z-2$ subsets. The denominator denotes the number of ways that $f(v)$ elements can be chosen from z elements. In the computation of the probability in (1), we have implicitly assumed that all arrangements of $f(v)$ elements out of z are equiprobable.

Generalizing to the case in which v has to wait for k epochs,

$$\Pr[d=k] = \frac{C(z-k-1, f(v)-1)}{C(z, f(v))}. \tag{2}$$

In the worst-case scenario, sensor v has to wait for $z-f(v)$ epochs before one of the subsets that contain it becomes active. This scenario occurs if v is part of subsets $\mathcal{D}_{z-f(v)+1}, \ldots, \mathcal{D}_z$. The value of $z-f(v)$ yields the worst case rotation delay for v. To compute the average delay, we sum over all possible delay values, multiple by the respective probability for each delay.

$$d_{ave} = \sum_{k=1}^{z-f} k \times \frac{C(z-k-1, f(v)-1)}{C(z, f(v))}.$$

□

G. Proof of Proposition 7

Proposition 7: The number of hops traversed by a real packet m originating from v until it reaches the sink μ is upper-bounded by $|p(v,\mu)| + 2rot$, where $|p(v,\mu)|$ is the shortest path length between v and μ and rot is the number of CDS rotations until m reaches μ.

Proof: Without loss of generality, let $\mathcal{D}_1$ be active during epoch W_k. Node $v \in \mathcal{D}_1$ initiates the relay of a real packet m to the sink μ. Assume that $w \in \mathcal{D}_1$ is the last node to forward m, before rotating to a CDS $\mathcal{D}_2$ in W_{k+1}. The transmission of w is overheard by a node $u \in \mathcal{D}_2 \cap \mathcal{N}_w$. First, we show via contradiction that paths $p(v,w)$ and $p(w,\mu)$ are also shortest paths between the respective sources and destination. We break the shortest path $p(v,\mu)$ into subpaths $p(v,w)$ and $p(w,\mu)$ and assume that $p(v,w)$ is not a shortest path between v and w. That is, there is a path $p^*(v,w)$ such that $|p^*(v,w)| < |p(v,w)|$. Hence, there exist a path $p^*(v,\mu) = p^*(v,w) \cup p(w,\mu)$ with $|p^*(v,\mu)| < |p(v,\mu)|$, which contradicts the assumption that $p(v,\mu)$ is the shortest path.

With the termination of the epoch W_k, packet m has traversed $|p(v,w)|+1$ hops to reach $u \in \mathcal{D}_2$. ($|p(v,w)|$ to reach w and one hop to reach u). The one-hop neighborhood of w can be divided to three possible subsets with respect to their hop distance to the sink. One subset at a distance of $|p(w,\mu)|$, one subset at a distance $|p(w,\mu)|+1$, and one subset at a distance $|p(w,\mu)|-1$, where $|p(w,\mu)|$ is the length of the shortest path between w and μ. Again, this claim can be shown via contradiction. If there was a neighbor $k \in N_w$ that could reach μ at fewer hops than $|p(w,\mu)|-1$, then w has a shorter path to μ via k, which contradicts the assumption that $p(w,\mu)$ is the shortest path. If there was a neighbor $k \in N_w$ with a shortest path to μ longer than $|p(w,\mu)|+1$ hops, then k could reach μ in fewer hops via w, which contradicts the fact that every CDS contains shortest paths to μ.

We therefore conclude that in Step 2 of Algorithm 9, when node $u \in \mathcal{D}_2 \cap \mathcal{N}_w$ continues to forward m to μ via the shortest path, $p(u,\mu)$ has one of the following sizes:

a. $|p(u,\mu)| = |p(w,\mu)| - 1$
b. $|p(u,\mu)| = |p(w,\mu)|$
c. $|p(u,\mu)| = |p(w,\mu)| + 1$.

The worst case occurs when $|p(u,\mu)| = |p(w,\mu)| + 1$. If m is delivered during $\mathcal{D}_2$, packet m traverses in the worst case

$$H = |p(v,w)| + 1 + |p(u,\mu)| + 1 = |p(v,\mu)| + 2$$

Therefore, the number of hops traversed by m due to the rotation of CDSs, increases up to 2 hops. The same process is repeated each time that a CDS rotation occurs. So in the worst case, the total number of hops traversed by m is upper-bounded by $|p(v,\mu)| + 2rot$, where rot is the total number of rotations required to delivered the packet. □

References

[1] J. Cong, A. Kahng, and K. Leung. Efficient algorithms for the minimum shortest path steiner arborescence problem with applications to vlsi physical design. *Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 17(1):24–39, 1998.

[2] T. Cormen, C. Leiserson, R. Rivest, C. Stein, et al. *Introduction to algorithms*, volume 2. MIT Press Cambridge, 2001.