Appendix A

The following presents the original description and proofs of DPSS, building off of the terms and definitions introduced in Section 2. It has been slightly modified from Ref. [11] to explicitly label certain theorems and lemmas. There have also been some minor changes for brevity and clarity, but the original errors have been preserved. We include this because it is our experience that the style of the algorithm descriptions and proof strategies are fairly representative of those found in multi-agent system literature, and so we believe it has educational value to formal methods practitioners looking to work with multi-agent system designers.

if UAV i rendezvous with neighbor j **then**
  Calculate team size $N = N_{R_i} + N_{L_i} + 1$.
  Calculate perimeter length $P = P_{R_i} + P_{L_i}$.
  Calculate UAV i's relative index $n = N_{L_i} + 1$.
  Calculate UAV i's segment endpoints:
    $\mathcal{S}_i = \left\{ \lfloor n - \frac{1}{2}(-1)^n \rfloor P/N, \lfloor n + \frac{1}{2}(-1)^n \rfloor P/N \right\}$.
  Communicate $\mathcal{S}_i$ to neighbor j and receive $\mathcal{S}_j$.
  Calculate shared border position $p_{i,j} = \mathcal{S}_i \cap \mathcal{S}_j$.
  Travel with neighbor j to shared border position $p_{i,j}$.
  Set direction to monitor own segment.
else if reached perimeter endpoint **then**
  Reverse direction.
else
  Continue in current direction.
end if

Alg. A: Neighbor escort from the perspective of UAV i.

Algorithm A Consider the protocol described in Algorithm A, where each UAV escorts any neighboring UAV it encounters to the shared boundary of their perimeter segments. Note that UAVs do not automatically turn around if they reach a segment boundary; they only turn around if they encounter a *perimeter* endpoint or start or stop a neighbor escort. Meeting and escorting neighbors to segment boundaries is key for both obtaining correct coordination variables and achieving the synchronization required for the optimum configuration. Also note that each UAV $i \in 1, \ldots, N$ continuously updates P_{R_i} and P_{L_i} based on the distance it travels, and UAVs are able to tell when they reach a perimeter endpoint.

Theorem 1. *Let the perimeter length P and the N UAVs in the system be fixed. If every UAV's coordination variables are correct, then Algorithm A achieves the optimal configuration within $2T$.*

Proof. UAVs can initially be positioned anywhere along the perimeter and can be traveling either left or right at constant speed V. Each UAV has correct coordination variables and so can accurately calculate its perimeter segment. Also, each UAV is guaranteed to meet its neighbors since each UAV only reverses direction at a *perimeter* (not segment) endpoint or when starting or stopping a neighbor escort.

For N UAVs monitoring a perimeter of length P, order segments of size P/N from the left edge of the perimeter as $1, \ldots, N$, so that UAV i is responsible for segment i. Label the left endpoint of segment 1 as p_1, the right endpoint of segment N as p_N, and the shared endpoint of segments i and j as $p_{i,j}$. Let T_s be the time for a UAV to travel once across a segment, which is the same for every UAV.

Consider first the actions of UAV 1. Once UAV 1 has escorted UAV 2 to $p_{1,2}$, then no UAV to the right of UAV 1 will ever enter segment 1 by crossing $p_{1,2}$ again. This is because UAV 1 will make the round trip from $p_{1,2}$ to p_1 back to $p_{1,2}$ in $2T_s$, whereas UAV 2 would require at least $2T_s$ to return to $p_{1,2}$ after reaching or passing $p_{2,3}$. This is because there are two cases for UAV 2, as depicted in Figure 4. UAV 2 will either meet UAV 3 on segment 2, escort it to $p_{2,3}$, and turn around and reach $p_{1,2}$ in exactly $2T_s$, or UAV 2 will meet UAV 3 to the right of $p_{2,3}$, turn around to escort it to $p_{2,3}$, then continue on toward $p_{1,2}$, which would require more time than $2T_s$. UAV 2 will therefore either meet UAV 1 at $p_{1,2}$, or it will meet UAV 1 to the right of $p_{1,2}$, where UAV 1 will escort it to $p_{1,2}$ and then UAV 2 will reverse direction back towad $p_{2,3}$. In either case, $p_{1,2}$ can now be regarded as a fixed perimeter endpoint for UAV 2, and the same argument can be repeated (once UAV 1 has completed escorting UAV 2) if we consider UAV 2 as the leftmost UAV in a set of $N-1$ UAVs. Therefore, there is a time τ after which all UAVs are constrained to their respective segments, and since UAVs only change direction when meeting their neighbors at their shared segment boundaries or a perimeter endpoint, the optimal configuration in Definition 1 is reached.

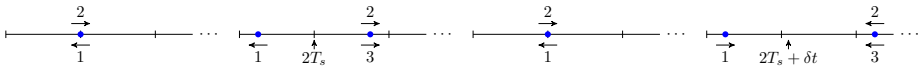

Fig. 4: Possible cases for the rendezvous between UAV 1 and UAV 2. They either meet in $2T_s$ at their shared segment boundary or to the right of it at $2T_s + \delta t$.

The worst case occurs when all UAVs are stacked at one end of the perimeter and are traveling toward the other end. Once T has passed, all UAVs are at the opposite end of the perimeter where they meet both of their neighbors. Each pair will travel to their shared borders, which for the farthest pair will require a travel time less than T. Therefore, the steady-state behavior will be achieved before time $2T$. □

1: **if** agent i (left) rendezvous with neighbor j (right) **then**
2: Update perimeter length and team size:
3: $P_{R_i} = P_{R_j}$
4: $N_{R_i} = N_{R_j} + 1$
5: Calculate team size $N = N_{R_i} + N_{L_i} + 1$.
6: Calculate perimeter length $P = P_{R_i} + P_{L_i}$.
7: Calculate relative index $n = N_{L_i} + 1$.
8: Calculate segment endpoints:
9: $\mathcal{S}_i = \left\{ \lfloor n - \frac{1}{2}(-1)^n \rfloor P/N, \lfloor n + \frac{1}{2}(-1)^n \rfloor P/N \right\}$.*
10: Communicate $\mathcal{S}_i$ to neighbor j and receive $\mathcal{S}_j$.
11: Calculate shared border position $p_{i,j} = \mathcal{S}_i \cap \mathcal{S}_j$.
12: Travel with neighbor j to shared border $p_{i,j}$.
13: Set direction to monitor own segment.
14: **else if** reached left perimeter endpoint **then**
15: Reset perimeter length to the left $P_{L_i} = 0$.
16: Reset team size to the left $N_{L_i} = 0$.
17: Reverse direction.
18: **else if** reached right perimeter endpoint **then**
19: Reset perimeter length to the right $P_{R_i} = 0$.
20: Reset team size to the right $N_{R_i} = 0$.
21: Reverse direction.
22: **else**
23: Continue in current direction keeping track of traversed perimeter length.
24: **end if**

Alg. B: Neighbor escort with coordination variable update from the perspective of UAV i. UAV j follows the same protocol but updates P_{L_j} and N_{L_j} in lines 3-4 instead. *This equation is incorrect. It requires an extra term to account for the fact the the left perimeter endpoint may not be at 0 or its estimate may not be at 0. For instance, it could be rewritten as $\mathcal{S}_i = \left\{ \lfloor n - \frac{1}{2}(-1)^n \rfloor P/N - \text{left\_endp\_est}, \lfloor n + \frac{1}{2}(-1)^n \rfloor P/N - \text{left\_endp\_est} \right\}$

Algorithm B Now consider Algorithm B, which is essentially Algorithm A with the additional steps of communicating and updating the coordination variables. We show that this protocol achieves the optimal configuration within $5T$ for arbitrary initial conditions of position, direction, and coordination variables for each UAV in the system. We first show that every UAV obtains correct coordination variables within $3T$, then by Theorem 1, the system takes an additional $2T$ to converge to the optimal configuration.

Lemma 1. *Using Algorithm B, every UAV will obtain correct coordination variables within* $3T$.

Proof. We first prove that all UAVs converge to correct coordination variables in finite time. Note that since a UAV only changes direction at perimeter endpoints or when starting or stopping a neighbor escort, all UAVs are guaranteed to meet their neighbors.

Label UAVs, segments, and endpoints as in Algorithm A and consider the actions of UAV 1. UAV 1 is guaranteed to visit p_1 either after an escort from UAV 2 or before having met any other UAVs, depending on initial conditions. Once UAV 1 has visited p_1, both N_{L_1} and P_{L_1} are correct. At the next meeting of UAV 1 and 2, UAV 2 updates its "left" coordination variables N_{L_2} and P_{L_2} through communication with UAV 1, thereby obtaining correct values for them. Note that repeated meetings between UAV 1 and 2 will not change the value of these coordination variables since N and P are fixed. Now consider UAV 2 as the leftmost UAV in a team of $N-1$ UAVs, and note that UAV 3 is similarly ensured to obtain correct left coordination variables. Since only one neighbor meeting is required after the leftmost UAV has obtained correct values for its left coordination variables, the number of UAVs needing correct "left" coordination variables is reduced at each stage, and since meetings are guaranteed to occur in finite time, every UAV in the system obtains correct values for its left coordination variables in finite time. Clearly, the same argument holds for the right end of the perimeter and right coordination variables.

During the transient period when the UAVs are learning the correct coordination variables, the calculation of shared segment boundaries is incorrect relative to the optimal configuration, but it is consistent between UAVs involved in the rendezvous and escort. This can be seen by noting that after both UAVs have communicated and updated their coordination variables with each other, they each have the same understanding of P and N and so calculate the same shared segment boundary position. So while they escort each other to the (ultimately) wrong position, they are still guaranteed to continue in the correct direction afterward to ensure that each UAV meets both its neighbors.

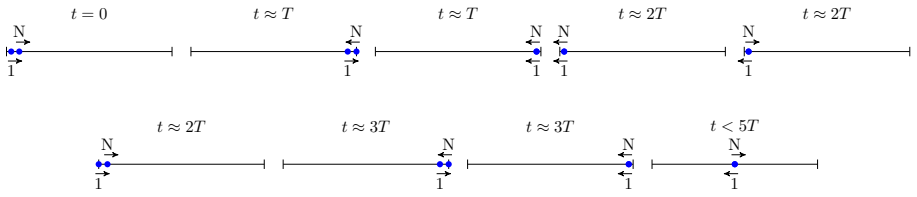

Fig. 5: Worst case convergence for Algorithm B demonstrated with 2 UAVs.

The worst case for obtaining correct coordination variables occurs when the overlap of the UAVs is the greatest, that is, when UAVs travel together rather than space out along the perimeter. Consider N UAVs stacked near $x = 0$ and headed right (Figure 5a). After time T has passed, UAV N will reach the right perimeter endpoint and update P_{R_N} and L_{R_N} to correct values (Figure 5b). Once UAV N has visited p_N, it will rendezvous with the rest of the UAVs in the system (Figure 5c). At this instant, the UAVs have arbitrary values for their "left" coordination variables. Suppose the estimated shared border position for

UAVs N and $N-1$ is at $x = \epsilon$ for $\epsilon << P$ and is less than that for all other UAVs. In this case, the entire team will again nearly travel the length of the perimeter to $x = \epsilon$ which requires less than another T units of time (Figure 5d). UAV N then separates from the team and begins heading toward the right end of the perimeter without learning about the left perimeter endpoint (Figure 5e). UAVs $1 \ldots N-1$ encounter the left endpoint of the perimeter and update to a completely correct set of coordination variables (Figure 5f). With correct coordination variables, UAVs begin to space out equally along the perimeter. Note that UAV $N-1$ will chase UAV N the entire length of the perimeter since it will escort UAV $N-2$ to their shared border position and then continue to the right. UAV $N-1$ then meets UAV N near the right end of the perimeter in T units of time since reaching the left endpoint (Figure 5g). Once they meet, UAV N will update to correct coordination variables (Figure 5h). At this point, $3T$ has passed. □

Theorem 2. *Let the perimeter length P and the N UAVs in the system be fixed. Then an upper bound for convergence of Algorithm B to the optimal configuration is $5T$.*

Proof. By Lemma 1, an upper bound on the amount of time for all UAVs to obtain correct coordination variables is $3T$. Then by Theorem 1, up to an additional $2T$ is needed for the UAVs to reach the optimal configuration. Therefore, an upper bound for the convergence of Algorithm B is $5T$. □