J Intell Robot Syst
DOI 10.1007/s10846-009-9333-y

Distributed Coordination Strategies for Wide-Area Patrol

Brandon J. Moore · Kevin M. Passino

Received: 14 April 2008 / Accepted: 20 April 2009

Abstract This paper addresses the problem of enabling a group of autonomous vehicles to effectively patrol an environment significantly larger than their communication and sensing radii. The environment is divided into smaller areas and special coordinator vehicles are designated to control the transfer of the other vehicles from one area to another. Our past work showed that by organizing the areas and coordinators into a ring topology, we could design a control algorithm that globally balanced the number of vehicles in all areas within a bounded length of time. This paper extends those results to a much broader class of area-coordinator topologies and this added flexibility can be used in implementation to reduce the time it takes to attain the globally balanced state.

Keywords Cooperative control · Autonomous systems · Optimization · Resource allocation

1 Introduction

This work is motivated by a mission scenario first proposed in [11] in which a group of autonomous vehicles with range-limited communications and sensing is tasked to cooperatively patrol a relatively large environment (i.e., an environment whose dimensions dramatically exceed the vehicles' maximum communication and sensing radii). The limited communication and sensing radii mean that the vehicles must

This work was supported by the AFRL/VA and AFOSR Collaborative Center of Control Science (Grant F33615-01-2-3154).

B. J. Moore (✉) · K. M. Passino
Department Electrical and Computer Engineering, The Ohio State University, Columbus, OH 43210, USA
e-mail: brandon.moore@gipsa-lab.inpg.fr, brandon.joseph.moore@gmail.com

K. M. Passino
e-mail: passino@ece.osu.edu

Published online: 06 May 2009

reach a compromise between distributing themselves throughout the environment (to reduce gaps in patrol coverage) and maintaining some cohesion as a group (because they will need to communicate at least periodically in order to cooperate with each other).

The approach to this problem we took in [11] is as follows. The environment was first broken down into a number of smaller *areas* to which individual vehicles could be assigned (see Fig. 1). The idea behind this spatial decomposition was that by restricting a vehicle to a small section of the environment, it should be easier to determine how that vehicle should behave in order to efficiently patrol that area. More specifically, if there are one or more locations in an area to which a vehicle can go to in order to communicate with another vehicle, then the small size of that area means that a vehicle can patrol a significant portion of that area in between visits to those locations and still communicate on a fairly regular schedule. That is to say that a vehicle can spend most of its time actually patrolling the area and still remain connected to the group through periodic communication. Of course, in order for a vehicle to communicate by going to one of these special locations, there must be another vehicle there. This brings us to the second part of our approach which was the establishment of a number of *coordinators*. Coordinators are simply a small number of the group's vehicles that remain along the border between two or more areas and provide the previously mentioned locations where the other vehicles can come to communicate. Exactly how the coordinators and the patrol vehicles can meet up to communicate is highly dependent on the nature of the vehicles and the specifics of the scenario and we direct the reader to [11] for more discussion on this topic.

The preceding formulation of the problem into areas and coordinators breaks the group of vehicles into a two-level hierarchy. At the bottom level we have the majority of the vehicles, and from this point on we will simply use "vehicle" and

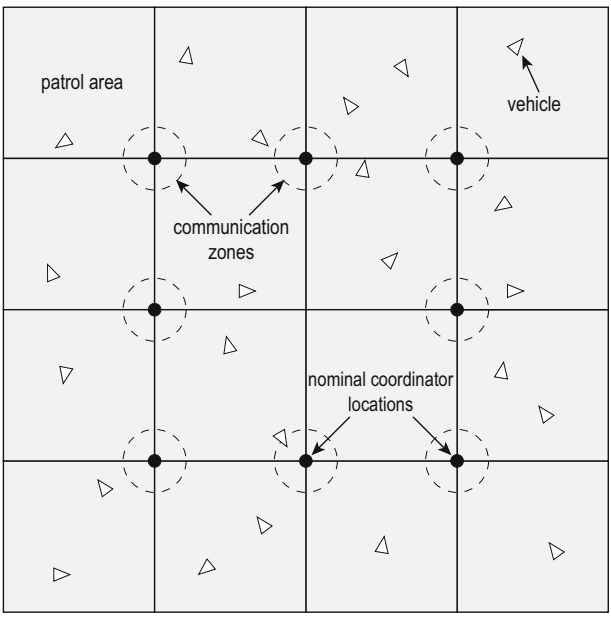

Fig. 1 A 16-area example patrol environment

"vehicles" to refer to the members of this group. Being confined to one area at a time, the vehicles' main job is simply to gather information from that limited region and report it back to the coordinators. Forming the upper level, the coordinators are responsible for collecting the information from the vehicles and passing it on (e.g., via the vehicles) to other coordinators and potentially a human operator, directing the vehicles' efforts within the area (i.e., telling them where to go look if a certain part of the area has something interesting in it or has not been observed for a while), and shifting vehicles between the areas they oversee (i.e., the ones they "connect" by virtue of their nominal location) in order to achieve the proper distribution of vehicles across all the areas. For now we will consider the proper distribution of vehicles to be one in which the number of vehicles in each area is *globally balanced*. By this we mean that the number of vehicles in any area differs from that of any other area by no more than one. The reasoning behind our desire for a globally balanced distribution is that unless we have specific information that leads us to believe it is better to concentrate more vehicles in certain regions, we want to spread them out as uniformly as possible across the environment to reduce the gaps in coverage. When the areas are all the same size (as would be the case with a regular gridding of the environment), then this corresponds to wanting the number of vehicles in each area to be as equal as possible (and unless the number of vehicles is evenly divisible by the number of areas, then the best we can do is to have any two areas differ by no more than one vehicle). Of course, in some cases it will be desirable to be able to specify a weighted balancing of the vehicles (i.e., when we want each area to have a proportion of group's vehicles that is equal to its relative *priority*).

Other work in cooperative control that shares some similarities with our scenario include those that deal with "pop-up" targets [8, 10] and those that deal with the spatial allocation of resources [5]. However, the most closely related work to ours is that concerning load balancing in distributed computing [1, 2, 7]. Our problem differs from the regular load balancing problem in that multiple coordinators will have control over any given area (as opposed to a single processor having control over its task load) and since we demand a globally balanced distribution (as opposed to a locally balanced distribution in which only neighboring processors are balanced to within one load block). The issue of achieving a globally balanced distribution has received only limited attention in the literature prior to [11]. Improvements on local balancing have been achieved both with message passing [4] (which guarantees balanced neighborhoods, but not global balancing) and non-deterministic algorithms [6] (which have an expected value for the global imbalance which does not generally go to zero as time goes to infinity). The algorithm presented in [9], which shares some commonalities with [11], does achieve a globally balanced distribution but is specific to applications involving a grid of parallel processors and does not address the information delays or asynchronism present in our scenario.

2 Global Balancing on a General Topology

2.1 Overview of Dynamics and Results

In order to better explain our work, we provide a high level overview of how we are able to achieve our results in this section before getting into the more technical

description in the sections that follow. As stated, this work is an extension of [11] which used a ring interconnection in which vehicles could only be transferred in one direction. While it does achieve global balancing, this unidirectional ring setup has the disadvantage of taking a considerable amount of time to converge to the globally balanced state in some circumstances. For example, if all the vehicles in the system initially start out in one area, then in order to put a vehicle in the area just preceding that area in the ring a vehicle has to be sent all the way around the ring (even though that area lies right next to that area physically). The purpose of this paper, then, is to extend the results of [11] to a less restrictive class of topologies. This new class of topologies includes all area-coordinator interconnections which have an underlying unidirectional ring, but in addition to being able to transfer resources along this ring, the coordinators will also be able to transfer vehicles across "shortcuts" to this ring. The effect of these shortcuts is to dramatically reduce the number of areas that a vehicle will have to traverse in order to get to where it is needed. More explanation of these topologies and how these shortcuts are established is given in Section 2.2 below.

The essential dynamics of the system presented in this paper are the same as those for the area ring set up of [11], although the presence of the shortcuts can result in much more complicated behavior and as a consequence, the control algorithm discussed in Section 2.4 is more complex than that of [11] in order to ensure that we can still achieve the globally balanced state. In order to help the reader better understand these dynamics we introduce the visualization aid shown in Fig. 2. In this figure, each area in the ring is drawn as a stack of identically sized blocks, with the number of blocks in this stack equal to the number of vehicles in that area. These area stacks are arranged according to the order specified by the area ring with the stack for an area drawn to the left of the stack for the next area in the ring and to the right of the stack for the preceding area. (The area ring eventually wraps back upon itself, but only a section of the ring is drawn in Fig. 2). The low spots in this picture correspond to the areas with the smallest number of vehicles and we call these areas *holes*. The relatively high spots (areas that have at least two more vehicles than the hole areas) we call *hills*. We may also have a *plateau* which is a group of areas, each with exactly one more vehicle than a hole, which is located between a hole on the right and a hill on the left (and we will sometimes refer to the area in a plateau that is most advanced along the ring as its *leading edge*). The directions *forward* and

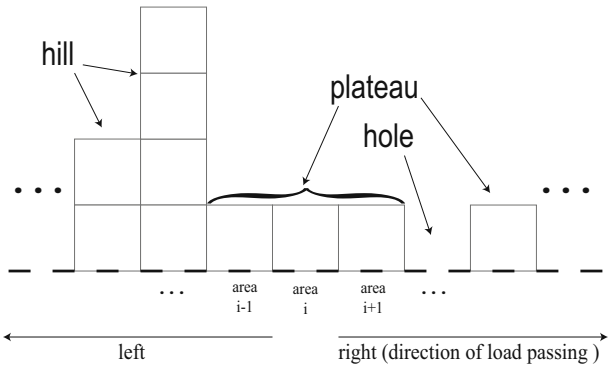

Fig. 2 Illustration of terms used in convergence proofs

backward refer respectively to the direction that vehicles may be passed around the ring and its opposite.

If we had a single centralized coordinator that always had accurate information about the number of vehicles in each area, then our goal of global balancing would not be a very hard one, and, in fact, with a little bit of calculation this coordinator could achieve it in an optimal manner (i.e., with the fewest number of vehicle transfers). What makes our problem interesting is that we must achieve global balancing via decentralized decision making based on local and inaccurate information. As a consequence it is unavoidable that the coordinators will make "mistakes" causing the number of vehicles is the areas to fluctuate. Our task then is to formulate a system structure in which the coordinators can make steady progress towards the globally balanced state in spite of these mistakes. Now, the way we organize the interconnection of the areas in Section 2.2 and the control algorithm of Section 2.4 enforce certain dynamics in the system. First, the minimum number of vehicles in any area cannot decrease at any time (i.e. it can only increase or stay the same). Second, the number of holes in the system can only increase at times when this minimum increases (i.e. no new holes can be created otherwise, although the location of the existing holes may change). Third, a main result of our analysis shows that whenever the areas are not globally balanced, then within a bounded amount of time at least one hill and hole will get closer together on the ring (i.e. the plateau between them will shrink by a least one area). In general the holes will move backwards on the area ring and/or the hills will move forward until they meet and the hill "fills in" the hole (although the holes may also be filled in by transfers of vehicles along the shortcuts to the ring). Thus holes are regularly eliminated from the system, and when the last hole at a given minimum number of vehicles per area is eliminated, then that minimum must increase. This process repeats until the minimum number of vehicles per area is raised as high as it can go and there are no more hills left to fill in the holes, and once there are no more hills then the areas must be globally balanced. Although this explanation of the system dynamics seems fairly simple, we remind the reader that we are only describing a subset of the system's dynamics and that many other vehicles transfers are being made in addition to ones involved in the above process. In fact, ensuring that these other transfers do not interfere with the filling in of holes is not at all trivial and will require a carefully designed distributed control algorithm.

2.2 Interconnection Design Requirements

In this section we describe the class of area-coordinator interconnections that will still allow us to achieve global balancing with an (extensive) modification of the setup seen in [11]. Although much more general than the unidirectional ring used before, there will still be a number of restrictions on the type of interconnections that may belong to this class. We will see in Section 4, however, that these restrictions do not pose much of a problem for typical scenarios.

To describe the area-coordinator interconnection, we start by letting $\mathcal{A} = \{1, \ldots, N_A\}$ denote the set of N_A areas and letting $\mathcal{C} = \{1, \ldots, N_C\}$ denote the set of N_C coordinators. To lessen confusion we will endeavor to use variants of a to denote specific areas and variants of c to denote specific coordinators. The area-coordinator interconnection is best described by a bipartite graph $(\mathcal{A} \cup \mathcal{C}, \mathcal{I})$ with vertices from $\mathcal{A}$ and $\mathcal{C}$ and a set of edges $\mathcal{I} \subseteq \mathcal{A} \times \mathcal{C}$ which contains the edge (a, c) if and only

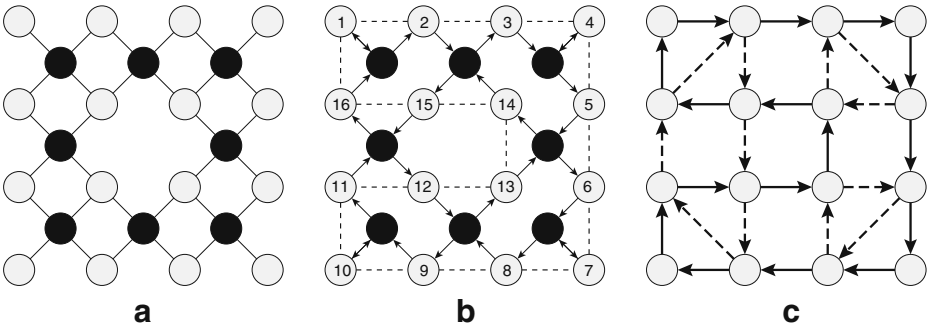

Fig. 3 The equivalent bipartite graph for the example interconnection of Fig. 1 shown in **a** with area vertices colored gray and coordinator vertices colored black. The final interconnection is shown in **b** with each area having exactly one coordinator that may remove vehicles (shown with an arrow from area to coordinator) and exactly one coordinator that may remove vehicles (shown with an arrow from coordinator to area). The number of the area vertices and the dashed line shows the underlying ring cycle established for this interconnection. Finally, in the graph in **c** shows the resulting unidirectional area network, with the ring connections and the shortcuts shown as solid and dashed arrows respectively. In this example, the diameter of the final network is only 5 (as compared to 15 for the ring alone)

if coordinator c connects to area a. For example, Fig. 3a shows the bipartite graph constructed from the physical layout in Fig. 1.

With notation for the interconnection in hand we can now state the main requirement that the design of our interconnection must satisfy, namely that it must be possible for a vehicle starting in any area to travel through the interconnection (via transfer from one area to another by a connecting coordinator) in such a manner that it visits each area once and only once before returning to its original location. To define this technically, let us first define the addition and subtraction operations on the area indices such that their output "wraps" into the set $\mathcal{A}$ so, for example, $a+1=1$ when $a=N_A$ and $a-1=N_A$ when $a=1$. We can technically define this operation as $a+i=\text{mod}(a+i-1, N_A)+1$, where $\text{mod}(p,q)=p-q\lfloor\frac{p}{q}\rfloor$ and $\lfloor\cdot\rfloor$ is the floor function (i.e., the largest integer no greater than its argument). Now our restriction on the interconnection $(\mathcal{A}\cup\mathcal{C},\mathcal{I})$ can be stated as follows: there must exist some numbering of the areas such that for every $a\in\mathcal{A}$ there exists a "ring" coordinator $C(a)$ that connects area a to area $a+1$.

The reader familiar with [11] will note that this condition we have placed on our area-coordinator interconnection is simply the ability to extract a ring (which may reuse coordinators but not areas), and if we were to limit coordinator $C(a)$ to transferring vehicles from area a to area $a+1$ then we could simply apply the algorithm in that paper in order to achieve global balancing of the vehicles. However, our intention here is to utilize the shortcuts to this ring.

For simplicity, assume from now on that the area indices have been assigned to satisfy the above requirement and that a specific coordinator $C(a)$ has been defined for each area a. (In actual implementation, the indexing of the areas is not so important. What is important is that for each area there is a unique "next" area and also that a special coordinator connects the two. Although somewhat confusing now, this concept should become clear when the control algorithm is discussed later.) In [11] each coordinator was restricted to transferring vehicles from one specific area

to another specific area, but in order to make the best use of the shortcuts, in our interconnection we will have to be more flexible than this. We cannot, however, simply let every coordinator for an area remove any vehicle it wants because this may result in unstable system behavior that prevents the system from balancing (i.e., the delays caused by our method of communication can result in undamped oscillations in the distribution of vehicles).

The approach we use for distributed coordination is to restrict the manner in which vehicles may be transferred from one area to another. For each area a we will only allow vehicles to be removed by its ring coordinator $C(a)$. This means that each coordinator c will have a set of *source* areas $S(c) \subset A$ from which it may remove vehicles (i.e., $S(c) = \{a \in A : C(a) = c\}$). For each coordinator c let us now define a set of *destination* areas $D(c)$ to which it may add vehicles. Specifically, if $a \in S(c)$ then $a + 1 \in D(c)$ (or, equivalently, $D(c) = \{a \in A : C(a - 1) = c\}$) which means that if coordinator c is allowed to remove vehicles from area a then it is allowed to add vehicles to $a + 1$. For example, Fig. 3b shows a possible area-coordinator interconnection derived from the bipartite graph is Fig. 3a. In this figure, the coordinator that may remove resources from an area is denoted by an arrow from that area to that coordinator, and the coordinator that may add resources to an area is denoted by an arrow to that area from that coordinator (i.e. the arrows represent how vehicles may circulate through the interconnection. As shown by the indexing of the areas and the dashed line, Fig. 3b and c demonstrate how we can preserve an underlying ring while also creating many shortcuts to that ring.

The reason we have chosen to limit vehicle transfers in the way just described is that very useful system properties are preserved when each area has exactly one coordinator that may remove vehicles and exactly one that may add vehicles. In [11] these properties allowed the system to achieve global balancing on a ring structure, and in this paper they will allow the coordinators to transfer vehicles along shortcuts to this ring in order to get faster balancing.

2.3 Estimation Methods and Properties

The fundamental feature of our distributed coordination problem that makes it interesting is that there is significant delay between the action of one coordinator (i.e., the transfer of a vehicle from one area to another) and another coordinator's recognition of that action. Specifically, a coordinator will not realize that a vehicle has been added to an area until that vehicle checks in with it and it will not realize that a vehicle has been removed from an area until the vehicle fails to check in for such a long time that the coordinator can assume it has been removed. Because of this, it is generally impossible for a coordinator to have accurate information about the number of vehicles in any particular area. The purpose of this section is to demonstrate how the coordinators can estimate these numbers and explore some important properties of these estimates that will allow the coordinators to make decisions about local vehicle transfers that will ultimately result in a globally balanced state.

Assume that we are given some length of time d which denotes the longest period of time a vehicle can go without checking in with any specific coordinator for the area to which it is assigned (and assume this value is known to all the coordinators). The value of d is determined by the physical layout of the area-coordinator

interconnection and the dynamics of the vehicles (e.g., their maximum speed) and must be chosen large enough so that a vehicle can check in with all the other coordinators for its assigned area and still have enough time left over to do some useful patrol work. Assume also that each vehicle is given an identification number that is unique. In order to avoid confusion, we will actually require that this identification number be reassigned every time a vehicle is transferred from one area to another and that the new identification number be unique from any identification number used in the past. This is easily implemented by having the coordinator that transferred the vehicle into a new area give that vehicle a new number consisting of that coordinator's index value and the date and time (according to the coordinator's local clock) that that vehicle was transferred. If simultaneous transfers of vehicles are allowed, then the coordinator can either add one more field to the identification number in order to distinguish between the vehicles or it can simply adjust the time field slightly to make each identification number unique (e.g., by adding a small number of seconds to each time field). Now, in order to generate an estimate for the number of vehicles in an area, a coordinator will simply keep a list of the vehicles it thinks are still assigned to that area. A vehicle gets added to this list the first time it checks in with that coordinator and removed whenever it has not visited that coordinator for a length of time greater than d. Obviously whenever a coordinator transfers a vehicle it can update its list for both the source and destination areas immediately. The coordinator's estimate of the number of vehicles in an area is then simply the length of this list, and in our system this estimate has a number of useful properties.

Now, each coordinator c will have to maintain an estimate for each area in the sets $S(c)$ and $D(c)$. One of our fundamental assumptions in this paper will be that no vehicles enter or leave the system for time $t \geq t_{init} - d$, where t_{init} is the initial time. This gives us two useful properties for coordinator c's estimates of the areas in $S(c)$ and $D(c)$. Specifically, for any area $a \in S(c)$ and any time $t \geq t_{int}$, it must be the case that coordinator c's list of vehicles it thinks are in area a differs from the true number only by those vehicles that were added by coordinator $C(a-1)$ in between time $t-d$ and t (because coordinator $c = C(a)$ is the only coordinator that can remove vehicles from area a). Thus we have that coordinator c's estimate for an area $a \in S(c)$ can never be greater than the true number of vehicles in a. In addition, if coordinator $C(a-1)$ has not added any vehicles to area a from time $t-d$ to time t, than coordinator c's estimate of the number of vehicles in area a must be correct because every vehicle added prior to $t-d$ will have checked in with coordinator c by time t. By a similar argument we also have that coordinator c's estimate for an area $a \in D(c)$ can never be less than the true number of vehicles in a and if coordinator $C(a)$ has not removed any vehicles from area a from time $t-d$ to time t, then this estimate must be correct.

2.4 Basic Model of Distributed Dynamics

To model our system, we will use a discrete event system (DES) model of the type described in [3]. This is a partially asynchronous model [1] in which the state of the system is updated according to the occurrence of certain events, with the current state of the system and certain assumptions about the system's behavior governing which events may occur at any given time. We start by defining our state variables, all of

which take values in the set of non-negative integers $\mathbb{N} = \{0, 1, 2, \ldots\}$. The number of vehicles in area a will be denoted by x_a and a coordinator c's estimate of that number will be denoted by $\tilde{x}_a^c$ (and for a particular area a this variable is only defined for a coordinator c if $c \in C(a) \cup C(a-1)$). Let x be the complete state of the system (i.e., the collection of the above x_a and $\tilde{x}_a^c$ values defined for each $a \in \mathcal{A}$ and applicable values of c) and let $\mathcal{X} = \mathbb{N}^{|x|}$ be the state space. Let $x(k)$ denote the state of the system at time step $k \in \mathbb{N}$, where k maps to a real time value $t(k)$ and this mapping satisfies both $k' > k \Rightarrow t(k') > t(k)$ and $\lim_{k\to\infty} t(k) = \infty$.

As mentioned, the state of the system is altered by the occurrence of particular events. Let us denote the set of all events that occur at time step k as $e(k)$. The set $e(k)$ contains one or more "partial" events that describe changes to the system such as the transfer of vehicles from one area to another and a coordinator updating its various estimate values. Let $e^{\alpha}_{a\to a^\star}$ denote the transfer of α vehicles from area a to area $a^\star$. Let $e^{+\beta}_{c,a}$ denote an increase of β to coordinator c's estimate for area a occurring when c adds a new vehicle to its list for a and let $e^{-\gamma}_{c,a}$ denote a decrease of γ to that estimate occurring when a vehicle on that list misses its check-in deadline and is removed. With these definitions we can also denote the set to which an event $e(k)$ must belong as the event space $\mathcal{E}$. Simply speaking, an element of $\mathcal{E}$ is any collection of partial events that are each properly defined (i.e., α, β, and γ must always be non-negative integers, $a^\star$ must be in $D(C(a))$ for transfer events, and c and a must correspond to a state variable $\tilde{x}_a^c$ for estimate change events).

With the above description of our event space, we can easily define the update to the state that occurs at each time step as follows. First, the state $x_a(k+1)$ is equal to $x_a(k)$ plus the net transfer of vehicles in and out of area a at time step k. Second, the estimate $\tilde{x}_a^c(k+1)$ is equal to $\tilde{x}_a^c(k)$ plus all the increases and minus all the decreases that occur at time step k.

Now, while the event space $\mathcal{E}$ lists all the possible events that could theoretically occur, the physical behavior of our system will greatly curtail both the number of events that may occur at a particular time step and the series of events that may occur over time. The event at time step k, $e(k)$, is constrained to be one of the events from a set valued enable function g which depends on the state of the system at time k (and it is this enable function that determines our control law). The purpose of this control law is to make the system act in the same general way as the ring of areas did [11] despite the shortcuts that are now permitted. That is to say that we want to the area holes and hills (i.e., those areas with $x_a = \min_{a'\in\mathcal{A}} x_{a'}$ and $x_a \geq 2 + \min_{a'\in\mathcal{A}} x_{a'}$ respectively) to move towards each other in order to guarantee that the holes are eventually eliminated. Specifically, every event $e(k) \in g(x(k))$ must satisfy the following conditions:

1. For each coordinator $c \in \mathcal{C}$ there exists at most one partial event event $e^{\alpha}_{a\to a^\star} \in e(k)$ with $C(a) = c$. This is equivalent to saying that a coordinator may only execute one transfer of vehicles at a time (i.e., from one specific area to another specific area). The purpose of this rule is to keep coordinator c from trying to fill in a hole with vehicles from two different areas simultaneously. Not having this rule could result in a situation in which a hole could avoid being eliminated indefinitely.
2. Let $c = C(a)$ for a partial event $e^{\alpha}_{a\to a^\star} \in e(k)$. Pursuant to our discussion in Section 2.2 concerning the restrictions on the way vehicles may be transferred, the destination area of the transfer, $a^\star$, must belong to the following set

$D(c) = \{a' \in \mathcal{A} : C(a'-1) = c\}$. We also require that $a^\star$ be one of the areas in $D(c)$ perceived to have the least number of vehicles, so

$$\tilde{x}^c_{a^\star}(k) = \min_{a' \in D(c)} \tilde{x}^c_{a'}(k) \tag{1}$$

Lastly, coordinator c is required to transfer vehicles to area $a+1$ if it has the smallest estimated number of vehicles of any area in $D(c)$, i.e.,

$$a^\star = a+1 \text{ if } \tilde{x}^c_{a+1}(k) \leq \tilde{x}^c_{a'}(k) \text{ for all } a' \in D(c) \tag{2}$$

Rule (1) is necessary in order to prevent coordinators from ignoring the need to fill in holes. Rule (2) serves the same purpose and is necessary because of the way parts 3(a) and 3(b) of the control algorithm is defined below.

3. Let a and $c = C(a)$ be the area and coordinator for a partial event $e^\alpha_{a \to a^\star} \in e(k)$. Possible values for α, the number of vehicles transferred in the event $e^\alpha_{a \to a^\star}$, are determined according to the following rules:

   (a) If $a^\star \neq a+1$ then

$$\alpha = 0 \qquad \text{if } \tilde{x}^c_a(k) \leq \tilde{x}^c_{a^\star}(k) + 1, \tag{3}$$

$$1 \leq \alpha \leq \left\lfloor \frac{\tilde{x}^c_a(k) - \tilde{x}^c_{a^\star}(k)}{2} \right\rfloor \text{ if } \tilde{x}^c_a(k) \geq \tilde{x}^c_{a^\star}(k) + 2, \tag{4}$$

   This portion of the control algorithm says that coordinator c can only transfer a number of vehicles from area a to area $a^\star$ that keeps $\tilde{x}^c_a(k+1)$ greater than or equal $\tilde{x}^c_{a^\star}(k+1)$ (and it must transfer a vehicle if it can do so while meeting this condition). This is equivalent to the standard load balancing rule (e.g., [1, 2]) which prevents a processor from taking any action that would make itself the least loaded processor in the system. As we will see shortly, this part of the control algorithm moderates vehicle transfers across short-cuts to the area ring in a way that is less "aggressive" than what is allowed in next case.

   (b) If $a^\star = a+1$ then

$$\alpha = 0 \qquad \text{if } \tilde{x}^c_a(k) \leq \tilde{x}^c_{a^\star}(k), \tag{5}$$

$$1 \leq \alpha \leq \left\lceil \frac{\tilde{x}^c_a(k) - \tilde{x}^c_{a^\star}(k)}{2} \right\rceil \text{ if } \tilde{x}^c_a(k) \geq \tilde{x}^c_{a^\star}(k) + 1, \tag{6}$$

   where $\lceil \cdot \rceil$ is the ceiling function (i.e., the smallest integer no less than its argument). This part of the control algorithm is more aggressive than part 3(a) in that it allows coordinator c to transfer enough vehicles from a to $a^\star$ to make $\tilde{x}^c_a(k+1)$ one less than $\tilde{x}^c_{a^\star}(k)$. Under certain circumstances this means that coordinator c may cause area a to become one of the areas with the least number of vehicles. Rules (5) and (6) are equivalent to those seen in [11] for the unidirectional ring.

As mentioned, the series of events that can occur over time is also limited by our system. Let $\mathcal{E}^{\mathbb{N}}$ be the set of all event trajectories and let the set of valid event trajectories $E_V \subset \mathcal{E}^{\mathbb{N}}$ contains all event trajectories $E = [e(0), e(1), \ldots]$ such that there exists a state trajectory $X = [x(0), x(1), \ldots] \in \mathcal{X}^{\mathbb{N}}$ satisfying $e(k) \in g(x(k))$ for all $k \in \mathbb{N}$. Whereas the enable function captures the dynamics of this system from

one time step to the next, we will need to define a set of allowed trajectories in order to fully describe its behavior over time. This subset of E_V, denoted E_A, consists of all event trajectories that meet the following conditions for some positive number B:

1. For every $a \in \mathcal{A}$, there exists no more than B time steps between two partial events of type $e^{\alpha}_{a \to a^\star}$. In other words, the coordinator for area a can go no longer than B time steps without attempting to balance the number of vehicles in that area with the number of vehicles in some other area $a^\star \in D(c)$ according to the rules of the control algorithm described by the enable function g.
2. The following inequalities hold for all $k \in \mathbb{N}$ as discussed in Section 2.3.

$$\tilde{x}_a^{C(a)}(k) \leq x_a(k) \text{ for all } a \in \mathcal{A} \tag{7}$$

$$\tilde{x}_{a+1}^{C(a)}(k) \geq x_{a+1}(k) \text{ for all } a \in \mathcal{A} \tag{8}$$

3. As discussed in Section 2.3, for all $a \in \mathcal{A}$, whenever coordinator $C(a)$ does not transfer any vehicles out of area a for B time steps the following holds for coordinator $C(a-1)$

$$\tilde{x}_a^{C(a-1)}(k) = x_a(k) \tag{9}$$

and whenever coordinator $C(a-1)$ does not transfer any vehicles into area a for B time steps then we have coordinator $C(a)$

$$\tilde{x}_a^{C(a)}(k) = x_a(k) \tag{10}$$

That such a constant B is guaranteed to exist for our system may not be intuitively obvious, and so we explain how one could find such a number. Specifically, B should be the largest number of events that may occur in a time interval of length d so that the properties listed in E_A correspond to the ones we discussed in Section 2.3. First of all, each event must be triggered by a vehicle checking in with a coordinator (coordinators can only transfer vehicles at these times and have no reason to update their estimates at other times). Second, the time between events triggered by the same vehicle can be bounded from below by a positive constant, say ϵ. At the very worst, there must be some minimum computation delay involved with a coordinator dealing with the checking in of a vehicle before it is ready to process another check in, but usually this delay will be much longer as the vehicle will be traveling through the area (to either patrol or visit another coordinator) and not repeatedly checking in with the same coordinator as often as the coordinator will let it. Therefore, the number of events that can be triggered in a time period d by a single vehicle is equal to $\frac{d}{\epsilon}$, and since we are assuming that there are a fixed number of vehicles, L, (and also assuming that there are no vehicle arrivals or departures) we can use for B any number greater than or equal to $\frac{dL}{\epsilon}$.

3 Convergence Results

In this section we provide the statements of our main theorems showing that the system modeled in Section 2.4 will converge to a balanced state within a finite number of events of known bound. Proofs are included in the Appendix.

Lemma 1 *Define a constant* $m \triangleq \left\lfloor \frac{L}{N_A} \right\rfloor$ *where* L *is the total number of vehicles in the system and* N_A *is the number of areas. The subset of the state space in which the number of vehicles in each area is as balanced as possible,*

$$\mathcal{X}_I = \left\{ x \in \mathcal{X} : x_a \in \{m, m+1\} \; \forall \, a \in A \right\} \tag{11}$$

is invariant (i.e., $x(k) \in \mathcal{X}_I \Rightarrow x(k+T) \in \mathcal{X}_I$ *for all* $k, T \in \mathbb{N}$*).*

Lemma 2 *The functions*

$$\underline{m}(k) \triangleq \min_{a \in A} x_a(k) \tag{12}$$

$$\overline{m}(k) \triangleq \max_{a \in A} x_a(k) \tag{13}$$

which denote the minimum and maximum number of vehicles in any area at time step k *are non-decreasing and non-increasing respectively.*

Lemma 3 *For all* $k \in \mathbb{N}$ *such that* $\underline{m}(k)$ *and* $\underline{m}(k+1)$ *are equal to some constant* $\underline{m}$ *the following relationship holds for all* $a \in A$*,*

$$x_a(k) \geq \underline{m} + 1 \text{ and } x_a(k+1) = \underline{m} \Rightarrow x_{a+1}(k) = \underline{m} \text{ and } x_{a+1}(k+1) \geq \underline{m} + 1 \tag{14}$$

In other words, so long as $\underline{m}(k)$ *does not increase from time step* k *to time step* $k+1$*, then in order for area* a *to become a hole it must be the case that a hole at area* $a+1$ *is eliminated (i.e., no new holes are created and those that already exist may only stay where they are, move backwards along the area ring, or be eliminated).*

Lemma 4 *For all* $k \in \mathbb{N}$ *such that* $\underline{m}(k)$ *and* $\underline{m}(k+1)$ *are equal to some constant* $\underline{m}$ *and such that there exists an area* $a \in A$ *with* $x_a(k) \geq \underline{m} + 2$*, at least one of the following statements must be true,*

$$x_a(k+1) \geq \underline{m} + 2 \tag{15}$$

$$x_a(k+1) = \underline{m} + 1 \text{ and } x_{a+1}(k+1) \geq \underline{m} + 2 \tag{16}$$

$$x_a(k+1) = \underline{m}+1 \text{ and } \left| \left\{ a' \in A : \; x_{a'}(k+1) = \underline{m} \right\} \right| < \left| \left\{ a' \in A : \; x_{a'}(k) = \underline{m} \right\} \right| \tag{17}$$

In other words, so long as $\underline{m}(k)$ *does not increase from time step* k *to time step* $k+1$ *then whenever area* a *is a hill at time step* k *but does not remain a hill at the next time step, then it must be the case that area* a *becomes (part of) a plateau and either area* $a+1$ *goes from being a plateau to being a hill or the number of holes decreases.*

Theorem 1 *For all* $k \in \mathbb{N}$ *such that* $x(k) \in \mathcal{X} - \mathcal{X}_I$*, there exists a finite number* $T \in \mathbb{N}$ *such that at least one of the following statements is true,*

$$\underline{m}(k+T) \geq \underline{m}(k) + 1 \tag{18}$$

$$\left| \left\{ a' \in A : \; x_{a'}(k+T) = \underline{m}(k+T) \right\} \right| < \left| \left\{ a' \in A : \; x_{a'}(k) = \underline{m}(k) \right\} \right| \tag{19}$$

In other words, eventually either the function $\underline{m}(k)$ *must increase or the number of holes must decrease. The value of* T *is also bounded from above by* $2B(N_A - 1)$*.*

Corollary 1 *For any initial condition $x(0) \in \mathcal{X}$, there exists a finite number $T \in \mathbb{N}$ such that $x(T) \in \mathcal{X}_I$. The value of T is also bounded from above by $2B(N_A - 1)^2 \lfloor \frac{L}{N_A} \rfloor$.*

4 Simulation Studies

The rationale for using the more complex algorithm of this paper over the one for the ring interconnection of [11] has been that the shortcuts should in most cases decrease the amount of time that it takes the system to converge to the globally balanced state. In this section we support that argument through two simulation studies. These simulations were run assuming the delay associated with the vehicle's inter-coordinator travel comes from a designated distribution, and because the system trajectories scale linearly with the value of the maximum delay, that maximum delay is always assumed to be equal to one and time is then expressed in dimensionless units. The distribution of delays was taken to be uniform on the interval $[0.9, 1]$ so that we would get close to worst case performance while maintaining some randomness. When initially adding vehicles to the system, we put them all in the same (randomly chosen) area in order to start off with the maximum imbalance possible, and they first check-in with one of that area's coordinators at some time in the interval $[0, 1]$. The time it takes the system to balance is averaged over multiple individual simulation runs for each interconnection tested.

For our first simulation study we generated a number of random[1] interconnections of 20 areas each, selected 60 of these to get a good distribution of graph connectivity as measured by the mean distance (number of graph edges) between areas, and then added the interconnections at either extreme (i.e., the unidirectional ring and the interconnection with just one coordinator). As clearly shown in the scatter plot of Fig. 4a, there is a direct relationship between the mean inter-area distance and the mean convergence time of the system when the number of vehicles is fixed. It is also apparent that the ring interconnection (the right most data point) has a worse average convergence time than any of the interconnections with shortcuts. Although the spread of data points in the middle of Fig. 4a makes it hard to determine if those points fit a line or a curve, it appears that the increase in convergence time with respect to the increase in mean distance is greater for small mean distances than for large ones.

To explore this further we generated seven different 64 area interconnections by repeatedly giving a single coordinator ring responsibility for more areas (where the specific areas added at each stage where selected to reduce the diameter and mean distance of the interconnection by as much as possible). Figure 4b clearly shows a logarithmic-like curve connecting the mean convergence times of these interconnections. Again, the ring interconnection (the right most data point) has a worse average convergence time than the interconnections with shortcuts. (Also shown for comparison is the optimal convergence time for each interconnection assuming the worst case initial location for the vehicles. This optimal time is thus equal to the diameter of the interconnection.) This sort of relationship can be roughly explained

[1]These interconnections were generated by process that started with a unidirectional ring and then transferred ownership of areas from one coordinator to another. We cannot claim that this process produced a uniform sampling of all possible 20 area interconnections.

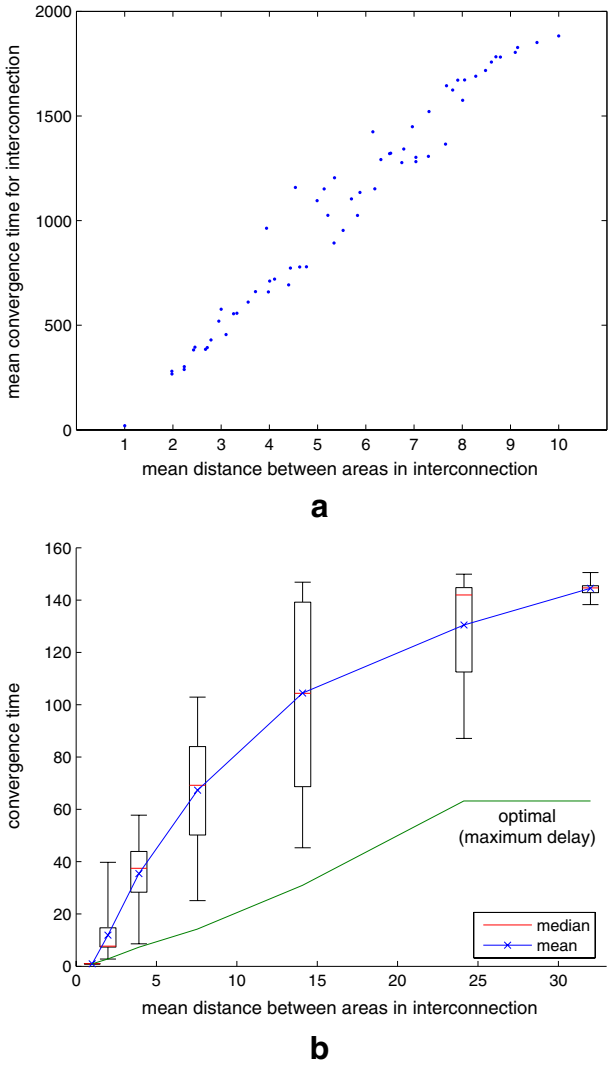

Fig. 4 Plot **a** gives mean distance between areas versus mean convergence time for 62 different 20 area interconnections using 80 vehicles (120 simulations per interconnection). Plot **b** shows the same relationship for seven specific 64 area interconnections (128 simulations per interconnection) plus box plots to illustrate the distribution of convergence times

with reference to the results of [11]. For interconnections that have fairly regular structure (i.e., where many coordinators occupy similar looking positions in the interconnection) then vehicles will usually get to the highly connected "central" areas quickly and then spread out from there. The more highly connected these central areas, the more paths the vehicles will have on which to spread out and the shorter those paths will be.

Now, even despite the restrictions our results place on the area-coordinator interconnections, we would like to note that there exist some straight forward ways of designing interconnections on typical spatial layouts (i.e., those based on some regular griding of the environment) that achieve good connectivity with a relatively small number of coordinators. As examples we present the layouts in Fig. 5.

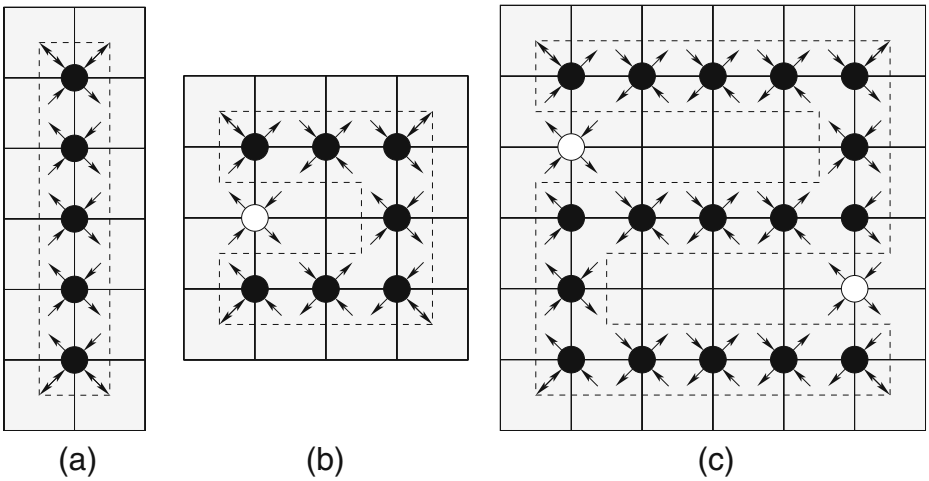

Fig. 5 A fairly efficient method for interconnecting a grid of areas (similar conventions to those in Fig. 3b). Layout **a** shows the basic form connected by (black) coordinators, while **b** shows how the basic form is "folded" into a four by four grid and connected at the ends by an additional (white) coordinator. Layout **c** shows one possibility for a six by six grid

5 Conclusion

In this paper we have extended the results of [11] to area-coordinator interconnection topologies other than a unidirectional ring. Permitting the use of highly connected topologies and still achieving a globally balanced state makes our algorithm superior to implementations of the traditional load balancing algorithm for our scenario with respect to both one-time and persistent disturbances. Although presented in terms of a cooperative patrol mission, our distributed control algorithm has application to a wide range of balancing problems involving uniformly sized resource units. Possible future directions include extending our results to the prioritized balancing case (where each area should have a certain fraction of the total number of vehicles) and performing an analysis of the system's behavior under a persistent disturbance.

Appendix

A.1 Basic Properties

We start by stating a number of simple properties that hold for all $k \in \mathbb{N}$ and that will be of general use. Proofs are omitted due to simplicity.

Property 1 A coordinator $c \in C$'s estimate of the imbalance between the number of vehicles in an area $a \in S(c)$ and the number in an area $a^\star \in D(c)$ is always less than the real imbalance, i.e.,

$$\tilde{x}_a^c(k) - \tilde{x}_{a^\star}^c(k) \leq x_a(k) - x_{a^\star}(k)$$

Property 2 A coordinator $c \in C$ will never transfer vehicles from an area $a \in S(c)$ to an area $a^\star \in D(c)$ if $x_a(k) \leq x_{a^\star}(k)$.

Property 3 A coordinator $c \in C$ will not transfer vehicles out of an area $a \in S(c)$ so long as $x_a(k) = \underline{m}(k)$, nor will it transfer any vehicles to an area $a^\star \in D(c)$ so long as $x_{a^\star}(k) = \overline{m}(k)$

Property 4 For coordinator $c \in C$ and any $a \in D(c)$ we have

$$\underline{m}(k) \leq x_a(k) \leq \tilde{x}_a^c(k)$$

Property 5 When $x_a(k) = \underline{m}(k) + 1$, then coordinator $C(a)$ can only transfer vehicles from area a "forward" along the ring of areas (i.e., to area $a + 1$). In addition, it is limited to transferring no more than one vehicle from area a to area $a + 1$ per time step (and only if $x_{a+1}(k) = \underline{m}(k)$).

A.2 Proof of Lemma 1

We take any $k \in \mathbb{N}$ such that $x(k) \in \mathcal{X}_I$ and show that this implies $x(k + 1) \in \mathcal{X}_I$ as well. By the definition of $\mathcal{X}_I$ we know that $x_a(k) \in \{m, m + 1\}$ for all $a \in A$, which tells us many things. First, there is at least one area with m vehicles. (If not, then we would have $L = (m + 1)N_A$ which would give us the contradiction $m = \left\lfloor \frac{(m+1)N_A}{N_A} \right\rfloor = m + 1$.) This in turn tells us that $\underline{m}(k) = m$ (and $\overline{m}(k) \leq m + 1$) and so Property 5 implies that all vehicle transfers occur only along the forward direction of the area ring while $x(k) \in \mathcal{X}_I$.

Take any $a \in A$ with controlling coordinator $c = C(a)$. If $x_a(k) = m$, then Property 3 tells us that no vehicles will be removed from area a while Property 5 tells us that at most one will be added, so it must be that $x_a(k + 1) \in \{m, m + 1\}$. Likewise, if $x_a(k) = m + 1$, then $x_a(k) = \overline{m}(k)$ and those same properties tell us that no vehicles will be added to area a and at most one will be removed, so it must be that $x_a(k + 1) \in \{m, m + 1\}$ again. Since $x_a(k + 1) \in \{m, m + 1\}$ holds for all $a \in A$, we have that $x(k + 1) \in \mathcal{X}_I$ and by extension $x(k + T) \in \mathcal{X}_I$ for all $T \in \mathbb{N}$. □

A.3 Proof of Lemma 2

Let us focus first on $\underline{m}(k)$ and show that $\underline{m}(k + 1) \geq \underline{m}(k)$ for all $k \in \mathbb{N}$. Take any $k \in \mathbb{N}$ and any $a \in A$ and consider the number of vehicles in area a at time $k + 1$. According to the statement of our update function, $x_a(k + 1)$ is equal to $x_a(k)$ minus the number of its vehicles transferred by coordinator $c = C(a)$ (to some area $a^\star \in D(c)$) and plus the number of vehicles transferred to a by coordinator $c' = C(a - 1)$ (from some area $a' \in S(c')$). Letting the number of vehicles removed and added to area a at time step k be denoted by $\alpha_a^-(k)$ and $\alpha_a^+(k)$ respectively, the previous statement is equivalent to

$$x_a(k + 1) = x_a(k) + \alpha_a^+(k) - \alpha_a^-(k) \tag{20}$$

Since $\alpha_a^+(k)$ and $\alpha_a^-(k)$ are non-negative, we can bound $x_a(k+1)$ from below as

$$x_a(k+1) \geq x_a(k) - \alpha_s^-(k) \tag{21}$$

We proceed by noting that in all cases of the control algorithm the value of α in an event $e^{\alpha}_{a\to a^\star}$ can be bounded from above by $\left\lceil \frac{\tilde{x}_a^c(k)-\tilde{x}_{a^\star}^c(k)}{2} \right\rceil$. Then,

$$x_a(k+1) \geq x_a(k) - \left\lceil \frac{\tilde{x}_a^c(k) - \tilde{x}_{a^\star}^c(k)}{2} \right\rceil \tag{22}$$

Using the inequality $\lceil q \rceil < q+1$, Eq. 22 becomes

$$x_a(k+1) > x_a(k) - \left(\frac{\tilde{x}_a^c(k) - \tilde{x}_{a^\star}^c(k)}{2} + 1 \right) \tag{23}$$

By employing Property 1 and the fact that both $x_a(k)$ and $x_{a^\star}(k)$ can be no less than $\underline{m}(k)$ we can proceed as follows

$$\begin{aligned} x_a(k+1) &> x_a(k) - \left(\frac{\tilde{x}_a^c(k) - \tilde{x}_{a^\star}^c(k)}{2} + 1 \right) \\ &\geq x_a(k) - \left(\frac{x_a(k) - x_{a^\star}(k)}{2} + 1 \right) && (24) \\ &= \frac{1}{2}\left(x_a(k) + x_{a^\star}(k)\right) - 1 && (25) \\ &\geq \frac{1}{2}\left(\underline{m}(k) + \underline{m}(k)\right) - 1 \;=\; \underline{m}(k) - 1 && (26) \end{aligned}$$

But since $x_a(k+1)$ takes on only integer values, $x_a(k+1) > \underline{m}(k) - 1$ implies that $x_a(k+1) \geq \underline{m}(k)$. Since this holds for all $a \in A$ and all $k \in \mathbb{N}$, $\underline{m}(k)$ must be non-decreasing. A parallel argument shows that $\overline{m}(k)$ is non-increasing and is omitted for brevity. □

A.4 Proof of Lemma 3

Take any $k \in \mathbb{N}$ such that $\underline{m}(k) = \underline{m}(k+1) = \underline{m}$. In order to prove this lemma, we start by showing that an area $a \in A$ with $x_a(k) \geq \underline{m} + 1$ cannot become a hole due to its coordinator $c = C(a)$ transferring vehicles to an area $a^\star \in D(c) - \{a+1\}$ (i.e., by a transfer of vehicles across one of the shortcuts to the area ring). We can start with Eq. 21 from the proof of Lemma 2 and use rule (4) from part 3(a) of the control

algorithm (since $a^\star \neq a+1$) in conjunction with the inequality $\lfloor q \rfloor \leq q$ to proceed in a familiar fashion,

$$\begin{aligned} x_a(k+1) &\geq x_a(k) - \alpha_s^-(k) \\ &\geq x_a(k) - \left\lfloor \frac{\tilde{x}_a^c(k) - \tilde{x}_{a^\star}^c(k)}{2} \right\rfloor && (27) \\ &\geq x_a(k) - \frac{\tilde{x}_a^c(k) - \tilde{x}_{a^\star}^c(k)}{2} && (28) \\ &\geq x_a(k) - \frac{x_a(k) - x_{a^\star}(k)}{2} && (29) \\ &= \frac{1}{2}\left(x_a(k) + x_{a^\star}(k)\right) && (30) \\ &\geq \frac{1}{2}\left((\underline{m}+1) + \underline{m}\right) = \underline{m} + \frac{1}{2} && (31) \end{aligned}$$

But $x_a(k+1) \geq \underline{m} + \frac{1}{2}$ implies that $x_a(k+1) \geq \underline{m} + 1$, and thus area a cannot become a hole in this manner.

It is the case, however, that area a can become a hole when coordinator c transfers vehicles from a to $a+1$. In this case we are interested in the relationship between the values of $x_a(k), x_a(k+1), x_{a+1}(k)$, and $x_{a+1}(k+1)$. We start by recalling the inequality (25) from the proof of Lemma 2 which gave us a lower bound on $x_a(k+1)$ based on the most aggressive type of transfer (i.e., one from a to $a+1$). Substituting $a^\star = a+1$,

$$x_a(k+1) > \frac{1}{2}\left(x_a(k) + x_{a+1}(k)\right) - 1 \tag{32}$$

and rearranging to see what values of $x_{a+1}(k)$ can result in $x_a(k+1) = \underline{m}$ given that $x_a(k) \geq \underline{m} + 1$

$$\begin{aligned} x_{a+1}(k) &< 2x_a(k+1) - x_a(k) + 2 && (33) \\ &\leq 2\underline{m} - (\underline{m}+1) + 2 && (34) \\ &= \underline{m} + 1 && (35) \end{aligned}$$

and of course $x_{a+1}(k) < \underline{m} + 1$ implies that $x_{a+1}(k) = \underline{m}$. Thus area a can only become a hole at time $k+1$ if area $a+1$ is a hole at time k, but since this only happens when coordinator c transfers vehicles into area $a+1$ and no vehicles will be transferred out of $a+1$ (Property 3), then it must be the case that $x_{a+1}(k+1) \geq \underline{m} + 1$. Thus the creation of a new hole at area a must be accompanied by the elimination of a hole at area $a+1$. □

A.5 Proof of Lemma 4

Take any $k \in \mathbb{N}$ such that $\underline{m}(k) = \underline{m}(k+1) = \underline{m}$ and any area a such that $x_a(k) \geq \underline{m} + 2$. If coordinator $c = C(a)$ does not transfer any vehicles out of a at time k, then clearly Eq. 15 holds. Let us examine what happens when coordinator c does transfer vehicles from a to some destination $a^\star$ by considering the cases $a^\star = a+1$ and $a^\star \neq a+1$ separately.

Case 1 $a^\star \in D(c) - \{a+1\}$. The inequality (30) in Lemma 3 is a lower bound on $x_a(k+1)$ for a vehicle transfer of this type, so we can simply proceed from there using $x_a(k) \geq \underline{m} + 2$,

$$\begin{aligned} x_a(k+1) &\geq \frac{1}{2}\left(x_a(k) + x_{a^\star}(k)\right) \\ &\geq \frac{1}{2}x_{a^\star}(k) + \frac{1}{2}\underline{m} + 1 \end{aligned} \tag{36}$$

Let us evaluate Eq. 36 in two cases that depend on the value of $x_{a^\star}(k)$.

Case 1a $x_{a^\star}(k) \geq \underline{m} + 1$. In this case $a^\star$ is not a hole, and Eq. 36 gives us $x_a(k+1) \geq \underline{m} + \frac{3}{2}$ which implies that $x_a(k+1) \geq \underline{m} + 2$ and so area a would remain a hill, thus satisfying condition (15).

Case 1b $x_{a^\star}(k) = \underline{m}$. Here Eq. 36 tells us that $x_a(k+1) \geq \underline{m} + 1$ and thus the hill at area a may disappear but the hole at area $a^\star$ gets eliminated, thus satisfying condition (17). (One might worry about the hole at $a^\star$ moving to $a^\star - 1$ at the same time step and thereby avoiding elimination, but recall that since $a^\star \in D(c)$, we have $a^\star - 1 \in S(c)$. Thus for this to happen, coordinator c would have to make two simultaneous transfers of vehicles and this is explicitly prohibited by part 1 of the control algorithm.)

Case 2 $a^\star = a + 1$. For this case we can use Eq. 32 from Lemma 3 to get a lower bound on $x_a(k+1)$ after a transfer of vehicles from a to $a+1$ when we have $x_a(k) \geq \underline{m} + 2$,

$$\begin{aligned} x_a(k+1) &> \frac{1}{2}\left(x_a(k) + x_{a+1}(k)\right) - 1 \\ &\geq \frac{1}{2}x_{a+1}(k) + \frac{1}{2}\underline{m} \end{aligned} \tag{37}$$

And we can evaluate Eq. 37 depending on the value of $x_{a+1}(k)$ using three cases.

Case 2a $x_{a+1}(k) \geq \underline{m} + 2$. Here Eq. 37 gives us $x_a(k+1) > \underline{m} + 1$ (implying $x_a(k+1) \geq \underline{m} + 2$) and thus area a would remain a hill and condition (15) is satisfied.

Case 2b $x_{a+1}(k) = \underline{m}$. In this case Eq. 37 tells us that $x_a(k+1) > \underline{m}$ (implying $x_a(k+1) \geq \underline{m} + 1$) so as in Case 1b it may be that the hill at a disappears but only because a hole at $a+1$ gets eliminated (so either condition (15) or condition (17) is satisfied).

Case 2c $x_{a+1}(k) = \underline{m} + 1$. This case is somewhat more complicated as it depends on the actions of coordinator $C(a+1)$. Inequality (37) gives us that $x_a(k+1) > \underline{m} + \frac{1}{2}$ (implying $x_a(k+1) \geq \underline{m} + 1$) so the hill at area a may disappear. Now, since $x_{a+1}(k) = \underline{m} + 1$, Property 5 tells us that coordinator $C(a+1)$ is limited to transferring at most one vehicle from $a+1$ to $a+2$. If it does not transfer a vehicle at time k, then $x_a(k+1) \geq \underline{m} + 2$ due to coordinator c's transfer of vehicles from a to $a+1$, and so a hill is created at area $a+1$, thus satisfying condition (16). On the other hand, if coordinator $C(a+1)$ does transfer a vehicle from $a+1$ to $a+2$ at time k it

must be because $a+2$ is a hole (Property 2), but since coordinator c is transferring at least one vehicle from a to $a+1$, area $a+1$ will experience no net loss and we have $x_{a+1}(k+1) \geq x_{a+1}(k) = \underline{m}+1$. Thus if the hill at a does disappear without creating a hill at $a+1$, it is because a hole at $a+2$ was eliminated, thus satisfying condition (17).

The case analysis above has covered all possibilities and has shown that at least one of the three conditions of Lemma 4 always holds. □

A.6 Proof of Theorem 1

The essence of Lemmas 3 and 4 was to define the possible behavior of the holes and hills, i.e., to establish that holes may move backwards on the area ring (or be eliminated) while the hills may move forward on the area ring (and can only be eliminated when a hole is eliminated at the same time). In what follows, we will show that holes and hills *must* move in this manner and that this movement ensures the elimination of at least one hole within a bounded length of time.

Take any $k \in \mathbb{N}$ such that $x(k) \in \mathcal{X} - \mathcal{X}_I$, and assume that $\underline{m}(k')$ is equal to some constant $\underline{m}$ for all $k' \geq k$ (otherwise Eq. 18 holds trivially). It is always the case that there exists at least one area which is a hole, and since $x(k) \notin \mathcal{X}_I$ it must also be the case that there is at least one area that is a hill (if not then $x_a(k) \in \{\underline{m}(k), \underline{m}(k)+1\}$ which implies $x(k) \in \mathcal{X}_I$). Furthermore, whenever there are both holes and hills there must exist either at least one hole $\underline{a}$ and one hill $\overline{a}$ such that the block of areas $\overline{a}+1, \overline{a}+2, \ldots, \underline{a}-2, \underline{a}-1$ is a plateau, or at least one hole and hill such that $\overline{a}+1 = \underline{a}$. For now, let us assume that there is no hill/hole pair $(\overline{a}, \underline{a})$ such that $\overline{a}+1 = \underline{a}$ at time step k and let us show that the system must evolve to the point where there is at least one such pair. Let $\overline{a}$ and $\underline{a}$ be the hill and hole guaranteed to exist with a connecting plateau $\overline{a}+1, \overline{a}+2, \ldots, \underline{a}-2, \underline{a}-1$ at time step k. Since $\overline{a}+1 \neq \underline{a}$ this plateau must contain at least one area. We will show next that the length of this plateau must decrease within a finite period of time (i.e. that one of the areas of the plateau will become either a hole or a hill within a finite number of time steps) or else a hole will be eliminated.

Now, if at any time step $k' \geq k$ one or more vehicles are added to an area a' in the plateau besides $\underline{a}-1$, then a' automatically becomes a hill at $k'+1$ (because Property 5 ensures that no vehicles will be moved out of any of these areas at time k'). If one or more vehicles is transferred to area $\underline{a}-1$ at time $k' \geq k$ then either $\underline{a}-1$ becomes a hill and/or the hole at area $\underline{a}$ moves to area $\underline{a}-1$ at the same time and is eliminated. Since both these situations result in one of our desired outcomes, we need only proceed with our argument for the case in which no vehicles are added to any of the plateau areas from time step k on.

Now according to Lemma 3, only one of three things may happen to the hole at $\underline{a}$ at each time step, 1) it remains a hole, 2) it moves to area $\underline{a}-1$, or 3) it gets eliminated (without the generation of a new hole at $\underline{a}-1$). Disregarding the third possibility (because it automatically gives us a desired result), it is our task to show that the second possibility eventually happens (i.e., that coordinator $C(\underline{a}-1)$ will eventually transfer a vehicle from $\underline{a}-1$ to $\underline{a}$). Now, if coordinator $C(\underline{a}-1)$ has not made such a transfer by time step $k+B$, then according to Eq. 10 from the description of the allowed trajectories E_A we will have

$$\tilde{x}_{\underline{a}}^{C(\underline{a}-1)}(k') = x_{\underline{a}}(k') = \underline{m} \quad \text{for all} \quad k' \geq k+B \tag{38}$$

And (under our assumption about no vehicles being added to the plateau areas) we also have that $x_{\underline{a}-1}(k') = \underline{m} + 1$ for all $k' \geq k$ and so by Eq. 9 we will have

$$\tilde{x}_{\underline{a}-1}^{C(\underline{a}-1)}(k') = x_{\underline{a}-1}(k') = \underline{m} + 1 \quad \text{for all} \quad k' \geq k + B \tag{39}$$

Therefore, for all $k' \geq k + B$ coordinator $C(\underline{a} - 1)$'s estimate of the imbalance between $\underline{a} - 1$ and $\underline{a}$, $\tilde{x}_{\underline{a}-1}^{C(\underline{a}-1)}(k') - \tilde{x}_{\underline{a}}^{C(\underline{a}-1)}(k')$, is accurate and equal to one. Thus when $C(\underline{a} - 1)$ attempts to balance area $\underline{a} - 1$ (which it must do within the next B time steps according to the definition of E_A), Property 5 reminds us that it can only transfer one vehicle to $\underline{a}$ and part 3(b) of the control algorithm tells us that it will indeed do this. Thus by time step $k + 2B$, the hole at $\underline{a}$ moves to $\underline{a} - 1$ (or one of our previous assumptions was violated at some time step k' between k and $k + 2B$ and either an area in the plateau became a hill or a hole was eliminated somewhere in the system).

With this information we can see that at time step $k + 2B$ there must be a hill and hole separated by a plateau that is at least one area shorter than the one we were considering at time step k. Since this process is guaranteed to keep repeating, we will eventually get to the point where we have a hill $\overline{a}$ and hole $\underline{a}$ such that $\overline{a} + 1 = \underline{a}$. Specifically, since a plateau can originally consist of no more than $N_A - 2$ areas (i.e., the total number of areas minus the areas $\overline{a}$ and $\underline{a}$) we will have this condition no later than $k + 2B(N_A - 2)$.

We now show that in the case were $\overline{a} + 1 = \underline{a}$, coordinator $C(\overline{a})$ will eventually transfer a vehicle from $\overline{a}$ to $\underline{a}$ thereby eliminating the hole at $\underline{a}$. According to Lemma 4, either 1) area $\overline{a}$ remains a hill, 2) the hill at $\overline{a}$ moves to area $\overline{a} + 1 = \underline{a}$, or 3) a hole is eliminated somewhere in the system. Again disregarding the third possibility (because it automatically gives us a desired result), it is our task to show that the second possibility eventually happens.

Property 3 ensures that no vehicles will be removed from area $\underline{a}$ as long as it remains a hole for $k' \geq k + 2B(N_A - 2)$. Until shown otherwise, assume coordinator $C(\overline{a})$ does not add any vehicles to area $\underline{a}$ for $k' \geq k + 2B(N_A - 2)$. Thus Eq. 10 gives us

$$x_{\underline{a}}^{C(\overline{a})}(k') = x_{\underline{a}}(k') = \underline{m} \quad \text{for all} \quad k' \geq k + 2B(N_A - 2) + B \tag{40}$$

So coordinator $C(\overline{a})$ has an accurate estimate of $x_{\underline{a}}(k')$ for all $k' \geq k + 2B(N_A - 2) + B$. Now, in order to show that coordinator $C(\overline{a})$ will eventually transfer a vehicle from $\overline{a}$ to $\underline{a}$, we first need to show that $\tilde{x}_{\overline{a}}^{C(\overline{a})}(k') \geq \underline{m} + 2$ for all k' greater than some value. If coordinator $C(\overline{a})$ does not transfer any vehicles out of area $\overline{a}$ from time step $k + 2B(N_A - 2)$ on, then Eq. 9 gives us

$$x_{\overline{a}}^{C(\overline{a})}(k') = x_{\overline{a}}(k') \geq \underline{m} + 2 \quad \text{for all} \quad k' \geq k + 2B(N_A - 2) + B \tag{41}$$

Unfortunately, coordinator $C(\overline{a})$ is not generally restricted from removing vehicles from area $\overline{a}$ and the logic leading up to Eq. 41 does not hold when coordinator $C(\overline{a})$ makes such a removal before $k + 2B(N_A - 2) + B$. We can, however, get an identical bound using another line of reasoning. Let $k^\star \geq k + 2B(N_A - 2)$ be some time step such that $e_{\overline{a} \to a^\star}^{\alpha} \in e(k^\star)$ for some $a^\star \in D(C(\overline{a}))$ and some $\alpha \geq 1$ (i.e., a time step when coordinator $C(\overline{a})$ transfers vehicles out of area $\overline{a}$). If $a^\star = \underline{a}$, then we have our desired

result. If $a^\star \neq \underline{a}$ then we can analyze the value of $\tilde{x}_{\overline{a}}^{C(\overline{a})}(k^\star+1)$. We start with the following equation which describes the update of coordinator $C(\overline{a})$'s estimate for $x_{\overline{a}}$

$$\tilde{x}_{\overline{a}}^{C(\overline{a})}(k^\star+1) = \tilde{x}_{\overline{a}}^{C(\overline{a})}(k^\star) - \alpha_{\overline{a}}^{-}(k^\star) + \beta_{\overline{a}}^{C(\overline{a})}(k^\star) \tag{42}$$

where $\alpha_{\overline{a}}^{-}(k^\star)$ is the number of vehicles $C(\overline{a})$ transferred out of area $\overline{a}$ at time step $k^\star$ and $\beta_{\overline{a}}^{C(\overline{a})}(k^\star)$ is the number of previously unseen vehicles that check in with coordinator $C(\overline{a})$ at time step $k^\star$. We know $\beta_{\overline{a}}^{c}(k^\star)$ is non-negative and since $a^\star \neq \underline{a}$ rule (4) from part 3(a) of the control algorithm determines the upper bound on $\alpha_{\overline{a}}^{-}(k^\star)$. We can thus proceed as follows,

$$\begin{aligned}
\tilde{x}_{\overline{a}}^{C(\overline{a})}(k^\star+1) &\geq \tilde{x}_{\overline{a}}^{C(\overline{a})}(k^\star) - \alpha_{\overline{a}}^{-}(k^\star) && (43)\\
&\geq \tilde{x}_{\overline{a}}^{C(\overline{a})}(k^\star) - \left\lfloor \frac{\tilde{x}_{\overline{a}}^{C(\overline{a})}(k^\star) - \tilde{x}_{a^\star}^{C(\overline{a})}(k^\star)}{2} \right\rfloor && (44)\\
&\geq \tilde{x}_{\overline{a}}^{C(\overline{a})}(k^\star) - \frac{\tilde{x}_{\overline{a}}^{C(\overline{a})}(k^\star) - \tilde{x}_{a^\star}^{C(\overline{a})}(k^\star)}{2} && (45)\\
&= \frac{1}{2}\left(\tilde{x}_{\overline{a}}^{C(\overline{a})}(k^\star) + \tilde{x}_{a^\star}^{C(\overline{a})}(k^\star)\right) && (46)
\end{aligned}$$

If area $a^\star$ is a hole at time $k^\star$, then this transfer eliminates a hole and we have a desired result (recall that if $a^\star$ was a hole, then it could not escape elimination by moving to $a^\star - 1$ at $k^\star$ because $C(a^\star - 1) = c$ and coordinator $C(\overline{a})$ is limited to transferring vehicles from at most one one area in $S(C(\overline{a}))$ per time step). Proceeding in the case where $a^\star$ is not a hole, we have $x_{a^\star}(k^\star) \geq \underline{m} + 1$ and thus Eq. 8 gives us that $\tilde{x}_{a^\star}^{C(\overline{a})}(k^\star) \geq \underline{m} + 1$ as well. In addition, since $\alpha_{\overline{a}}^{-}(k^\star) \geq 1$ it must be the case that the condition of rule (4) is met at time step $k^\star$, so we also have $\tilde{x}_{\overline{a}}^{C(\overline{a})}(k^\star) \geq \tilde{x}_{a^\star}^{C(\overline{a})}(k^\star) + 2$. We can use these two inequalities to evaluate Eq. 46 as follows,

$$\begin{aligned}
\tilde{x}_{\overline{a}}^{C(\overline{a})}(k^\star+1) &\geq \frac{1}{2}\left(\tilde{x}_{\overline{a}}^{C(\overline{a})}(k^\star) + \tilde{x}_{a^\star}^{C(\overline{a})}(k^\star)\right)\\
&\geq \frac{1}{2}\left((\tilde{x}_{a^\star}^{C(\overline{a})}(k^\star) + 2) + \tilde{x}_{a^\star}^{C(\overline{a})}(k^\star)\right) && (47)\\
&= \tilde{x}_{a^\star}^{C(\overline{a})}(k^\star) + 1 && (48)\\
&\geq \underline{m} + 2 && (49)
\end{aligned}$$

The implication of Eq. 49 is that if coordinator $C(\overline{a})$ transfers some vehicles to an area $a^\star \neq \underline{a}$ at time step $k^\star \geq k + 2B(N_A - 2)$, then for all $k' \geq k^\star + 1$ coordinator $C(\overline{a})$'s estimate for area $\overline{a}$ satisfies $\tilde{x}_{\overline{a}}(k') \geq \underline{m} + 2$. This is because events $e_{C(\overline{a}),\overline{a}}^{+\beta}$ after time step $k^\star$ cannot decrease $\tilde{x}_{\overline{a}}^{C(\overline{a})}$ and another event $e_{\overline{a}\to a^\star}^{\alpha}$ just gives us Eq. 49 again.

In summary, coordinator $C(\overline{a})$ must try and balance area $\overline{a}$ at some time k' such that $k + 2B(N_A - 2) + B \leq k' \leq k + 2B(N_A - 2) + 2B - 1$ and when it does it is either the case that it has not transferred any vehicles out of $\overline{a}$ from k to $k' - 1$ (in which case Eq. 41 applies) or that it transferred vehicles out of $\overline{a}$ at least once in that interval (in which case Eq. 49 applies). Since both Eqs. 41 and 49 give us $\tilde{x}_{\overline{a}}^{C(\overline{a})}(k') \geq \underline{m} + 2$, and we have already shown that $\tilde{x}_{\underline{a}}^{C(\overline{a})}(k') = \underline{m} + 1$ for the values of k' under consideration, we have that $C(\overline{a})$ must transfer at least one vehicle from $\overline{a}$ to

$\underline{a}$ (given rule (6) in part 3(b) of the control algorithm) Thus the hill at $\overline{a}$ moves to $\underline{a}$ by time step $k + 2B(N_A - 1)$ (or a hole somewhere in the system was eliminated at an earlier time step).

In conclusion, the above proof has shown that a hole is eliminated in the (unbalanced) system at least once every $2B(N_A - 1)$ time steps (and if all the holes at a given minimum vehicle level are eliminated, that minimum must rise). Thus we have shown that for some $T \leq 2B(N_A - 1)$, either the condition (18) or condition (19) of theorem must hold. □

A.7 Proof of Corollary 1

Assuming that the function $\underline{m}(k)$ remains constant until proven otherwise, we can see that since the time it takes to eliminate a is hole bounded from above by $2B(N_A - 1)$ (Theorem 1), then all the holes that exist at a time k must be eliminated no later than time $k + 2B(N_A - 1)^2$ because there can be at most $N_A - 1$ holes whenever the state of the system is not in $\mathcal{X}_I$. When all the holes are eliminated, the minimum number of vehicles per area must increase by at least one, so we have that $\underline{m}(k + 2B(N_A - 1)^2) \geq \underline{m}(k) + 1$ for all $k \in \mathbb{N}$. The initial value for this function $\underline{m}(0)$ is at least zero and it cannot increase to more than $m = \lfloor \frac{L}{N_A} \rfloor$ because of our assumption that the total number of vehicles is fixed at L. Thus the system must converge to $\mathcal{X}_I$ in no more than $T = 2B(N_A - 1)^2 \lfloor \frac{L}{N_A} \rfloor$ time steps. □

References

1. Bertsekas, D.P., Tsitsiklis, J.N.: Parallel and Distributed Computation: Numerical Methods. Athena Scientific, Bellmont (1997)
2. Burgess, K.L., Passino, K.M.: Stability analysis of load balancing systems. Int. J. Control **61**(2), 357–393 (1995)
3. Burgess, K.L., Passino, K.M.: Stability Analysis of Discrete Event Systems. Wiley, New York (1998)
4. Cortés, A., Ripoll, A., Cedó, F., Senar, M.A., Luque, E.: An asynchronous and iterative load balancing algorithm for discrete load model. J. Parallel Distrib. Comput. **62**, 1729–1746 (2002)
5. Cortés, J., Martinez, S., Bullo, F.: Coordinated deployment of mobile sensing networks with limited-range interactions. In: 43rd IEEE Conference on Decision and Control, pp. 1944–1949, Paradise Island (2004)
6. Els, R., Monien, B.: Load balancing of unit size tokens and expansion properties of graphs. In: SPAA '03: Proceedings of the Fifteenth Annual ACM Symposium on Parallel Algorithms and Architectures, pp. 266–273. ACM, New York (2003)
7. Finke, J., Passino, K.M., Sparks, A.: Cooperative control via task load balancing for networked uninhabited autonomous vehicles. In: 42nd IEEE Conference on Decision and Control, pp. 31–36, Maui (2003)
8. Frazzoli, E., Bullo, F.: Decentralized algorithms for vehicle routing in a stochastic time-varying environment. In: 43rd IEEE Conference on Decision and Control, pp. 3357–3363, Paradise Island (2004)
9. Henrich, D.: The liquid model load balancing method. J. Parallel Algorithms Appl. **8**, 285–307 (1996) (Special Issue on Algorithms for Enhanced Mesh Architectures)
10. Liu, Y., Cruz, J.B., Sparks, A.G.: Coordinating networked uninhabited air vehicles for persistent area denial. In: 43rd IEEE Conference on Decision and Control, pp. 3351–3356, Paradise Island (2004)
11. Moore, B.J., Passino, K.M.: Decentralized redistribution for cooperative patrol. Int. J. Robust Nonlinear Control **18**, 165–195 (2008)