Task selection for radar resource management in dynamic environments

Jinwoo Seok, Pierre Kabamba, Anouck Girard

Department of Aerospace Engineering, University of Michigan, Ann Arbor, Michigan, USA
E-mail: sjinu@umich.edu

Published in *The Journal of Engineering*; Received on 7th June 2017; Accepted on 6th November 2017

Abstract: A task selection method for multi-faced static phased array radar resource management in dynamically changing environments using recomposable restricted finite state machines is presented. Restricted finite state machines allow the design of a composed finite state machine with resource limitations by restricting some of the inputs. Recomposable restricted finite state machines allow the state space of a finite state machine to change dynamically, which allows the modelling of a dynamically changing environment. Applying dynamic programming to restricted finite state machines yields optimal policies for a given cost function and applying breadth-first search or limited breadth-first search with fixed depth yields suboptimal solutions for the current state. The authors model a task selector for the radar in an overloaded battlefield situation using recomposable restricted finite state machines and obtain a radar resource allocation policy using dynamic programming when the environment changes dynamically and the resources are limited. The suboptimal solution for the current state is obtained using heuristic methods: breadth-first search, or limited breadth-first search in the task selector for large-scale problems. Furthermore, the authors consider distributed architectures for multi-radar systems with communication channels. The results show that their approach performs well from the standpoints of both computational time and performance.

1 Introduction

In this paper, we study multi-faced static phased array radar resource management in dynamic environments. By management, we mean task assignment that considers the finite resources. By dynamic environments, we mean that the environment may change unpredictably in time. A phased array radar is advantageous compared with the traditional rotating radar. For instance, the phased array radar is multifunctional, and does not have rotating parts; instead, it uses electronically steered radar beams, can perform track/search tasks simultaneously, and so on [1–3]. We develop a multi-faced static phased array radar task selector and test it for single radar and multi-radar scenarios. We focus on making decisions of how to allocate finite radar resources when the resources are not sufficient to perform all tasks, in a dynamically changing environment. Thus, we assume that the radar only can use a fixed amount of energy at each time step, and we do not consider detailed radar control and scheduling. Here, we mainly focus on allocation of radar energy to area search and threat tracking for discrimination subject to given constraints, and on implementation of our own original methodology for dynamic modelling and efficiency. By discrimination, we mean collecting information to identify the threat. We use finite state machines (FSMs) to design a task selector for the radar and apply dynamic programming (DP) to obtain the policy, or heuristics to obtain the policy/solution. By solution, we mean the sequence of decisions at the current state. By policy, we mean the solutions for all the states.

1.1 Motivation

Our motivating problem is as follows. Consider a geographic area in which two enemy forces operate. The red force consists of multiple threats (used to represent either missiles or aircraft). The blue force consists of ships that carry multi-function phased array radars. The ships are deployed throughout the area; they use their radars to detect threats, and can communicate among themselves. The goal of the red force is to destroy the blue force. The goal of the blue force is to detect and destroy the red force. Our goal is to provide algorithms to determine the set of the tasks to perform when the available energy is not enough to perform all the tasks at each discrete time step to detect, discriminate, and obtain as much information as possible on threats, which is then sent to an interceptor unit that processes (attempts to destroy) the threats. This is realistic in view of the standard separation of responsibilities on battleships. We refer to the tasks performed by our algorithms as radar resource management.

Our designs do not contain all the physics and details necessary for individual radar control. However, they were suggested by radar modelling experts at Office of Naval Research and appropriate for the study of distributed policies/solutions and communication strategies. A distributed system can achieve cooperation of radars using communications. Distribution of tasks across the fleet (to avoid duplication of effort) and communication of information between radars may be advantageous. For distributed fleet-level radar systems, each radar uses its own task selector rather than one global task selector. A distributed architecture is advantageous compared with a centralised one from the standpoints of computational time and flexibility of organisation.

In addition, the radar resource management must deal with a dynamic environment. The blue force does not know a priori how many threats the red force has at its disposal, or their nature. Every time a new threat is detected, the situation awareness must be updated, and the tasks for the radar are initiated based on the available resources.

1.2 Literature review

There have been many approaches to the problem of modelling and control of battlespaces. In [4], time-based state-space models are used to represent a stochastic system. Discrete system based modelling for adversarial situations can be found in [5] using FSMs and in [6] using hybrid automata.

A number of papers have studied allocation of multi-function radar resources. Butler [7] compares a rotating phased array radar system with a static phased array radar system and develops the control and scheduling strategies for the rotating phased array radar system. Watson and Blair [8] calculate a revisit time to track manoeuvering targets using the interacting multiple model. Blair *et al.* [9] propose a benchmark problem for highly manoeuvrable targets. The purpose is to obtain beam pointing control of a phased array radar against highly manoeuvrable targets in the presence of false alarms and electronic counter measures. Gosh *et al.* [10] propose a phased array radar model that does resource

allocation and scheduling using a quality of service (QoS)-based resource allocation model optimiser and improper nesting of the radar dwells. They allocate the radar resources to each task and also schedule the allocated resource to the radar to meet a jitter requirement. These works are focused on detailed radar control and the level of modelling is not appropriate for multi-radar systems.

Many of multi-function radar task scheduling works are based on prioritisation. Filippi and Pardini [11] propose a multi-function rotating phased array radar control scheduling architecture based on the priorities. Miranda *et al.* [12, 13] allocate the radar resources using task scheduling and compare the scheduling algorithms. The scheduling depends on the pre-assigned priorities of the tasks allocated to the radar. In [14], the authors develop an adaptive prioritisation of the tasks based on a fuzzy-reasoning-based algorithm in dynamically changing tactical environments and compare their design with other prioritisation methods. Jiménez *et al.* [15] model a radar task scheduler using three stages: task priorisation, a scheduling algorithm, and temporal planning, and compare different scheduling algorithms. Duron and Proth [16] maximise the number of useful tasks performed, taking into account their priorities. Moo [17] develops a method for the scheduling of prioritised tracking and surveillance tasks based on a two-slope benefit function where tracking and high priority surveillance tasks are scheduled first, then lower priority surveillance tasks are scheduled.

Many radar control schemes do not consider overloaded situations, which are important in our case. Many other radar control schemes focus on how to allocate the radar resources to maximise the radar usage while we focus on how to choose the tasks to perform when the radar cannot perform all the tasks because of resource limitations. Tumová *et al.* [18] consider instances when all of the given specifications cannot be reached simultaneously due to their incompatibility or environmental constraints, which is an overloaded situation. They find the least violating control in the environment with respect to the given set of mission specifications using a Büchi automaton and a nested depth-first search, which is computationally advantageous compared with exhaustive search. However, unlike [18], we do not have strict behaviour rules nor final constraints. In overloaded situations, some references allow performance degradation [13] or use pre-defined rules to choose the tasks to perform [17]. However, in our approach, rules are embedded in the cost function, so by minimising or maximising the objective function online, we obtain a policy/solution to choose the tasks to perform online in overloaded situations.

Moo and Ding [19] consider a multi-radar configuration, where the networked phased array radars are connected by a communication channel and share the tracking and detection data, and verify a benefit over the independent radars configuration. Severson and Paley [20] describe a distributed multi-radar system. They calculate the optimal position and search radius of the radars to maximise the unified searching area among the radars in a given environment. Also, they allocate the tracking tasks to each radar to balance each radar's utilisation. However, there are limitations for dynamic situations, because of assumptions such as the number of targets that the radar can track. These assumptions are not suitable for dynamic environments scenarios because in practice the number of threats is unpredictable.

In our previous work [21, 22], we developed a framework for phased array radar model control using FSM and we applied DP to the FSM. Especially, we introduced the dynamic FSM (DyFSM). In this paper, we reformalise the DyFSM as a recomposable FSM (ReFSM) and define the ReFSM. We introduce a restricted FSM (RFSM), which is a composed FSM with restricted transitions and directly takes resource limitations of the system into account. Then, a recomposable RFSM (ReRFSM) is defined and handles a dynamic environment in the presence of resource limitations. We also develop a more sophisticated radar task selector based on our prior work to obtain the policy/solution and test a distributed architecture with multiple radars.

1.3 Original contributions

The original contributions of this paper are as follows:

(i) *Radar task selector design using ReRFSM*: We develop a multi-faced static phased array radar task selector based on FSM. We define RFSMs that allow the consideration of limited (radar) resources. RFSMs can take into account limited resources by restricting some transitions during the composition. We then define ReRFSMs that allow the state space of an RFSM to change dynamically; they are suitable to model dynamically changing environments.

(ii) *Policy generation for radar resource management in dynamic environments using DP*: We generate policies for a phased array radar system to allocate the radar resources in dynamic environments by applying DP to ReRFSM. The resulting policy allows us to find the radar resource allocation. DP yields a policy for every state, so even though a state may jump to another state unexpectedly, we always have a policy for the resulting state. In other words, if the current decision is interrupted, the policy gives an alternative solution within the state space.

(iii) *Solution generation for radar resource management in dynamic environments using heuristic methods*: We generate solutions for a phased array radar or a multi-radar system to allocate radar resources in dynamic environments even if the radar system encounters an overloaded situation; breadth-first search (BFS) and limited BFS (LBFS) are considered.

(iv) *Distributed multi-radar systems and communication*: We develop a distributed architecture for fleet-level radars over communication networks that improves the overall system's performance compared with a decentralised system, without a significant trade-off such as computational time.

To model dynamically changing environments is important because many real-world situations are dynamically changing. However, modelling dynamic environments is not easy because it is not possible to perfectly predict dynamically changing environments. Thus, a method is needed to capture dynamically changing environments easily and precisely. Resource allocation is also important because in practice, resources are limited, so not all desired tasks are performed at the same time step.

2 Theoretical approach

2.1 Finite state machines

An FSM constructs an output signal one symbol at a time by observing an input signal one symbol at a time [23–25]. An FSM is a six-tuple

$$\text{FSM} = (X, U, Y, f, g, x_0), \tag{1}$$

where X, U, and Y are sets, f and g are functions, and $x_0 \in X$. X is the state space, U is the input alphabet, Y is the output alphabet, f: $X \times U \to X$ is the next state function, g: $X \times U \to Y$ is the output function, and $x_0 \in X$ is the initial state.

The interpretation of f and g is as follows: if $x(k) \in X$ is the current state at step k and $u(k) \in U$ is the current input signal, then the current output symbol $y(k)$ and the next state $x(k+1)$ are given by

$$x(k+1) = f(x(k), u(k)), \tag{2}$$

$$y(k) = g(x(k), u(k)), \tag{3}$$

and $x(0)$ is x_0.

To represent an FSM, we can use the sets and functions models as given above, or state transition diagrams.

2.2 Finite state machine composition and pruning

We consider the following composition operation on FSMs. Given a number w of FSMs, $\text{FSM}_1 = (X_1, U_1, Y_1, f_1, g_1, x_{10})$, $\text{FSM}_2 = (X_2, U_2, Y_2, f_2, g_2, x_{20})$, …, $\text{FSM}_w = (X_w, U_w, Y_w, f_w, g_w, x_{w0})$, we define the composition of FSM_1, FSM_2, …, FSM_w, denoted $\text{FSM}_1 \times \text{FSM}_2 \times \cdots \times \text{FSM}_w$, as the FSM

$$\begin{aligned}&\text{FSM}_1 \times \text{FSM}_2 \times \cdots \times \text{FSM}_w \\ &\quad = (X_1 \times X_2 \times \cdots \times X_w, U_1 \times U_2 \times \cdots \times U_w, \\ &\quad Y_1 \times Y_2 \times \cdots \times Y_w, (f_1, f_2, \ldots, f_w), \\ &\quad (g_1, g_2, \ldots, g_w), (x_{10}, x_{20}, \ldots, x_{w0})),\end{aligned} \tag{4}$$

where FSM_1, FSM_2, …, FSM_w are called the components of the composed FSM.

We also consider the following pruning operation on composed FSMs. If one of the component FSMs is not needed, we remove that FSM as follows. Assume that in (4), FSM_i is not needed. Then we define the pruning of FSM_i from the FSM, denoted FSM/FSM_i, as the FSM

$$\begin{aligned}&\text{FSM}/\text{FSM}_i \\ &= \text{FSM}_1 \times \cdots \times \text{FSM}_{i-1} \times \text{FSM}_{i+1} \times \cdots \times \text{FSM}_w.\end{aligned} \tag{5}$$

2.3 Restricted finite state machines

In the presence of a resource limitation, some of the transitions may be restricted during the composition in (4) due to the limitation. We call this composed FSM an RFSM. Thus, the RFSM is a composed FSM with the component FSMs defined as (4) but the input alphabet, U

$$\begin{aligned}&U \subset U_1 \times U_2 \times \cdots \times U_w, \\ &\text{where } U \neq U_1 \times U_2 \times \cdots \times U_w,\end{aligned} \tag{6}$$

and/or

$$\exists x \in X, u \in U : f(x(k), u(k)) = g(x(k), u(k)) = \emptyset, \tag{7}$$

where $\emptyset$ means empty, so the transition is not available. In other words, we disable transitions that correspond to insufficient or excessive resource or capability use. This is an example of supervisory control [24, 26]. The limitations are different for different situations/problems, so restriction rules for the transitions are problem dependent. In general, for the case of composition of w component FSMs with input restrictions, the RFSM is denoted as

$$\text{RFSM} = \text{FSM}_1 \bar{\times} \text{FSM}_2 \bar{\times} \cdots \bar{\times} \text{FSM}_w, \tag{8}$$

and the case of pruning of FSM_i from the RFSM is denoted as

$$\text{RFSM} = \text{RFSM} \bar{/} \text{FSM}_i. \tag{9}$$

2.4 Recomposable restricted finite state machines

An ReRFSM is an RFSM that has three modes of operation: normal, composition, and pruning.

(i) In the normal mode of operation, the ReRFSM operates as an RFSM.

(ii) The composition mode happens when a new component FSM arises. Assume that this happens at step k; let $\text{RFSM}(k)$ be the RFSM in which the ReRFSM operates at step k, and let FSM_{new} be the new component FSM that arises at step k. Then, we have

$$\text{RFSM}(k+1) = \text{RFSM}(k) \bar{\times} \text{FSM}_{\text{new}}. \tag{10}$$

(iii) The pruning mode happens when a component FSM is not needed. Assume that at step k, component FSM_i is not needed. Then, we have

$$\text{RFSM}(k+1) = \text{RFSM}(k) \bar{/} \text{FSM}_i. \tag{11}$$

3 Radar modelling

We consider a two-dimensional geographic area for the phased array radar model. The radar is physically at the centre of a circular disk divided into sectors of equal aperture (Fig. 1). Each sector may contain a number of threats. The radar is capable of searching sectors and focusing attention on specific threats, and does so using energy. Accordingly, the radar resource is energy, and the radar can use a maximum energy of E_{total} at each time step. Then, the resource constraint of the radar at each time step k is

$$\begin{aligned}&s_1(k) + s_2(k) + \cdots + s_m(k) + t_1(k) + t_2(k) \\ &\quad + \cdots + t_n(k) \leq E_{\text{total}}, \quad \text{for all } k,\end{aligned} \tag{12}$$

where s_i is the assigned energy to sector i, m is the total number of radar area sectors, t_j is the assigned energy to threat number j, and n is the total number of threats.

For simplicity, assume that the energy is computed by multiplying the power by a single unit time at each time step k, in other word, energy = power × 1 at each time step k. Then, the quantity of energy for s_i and t_j is the same as the quantity of power for s_i and t_j, respectively, at each time step. Then, the integers s_i and t_j have the following constraints at each time step k:

$$\begin{aligned}&s_i(k) = 0 \text{ or } Bs \leq P_{\text{trans}_{\max}}, \quad i = 1, 2, \ldots, m, \\ &0 \leq t_j(k) \leq P_{\text{trans}_{\max}}, \quad j = 1, 2, \ldots, n,\end{aligned} \tag{13}$$

where Bs is the transmitted power required for searching each radar area sector and $P_{\text{trans}_{\max}}$ is the maximum transmitted power that the radar can use for one task. To ensure zero leakage, the radar should search each sector at least once every maximum revisit time ($\text{REV}_{\max}$) to acquire the threat in a certain range. Assume that the scanning rotation rate is fixed. Note that the phased array radar electronically steers radar beams, so there are no mechanically rotating parts. Then, threat acquisition happens when a certain QoS is satisfied for s_i. The QoS for each threat j in sector i at each time step k is given by

$$\text{QoS}_j(s_i, r_j, \sigma_j, k) = \frac{Q_0 s_i(k) \sigma_j(k)}{r_j(k)^4}, \tag{14}$$

where r_j is the distance between the radar and threat j, σ_j is the radar cross-section of threat j, and Q_0 is a normalisation constant. Once

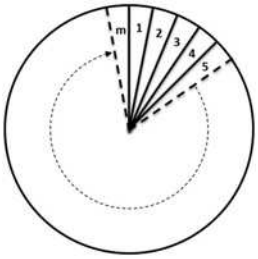

Fig. 1 *Example of radar area sector: a radar can search within a circular area with radius $r_{\min}$ and the area is divided into m radar area sectors*

the desired QoS ($\text{QoS}_{\text{desired}}$) is determined, which ensures the acquisition of the target in the radar area sector, Bs can be determined as follows. Assume that we know Q_0. If we have the minimum radar range ($r_{\min}$) and the minimum radar cross-section ($\sigma_{\min}$) of the threat that the radar would acquire at the minimum radar range, we can calculate Bs for $\text{QoS}_{\text{desired}}$ using the following equation:

$$Bs = \text{ceil}\left(\frac{QoS_{\text{desired}} r_{\min}^4}{Q_0 \sigma_{\min}}\right), \tag{15}$$

where the function 'ceil' rounds up to the next integer and $Bs \leq P_{\text{trans}_{\max}}$ as in (13).

Note that s_i should be binary, that is either 0 or Bs, to always satisfy the desired QoS. Then, $\text{QoS}_{\text{desired}}$ is always satisfied for any threat in a disk of radius $r_{\min}$ that has radar cross-section larger than $\sigma_{\min}$. This allows the radar to ensure the acquisition of the threat for $r_{\min}$ and $\sigma_{\min}$.

To discriminate the threat, we need to track the threat longer than the discrimination time (T_d) with return power (P_{return}) larger than the minimum required return power $\left(P_{\text{return}_{\min}}\right)$ at every time step during tracking. The return power for threat j at each time step k is

$$P_{\text{return}_j}(k) = \frac{t_j(k) K_G \sigma_j(k)}{r_j(k)^4}, \tag{16}$$

where K_G is the radar constant. Assume that we are given $P_{\text{return}_{\min}}$. Then, we can compute the required transmitted power to the threat to satisfy $P_{\text{return}_{\min}}$ as function of r

$$t_{\text{req}}(r) = \frac{P_{\text{return}_{\min}} r^4}{K_G \sigma}, \tag{17}$$

where $t_{\text{req}}(r) \leq P_{\text{trans}_{\max}}$ as in (13).

4 Problem formulation

The problem is formulated as follows. Assume that the phased array radar (or radars) is (are) located at the centre of a Manhattan grid along which all threats move. The radar has an amount of resources, E_{total} at each time step k as in (12) and the radar can use a maximum of $P_{\text{trans}_{\max}}$ for each threat or sector. The geographic area is divided into m radar area sectors as in Fig. 1. Each sector has a revisit deadline, $\text{REV}_{\max}$ (19), and desired QoS $\text{QoS}_{\text{desired}}$. The radar has constant K_G, normalisation constant Q_0, minimum distance to search $r_{\min}$, and minimum cross-section to search $\sigma_{\min}$. There are nt threats moving towards the radar with different constant speeds from different locations. The minimum required return power for tracking is $P_{\text{return}_{\min}}$ and the time required to discriminate threats is T_d. Given the above data, we want to find a policy/solution for the radar to allocate its resources by minimising the cumulative cost.

5 Task selector design

The assumptions are: (i) events are discrete, i.e. events happen slowly enough relative to the time constants of the radar that we can eliminate monitoring of continuous time variables. This is the basis for employing logic-level models (FSMs). (ii) We consider a two-dimensional geographic area. (iii) We assume that the radar is not allowed to have resources left at any time step if there is any task left. (iv) We assume perfect communications between the radars of the multi-radar system. By perfect communication, we mean no information loss, no delay, and no fail.

As previously mentioned, the phased array radar electronically steers the radar beam, so there are no mechanically rotating parts, which means the radar can perform track/search tasks simultaneously. Thus, we assume that the radar can perform the set of tasks chosen by the task selector simultaneously at each time step. Note that our focus is on overloaded situations where the radar cannot perform all the given tasks because of limited resources in dynamic environments, so the task selector has to choose which tasks to perform first when the tasks are changed without any prediction.

5.1 Sector finite state machine

We keep track of the revisit time for each sector i. The revisit time for each sector i at time step $k(\text{REV}_i(k))$ satisfies

$$\begin{aligned} &\text{REV}_i(k+1) = \text{REV}_i(k) + 1, \quad \text{if} \quad s_i(k) = 0, \\ &\text{REV}_i(k+1) = 0, \quad \text{if} \quad s_i(k) > 0. \end{aligned} \tag{18}$$

Then, the constraint to ensure zero leakage is

$$\text{REV}_i(k) \leq \text{REV}_{\max}, \quad \text{for all } k, \tag{19}$$

where $\text{REV}_{\max}$ is the maximum revisit time for all sectors that the radar should search before the sectors reach $\text{REV}_{\max}$. This constraint makes the radar search each sector at least once every $\text{REV}_{\max}$ time steps. We want to discriminate as many threats as possible with respect to the constraints described above.

We use the formalism of FSMs to model the radar area sectors. Fig. 2 shows the FSM for a radar area sector when the revisit deadline ($\text{REV}_{\max}$) is four.

5.2 Threat finite state machine

As we stated in Section 3, to discriminate the targets, we need to keep a record of tracking time for each threat. The consecutive tracking time for each threat j at time step k ($TT_j(k)$) always starts from zero ($TT_j(0) = 0$), and satisfies

$$\begin{aligned} &TT_j(k+1) = TT_j(k) + 1, \quad \text{if} \quad P_{\text{return}_j}(k) \geq P_{\text{return}_{\min}}, \\ &TT_j(k+1) = 0, \quad \text{if} \quad P_{\text{return}_j}(k) < P_{\text{return}_{\min}}. \end{aligned} \tag{20}$$

As we describe in Section 3, t_{req} guarantees $P_{\text{return}} \geq P_{\text{return}_{\min}}$. Fig. 3 shows an FSM for tracking time for threats when the discrimination time (T_d) is four.

5.3 Mission restricted finite state machine

By composing the sector FSM and the threat FSM, we obtain an RFSM for the mission. Assume that we have m sectors and n threats. Then, during the composition, the input alphabet of the mission RFSM, U, is defined as follows:

$$\begin{aligned} U = \{u \in U_1 \times U_2 \times \cdots \times U_n \times U_{n+1} \\ \times U_{n+2} \cdots \times U_{n+m} : \text{sum}_E(u) \leq E_{\text{total}}\}, \end{aligned} \tag{21}$$

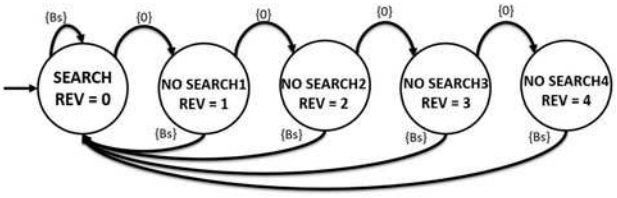

Fig. 2 *Sector FSM when* $\text{REV}_{\max} = 4$*:Bs means the sector is searched and 0 means the sector is not searched. Each sector only allows four time steps of not being searched. Wherever the state is, if the input is Bs, REV is reset to zero*

doi: 10.1049/joe.2017.0236

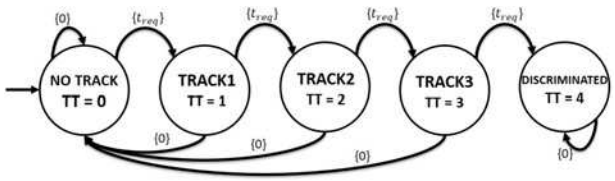

Fig. 3 *Threat FSM when $T_d = 4{:}t_{\text{req}}$ means the threat is tracked, 0 means the threat is not tracked. Each threat is discriminated after being tracked for four consecutive time steps. Wherever the state is, if the input is 0, TT is reset to zero except in the last state because the threat is already discriminated*

where $\text{sum}_E(u)$ means sum of the required power to perform the input u. The input alphabet is restricted by (12).

The representations of sector FSM and threat FSM are convenient because composition of these FSMs allows the representation of all states of the system.

5.4 Cooperation

For the distributed system with communication, consider the formation of three phased array radars shown in Fig. 4. The areas monitored by the individual radars overlap. We set a threshold such that if one radar sub-area overlaps with at least a certain percentage (say, 90%) of another radar sub-area, we assume that the two sub-areas are the same. Under this assumption, we can say that sub-areas 4A and 2B are the same, as are sub-areas 4B and 2C. These sub-areas are defined as redundant sub-areas.

We introduce a distributed architecture over a communication channel as follows. The radars share the revisit times of redundant sub-areas. For example, in Fig. 4, the first radar and second radar share the revisit time for sub-area 4A = 2B, and the second radar and third radar share the revisit time for sub-area 4B = 2C. Thus, if the first radar searches the sub-area 4A, the state of both mission RFSMs of the first and second radars are updated accordingly. In other words, sharing the revisit times of redundant sub-areas can update the states of the relevant radars. In addition, all the radars share information about detected and discriminated threats. Thus, the states of the mission RFSMs of the relevant radars are updated based on the information. As a consequence, the system does not track a threat if it has been discriminated by one of the radars. Furthermore, the tasks are assigned to the idlest radar first, so the radars that have fewer tasks than the neighbouring radars are able to help the neighbouring radars by searching the redundant sub-areas. This allows radars to save resources and improves the overall system performance.

6 Policy and solution approach

Given a ReRFSM, different methods can be applied to obtain the policy and the solution. Each component of the ReRFSM is assigned a transition cost. The transition cost is assigned by a heuristic using REV, TT, r, and v of the threat at the time when the FSMs are created. The transition cost of the ReRFSM is the sum of the transition costs of its components. When composition occurs, the transition cost of the new component is added to the transition cost of the ReRFSM. When pruning occurs, the transition cost of the pruned FSM is subtracted from the transition cost of the ReRFSM.

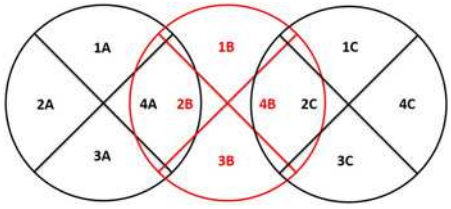

Fig. 4 *Multi-radar sectors example: each radar has the same $r_{\min}$ and the same $m = 4$. Sub-area 4A = 2B and sub-area 4B = 2C are defined as redundant sub-areas*

6.1 DP for ReRFSM

Note that the optimal policy of a composed FSM is obtained by the union of the optimal policies of its component FSMs. However, for RFSM, this does not hold due to the restricted inputs in RFSM. Thus, the optimal policy of RFSM has to be obtained by applying DP directly to the whole RFSM.

ReRFSMs allow the state space of an FSM to change dynamically as defined in Section 2.4. For example, in our scenario, when the radar detects new enemies, threat FSMs are created and composition occurs. If a threat is discriminated, the FSM of that threat reaches an absorbing state and pruning occurs when needed.

In normal mode, the ReRFSM operates as a composed FSM with associated transition cost. DP is applied to RFSM to provide a policy. We use that policy until composition occurs. Then the ReRFSM and cost are updated and we recompute the policy by applying DP.

6.2 Heuristics for ReRFSM: BFS and LBFS for ReRFSM

The motivation for this section is that DP suffers from the curse of dimensionality especially if $\text{REV}_{\max}$ and T_d are large. The first idea is to only use the inputs that use more than a certain threshold of power, P_{lb}, because in an overloaded situation, using as many resources as possible can take many tasks. Furthermore, if the environment is rapidly changing, for example, the number of detected threats is rapidly changing, the ReRFSM has to be recomposed frequently, so the previous policy is useless. In such situations, we may not need DP for ReRFSM for the policy; instead we may want to find the solution for the current state. We may lose robustness of the policy, but if the solution is obtained fast enough, we can quickly generate the solution for any state.

Thus, we use the BFS algorithm with fixed depth (*fd*) to get the solution for the current state. ReRFSM is recomposed at every time step to take account for environment changes.

However, if the state space and input space for the ReRFSM are very large, BFS with *fd* may not be feasible. Thus, we set the upper bound for the computational cardinality, CC_{ub}, for each node for BFS as follows:

$$\text{number of next states} \times \text{number of inputs} \leq CC_{\text{ub}}. \tag{22}$$

In other words, the computational cardinality at each node for BFS has to be less than the upper bound, CC_{ub}. We call this LBFS. Once we have the number of inputs, the number of next states is obtained based on (22), and the next state is decided based on the cost, highest first for maximising, lowest first for minimising. Consequently, LBFS with *fd* ensures a small amount of computation time.

7 Task selector architecture

In this section, we describe the task selector architecture used to perform radar resource management. The task selector consists of subsystems (a) to (g), as shown in Fig. 5. The descriptions of and relationships between the subsystems are as follows.

(a) Environment evaluator (EE): EE evaluates environment situations based on the inputs from the radar, such as number of detected threats, to decide whether the system needs to generate a new policy/solution or use the previous policy/solution. If a new policy/solution is needed, go to (b): otherwise, go to (d).

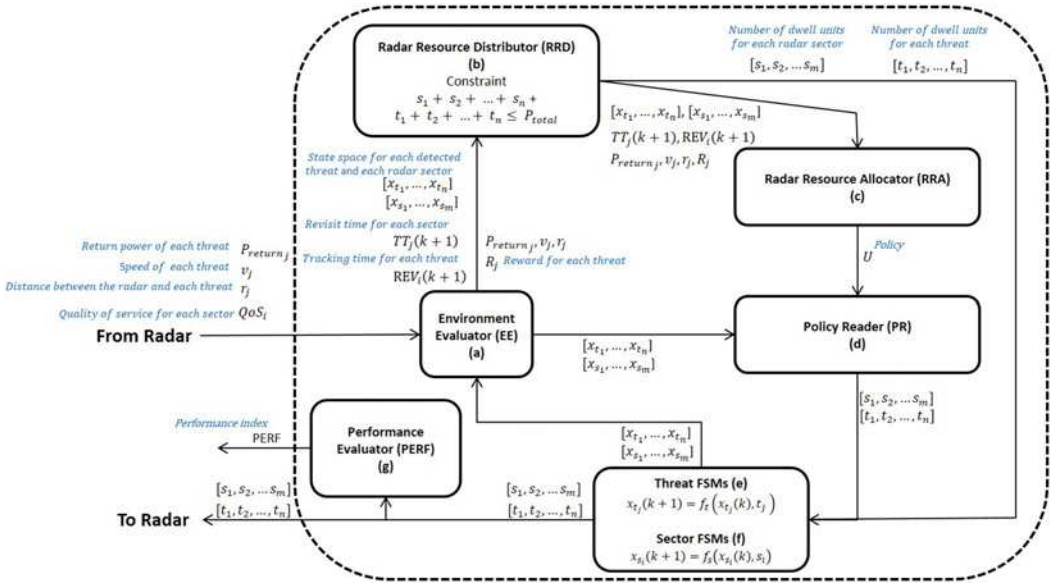

Fig. 5 *Task selector diagram: the inputs of the task selector from the radar are return power, speed of detected threats, distance between detected threats and the radar, and QoS. The outputs to the radar are the assigned energy for each threat and sector*

Table 1 Performance metrics

Metrics	Descriptions
$time_{max}$	maximum computation time: the longest time for DP, BFS, or LBFS to generate the policy/solution
FAIL	number of fail cases where the red force wins, that is, a threat reaches the radar before detection or discrimination
REV_{avg}	average revisit time of all sectors: a large value means that the radar does not search the sectors often and evenly, so a small value is desired
REV_{high}	highest revisit time of the sectors: a smaller value than REV_{max} is desired
TI_{avg}	average time to impact when the threat is discriminated: a small value means that the radar discriminates the threats late, so a large value is desired
TI_{low}	minimum time to impact when the threat is discriminated: a small value means that the radar discriminates the threats late, so a large value is desired

(b) Radar resource distributor (RRD): Assigns energy to all the sectors and all the threats. If there is not enough available energy to do this assignment, go to (c): otherwise, go to (e) and (f).
(c) Radar resource allocator (RRA): Use DP for ReRFSM to obtain a policy, or use BFS for ReRFSM or LBFS for ReRFSM to obtain solution.
Note that in the above sequence of actions, we generate the policy/solution only at (c) in the sequence.
(d) Policy reader (PR): PR interprets the policy/solution generated by RRA in terms of radar resources for each sector and threat, then gives the result to the radar.
(e) Threat FSMs: Threat FSM takes assigned energy for each threat as inputs and updates the states.
(f) Sector FSMs: Sector FSM takes assigned energy for each radar sector for searching as inputs and updates the state.
(g) Performance evaluator (PERF): PERF generates an index to evaluate radar performance.

8 Simulations and results

For the simulation, we consider a four-faced phased array radar with four sectors, so $m = 4$. We simulate two scenarios: a one radar case and a three-radar case. For each simulation, we use performance metrics to compare each method as shown in Table 1.

8.1 One radar case: small scale

We first run small-scale simulations that the radar handles easily to compare the solution approaches (DP, BFS, and LBFS) and their simulation times. A snapshot of a one radar scenario is shown in Fig. 6. For the simulations, we set the variables as shown in Table 2.

We run 30 simulations using the same variables but with different initial positions and speeds for each threat. The average performance metrics for the 30 simulations are shown in Table 3.

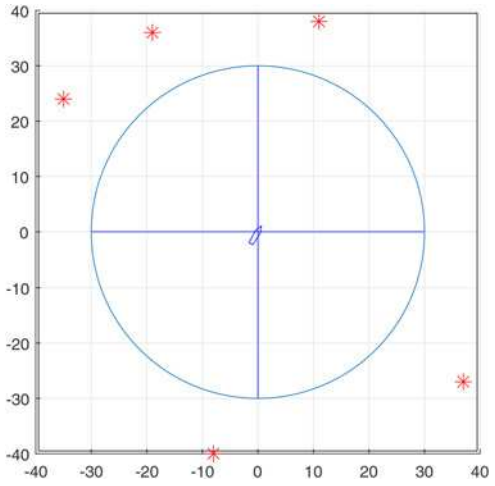

Fig. 6 *Snapshot of one radar scenario. The ship and radar sectors are indicated in blue. Threats are indicated by red stars*

doi: 10.1049/joe.2017.0236

Table 2 One radar scenario simulation variables

P_{total}	200	Q_0	15,000
$P_{trans_{max}}$	64	r_{min}	30
m	4	σ_{min}	2
REV_{max}	2	nt	5
$QoS_{desired}$	2	T_d	2
K_G	20,000	P_{lb}	140
fd	4	CC_{ub}	35,000
$P_{return_{min}}$	5	—	—

Table 3 Average performance metrics for small-scale simulations for one radar

	DP	BFS	LBFS
$time_{max}$	7.74	0.60	0.44
FAIL	0	0	0
REV_{avg}	0.36	0.33	0.33
REV_{high}	2	2	2
TI_{avg}	12.39	12.53	12.53
TI_{low}	5.22	5.32	5.31

Table 4 One radar scenario simulation variables

P_{total}	200	Q_0	15,000
$P_{trans_{max}}$	64	r_{min}	30
m	4	σ_{min}	2
REV_{max}	3	nt	30
$QoS_{desired}$	2	T_d	4
K_G	20,000	P_{lb}	140
fd	4	CC_{ub}	35,000
$P_{return_{min}}$	5	—	—

In terms of maximum computation time, the LBFS is the best, by a factor of 18 times compared with DP. No method encounters a fail case. The other performance metrics are not significantly different between each method. BFS and LBFS yield slightly better performance metrics (smaller REV_{avg} and larger TI_{avg} and TI_{low}) than DP. It is because the BFS and LBFS update the ReRFSM at every time step, which allows them to use the latest information on the threats, while DP only updates ReRFSM when a new threat FSM is added because the policy can be used before a new threat FSM arises. As we mentioned earlier, if the environment changes are frequent, we may not need the full policy; instead, we need the solution for the current state, as the simulation results suggest.

Therefore, we can conclude that the LBFS performs as well as DP on the metrics that were evaluated, and significantly reduces computation time. In addition, LBFS yields better decisions as DP, so we can save much computation time by using LBFS.

8.2 *One radar case: large scale*

We then simulate larger scale problems that the radar does not easily handle by increasing nt, REV_{max}, and T_d. For the large-scale simulations, we set the variables as shown in Table 4.

Then, our method is compared to the basic rule-based greedy method, nearest first (NF), where NF searches the sector only if the sector reaches the revisit deadline, and tracks the nearest threats first. We run 30 simulations using the same variables but where initial positions and speeds for each threat are different. The comparisons of average performance metrics of the 30 simulation results are shown in Fig. 7 and Table 5. In Fig. 7, the available number of tracking task indicates the number of threats in the minimum radar range (r_{min}), so the radar is able to track the threats, but the radar may not be aware of the threats because the radar may not search the sector yet. The perceived number of tracking tasks indicates the number of threats that the radar is aware of by searching the sectors, so the radar can perform the tracking tasks for the threats. Thus, perceived number of tracking tasks is always equal or less than the available number of tracking tasks. Finally, the performed number of tracking tasks indicates the tracking tasks that the radar is performing. Thus, the performed number of tracking tasks is always equal or less than the perceived number of tracking tasks because of the resource limitations.

As the number of threats is large for our configuration (30), there are some fail cases. Consequently, TI_{low} is small, which means that the radar discriminates some threats at very close range to the radar. However, TI_{avg} for LBFS is 9.61, which means that the radar still discriminates most threats far enough from the radar if we compare this to the small-scale problem (where the range was 12.39–12.53). As shown in Fig. 7, the profiles of number of total perceived tracking tasks and number of total performed tracking tasks of LBFS are more uniform than those of NF, which indicates more stable operations. Furthermore, as the number of total available tracking tasks is increased, the number of total perceived tracking tasks is increased for both LBFS and NF, but NF has generally lower number of total perceived tracking tasks than of LBFS, which indicates the LBFS has better awareness of threats in the radar range. Thus, both TI_{low} and TI_{avg} for LBFS are better than NF, and LBFS generates fewer fail cases.

For the revisit time, LBFS has better REV_{avg} than NF, and does not violate revisit deadlines where NF violates the revisit deadline as indicated by REV_{high}. As shown in Fig. 7, the revisit time is increased as the number of tracking tasks is increased, and NF has larger and more unstable revisit time profile than LBFS. Consequently, LBFS has better and more stable threat detection and tracking.

Thus, LBFS can handle more threats than NF and ensure revisit deadlines for the sectors, so LBFS generates fewer fail cases, that is, LBFS outperforms NF. The results indicate that the radar can handle many threats using LBFS but may fail on some threats because of limited resources.

8.3 *Three radar case: large scale*

For a multi-radar system, we compare a decentralised system (without communication) and a distributed architecture (with communication) as described in Section 5.4. Both systems are built based on LBFS. A snapshot of a three radar scenario is shown in Fig. 8. We simulate larger scale problems that the radar does not easily handle by increasing nt, REV_{max} and T_d. For the three radar scenario, we set the variables as shown in Table 6.

We run 30 simulations using the same variables, but with different initial positions and speeds for each threat. The average performance metrics for the 30 simulation runs are shown in Table 7.

We observe that in the distributed architecture, radars mostly have better performance metrics than in the decentralised system. The TI_{avg} and TI_{low} values of the first and third radars for the distributed architecture are slightly worse than those of the decentralised system, but TI_{avg} and TI_{low} values of the second radar for the distributed architecture are improved over those of the decentralised system. It is because the radars in the distributed architecture share information about detected and discriminated threats, so the first and the third radars help the second radar based on the shared information: the second radar faces the most overloaded situation as indicated by lowest TI_{avg} and TI_{low} values, and highest REV_{avg} and REV_{high} values in the decentralised system, so the second radar needs to be assisted. Also, the radars in the distributed architecture have smaller REV_{avg} and REV_{high}. This means that, in the distributed architecture, the radars help each other by searching redundant sub-areas. The distributed system is very useful because only simple communications (sharing only revisit time and

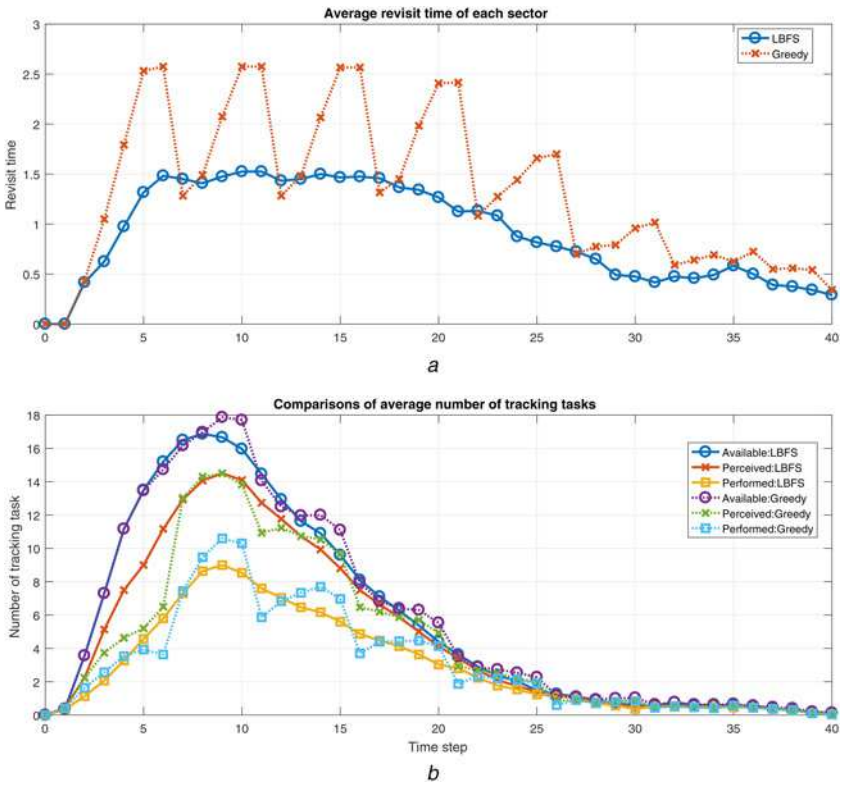

Fig. 7 *Comparison of NF and LBFS*

Table 5 Average performance metrics for large-scale simulations for one radar

	NF	LBFS
FAIL	0.1	0.03
REV_{avg}	1.34	0.92
REV_{high}	4	3
TI_{avg}	9.43	9.61
TI_{low}	0.71	1.17

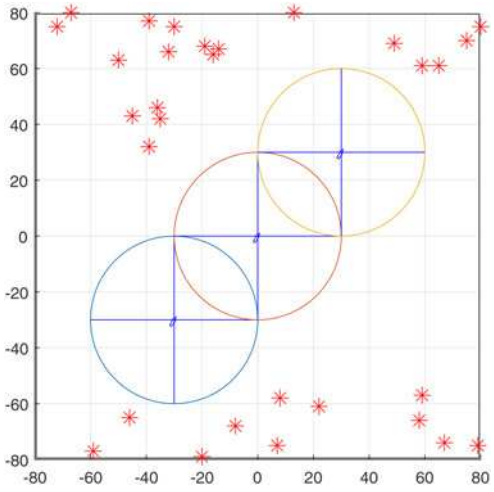

Fig. 8 *Snapshot of three radars scenario. Ships and radars are located at the centre of each circle, and the threats are indicated by red stars*

information on detected/discriminated threats) improve the overall performance a great deal. A centralised system should have better results than the distributed architecture. However, it is nearly

Table 6 Three radar scenario simulation variables

P_{total}	200	Q_0	15,000
$P_{trans_{max}}$	64	r_{min}	30
m	4	σ_{min}	2
REV_{max}	3	nt	30
$QoS_{desired}$	2	T_d	3
K_G	20,000	P_{lb}	140
fd	4	CC_{ub}	35,000
$P_{return_{min}}$	5	—	—

Table 7 Average performance metrics for three radar case

	Decentralised LBFS	Distributed LBFS
FAIL	0.23/0/0.27	0/0/0
REV_{avg}	0.38/0.69/0.36	0.25/0.15/0.24
REV_{high}	2.5/3/2.37	2.23/2.57/2.10
TI_{avg}	13.30/9.17/12.02	13.24/10.98/11.97
TI_{low}	6.03/2.59/5.46	5.87/3.47/5.63

impossible to get a solution because of the curse of dimensionality of the resulting composed FSM, even in small-scale problems. Thus, in practice, the distributed architecture is a good method to implement radar resource management with proper choice of communications.

9 Conclusions and future work

In this paper, we design a task selector for a multi-faced static multi-function phased array radar using ReRFSMs for radar resource allocation in dynamically changing environments. We define ReRFSMs that allow the state space of an FSM to change dynamically. RFSM allows one to design a composed FSM in the presence

doi: 10.1049/joe.2017.0236

of resource limitations by restricting some of the inputs. We develop a phased array radar resource allocation algorithm using DP for ReRFSM that handles situations where the set of tasks to be performed changes dynamically and the ability to perform tasks is resource-constrained. DP yields a policy for every state, so we always have alternative decisions within the state space although the states may change unexpectedly. However, DP is not feasible for large-scale problems and we may not need the full policy when environment changes are frequent. Thus, we design heuristic methods: BFS for ReRFSM, and LBFS for ReRFSM, and implement them in the task selector for large numbers of threats and large numbers for REV_{max} and T_d for highly dynamic environments. Our results show that LBFS generates better radar performance compared with the other methods, so our approach using LBFS is effective. We also develop a distributed architecture using communication for fleet-level radar systems. The distributed architecture performs better than the decentralised one, as shown by the facts that the distributed architecture has better overall performance metrics than the decentralised one, and that the distributed architecture can handle more threats than the decentralised one in the same battle situation.

10 Acknowledgment

This research was partially funded by the Office of Naval Research, U.S. Naval Surface Warfare Center, under grant number BAA N00178-12-C-2003.

11 References

[1] Hommel H., Feldle H: 'Current status of airborne active phased array (AESA) radar systems and future trends'. IEEE MTT-S Int. Microwave Symp. Digest, June 2005, pp. 1449–1452

[2] Agrawal A.K., Kopp B.A., Luesse M.H., *ET AL.*: 'Active phased array antenna development for modern shipboard radar systems', *Johns Hopkins APL Tech. Dig.*, 2001, **22**, (4), pp. 600–613

[3] Parker D., Zimmermann D.C.: 'Phased arrays - part 1: theory and architectures', *IEEE Trans. Microw. Theory Tech.*, 2002, **50**, (3), pp. 678–687

[4] Cruz J.B., Simaan J.M.A., Gacic A., *ET AL.*: 'Modeling and control of military operations against adversarial control'. Proc. IEEE Conf. on Decision and Control, December 2000, pp. 2581–2586

[5] Hill R.R., Mclntyre G.A., Miller J.O.: 'Application of discrete event simulation to modeling military problems'. Proc. Winter Simulation Conf., February 2001, pp. 780–788

[6] Faied M., Girard A.: 'Modeling and optimization of military air operations'. Proc. IEEE Conf. Decision and Control, December 2009, pp. 6274–6279

[7] Butler J.M.: 'Tracking and control in multi-function radar'. PhD thesis, University College London, 1998

[8] Watson G.A., Blair W.D.: 'Revisit calculation and waveform control for a multifunction radar'. Proc. IEEE Conf. on Decision and Control, December 1993, pp. 456–461

[9] Blair W., Watson G.A., Kirubarajan T., *ET AL.*: 'Benchmark for radar allocation and tracking in ecm', *IEEE Trans. Aerosp. Electron. Syst..*, 1998, **34**, (4), pp. 1097–1114

[10] Ghosh S., Hansen J., Rajkumar R., *ET AL.*: 'Integrated resource management and scheduling with multi-resource constraints'. Proc. IEEE Int. Real-Time Systems Symp., December 2004, pp. 12–22

[11] Filippi R., Pardini S.: 'An example of resources management in a multifunctional rotating phased array radar'. Proc. IEE Colloquium on Real-Time Management of Adaptive Radar Systems, June 1990, pp. 1–3

[12] Miranda S.L.C., Baker C.J., Woodbridge K., *ET AL.*: 'Phased array radar resource management: a comparison of scheduling algorithms'. Proc. IEEE Radar Conf., 2004, pp. 79–84

[13] Miranda S.L.C., Baker C.J., Woodbridge K., *ET AL.*: 'Comparison of scheduling algorithms for multifunction radar', *IET Radar Sonar Navig..*, 2007, **1**, (6), pp. 414–424

[14] Miranda S.L.C., Baker C.J., Woodbridge K., *ET AL.*: 'Fuzzy logic approach for prioritisation of radar tasks and sectors of surveillance in multifunction radar', *IET Radar Sonar Navig..*, 2007, **1**, (2), pp. 131–141

[15] Jiménez M.I., Del Val L., Villacorta J.J., *ET AL.*: 'Design of task scheduling process for a multifunction radar', *IET Proc., Radar Sonar Navig.*, 2012, **6**, (5), pp. 341–347

[16] Duron C., Proth J.: 'Multifunction radar: task scheduling', *J. Math. Model. Algorithms*, 2002, **1**, (2), pp. 105–116

[17] Moo P.W.: 'Scheduling for multifunction radar via two-slope benefit functions', *IET Radar Sonar Navig..*, 2011, **5**, (8), pp. 884–894

[18] Tumová J., Reyes-Castro L.I., Karaman S., *ET AL.*: 'Minimum-violating planning with conflicting specifications'. Proc. American Control Conf., June 2013, pp. 200–205

[19] Moo P.W., Ding Z.: 'Coordinated radar resource management for networked phased array radars', *IET Radar Sonar Navig..*, 2015, **9**, (8), pp. 1009–1020

[20] Severson T.A., Paley D.A.: 'Distributed optimization for radar mission coordination'. Proc. American Control Conf., June 2012, pp. 5102–5107

[21] Seok J., Zhao J., Selvakumar J., *ET AL.*: 'Radar resource management: dynamic programming and dynamic finite state machines'. Proc. European Control Conf., July 2013, pp. 4100–4105

[22] Zhao J., Seok J., Selvakumar J., *ET AL.*: 'A greedy policy for fleet-level radar resource management'. Proc. IEEE Conf. Decision and Control, December 2013, pp. 3160–3165

[23] Epp S.S.: 'Discrete mathematics with applications' (Cengage Learning, Boston, MA, USA, 2010, 4th edn.)

[24] Cassandras C.G., Lafortune S.: 'Introduction to discrete event systems' (Springer, 2007, New York, NY, USA, 2nd edn.)

[25] Lynch N.A., Tuttled M.R.: 'An introduction to input/output automata', *CWI Q.*, 1989, **2**, (3), pp. 219–246

[26] Ramadge P.J., Wonham W.H.: 'Supervisory control of a class of discrete event processes', *SIAM J. Control Optim.*, 1987, **25**, (1), pp. 206–230