Reasoning and Resource Allocation for Sensor-Mission Assignment in a Coalition Context

A. Preece*, D. Pizzocaro*, K. Borowiecki*, G. de Mel[†], M. Gomez[†], W. Vasconcelos[†],
A. Bar-Noy[‡], M. P. Johnson[‡], T. La Porta[§], H. Rowaihy[§], G. Pearson[¶], T. Pham[||]
*Cardiff University, UK; contact email: a.d.preece@cs.cardiff.ac.uk; [†]University of Aberdeen, UK
[‡]City University of New York, USA; [§]Pennsylvania State University, USA
[¶]DTSL, Great Malvern, UK; [||]US Army Research Laboratory, Adelphi MD, USA

***Abstract*—We consider the problem of sensor-mission assignment as that of allocating a collection of intelligence, surveillance and reconnaisance (ISR) assets (including sensors and sensor platforms) to a set of mission tasks in an attempt to satisfy the ISR requirements of those tasks. This problem is exacerbated in a coalition context because the full range of possible ISR solutions is not easy to obtain at-a-glance. Moreover, the operational environment is highly dynamic, with frequent changes in ISR requirements and availability of assets. In this paper we describe a solution for the sensor-mission assignment problem that aims to maximize agility in sensor-mission assignment, while preserving robustness. The search space of potential solutions is reduced by employing a semantic reasoner to work out the types of sensor and platform bundles suitable for a given set of ISR tasks. Then, an efficient resource allocation algorithm is used to assign bundles of sensor/platform instances to satisfy each task, within the search space determined by the reasoner. The availability of instances takes into account access rights on those instances across the coalition's inventory. We describe a proof-of-concept implementation of this approach, in the form of a decision support tool for ISR planning. We illustrate the approach in the context of a coalition peace support operation scenario.**

I. Sensor-Mission Assignment

Dynamic, mission-focussed intelligence, surveillance and reconnaisance (ISR) requires agile management of information-provisioning capabilities. This includes rapid assembly of sensing systems, highly efficient resource management, and an ability to configure and reconfigure, task and retask, ISR systems in a robust way [1]. We consider the problem of *sensor-mission assignment* as that of allocating a collection of ISR assets (including sensors and sensor platforms) to a set of *tasks* comprising a mission, in an attempt to satisfy the ISR requirements of those tasks. We assume that tasks originate from ad hoc communities of interest (CoIs) within the coalition [1], and that coalition members share ISR assets to some extent. The sensor-mission assignment problem is hard for several reasons. First, the information requirements of the full set of tasks typically exceeds what the inventory of assets can provide, necessitating complex resource allocation choices [2]. Second, the full inventory of ISR assets potentially available is not easy to obtain at-a-glance, and so CoIs engaged in ISR planning will not have ready access to the full range of options [3]. Third, the operational environment is highly dynamic: ISR requirements change in response to the emerging situation, and the availability of assets needs constant updating due to factors such as technical failures, changing weather and provision of new assets.

In this context, the goal of our work is: *to maximize agility in sensor-mission assignment, while preserving robustness.* In terms of maximising agility, we aim to provide as much automation as possible in the assignment of sensing assets to tasks. This involves attempting to capture the information requirements of CoIs in a manner that is as independent as possible of the capabilities of specific types of sensor or platform, to allow multiple degrees of freedom in allocation and reallocation of assets. Note that we deal with heterogeneous task and sensor types, and there is a many-to-many relationship between these: the same kind of task can be accomplished in several different ways; the same type of asset can serve many different kinds of tasks. This enables flexibility at mission planning time, but also at run-time, if it is possible to (re)configure and (re)task packages of assets in synch with mission tempo. Emerging service-oriented sensor architectures (e.g. [4]) are a key enabler here, supporting late-binding of sensors to tasks at mission run-time. We preserve robustness in a variety of ways, by ensuring that the association of task to asset types is sound, transparent, and explainable, and that the allocation mechanisms are fair, efficient, and effective. Rights of access to assets are incorporated; for example, the full capabilities of ISR assets may not be revealed to all parties in the coalition, or tasking rights may be limited. Moreover, various other kinds of policy can be incorporated into the approach, in line with the respective rules of engagement for coalition operations.

Figure 1 shows our approach to the sensor-mission assignment problem, as described in this paper. We assume that the information requirements of mission tasks can be captured in a machine-processable way and that these can be "fitted" to available types of assets, to yield a set of fit-for-purpose types of ISR solutions. We envisage this normally happening at mission-planning time. Then, individual instances of the assets are allocated to the tasks: this can happen at planning time, but for maximal agility should be done at run-time. Most importantly, the approach supports run-time *re*allocation of assets, to cope with the emerging situation, guided by the range of feasible alternative solutions determined at planning time. This process requires a highly reconfigurable sensor network environment, which also delivers the collected information, and feeds back monitoring data on the status of the assets. Changes in the task and asset sets may result from this received information and data: new information requirements may be generated, or monitoring may reveal that a particular sensor has become defunct.

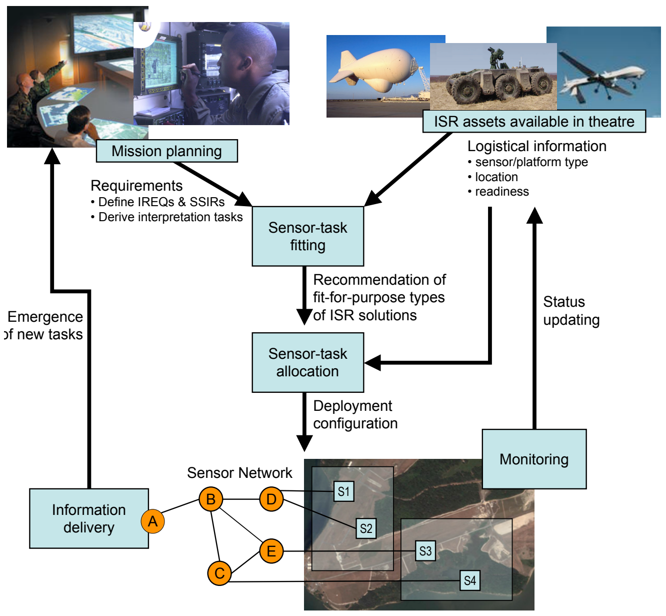

Fig. 1. An approach to sensor-mission assignment

The remainder of this paper is organized as follows: Section II introduces our formulation of the sensor-mission assignment problem in terms of three elements: (1) tasks, (2) bundles of ISR assets of particular kinds, and (3) the individual assets (sensors and platforms). The following three sections detail each element of this model: Section III describes our formalisation of information requirements and sensing tasks; Section IV covers the generation of bundle types to meet task requirements; then Section V shows how assets are allocated to bundles. Section VI describes the status of our proof-of-concept implementation, for a small-scale UK/US coalition. Finally, Section VII provides a concluding discussion.

This paper builds on our earlier work: in [5] and [6] we introduced the reasoning approach to sensor-mission fitting; Section IV now develops this further by showing how this mechanism yields bundle types. Asset allocation using task-related utility models was first presented in [7], but without showing explicitly how this related to the sensor-mission fitting mechanism. Sections IV and V now make this linkage explicit. Finally, [8] introduced the specification of tasks in terms of information requirements, but without fully detailing the subsequent fitting and allocation steps. Thus, this paper gives the first full account of the whole sensor-mission assignment process, from tasks, to bundle types, to allocation of assets.

II. Problem Formulation: Tasks-Bundles-Assets

We formulate the sensor-mission assignment problem as a graph; an example is shown in Figure 2. *Tasks* are representations of the information requirements needed by some CoI within the coalition, and *assets* represent the individual sensor and platform resources. Satisfying a task may involve the allocation of multiple assets, so we introduce the notion of *bundles*. Each bundle is composed of several assets and, depending on its type, may be suitable for several tasks. Assets may be suitable for assignment to several bundles, again depending on the bundle type. A solution to the sensor-mission assignment problem is an assignment of bundles to tasks, subject to the constraints: each task can have at most one bundle assigned to it, a bundle may be assigned to at most one task, each asset may be assigned to at most one bundle. Note that the thin arcs in Figure 2 show possible assignments, while the bold arcs show one actual assignment.

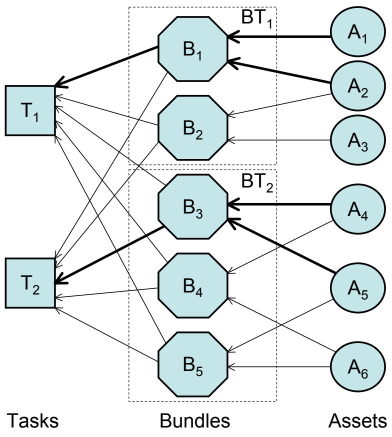

Fig. 2. Example sensor-mission assignment problem as a graph

A concrete example will illustrate this. In a peace support operation [1], UK and US bases have been established to detect/deter insurgent activity on a border. The bases rely on a main supply route (MSR) which must be surveilled and protected. Surveilling the border will likely involve, among other things, detection of suspicious vehicle activity near it: vehicle detection can be formalized as an information requirement task T_1. This may be accomplished by a variety of means, depending on the kinds of assets available. We assume these include flying a UAV over the border to gather IMINT, or using acoustic sensing to identify types of vehicles likely to be used by insurgents. Further, we assume that a single UAV can cover the area of interest (AoI), but that at least two acoustic arrays will be needed. Each of these options is represented as a type of bundle: $BT_1 = \langle\{\textsf{UAV}, \textsf{IMINT-sensor}\}\rangle, BT_2 = \langle\{\textsf{AcousticArray}\}, \{\textsf{AcousticArray}\}\rangle$. Assume there are multiple UAV assets available (A_1 and A_3) but only one functioning IMINT payload (A_2) that can be mounted on either A_1 or A_3. Assuming there are three acoustic arrays deployed in the area (A_4, A_5, A_6), then we can meet the requirements of T_1 by assigning UAV-IMINT bundles $\{A_1, A_2\}$ or $\{A_2, A_3\}$ or pairs of acoustic arrays (e.g. $\{A_4, A_5\}$, $\{A_4, A_6\}$, $\{A_5, A_6\}$).

However, it is likely that there will be other tasks, potentially in competition for these assets; for example, a requirement to detect vehicles posing a potential threat to the MSR (T_2) may be accomplished by the same types of bundle as T_1 but, because the areas of interest (MSR and border) do not intersect, it will not be possible to share assets between these tasks. So, for example, if we assign the UAV to T_2 then we will have to satisfy T_1 by means of acoustic intelligence (ACINT).

III. Specifiying Tasks

As we introduced in [8], high-level information requirements (IREQs) are defined informally, using natural language. For example: "Is there suspicious activity on the MSR road?" Each high-level IREQ must be broken down into a set of scenario-specific information requirements (SSIRs), again described informally using natural language, for example:

- "Are there suspicious vehicles on the road?"
- "Is there suspicious pedestrian activity along the roadside?"
- "Are there suspicious objects located near the road?"

Defining SSIRs is a key step in providing highly mission-focussed information and hence handling information overload. In [8] we consider the relationship between this step and the filtering and dissemination of information; for space reasons we do not consider this issue further here. The SSIRs need to be broken down further before they can be matched to types of ISR asset, in order to identify the *interpretation tasks* within each: what kinds of things does the CoI need to detect, identify, distinguish, etc. In our example, the SSIRs require detection of physical things (vehicles, people, objects) and also some characterisation of intent ("suspicious"). The results of this breakdown resembles a set of database queries:

- "detect vehicles where vehicle type or behaviour is suspicious"
- "detect people where person type or behaviour is suspicious"
- "detect object where object type is suspicious"

Once we have identified the interpretation tasks within each SSIR, we need to know the kinds of data that are interpretable to answer these: for example, visible imaging, radar, acoustic, etc. An established way to do this in Collection Co-ordination and Intelligence Requirements Management (CCIRM) is to use the National Imagery Interpretability Rating Scale (NIIRS) for various kinds of imagery intelligence[9]. For example, detection of vehicles of particular types is achievable by Visible NIIRS 4 and Radar NIIRS 6. As part of our sensor-mission fitting approach, we have built a proof-of-concept knowledge base (KB) for part of NIIRS, allowing a user to select interpretation tasks in terms of three task types (detect, identify, distinguish) and a range of "detectables" (e.g. ground, air and maritime vehicles, buildings). The KB infers which NIIRS ratings are appropriate for each interpretation task. (In practice, the need to detect "suspicious" activity at a border could involve establishing "normal" activity by gathering lower-resolution data over a period of time, then cueing higher-resolution assets to identify and track potential targets. We return to the issue of supporting sensor cueing in Section VII.)

Note that we have now moved from a set of "soft" (human-interpretable) information requirements to a set of "hard" (machine-processable) requirements, to enable the subsequent reasoning, allocation, and deployment processes. It is also necessary to identify "non-functional" requirements at this stage, which will have bearing upon the choice of assets, for example those concerned with the expected weather conditions (e.g. fog penetration), the operational environment [10] (e.g. foliage penetration), and any policies on asset deployment such as airspace control (which could rule-out use of UAVs, for example) [1]. Note that we envisage that this process be carried out on a per-CoI basis, rather than being centralized for the coalition as a whole. The CoI that "owns" each task will determine the resources that can be assigned to meet the requirements of the task, as we show later in Section VI.

IV. Generating Bundle Types

Our approach to sensor-mission fitting is founded on the use of *ontologies* to represent the capabilities required by tasks and provided by assets, and reasoning to determine logically-sound matches [6]. Ontologies define formally the semantics of a set of terms, allowing automatic reasoning to be performed using the terms, in a manner consistent with their real-world interpretation [11]. There is already a sizeable amount of work done in providing descriptive schemas and ontologies for sensors, sensor platforms, and their properties (e.g. [4], [12], [13]). There are also several well-known structured descriptions of tasks in the military missions context, e.g. the US Universal Joint Task List (UJTL)[14] and the UK JETL/METL task lists. Moreover, ontologies allow the results of the fitting process to be explainable in meaningful terms [15].

An ontology is typically structured as a set of definitions of *classes* (denoting "concepts") and *properties* (denoting "relationships"). The *subclass* relation denotes that one class is a specialisation of another. For example, the class UAV is a subclass of Aircraft, which in turn is a subclass of Vehicle. Ontologies are expressed using a meta-ontology language, the most commonly-used of which is now the Web Ontology Language (OWL) [15]. OWL provides ontology developers with the means to define new concepts and properties based on existing ones; OWL-supporting tools allow ontologies to be checked for consistency, and a number of reasoning engines exist to process OWL ontologies to draw inferences and answer queries. OWL is a suite of interrelated languages, with various levels of expressive power; in our work we use OWL DL (description logic), which offers a trade-off between expressivity and tractable reasoning [16].

The following is an example fragment of an ontology for sensor platforms:

$\mathsf{Aircraft} \equiv (\mathsf{Platform} \sqcap \exists \mathsf{hasRealm.Atmosphere})$
$\mathsf{UnmannedVehicle} \equiv (\mathsf{Platform} \sqcap \exists \mathsf{hasQuality.WithoutCrew})$
$\mathsf{UAV} \equiv (\mathsf{Aircraft} \sqcap \mathsf{UnmannedVehicle})$
$\mathsf{CombatUAV} \equiv (\mathsf{UAV} \sqcap \exists \mathsf{providesCapability.Firepower})$
$\mathsf{MALE} \equiv (\mathsf{UAV} \sqcap \exists \mathsf{providesCapability.MediumAltitude} \sqcap \exists \mathsf{providesCapability.LongEndurance})$
$\mathsf{Predator} \sqsubseteq (\mathsf{MALE} \sqcap \mathsf{CombatUAV})$

The example uses DL notation which can be read informally as follows: $\equiv$ "is equivalent to", $\sqcap$ "and", $\sqcup$ "or", $\neg$ "not", $\sqsubseteq$ "subset of". The semantics of description logics is defined by interpreting classes as sets of individuals and properties as sets of pairs of individuals; $\exists p.C$ denotes an (unnamed) class of individuals that are related by property p to at least one individual of class C. So, for example, the class CombatUAV is defined as being a UAV that is related by the providesCapability property to a Firepower capability.

The ontology definitions allow a reasoner to perform classification by subsumption [15]. For example, we could ask a reasoner to determine which kinds of platform provide long-endurance and firepower capabilities: $\mathsf{Platform} \sqcap \exists \mathsf{providesCapability.LongEndurance} \sqcap \exists \mathsf{providesCapability.Firepower}$. From the above definitons a reasoner can conclude that Predator satisfies these criteria.

Based largely on pre-existing ontologies, we have identified a collection of concept hierarchies relevant to the ISR domain. A sample of these is shown in Figure 3, including definitions of (a) platforms and (b) sensors. Note that the set of concept hierarchies is extensible: since creating the original version described in [6], we have added concepts representing NIIRS rating capabilities (e.g. Visible-4, Radar-6).

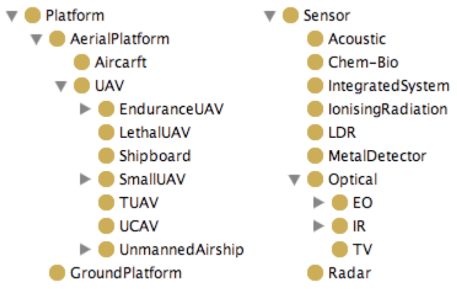

Fig. 3. Sample concept taxonomies relevant to the ISR domain

A *bundle type* is an intensional definition of a set of bundles of assets that can satisfy a task. The essential part of a bundle type is the specification of required sensor and platform types needed to provide the capabilities to satisfy the task. For this, we define a *platform configuration* $\pi = \langle P, \mathcal{S} \rangle$, where P is a type of platform, and $\mathcal{S} = \{S_1, \ldots, S_m\}$ is a set of sensor types that can be mounted in P simultaneously. Given a task T with a set of required ISR capabilities $\mathcal{C}^T = \{C_1, ..., C_n\}$, a single platform configuration is a *valid match* for T if the combined capabilities of P and $\mathcal{S}$ satisfy $\mathcal{C}^T$. Formally, $\mathcal{V}(T) = \{\pi_1, ..., \pi_n\}$ is the set of valid matches for task T iff $\langle P, \mathcal{S} \rangle \in \mathcal{V}(T) \iff \forall C_i \in \mathcal{C}^T$ where $(P \sqsubseteq \exists \mathsf{providesCapability}.C_i) \sqcup (S_i \in \mathcal{S} \sqsubseteq \exists \mathsf{providesCapability}.C_i)$.

More commonly, the requirements of a task will not be satisfied by a single platform configuration. We define a *package configuration* $\Pi = \{\pi_1, ..., \pi_n\}$, where π_i is a platform configuration. Π is a *valid match* for T if collectively the platforms and sensors in $\{\pi_1, ..., \pi_n\}$ satisfy the capabilities in $\mathcal{C}^T$, and Π is minimal with respect to $\mathcal{C}^T$ (there does not exist any subset of Π that is a *valid match* for T).

Bundle types are created by post-processing the package configurations, to add cardinality constraints, using pre-defined configuration knowledge, for example:

- at least 1 UAV with at least 1 Camera
- at least 2 AcousticArrays with exactly 2 ACINTSensors

V. Allocating Asset Instances

The aim here is to find an optimal allocation of individual asset instances to tasks while ensuring that they are correctly grouped into *bundle instances* (or simply *bundles*). A bundle instance is an instantiation of a bundle type, i.e. a set of real assets available in the field that are compatible with the bundle type defined in the previous step. We are thus moving from a "type level" fitting to an "instance level" allocation, that allows us to factor in logistical information (such as location, cost, readiness, etc) related in particular to asset instances available in the theatre. As previously noted in Section II, tasks might compete for the exclusive usage of the same asset instance. Our goal is to allocate specific assets to the tasks in order to maximize the *utility* of the sensor network.

In general for each single task we can choose among many different bundle instances. Therefore we need a way to compute the joint utility of a particular bundle instance to choose the best bundle to allocate to the task. Each task can be classified as belonging to a particular *task type*. The task/bundle types associated to a task localized in a particular AoI are used to decide the *joint utility model* (JUM) with which to compute the joint utility that a particular bundle of asset instances will bring to a task.

For example, in our scenario from Section II we have two tasks: T_1 event detection along the border, and T_2 event detection along the MSR. Both tasks belong to the same task type, *event detection*. For this particular task type we identified in [7] a joint utility model called Cumulative Detection Probability (CDP), which is able to compute the joint utility of a bundle regardless of the bundle type from which it was instantiated. In the scenario, we can choose between two bundle types for each task: $BT_1 = \langle \{\mathsf{UAV}, \mathsf{IMINT\text{-}sensor}\} \rangle, BT_2 = \langle \{\mathsf{AcousticArray}\}, \{\mathsf{AcousticArray}\} \rangle$. In CDP, the joint utility of a bundle is equal to a non-additive but submodular combination of the detection probability of each asset in the bundle. Every sensor asset can be associated with a detection probability and therefore taken into consideration when computing the cumulative detection probability.

This implies that in a scenario in which there are only event detection tasks, we can formally define the CDP maximization problem (MaxCDP) [7], where to globally maximize the utility we have to maximize the CDP. The utility of a sensor asset to a task is the probability that it will successfully detect the event if it occurs (we assume there are no false positives), which might be based on distance. Let $A_i \rightarrow T_j$ indicate that asset i is assigned to task j. The objective function is then to maximize the sum of detection probabilities (weighted by the task "profit" p_j, representing the "value" of achieving that task), given the probability u_{ij} that a single (sensor) asset A_i detects an event for T_j:

$$\sum_j p_j (1 - \prod_{A_i \rightarrow T_j} (1 - u_{ij})) \qquad (1)$$

As we showed in [7], MaxCDP is NP-hard even for geometric instances, even if sensors and tasks lie on a line.

To solve this problem, we propose a distributed approach which consists of a bidding protocol where sensor assets bid for tasks, described in Figure 4. A distributed approach is preferable because it does not require any central node to make the allocation decisions. This allows the protocol to leverage run-time status information about assets which are operational and which may currently be assigned to other tasks. Such a protocol is also scalable and more efficient in terms of communication cost compared to centralized approaches. The approach also assumes a dynamic system, in which the tasks may arrive and depart over time.

```
initialize each $e_{ij}$ as the detection probability of $A_i$ for $T_i$
initialize each task cumulative detection probability $u_j \leftarrow 0$
initialize number of assigned assets to $T_j$, $n_j \leftarrow 0$
initialize $N$ to maximum number of assets a task is allowed to have
initialize $R$ to number of rounds

Protocol for Task Leader ($T_j$):
  announce presence of $T_j$ to each neighboring asset $A_i$
  for round = 0 to $R$ do
    if $n_j < N$ then
      among responding assets $G$, choose $i \leftarrow \arg\max_i \{e_{ij} : A_i \in G\}$
      update $u_j \leftarrow u_j + e_{ij}$
      send accept messages and announce new $u_j$
    else done

Protocol for Asset ($A_i$):
  wait for task requests
  among requesting tasks $Q$, choose $j \leftarrow \arg\max_j \{e_{ij} p_j : T_j \in Q\}$
  send offer to $T_j$ including location
  if accepted then
    $A_i$ is assigned to $T_j$
    done
  else
    listen to current $u_j$ values for requesting tasks
    update detection probability based on new $u_j$'s
    $e_{ij} \leftarrow 1 - (1 - u_j)(1 - e_{ij}) - u_j$
  repeat
```

Fig. 4. Protocol for event detection (from [7])

In this protocol, an asset called the *task leader* is chosen for each task. Currently, the closest asset to the task's location in chosen for this, to minimise communication cost. The task leaders are informed about their tasks' types, locations and priorities by a base station. Each task leader runs the protocol locally to match nearby assets to the requirements of the task. Since the utility a sensor can provide to a task is limited by a finite sensing range, only nearby assets are considered. The leader advertises its task information to nearby assets; assets hearing this advertisement will offer their services to the task with their locations. To minimize interruption of other ongoing tasks, the leader of the arriving task always tries to satisfy the requirements of the task using only assets that are not currently assigned. If the task still requires more assets after trying all available assets, the task leader selects the best match among assigned assets. An asset that is assigned to a task that is not sensitive to preemption may be reassigned to another task if doing so will increase the total utility.

When a task ends, the leader sends out a message to announce that the task has ended and all its allocated assets are released. Because the system is dynamic, tasks that are not satisfied after the first allocation process will try to obtain more sensors once they learn there may be more available. This information can be obtained either from the base station or by overhearing the message announcing the end of a task.

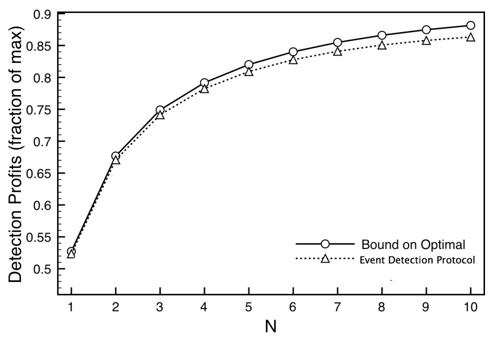

Fig. 5. Effect on performance of varying the number of assets in detection tasks

As described in [7], we implemented a simulator to test our event detection protocol on randomly-generated problem instances. Considering a network where only event detection tasks are present, we studied the effects of changing the maximum number of sensors that can be assigned to a detection task on the detection quality ("profit"). In Figure 5, the performance of the detection schemes is measured as N (the maximum number of assets a task is allowed to have) increases from 1 to 10. Note that a higher value of N means that more assets can be assigned to each detection task, which will increase the cumulative detection probability. The increase is rapid in the beginning but slows down due to the submodular nature of our cumulative detection function (1). The upper bound on the optimal solution is found by selecting the best N assets.

VI. Implementation and Deployment

We have implemented a pilot application called SAM (Sensor Assignment to Missions) as a proof-of-concept to test and refine our approach: see Figure 6. A user logs-in as a member of a particular CoI. In our simplified example, there are two, corresponding to the "UK" and "US" members of a two-country coalition. The user is able to select one or more AoIs on a map (left panel) and, for each, to select multiple ISR requirements (top-right panel) from the capability ontologies. SAM is implemented as a Web application in Java and uses the Pellet reasoner [17] to infer a set of package configurations, each of which can satisfy all the requirements (bottom-right). Before performing reasoning, the SAM application queries inventory catalogues to determine what types of platforms and sensors are actually available, so it can recommend package configurations including the most specific types that are potentially deployable. In doing this, it takes into account access policies on these resources, including ownership (note that the figure shows UK/US ownership of assets in the bottom-right panel) and whether the user has sufficient privileges to task those assets (shown by the "lock" icon next to the asset types). While simple, this mechanism is intended as an expansion point to allow the incorporation of more sophisticated access policies in future, such as those described in [1]. Figure 7 shows how the SAM application

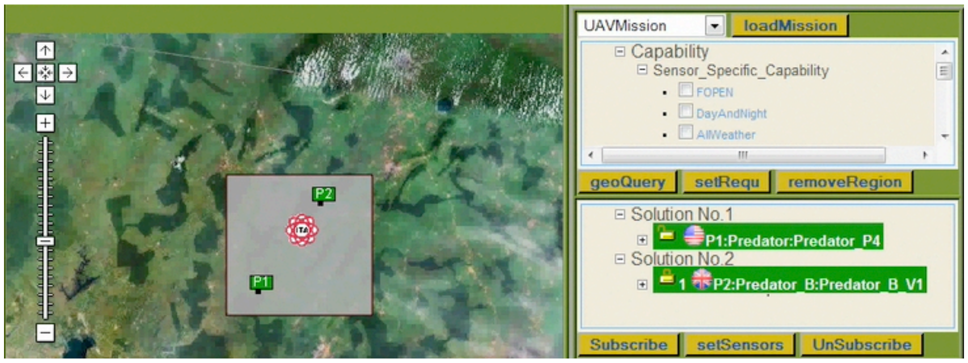

Fig. 6. Screenshot of the Sensor Assignment to Missions (SAM) application

works in a distributed fashion in a multiple-CoI context, with separate UK and US users, each having access to a private catalogue of ISR assets (Cat_{UK} and Cat_{US} respectively) and also a coalition-wide catalogue of shared assets (Cat_{Co}).

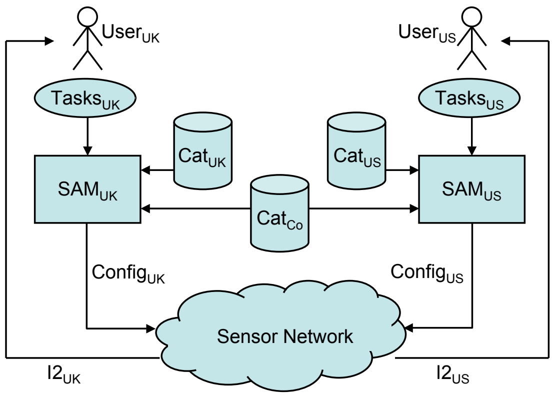

Fig. 7. A simple coalition use of the SAM application

The reasoning procedure is an exponential-time algorithm [16] and so the time required to compute package configurations increases rapidly with the number of classes in the ontology. However, in practice many optimisations and heuristics are possible to speed this up[1] and we expect that there will normally be far fewer asset classes than asset instances (as there will be many instances of each class). Also, the fitting step is simpler in the sense that it operates on a per-task basis, rather than considering many simultaneous tasks. In general, however, the exponential-time fitting algorithm will perform worse than the polynomial-time algorithms used in instance allocation, which is why we envisage that the fitting step will typically be carried out at planning-time, while allocation can be done dynamically at run-time.

Dynamic allocation (and potentially reallocation) of asset instances at run-time depends on having a dynamic sensor deployment environment. A sizeable amount of work is currently being put into developing such environments, generally following the principles of service-oriented architectures, allowing sensors to be described, discovered, configured, subscribed-to, and queried (e.g. [4], [18], [19]). Our prototype implementation uses the ITA Sensor Fabric [18], allowing the SAM application to dynamically configure a set of selected sensors on the network, and allowing the user to subscribe to these in order to receive the required information and intelligence (I2). This is shown in the lower part of Figure 7.

[1]The current prototype takes only a few seconds for an ontology of 180 types.

VII. Discussion and Conclusion

The SAM tool has been demonstrated to stakeholders in the UK MoD and US ARL, with positive feedback. The ability of the tool to generate explanations and support “what-if” explorations of potentially-available ISR solutions has been highlighted as particularly desirable, as has the incorporation of various kinds of policy. The use of ontologies offers potential for restricted “advertising” of assets within the coalition. As highlighted in [1], it is conceivable that a coalition member may not want to expose the full capabilities of its sensors. So, for example, a partner could share with the coalition that they have IMINT-capable UAVs, but withhold specific features of the IMINT package (perhaps advertising it at a lower NIIRS rating than it actually has).

The task-bundle-asset model provides two main degrees-of-freedom in allocating and reallocating assets to tasks. At planning time, a range of options is identified for each task by the fitting algorithm, based on potentially-available types of assets. Assets are dynamically and efficiently allocated into bundles (as specified by the bundle types) at run-time, and can be reallocated as the need arises. However, it is also feasible to choose an alternative task-bundle pairing at run-time without incurring the cost of re-running the fitting reasoning. This is trivial when the set of available assets is unchanged; it is also straightforward to prune the sets of recommended bundle types if assets become unavailable (e.g. destroyed). The potential also exists to explore incremental updates of the recommended bundle type sets when new asset types become available.

Our work so far takes a deliberately simple approach to resource scheduling: tasks are associated with a particular AoI, and the allocation algorithms use locality as part of their decision-making process. We envisage the allocation algorithms being run dynamically at discrete timesteps throughout operations, in synch with mission tempo, allowing late-binding of sensors to tasks, and hence increasing agilty. We accept, however, that resource scheduling approaches may be beneficial, and indeed have developed constraint-based techniques for this in the past [20]. These allow a resource-manager to maintain a schedule of resource commitments over time, and to decide, in a transparent and explainable way, if and when to break existing commitments in order to adopt new ones.

The approach described above assumes that an asset is assigned exclusively to one bundle, and a bundle is assigned exclusively to one task. We intend to relax this restriction in future work, to allow sharing of bundles among tasks, and assets among bundles. This reflects the reality that the same sensed data can contribute to the satisfaction of more than one task. Possibilities under consideration here include the preprocessing of task descriptions to identify sets of tasks where information can be shared among them, and richer modelling of sensor and platforms to allow timesharing of assets where a single asset instance to contribute part of its "capacity" to multiple tasks.

Finally, we observe that use of the bundle formalism allows us potentially to cope with three additional important aspects of CCIRM: assignment of complementary sensing types to the same task, assignment of assets other than sensors and platforms, and sensor cueing. It is straightforward to extend the NIIRS knowledge base described in Section III to assert that complementary sensing types are required to confirm findings. So, for example, in the vehicle identification case we might require: $\mathsf{Visible\text{-}4} \sqcap \mathsf{Radar\text{-}6}$ instead of $\mathsf{Visible\text{-}4} \sqcup \mathsf{Radar\text{-}6}$. In a similar way, we can add types of asset other than sensors and platforms (for example, kinds of HUMINT or OSINT) to a bundle, and can require the provision of assets in a bundle that support sensor cueing, capable for example of allowing scan-cue-focus networked operation [21].

Acknowledgments This research was sponsored by the US Army Research Laboratory and the UK Ministry of Defence and was accomplished under agreement W911NF-06-3-0001. The views contained in this paper are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the US ARL, the UK MoD, or the US or UK Governments. The US and UK Governments are authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

References

[1] T. Pham, G. Cirincione, D. Verma, and G. Pearson, "Intelligence, surveillance, and reconnaisance fusion for coalition operations," in *Proc 11th International Conference on Information Fusion*, 2008.

[2] Joint Publication, "2-01: Joint and national intelligence support to military operations," 2004. [Online]. Available: http://www.dtic.mil/doctrine/jpintelligenceseriespubs.htm

[3] R. Desimone and D. Charles, "Towards an ontology for intelligence analysis and collection management," in *Proc 2nd International Conference on Knowledge Systems for Coalition Operations (KSCO 2002)*, 2002, pp. 26–32.

[4] M. Botts, A. Robin, J. Davidson, and I. Simonis, "OpenGIS sensor web enablement architecture document," Open Geospatial Consortium Inc, Tech. Rep., 2006, OpenGIS Discussion Paper.

[5] M. Gomez, A. Preece, M. Johnson, G. de Mel, W. Vasconcelos, C. Gibson, A. Bar-Noy, K. Borowiecki, T. L. Porta, D. Pizzocaro, H. Rowaihy, G. Pearson, and T. Pham, "An ontology-centric approach to sensor-mission assignment," in *Proc 16th International Conference on Knowledge Engineering and Knowledge Management (EKAW 2008)*, 2008.

[6] A. Preece, M. Gomez, G. de Mel, W. Vasconcelos, D. Sleeman, S. Colley, G. Pearson, T. Pham, and T. L. Porta, "Matching sensors to missions using a knowledge-based approach," in *SPIE Defense Transformation and Net-Centric Systems 2008 (SPIE Defense and Security Symposium)*, 2008.

[7] H. Rowaihy, M. Johnson, D. Pizzocaro, A. Bar-Noy, T. L. Porta, and A. Preece, "Sensor-task assignment protocols," International Technology Alliance, Tech. Rep., 2008.

[8] A. Preece, M. Jackson, G. Pearson, and T. Pham, "Sensor-mission assignment: A scenario-driven walkthrough," in *Second Annual Conference of the International Technology Alliance (ACITA 2008)*, 2008.

[9] National Imagery Interpretability Rating Scale (NIIRS), http://www.fas.org/irp/imint/niirs.htm.

[10] D. Wilson, C. Pettit, M. Lewis, S. Mackay, and P. Seman, "Propabilistic framework for characterizing uncertainty in the performance of networked battlefield sensors," in *SPIE Defense Transformation and Net-Centric Systems 2008 (SPIE Defense and Security Symposium)*, 2008.

[11] T. R. Gruber, "Toward principles for the design of ontologies used for knowledge sharing," *Journal of Human Computer Studies*, vol. 43(5/6), pp. 907–928, 1994.

[12] D. Russomanno, C. Kothari, and O. Thomas, "Building a sensor ontology: A practical approach leveraging ISO and OGC models," in *Proceedings of the International Conference on Artificial Intelligence*, 2005, pp. 637–643.

[13] L. Bermudez, J. Graybeal, and R. Arko, "A marine platforms ontology: Experiences and lessons," in *Proceedings of the ISWC 2006 Workshop on Semantic Sensor Networks*, Athens GA, USA, 2006.

[14] UJTL Ontology, http://orlando.drc.com/semanticweb/daml/ontology/condition/ujtl/condition-ont.

[15] G. Antoniou and F. van Harmelen, *A Semantic Web Primer, 2nd edition*. MIT Press, 2008.

[16] F. Baader, I. Horrocks, and U. Sattler, "Description logics," in *Handbook on Ontologies*. Springer, 2004, pp. 4–28.

[17] Pellet OWL DL Reasoner, http://pellet.owldl.com/.

[18] F. Bergamaschi, D. Conway-Jones, C. Gibson, and A. Stanford-Clark, "A distributed test framework for the validation of experimental algorithms using real and simulated sensors," in *First Annual Conference of the International Technology Alliance (ACITA 2007)*, 2007.

[19] E.Bouillet, M.Feblowitz, Z.Liu, A.Ranganathan, A.Riabov, and F.Ye, "A semantics-based middleware for utilizing heterogeneous sensor networks," in *Distributed Computing in Sensor Systems (LNCS 4549)*. Springer, 2007, pp. 174–188.

[20] S. Chalmers, A. D. Preece, T. J. Norman, and P. Gray, "Commitment management through constraint reification," in *3rd International Joint Conference on Autonomous Agents and Multi Agent Systems (AAMAS 2004)*, 2004, pp. 430–437.

[21] G. Pearson, "A vision of network-centric ISTAR and the resulting challenges," in *SPIE Defense Transformation and Net-Centric Systems 2008 (SPIE Defense and Security Symposium)*, 2008.

Appendix - Glossary of Acronyms

ACINT	Acoustic Intelligence
AoI	Area of Interest
CCIRM	Collection Co-ordination and Intelligence Requirements Management
CoI	Community of Interest
CDP	Cumulative Detection Probability
HUMINT	Human Intelligence
I2	Information and Intelligence
IREQ	Information Requirement
ISR	Intelligence, Surveillance and Reconnaissance
IMINT	Imagery Intelligence
JETL/METL	Joint/Military Essential Task List
MALE	Medium Altitude Long Endurance (UAV)
MSR	Main Supply Route
NIIRS	National Imagery Interpretability Rating Scale
OSINT	Open Source Intelligence
SSIR	Scenario-Specific Information Requirement
UAV	Unmanned Aerial Vehicle
UJTL	Universal Joint Task List