Cooperative Situation Assessment in a Maritime Scenario

Alessandro Farinelli[*], Daniele Nardi[†], Roberta Pigliacampo[‡], Mirco Rossi[§], Giuseppe Paolo Settembre[†]

Abstract

In large-scale, complex domains such as space defense and security systems, situation assessment and decision making are evolving from centralized models to high-level, net-centric models. In this context, collaboration among the many actors involved in the situation assessment process is critical in order to achieve a prompt reaction as needed in the operational scenario.

In this paper, we propose a multi-agent based approach to situation assessment, where agents cooperate by sharing local information to reach a common and coherent assessment of situations. Specifically, we characterize situation assessment as a classification process based on OWL ontology reasoning, and we provide a protocol for cooperative multi-agent situation assessment, which allows the agents to achieve coherent high level conclusions. We validate our approach in a real maritime surveillance scenario, where our prototype system effectively supports the user in detecting and classifying potential threats; moreover, our distributed solution performs comparably to a centralized method, while preserving independence of decision makers and dramatically reducing the amount of communication required.

1 Introduction

In modern applications, decision making processes often require to coordinate several actors. For example, in large-scale, complex domains like space explorations, disaster response, real-time monitoring, defense and security systems, collaboration is critical in order to achieve a prompt reaction to many unexpected situations that may happen. In particular, among the

[*]A. Farinelli is with the CS Department, University of Verona, Italy.
[†]D. Nardi and G. P. Settembre are with the DIS Department, University "Sapienza" of Rome, Italy.
[‡]R. Pigliacampo is with SESM s.c.a.r.l. - a Finmeccanica company.
[§]M. Rossi is with SELEX S.I. - a Finmeccanica company.

several challenges, a current objective, from both a scientific and industrial perspective, is to remove the dependency from a centralized decision entity, relying instead upon mechanisms which allow for decision-making in a distributed way.

In this context, Situation Assessment [16] is the process that "dynamically attempts to develop a description of current relationships among entities and events in the context of their environment". In a complex scenario, which deals with several actors and events, like the above mentioned ones, Situation Assessment requires fusing information from different sources to recognize high-level relationships and, moreover, provide a classification of the situation that supports the detection of anomalies and threats.

While many approaches allow current systems to achieve effective performances on low level data analysis (i.e. level 0 and 1 of the JDL model [27]), there is a strong need for techniques that can process data at a higher level thus performing Situation Assessment. Many Situation Assessment approaches have been proposed in the literature [22, 26, 5] but they usually propose a centralized architecture that can not be easily adapted to decentralized settings such as the ones we are interested here.

In contrast in this paper, we present a multi-agent approach to Situation Assessment. Such solution is key whenever the application domain is endangered to become unmanageable for a centralized decision making, due to the foreseeable increase of entities involved (e.g. maritime surveillance, air traffic control [1]). In order to achieve this goal, it is necessary to devise techniques which can convey pieces of information towards those destinations where they are strategic for decision making. *"The right information, in the right place, at the right time"*.

Specifically, our approach provides the following innovative contributions: (1) We characterize Situation Assessment as a classification process, based on each agent's knowledge base (KB) and reasoning capabilities; (2) we apply and adapt a coordination protocol for multi-agent situation assessment, in order to achieve high level conclusions; (3) we present a validation of the approach in a real case maritime surveillance scenario.

Our first contribution is the formalization of the problem by suitably defining a symbolic representation of the domain that enables for the classification of the situations of interest. More specifically, we build a representation of the domain and the occurring events (perceived with noise) in OWL-DL [34], a well known standard for ontology representation (ontology web language) and use the reasoning services provided by Description Logics (DL) [36], which allow each agent for the classification of high level situations over a set of events.

Our second contribution is at the core of multi-agent Situation Assessment: enable the agents assess obtained conclusions, sharing just the necessary amount of knowledge about the situation at hand. The proposed solution relies on a distributed algorithm for situation assessment [49], which solves disagreements among agents, by using sequences of one to one interactions. While, in [49], agents argue only about their interpretations of events, here we frame the distributed algorithm in the context of a OWL-DL knowledge bases, thus allowing for a greater expressiveness and a standard representation for high level situations.

A validation in a real case context of maritime surveillance has been conducted in collaboration with Selex-SI [47], the system integrator house within the Finmeccanica group, which started addressing a net-centric architecture for harbor protection in the early 2000s, and is currently exploring multi-agent technologies in order to extend their system with more sophisticated decision making. Experiments in the maritime scenario show that our approach requires a smaller amount of knowledge to be exchanged among agents than other solutions that broadcast each assertion, thus preserving locality and dramatically limiting communication requirements.

The paper is organized as follows. In Sec.2 we analyze approaches to Situation Assessment currently present in literature. In Sec.3, we present our overall multi-agent approach to Situation Assessment and focus on the knowledge base construction and the classification of situations by a single agent; in Sec.4, we present our distributed algorithm for Situation Assessment, first defining the interaction protocol and then the knowledge management required by the assessment process; in Sec.5, we present the set of experiments performed in the context of maritime surveillance; conclusions and future work are addressed in Sec.6.

2 Related Work

The development of new approaches for Situation Assessment has recently become a key issue for the maritime environments, that represent our reference domain [18]. Specifically, such novel approaches should be able to deal with the increasing complexity and dynamics of such scenarios as well as the need of a good and reactive understanding of the situation at hand by the human operators in high workload conditions.

In fact, classic approaches to maritime surveillance are usually based on a centralized architectures, and they mainly rely on human operators to understand complex relationships among relevant entities (e.g., vessels that navigate in critical zones such as harbors). Specifically, several works show how operator's ability to recognize abnormal behaviors decreases with the system complexity [13, 38] and how building a model of object behaviors may help simplifying the access to relevant information [41]. These works generally focus on human-computer interaction issues, but highlight the importance of supporting the human operators with automatic situation interpretation systems.

Few experimental frameworks try to exploit automatic systems to assess high-level conclusions, in particular malicious behaviors, using for example learning techniques [40, 3] or clustering algorithms [25]. These approaches are particularly useful when only homogeneous data are available for classification; whenever a significant amount of contextual information should be used for classification, these techniques generally show their limits, and an inference process seems more appropriate. Contextual information are crucial to handle complex environments, typical of practical applications. For example a system designer might want to specify different methods to perform situation assessment with respect to different operational conditions (e.g. if it is raining and perceptions are very noisy, take extra care before signaling an alarm). Many works report the presence of large amounts of additional information taken from the context, which are strategic to classify actual situation, but usually are not included in any in-use system for situation understanding [37].

In this context, agent-based methods appear to be a promising approach to situation assessment and several works exploit agent-based technology [6, 53, 46]. However, in these approaches, agents are not used as effective actors in the process of situation assessment, but rather as a modeling tools to simulate strategies that aim at preventing dangerous or harmful situation (e.g., a small boat attacking a vessel).

Centralized situation assessment

Most of the "classical" approaches to situation assessment focus on building a high-level representation of the world, by providing a formalization of situation, actions and events, and by interpreting and contextualizing data with respect to the world model. However, most of the time, they rely upon centralized architectures and cannot easily be adapted to distributed ones. Moreover, when deployed in a real application, these approaches usually suffer from the typical problems of a centralized architecture (single point of failure, massive communications, scalability issues).

The work by Kokar and colleagues focuses on providing a formalization of Situation Awareness [31, 32, 33, 22], placing the problem of reaching Situation Awareness in the third layer of the JDL data fusion model as defined in [16]. The proposed architecture uses a knowledge based agent, which is able to draw high level conclusions in the application domains. Such approach, which inspired part of our work, is based on: i) the use of a domain ontology to represent the relevant elements of the scenario, including interesting high level relationships; ii) the use of a rule propagation system to populate instances of relations, whenever domain-specific conditions are verified; iii) an explicit representation of time intervals, thus allowing the representation of properties which may dynamically change their state. Similarly, we characterize events as objects that build/enable situations and represent the agent model using OWL [34], the standard proposed by W3C for ontology description based on Description Logics inference. The most distinguishing feature of our approach is that it deals with the the problem of different assessment, that arises when situation management is concurrently executed by several agents.

Little and Rogova address the symbolic representation of relationships among entities in the disaster management domain [26], focusing on the trade-off between the generality of the representation and the possibility to include domain-specific characteristic. These two aspects are critical when ontologies are used to reason over relationships among entities in a complex scenario.

High level fusion is also motivated by the impossibility to capture non-real-time data (NRT) inside standard techniques for low level data fusion. In fact, most of the time, NRT data are simply assumed to be static, or their inclusion in the fusion process is left to human operators. Sycara et al. advocate the use of a symbolic representation for contextual data to support several high level processes, like force structure recognition, intent inference [52].

Capraro et al. use symbolic reasoning to refine perceptions at data level [5]: instead of allowing for higher level conclusions, a priori knowledge related to the specific sensor attitude is used to customize and tune the feature extraction process, thus reducing the effects of the uncertainty of information sources. With respect to these approaches, our research will pursue the goal of giving high level events a unique symbolic representation, and obtaining new conclusions with logical reasoning, distributing the process over a team of agents.

Whenever symbolic reasoning needs to be used to obtain conclusions from sensors, the system must explicitly deal with the uncertainty of information. Therefore, standard logic-based frameworks have often been extended with uncertainty representation and features (e.g. [14, 2, 42]). Also OWL is provided with several probabilistic extensions [8, 17, 19, 23, 28]. In these approaches, usually every assertion about individuals is given with a certain degree of certainty, allowing the system to know the reliability of the inferred conclusions. In our approach, we deal with uncertainty separately from situation management, assuming the information of the feature extraction to be the best assessment currently available. This choice preserves the flexibility of the current solutions allowing us to operate in a well known framework with a well defined semantics. Moreover, we can use state of the art technologies [50, 15, 21] for reasoning and thus focus our effort only on relevant aspects of situation understanding.

Assessment in distributed settings

The approaches developed to address situation assessment in the context of multi-agent systems usually take advantage of the distributed architecture (no single point of failure, decentralized computational load), but they are often restricted to low level fusion (e.g. [29, 54, 49, 52]), and hardly usable for high-level understanding of the situation. Very limited attention has been devoted to the problem of situation assessment in a distributed setting, where each agent has the reasoning capability to classify the situation and act accordingly. Nonetheless, several techniques appear in the literature, which try to decentralize part of the process, in order to achieve some specific improvement. For example, several approaches (e.g., [30, 24]) decentralize only the sensing process, but the situation assessment and decision making always takes place in a single point of the system. Similarly, Museux et al.[35] propose a distributed fusion approach, where semantics of information is considered; however, the distributed information sources only perform the feature extraction process, while fusion still relies on a single system entity. The main limitation of these approaches is that a high number of messages have to be exchanged, in case many features are relevant to draw a conclusion, and the distributed entities are still dependent from the central unity to make decisions. With respect to these approaches, we are aiming at decentralizing also the situation assessment process.

For applications involving multi-robot platforms or sensor networks, information is frequently integrated at data level using a distributed approach across the robots [29] or sensors [54]. Specifically in multi-robot systems, robots are often forced to use a complex world model and to cooperate in order to initiate a course of actions. Their on-board sensors and different location in the environment, provide them with different perceptions and executing a complex plan often involves task to be executed by several robots. Hence, in this settings cooperative perception is often used to deal with perception limitation of the single robot [7, 43, 51]. The general idea is to exchange sensor readings and aggregate them using different filtering techniques to reduce the uncertainty before deciding how to act. Other techniques explicitly deal with the uncertainty when choosing a course of actions; for example COM-MTDPs [39]. However, such approaches often require to exchange large amounts of data among robots, since typically, each robot attempts to maintain an accurate model of the complete state. On the contrary, in practice, only a small part of the overall state might be relevant to its activities. Some work explore this possibility [44], but current results are still limited to small number of agents.

Finally, Buford et al. addresses distributed situation management by proposing a coupling of agent platforms (AP) and P2P overlay networks which address interoperability and high scalability requirements [4]. In order to share events and situations between peers, a two-phase semantic discovery mechanism is supported: the protocol uses a Distributed Hash Table (DHT) to discover an AP of interest, followed by sending a semantic query to the directory service on the AP. However, the situation recognition process is still localized on each AP and performed by a single agent (Situation Manager), which relies only on local information and event correlation to create and update situations (instances of classes defined in some domain ontology) and notifies each agent which has previously subscribed to situations of that category (mostly related to its functions).

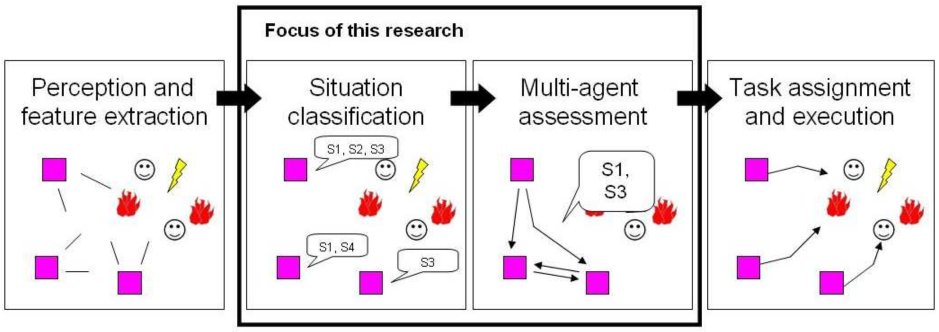

Figure 1: Description of a multi-agent situation assessment process.

3 Agent-Based Situation Assessment

In Fig.1, the main steps of a multi-agent approach to Situation Assessment are shown. Squares represent the agents that take part to the process. Fires, emoticons and lightnings represent dynamic events which are imperfectly perceived by agents.

In the first step of the process, denoted as "Perception and feature extraction", distributed sensing is performed, where each agent separately performs perception with his own sensors, and extracts some relevant features from sensor readings. This part of the process includes also data association (or object assessment).

The second part of the process, denoted as "Situation Classification", consists in obtaining, from a single agent perspective, a high-level assessment hypothesis of situations which are of interest in the perceived scenario.

Each agent may estimate that several (relevant) situations are currently present in the scenario[1]. Due to different agent's perceptions, agents may have different assessment at the end of this phase.

The third part of the process, denoted as "Multi-agent assessment", consists in the agents arguing, generally through message exchange, about their assessments, aiming at reaching an agreement on each of their previous conclusions. At the end of this phase, the team as a whole has somehow solved the conflicting assessments. In general, it is not necessary that each agent knows the assessment of each situation, because it may not be interested in having information

[1]The presence of several situations in the agents' balloon may be misleading for the reader who is familiar with the term *situation* in formal logics, because having several situations would represent that an agent has several alternative models for the KB. This is *not* the intended meaning: speaking in terms of formal logics, we may say that we indicate with different terms independent conclusions extracted from the KB typically corresponding to a single model (the global situation).

about certain situations at all.

The last part of the process, denoted as "Task assignment and execution", involves planning the best action, assigning tasks to the agents and executing them. Some parts of this phase may again be performed in a distributed way.

The research presented in this paper focuses precisely on the two central parts of the process: in the rest of this section, we address the first of them, namely the classification of the situations from the standpoint of a single agent, as a basic tool supporting situation assessment. More specifically, we first provide a formal characterization of the problem, then discuss the population of the knowledge base, given the events perceived by the agent, and finally we present an example of situation classification. The extension of the proposed approach in a multi-agent system is discussed in the next section.

3.1 Problem formalization

We consider a certain number of observed entities pursuing some unknown private goals, and a set of agents $A_1, ..., A_m$. Each agent has a world model, its own perceptions, it communicates with the other agents, and takes part pro-actively in the classification process. The agent's world model is an *ontology*, which is a symbolic representation of the organization of concepts and their relevant relationships into the domain (intensional knowledge, TBox); the TBox is the same for all the agents, and constitutes a common language for communication.

The specific scenario where the agent is operating is represented by a set of *assertions* (extensional knowledge, ABox), that are built based on the events perceived by the agent. We use the term *event* to refer to any property/object that results from sensor interpretation and corresponds to an assertion in the ABox. More specifically, we have to take into account that each event can be perceived at different times (possibly by multiple agents). Thus, we need a method for combining multiple, possibly uncertain observations into a belief about the event under consideration. In this way, we handle the connection between a numeric representation of uncertain perception and the symbolic information (facts) to be asserted in the extensional portion of the knowledge base (ABox).

In the next subsection, we describe a simple solution to this problem, called *event assessment*, that constitutes a building block of our approach to situation assessment.

In order to focus on the aspects of interest within the perceived events, we consider the set of the *relevant situation classes* $S_{CL} = \{S_1, \ldots, S_n\} \cup S_\top$, in which each S_i identifies a type of situations, of interest with respect to the context. Basically, each relevant situation class S_i represents a group of semantically equivalent circumstances of the world (for the purpose of Situation Assessment). A symbolic definition of each S_i is given in terms of the type of events which are observable in the environment. In the following, the situation classes are defined in Description Logics (DL) [36]. With $S_\top$ we denote the most generic situation class, which includes all possible situations. A situation class S_i is *more specific* than S_j, iff every instance of S_i is also an instance of S_j. In DL, this is denoted by $S_i \sqsubseteq S_j$.

A situation instance s_i is an individual characterized by a set of events, which are actually observed by an agent on the scenario. From the point of view of the agent's knowledge, with $K_p(s_i \in S_j)$ we denote that an agent A_p *knows that* $s_i \in S_j$, which means that a set of events, which characterize s_i as in the definition of a class S_j, are known to A_p: in DL, this simply means that in A_p's knowledge base, s_i is classified as an instance of S_j.

We can now provide a formal characterization of the Situation Assessment process.

Definition 1 We say that s_i *is assessed as* S_i by an agent A_p iff

- $K_p(s_i \in S_i)$
- $\nexists S_j \ s.t. \ K_p(s_i \in S_j) \wedge S_j \sqsubseteq S_i$.

Thus, given the above formalization, *Situation Assessment (SA)* is the *non-trivial* assessment of all the instances s_i. A classification of s_i as S_i is *non-trivial* if $S_i \neq S_\top$.

In our approach, this formalization has been realized through ontologies, written in the OWL language, which is based on the Description Logics formalism (see [36]). A distinguishing feature of our approach is that we classify the situations of interest as instances of the relevant situation concepts, that are represented in a *taxonomy* of situations. An example TBox representing the taxonomy of situations for maritime domain is shown in Fig.2. The proposed approach allows to use the reasoning services provided by Description Logics: we can compute subsumption of situations, as well as other relations of interest (e.g.distinct_from, overlaps, covers, covered_by, equal, contains). Previous approaches [32] need to introduce a rule propagation engine to reason on situations.

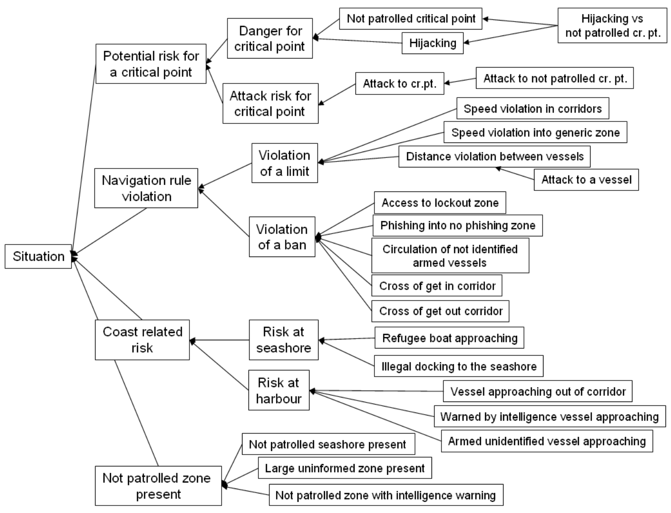
Situation
Potential risk for a critical point
Danger for critical point
Not patrolled critical point
Hijacking
Hijacking vs not patrolled cr. pt.
Attack risk for critical point
Attack to cr.pt.
Attack to not patrolled cr. pt.
Navigation rule violation
Violation of a limit
Speed violation in corridors
Speed violation into generic zone
Distance violation between vessels
Attack to a vessel
Violation of a ban
Access to lockout zone
Phishing into no phishing zone
Circulation of not identified armed vessels
Cross of get in corridor
Cross of get out corridor
Coast related risk
Risk at seashore
Refugee boat approaching
Illegal docking to the seashore
Risk at harbour
Vessel approaching out of corridor
Warned by intelligence vessel approaching
Armed unidentified vessel approaching
Not patrolled zone present
Not patrolled seashore present
Large uninformed zone present
Not patrolled zone with intelligence warning

Figure 2: An example of situation taxonomy for maritime domain

3.2 Event assessment

In order to populate the agent's ontology with data extracted from the agent's sensor processing, we have to explicitly deal with the uncertainty of event perception.

More specifically, given a set of perceptions that refer to the same property of the scenario, we need to build the belief of the agent represented as an assertion in the ABox. There are several techniques that can be applied to address this problem (see for example [16]); we chose to rely on a simple Bayesian filter, since our focus is on the upper layers of the fusion process. Obviously, more sophisticated techniques can be deployed, depending on the specific setting addressed.

Each time a new sensor reading is provided, features are extracted about specific circumstances of interest, and they are abstracted as positive or negative observations of certain events at specific time instances.

Algorithm 1 presents the pseudo-code executed by an agent, when a new observation is received (directly from sensors or another agent). Each observation will be denoted, within the code, with the following structure:

$obs =< ag, event, value, time >$

where

- ag = the ID of the agent who performed the observation

- *event* = event identification
- *value* = $\{true, false\}$, indicates whether *event* is confirmed or negated by this observation.
- *time* = the time which the observation refers to

. The Bayesian filter considers all the observations in the past[2]. When a reading referring to a past time is obtained, the filter is reinitialized, and the belief recalculated, starting from the time of the oldest available observation.

Notice that, we do not address the data association problem regarding observations. While this problem must be solved somehow to allow agents to combine perceptions about events this is outside the scope of the present paper and we rely on standard techniques to address this problem [16].

Algorithm 1 INTEGRATEOBSERVATION

```
Input: an observation obs and its corresponding event
Output: update of agent's list of observations and ABox

 1: $event\_Obs \leftarrow$ getObs($event$)
 2: if $obs \notin event\_Obs$ then
 3:    $event\_Obs \leftarrow event\_Obs \cup obs$
 4:    $belief \leftarrow$ evolve_Filter($event\_Obs$)
 5:    if $belief > k_1$ then
 6:       ASSERT( $event$ )
 7:    else
 8:       if ($belief < 1 - k_1$)$\wedge$ Concept($event$) then
 9:          ASSERT( $\neg event$ )
10:       else
11:          REMOVE( $event$ )
12:       end if
13:    end if
14: end if
```

Algorithm 1 allows each agent to know whether considering an event to be true, false or unknown, given his available observations. Thus, in our approach, *each ABox assertion is "justified" by a list of observations, which can be retrieved at any time, if needed.* The function GETOBS retrieves from the agent's memory the list of observations related to a certain assertion. The function EVOLVEFILTER, computes the likelihood of an event (i.e., the belief) given the list of observations, based on the Bayesian filter already used in [12]. The event is considered true

[2]Maintaining the history of observation clearly has a cost in terms of memory that grows with time. To limit such cost, a valid time window T is used that goes from the current time t_c back to $t_c - T$. An appropriate time window can be chosen by considering the evolution model of the environment.

and inserted in the agent's knowledge base if the belief is greater than a threshold k_1, while the falsity of the event is asserted if the belief is less than $1 - k_1$ and the assertion refers to a concept. In fact, we can rely on negation for concepts, but for role assertions we do not have negation, and we can only represent lack of knowledge about the event.

The functions ASSERT/REMOVE are used to add/remove assertions from the agent's knowledge base (ABox). Through ASSERT inconsistent information can be introduced into the agent's KB; moreover, REMOVE would, in general, require to deal with the problem of belief revision (see for example [9]). In our implementation, we simply check for ABox consistency, and use heuristics to solve inconsistent sets of assertions, as typically done in most knowledge based systems. This issue comes up also in the knowledge management operations that are discussed in Subsection 4.2.

Algorithm 1 is executed when a new observation is acquired by an agent either through direct perception or by the distributed assessment process described in the next sections. The result is a consistent knowledge base, resulting from the beliefs supported by the new observation.

3.3 An example

The taxonomy that refers to our maritime domain has been illustrated in [48]. Each agent is provided with a taxonomy of the relevant events of the domain, and one which includes the definitions of the relevant situation classes. Below, we show a simple example taken from our working domain.

We define the following concept names: `FREEACCESSAREA` is a free access sea area, `LOCKEDAREA` is a lockout sea area, `NONAVIGATIONAREA` is a not navigable area, `DANGEROUSAREA` is a dangerous area, `UNKNOWNVESSEL` is the set of unidentified vessels perceived in the scenario, `ARMEDVESSEL` is the set of vessels carrying weapons on-board.

Moreover, `Position` is the property which relates a vessel with its position and `hasObject` is the property which relates situation instances and objects that are involved in the situation. We consider the following TBox assertions:

(1) Free access areas are disjoint from lockout areas

(2) Lockout areas are the union of not navigable and dangerous areas

(3) A vessel has a unique position

(4) SITUNKVESSEL are situations with at least one unidentified vessel

(5) SITLOCKAVESSEL are situations where a vessel is detected in a lockout zone

(6) SITUNKARMEDVESSEL are situations with vessels that are both unidentified and armed

Assertions (1),(2),(3) refer to a simple taxonomy of the scenario, while (4),(5),(6) are a simple taxonomy for situation classes. In Description Logics, the above sentences are represented as follows:

(1) $\texttt{FREEACCESSAREA} \equiv \neg\texttt{LOCKEDAREA}$

(2) $\texttt{LOCKEDAREA} \equiv \texttt{NONAVIGATIONAREA} \sqcup \texttt{DANGEROUSAREA}$

(3) $\top \sqsubseteq \leq 1\ \texttt{Position}$

(4) $\texttt{SITUNKVESSEL} \equiv \exists\texttt{hasObject}.\texttt{UNKNOWNVESSEL}$

(5) $\texttt{SITLOCKAVESSEL} \equiv \exists\texttt{hasObject}.\exists\texttt{Position}.\texttt{LOCKEDAREA}$

(6) $\texttt{SITUNKARMEDVESSEL} \equiv \exists\texttt{hasObject}.\texttt{UNKNOWNVESSEL} \sqcap \texttt{ARMEDVESSEL}$

In order to generate the correct number of situation instances, we identify a subset of events, which trigger the generation of a new situation instance. Each situation class has at least one triggering event. When a new triggering event is detected, a new situation instance s_i is created, and asserted to belong to the most general situation class SITUATION, corresponding to $S_\top$ of our formalization. Moreover, an assertion is generated establishing the relation hasObject between the situation instance s_i and the event e_i, since the situation instances are generated based on the detection of events in different locations of the scenario.

The situation instances are then classified and assessed using standard DL inference capabilities, specifically *realization*, which returns the most specific concept an individual is an instance of [36]. As a result, s_i will be classified as member of S_k, written $K_p(s_i \in S_k)$ in our formalization. The classification process is also explained in details in [48].

Let us consider for example the scenario where the agent receives enough new observations that allow for the following assertions:

(7) p_1 is a position in a free access zone

(8) a vessel v_1 is detected in position p_1

(9) vessel v_1 is unidentified

The detection of an unidentified vessel, for example v_1 is a triggering event. Such an event can give rise to two assertions, the first one stating that v_1 belongs to `UNKNOWNVESSEL` and the second one specifying its position through the role `Position`. The corresponding agent's ABox is:

(7) `FREEACCESSAREA`(p_1)

(8) `Position`(v_1, p_1)

(9) `UNKNOWNVESSEL`(v_1)

Since the detection of an unidentified vessel is a triggering event, the following assertions are added to the ABox:

(12) `SITUATION`(t)

(14) `hasObject`(t, v_1)

By applying realization on the situation instance t we conclude that t is an instance of class `SITUNKVESSEL`. In fact, t participates in relation `hasObject` with a concept of class `UNKNOWNVESSEL` as the definition of `SITUNKVESSEL` requires; moreover, the agent is not able to classify t as an instance of `SITUNKARMEDVESSEL` which would have been more specific than `SITUNKVESSEL`. `SITUNKVESSEL` is therefore the best current available assessment.

4 Distributed situation assessment

In this section, we move to the scenario where multiple agents, deployed in the environment, independently detect events and need to turn perceptions into knowledge, by making a suitable assessment of the situation. The single agent approach that we have described in the previous section needs to be generalized in several respects.

In particular, we need to establish a protocol that allows the agents to share classifications of situations, as supported by their beliefs, in order to reach consensus about the situation assessment. More specifically, our aim is to define a distributed process to classify situations, that achieves the performance of a centralized one, while minimizing the information flow among the agents. In Subsection 4.1 we present a protocol that allows to reach consensus on a specific situation instance. Such a protocol relies on specific reasoning services that are described in more detail in Subsection 4.2. In particular in this paper we extend the coordination protocol proposed in [49], which allows for the distributed assessment of individual events.

It is worth emphasizing that in order to achieve our goal several problems, that are deeply investigated in the literature, must be tackled. The novelty of our proposal is in putting together an integrated framework by combining existing solutions, and introducing in some cases ad-hoc solutions to manage the complexity.

4.1 Distributed Assessment Protocol

Since agents may reach different conclusions, due to their partial (and noisy) perceptions, a coordination technique is needed, to be used by the agents in order to reach an agreement about the situation classification.

The proposed solution is inspired to the Token Passing algorithm for task assignment [45] which is the one that better fits our goals. In fact, it takes into account all our crucial requirements, guaranteeing good performance on dynamic unpredictable change in the environment [11], the absence of conflicts in task assignment [10], and avoiding massive broadcast communication among the team members [12]. The token mechanism in our approach will be used as a mean to allow arguing among different proposals and exchanging observations among agents, whose overall goal is to pro-actively perform Situation Assessment. The main challenge is to maintain a shared knowledge of the evolution of the situation, without relying on simple flooding of updating messages, but exploiting the shared model of the world and the reasoning capability of each agent, as in [54].

The key idea of the approach is that each agent will send its situation assessment proposals to team mates, based on his current information on the world, receives challenges from disagreeing agents, updates the situation estimate and reacts accordingly. The information that agents exchange are the observations that they acquired and that they consider relevant to the current classification proposal.

More in detail, when an agent is able to formulate locally a non-trivial situation assessment, it creates a proposal for that conclusion. The proposal is then sent to a randomly chosen team mate to start the assessment process. When an agent receives an assessment proposal, it evaluates the proposal (EVALARGS) and, if it agrees with such proposal (i.e., if its local knowledge does not refute the proposal) it forwards the message on to some other agent. In case the agent disagrees, it challenges the proposal, sending back the observations that would refute it (GETCHALLENGINGARGS). When an agent receives a challenge about a previously processed proposal, it integrates the observations that would refute the proposal into its own beliefs and reconsiders the classification. In case the additional observations do not cause a change in the assessment, the agent sends to the challenger the list of observations (GETSUPPORTINGARGS) that are needed solve the conflict; vice-versa, if the agent changes its assessment, it sends the proposal back, attaching relevant observations and asking again for agreement; the challenge continues until it is solved, or the proposal destroyed. Once a sufficient number of agents agree with the proposal, the assessment is completed.

In particular, when a agent receives a proposal message it executes the procedure `OnMsgReceived` specified in Algorithm 2. The agent first checks whether its local observations agree with the proposed assessment, say P_x (line 2). If the agent agrees with the assessment P_x, then it will just forward the message randomly to another agent, increasing $\sharp agree$ (lines 3-6). If the agent disagrees, *status* is changed to “CHALLENGE”. Observations relevant to justify the different classification are inserted into *obsList*, and the agent sends the message back to the sender agent, to search for a new agreement, that will take into account also its own observations (lines 12-14). The function GETCHALLENGINGARGS retrieves observations that are relevant to challenge the proposal from the agent’s knowledge base and will be discussed in more details in the next subsection.

The agent receiving a challenge integrates the observations in the message into its own memory and reconsiders the choice of assessment. The integration gives rise to two alternatives: (i) The additional observations did not sufficiently change the agent’s conclusions, to cause it to believe that a different assessment is preferable; (ii) the agent now believes that a different

Algorithm 2 Algorithm executed by each agent

```
    OnMsgReceived($msg$)
 1: IntegrateBeliefs(msg.obs)
 2: $agree \leftarrow$ evalArgs(msg.prop)
 3: if $msg.status == PROPOSAL$ then
 4:   if $agree$ then
 5:     $msg.\sharp agree \leftarrow msg.\sharp agree + 1$
 6:     if $\sharp$ msg.agree $<$ TTL then
 7:       send($msg$,nextAgent())
 8:     else
 9:       instantiatePlan($msg.prop$)
10:     end if
11:   else
12:     $msg.status \leftarrow CHALLENGE$
13:     $msg.obs \leftarrow$getChallengingArgs(msg.prop)
14:     send($msg$,$msg.sender$)
15:   end if
16: else
17:   /* $msg.status == CHALLENGE$ */
18:   if $agree$ then
19:     $msg.status \leftarrow PROPOSAL$
20:     $msg.obs \leftarrow$getSupportingArgs(msg.prop,msg.obs)
21:     send($msg$,origMsg.nextAgent())
22:   else
23:     if $origMsg.prevAgent() \neq null$ then
24:       $msg.obs \leftarrow$getChallengingArgs(msg.prop)
25:       send($msg$,origMsg.prevAgent())
26:     else
27:       destroy($msg$)
28:     end if
29:   end if
30: end if
```

situation is the best assessment. In case (i), the agent clears the *obsList*, changes the status back to “PROPOSAL”, and forwards it. (lines 18-21). The function getSupportingArgs retrieves the list of events whose observations are needed by the challenger to be persuaded by the assessment; its explanation is again deferred to the next subsection. In case (ii), the agent attaches any additional observations to the message and sends it back to where it received it (lines 22-25). If it was the agent that initiated the proposal, the classification proposal is destroyed (line 27). Finally, an assessment is instantiated when the number of agreeing agents with a proposed classification reaches a predefined threshold (TTL) ($\sharp agree == TTL$)(line 9)

Notice that since only contradicting observations are sent and only to the agent that disagrees, this message format scales with all key environmental and team variables. Moreover, it is possible to show (see [49]) that the protocol always terminates after a finite number of message exchange, if the agents do not acquire further observations while executing the protocol, and if the threshold on the number of agents that must agree on a proposal (TTL) does not change during the protocol execution.

In general,the above termination property holds because when a challenge is resolved, agents share all relevant information about events involved in that proposal. More in detail, as mentioned above, a challenge can be resolved in two ways: i) the proposing agent evaluates the challenge and decides the proposal is still valid, it then attaches to the message the supporting arguments in favor of the proposal and forwards it; since the knowledge base of this agent includes the information provided by the challenging agents, the two agents will share the same relevant information about this proposal and, since all agents share the same intensional knowledge (e.g the TBox), and the same Bayesian model for integration of new evidence, they will necessarily arrive to the same conclusion; ii) the proposing agent after integrating observations from the challenging agent, decides to cancel the proposal, in this case no further message will be sent about this proposal.

4.2 Distributed Knowledge Assessment

In this subsection, we focus on the knowledge base management aspects related to the distributed assessment described in the previous section, where each agent has a knowledge base representing his view on the scenario. Specifically, we focus on the operations denoted as EVALARGS, to check whether a given proposal is aligned with the knowledge base of an agent, GETCHALLENGINGARGS, to extract from the KB the knowledge that is challenging a proposal, and GETSUPPORTINGARGS to extract the knowledge to support a proposal.

The knowledge to support/challenge a proposal is given by a set of ABox assertions. We adopt the term *support set* for this set of assertions, since its role is similar to the corresponding logical notion. Therefore, the support/challenge of a proposal requires to identify a set of assertions together with the evidence accumulated on the corresponding events. Consequently, it is useful to explicitly introduce the representation of the association between assertions and events, which is also at the basis of the process of event assessment, described in subsection 3.2.

We call *arguments* the elements of a support set and denote them as $\langle C(i_1), \mathit{obsList}\rangle$ or $\langle R(i_1, i_2), \mathit{obsList}\rangle$, where C is a concept name, R is a role name, i_1 and i_2 are instance names, *obsList* is a list of sensor feature observations. The intuitive meaning of an argument $\langle C(i_1), obsList\rangle$ ($\langle R(i_1, i_2), obsList\rangle$) is that $C(i)$ ($R(i_i, i_2)$) is a valid assertion into an agent's KB, and it is supported by the observations in *obsList*. Moreover, when $obsList = \emptyset$, the argument indicates that the fact is unknown to the agent. The agent receiving a set of arguments integrates the corresponding observations into its own beliefs, as explained in Sec.3.2.

We are now in the position to discuss the required KB operations.

evalArgs takes in input an assessment proposal, specifying that s is an instance of situation S_i, and outputs *true* if the *assessment* of s is S_i, *false* otherwise.

The implementation of evalArgs is fully supported by the basic DL reasoning techniques for classification and consistency checking. Notice that, in general there might be different sets of arguments that lead to the same conclusion, however here we are only interested in checking whether the agents agree on the proposal. On the other hand, there may be several reasons for disagreement including a more general classification, a more specific and also an inconsistent one. In any of those cases a challenge is generated with a support set in order to reach a full agreement in the assessment S_i of s.

In our example in Section 3, the agent knows that s is a situation with an unidentified vessel and, therefore, he would make a different classification of s if he receives a proposal which states that `SITLOCKAVESSEL` (s). In such a case, he must compute a support set of arguments to send back to challenge the proposal. In our scenario, situations instances are typically characterized by the location, and therefore we can assume that the association problem is solved for situation instances. If this assumption is unfeasible it may still be possible to make associations, based on the underlying ability to associate observations, but this step can be rather challenging.

getChallengingArgs takes in input an assessment proposal, specifying that s is an instance of situation S_i, and outputs a support set of arguments.

This function is executed whenever an agent receives an assessment proposal and EvalArgs returns *false*. In these cases, the agent believes that the situation s is not an instance of S_i but of some other situation class S_j. The challenging arguments are thus the support set of the classification S_j plus the support set of S_i.

For example, if the agent receives `SITLOCKAVESSEL`(s), he must raise a challenge, since his assessment is $\texttt{SITUNKVESSEL}(s)$. The challenge should provide the arguments supporting the different view. In this case we have: $\{\langle \texttt{UNKNOWNVESSEL}(v_1), obsList_{v_1}\rangle, \langle \texttt{hasObject}(s, v_1), obsList_{v_1}\rangle\}$.

In the context of a complex DL ontology, retrieving the minimal support set can be rather complex, and is currently outside the scope of implemented systems. There are, however, specific proposals: for example Kalyanpur in [20] presents algorithms to extract the minimal set of TBOX and ABox assertions, that support a given conclusion. We overcome this difficulty by hard-coding the set of assertions that should be associated with each situation class, in order to obtain the corresponding support set. In this way, the support set is built by finding the instances of the assertions in the predefined support set in the agent ABox.

getSupportingArgs takes in input an assessment proposal (specifying that s is an instance of situation S_i) with a support set of arguments and outputs a new support set of arguments. This function is executed by an agent, when receiving a challenge to a proposal and the related set of arguments. The aim of getSupportingArgs is to retrieve the arguments that provide further support for the current proposal. The proposal is again forwarded with the new support set for the classification, to the challenger agent. The challenger agent, will now re-evaluate the proposal integrating also the additional evidence provided an will now agree with the proposal[3].

A couple of final notes are needed to clarify the overall method for distributed situation assessment. First, proposals are generated whenever an event triggers the classification of an instance of a relevant situation in the taxonomy. It is clearly possible that the same event generates different or even, contradictory proposals. The proposed algorithm for cooperative situation assessment relies on the fact that the accumulation of new evidence from other agents will lead either to confirm a proposal or to reject it, thus resolving any potential conflict arising from perception errors causing incorrect or insufficiently refined proposals. Second, the termination of the process is ensured when agreement between two agents is reached through two message: from the current receiver agent (i.e. challenger) to the proposer agent and, if needed, back to the current agent. This property is achieved by relying on the features of the integration of new evidence in the Bayesian model and on the collection of arguments that are associated with challenges and proposals. It turns out that directly computing the support set for the

[3]As mentioned before this is guaranteed by the fact that all agents share the same intensional knowledge and the same Bayesian model for integration of new evidence

classification by using basic DL reasoning techniques may not be sufficient to ensure the above property, and additional knowledge about the inability to make a classification must be taken into account. While this issue is solved in our ad-hoc coding of the support sets, a general solution would make the task of the ontology designer much easier.

5 Experiments and results in a seacoast surveillance scenario

This section reports experimental results of our approach evaluated in a seacoast surveillance scenario.

The proposed approach to distributed situation assessment presented in Section 4 was extensively evaluated on an abstract simulation environment [49]. Such a simulator abstracts the low level details of agent perceptions and focuses on coordination issues, thus allowing to run experiments with large number of agents (up to 120), under different environmental conditions. Results show that the approach achieves good performance, by selecting the correct plan for the current situation and keeping a very low coordination overhead, in terms of number of message and bytes per time unit. Specifically, the performance level is close to a centralized approach, while reducing the coordination overhead by an order of magnitude. Moreover, the approach was able to maintain good performance (close to the centralized benchmarking method) across a wide experimental parameter settings, such as increasing perception noise, increasing world size[4] and increasing dynamism (rate of changes that happen in the world).

After evaluating the performances of our approach in a simulated environment, we validated it in a real maritime surveillance application in collaboration with Selex-SI [47]. In the following we report results of such a validation.

5.1 Seacoast Surveillance Experimental Setting

Our approach has been evaluated on a multi-agent platform for maritime surveillance in a real-time scenario. Specifically, the multi-agent platform is a component of a general architecture of a system for harbor protection (developed by Selex-SI), which acquires the perceptions used in the situation assessment process from a set of heterogeneous sensors. In this context, agents represent patrol-ships and command and control workstations, while perceptions are information

[4]Since we fix the number of agents this results in agents having less mutual observation of the same features

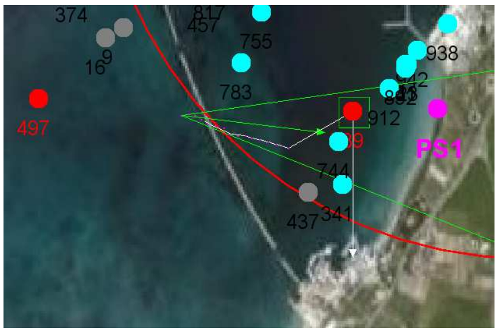

Figure 3: A splitting situation. Two boats (red circles) performed a split close to a surveillance area (red arc) and one of them is directed to the critical point (in purple).

about vessels with an uncertain estimated position and a unique (possible incorrect) ID; other inputs (such as vessel type or self-identification of the vessel) are available through database access. On the output side, the software architecture provides a graphical interface, which shows every assessed situation as a warning on a display, to allow for human decision making. The multi-agent architecture can be run on a laptop with single core processor, with small sized teams of cognitive agents (up to 5 members) and about 100 external entities moving. The number of agents is limited by the amount of radar data received and of requests of reasoning services over the ontologies.

With respect to the quality of the input data, the noise in the radar data can not be eliminated, therefore it will be always present in our experiments. In order to evaluate the behavior of our approach with different levels of sensor noise, we consider three different experimental settings: we call "high quality perception" the setting where we consider only radars' noise. Then, we reduce the quality of the feature extraction process by an exponential decay factor, which increases observations' noise in relation to the distance from the perceived event: "medium quality perceptions" refers to a factor of 0.01, and "low quality perceptions" of 0.1.

We focus on various suspect operations to be detected, among which:

splitting: the maneuver of remaining hidden staying close to another vessel, then suddenly move away directed to a critical area. (see Fig.3).

suspect approach: a suspect vessel approached by other (at least two) vessels. A suspect vessel is a vessel whose identification is not known, which is located near the border of a patrolled zone.

Other situation classes are present. For example, there are some partially specified situa-

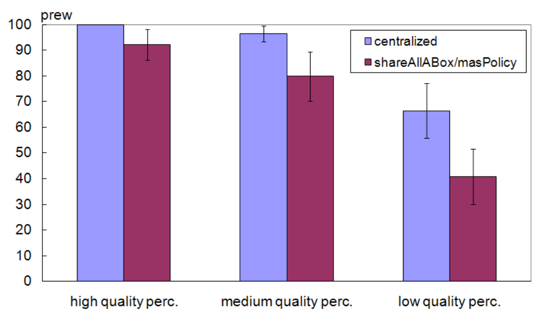

Figure 4: Performances varying quality of perceptions.

tions, which are generalizations of the above two. During the detection process of high level situation, agents may have a disagreement on different assertions, like the region were the vessel first appeared, the average direction of the vessel (which is obtained reasoning also on past perceptions) or the number of vehicles involved in the procedure.

5.2 Results

Our approach (referred as *mas_Policy*) was compared to two different strategies. The first one, *centralized*, represents an approach where the high level fusion process is performed by a specific agent (which represents a command and control center) and patrol-ships which are distributed in the sea area considered, are used only as information sources. The second benchmark policy, *share_all_ABox*, represents the multi-agent solution, where agents execute the proposed algorithm to reach agreements, solving their disagreements by attaching all assertions in the ABox.

	high quality perc.		medium quality perc.		low quality perc.	
	assert.	assert.shared	assert.	assert.shared	assert.	assert.shared
centralized	177	177 (100%)	673	673 (100%)	596	596 (100%)
shareAllABox	110	110 (100%)	151	151 (100%)	423	423 (100%)
mas_Policy	110	12 (11%)	148	31 (21%)	439	123 (28%)

Table 1: Measuring information locality. The number of assertions and of shared assertions of each agent is shown.

Figure 4 shows the quality of the overall team assessment, comparing the three policies, under varying noise conditions. Specifically, the x axis reports the perception quality, while the y-axis reports an indicator of performances *prew* (% of reward). *prew* uses a utility function which gives rewards and costs to correct/wrong assessments and partially correct/wrong ones.

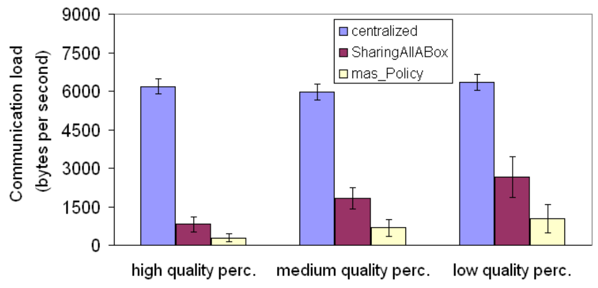

Figure 5: Communication costs of 3 different agent policies.

Then, $prew = \frac{u^*}{u_{max}}$, where u^* is the utility that the team achieves through its classifications, and u_{max} is the maximum achievable utility, considering the set of classifications of the *centralized* policy, in the "high quality" configuration, as ground truth (correct assessment). In this case, the *mas_Policy* and *share_All_ABox* are indistinguishable, because all the relevant observations are shared in both policies.

These results highlight three aspects: first, the *centralized* approach does not get always the best result, because it suffers from the quality of received observations. Second, the centralized approach always out-performs the multi-agent policy, as expected. This happens because the *centralized* approach has always instantaneous access to all available observations. Third, in case of low level perceptions the quality of results of the multi-agent policy significantly decreases: in this case, the approach suffers from the fact that a small number of agents is used, and loosing some observations may cause an incorrect classification. Increasing the density of agents would improve performances, as it is shown in the evaluation of the distributed situation assessment protocol [49]; also, exploiting the locality of information to forward proposals to the best informed agents would enhance the performances with respect to the actual random forwarding policy.

In addition, we evaluated communication costs, in the three different policies. Obviously, a reduced communication is beneficial not simply in terms of traffic load reduction, but mainly in terms of robustness to communication failures and security. We analyzed the amount of bandwidth used, which is a better indicator of the costs than the number of messages, because messages have a different size in the various policies, due to the possible presence (and the number) of observations attached. We considered a fix size of 100 bytes for each observation. Figure 5 shows these results: the x-axis reports the quality of observations, while the y-axis indicates the bandwidth used, measured in bytes per second. We can observe that *centralized* has severe bandwidth requirements, while the two multi-agent approaches significantly reduce the amount of bandwidth used. The used bandwidth of the approaches with distributed fusion is slightly influenced by the quality of perceptions, since the number of conflicts in the assessment increases as perception quality decreases. Even for the worst case, with very noisy observations, the policies with distributed fusion largely reduce (more than half) the amount of necessary bandwidth. Moreover, comparing the *share_all_ABox* and the *mas_Policy* approaches, the second one further reduces (about one order of magnitude less than *centralized*) the bandwidth used, because the minimal amount of events to be challenged is evaluated.

A major reason to compute and share minimal information among agents is to preserve information locality. Maintaining information locality is a key aspect to reduce the amount of computation each agent must perform, thus allowing to scale the systems to large number of agents and observations typical of real applications. To evaluate this attribute, in Table 1 we related it to the amount of ABox assertions which are shared among the agents. In particular, we use the ratio between the number of assertions shared and the total amount of assertions in the KB (both averaged among the agents). We count as ABox assertions, only those that derive from observations (and not, for example, static facts). We consider an assertion to be shared, whenever an observation related to that assertion is sent. Results are shown in the table, where we may conclude that, with our policy, each agent shares less than 30% of its KB. When assertions are not shared, they are either useless for current conclusions, or they are already agreed upon (perhaps for different reasons) among the agents. Obviously, we see also that the *share_all_ABox* and the *centralized* policy are worse with respect to locality, since they share all the local assertions.

In conclusion, the experimental evaluation in the maritime scenario shows that the proposed approach to distributed situation assessment achieves the design goal of keeping the information at the right place, since the quality classifications are comparable to the centralized case, while avoiding unnecessary communication.

6 Conclusions

The work presented in this paper addresses the key aspects of Situation Assessment in scenarios where several actors have to monitor many events of interest. The target applications for the above class of systems can be manifold, for example air traffic control, emergency operation after a disaster, or maritime surveillance that is specifically taken here as a case study. More in particular, our goal is to achieve a distributed solution to Situation Assessment, that overcomes the limitations in terms of robustness, scalability and security of a centralized approach.

The approach presented in this paper is an important step forward. Specifically, the key contributions of this work are: i) a general framework for situation assessment that exploits the inferential capabilities of ontological representations; ii) a multi-agent approach to Situation Assessment that, by sharing a minimal amount of knowledge using dialogues between agents, achieves a classification of situations comparable with the centralized case, while substantially reducing the communication; iii) a validation of our approach in a real case scenario of maritime surveillance showing that the design goals of the distributed approach can be successfully achieved.

Several aspects of the overall process could be improved. We are concerned specifically on the agent interaction leading to high level assessment. For example, a future work in this area will focus on the possibility to compute the set of justifications for each relevant situation class, using results from belief revision theory [20]. Another possible direction to investigate is whether an agent may solve also inconsistencies (and not only disagreements) in its own KB by asking for an agreement to team mates, attaching its subset of inconsistent assertions.

Acknowledgment

The work has been carried out within the project "Iniziativa Software" supported by Finmeccanica and CINI. The authors would like to thank Luca Izzotti and Francesco Canini for their support in the earlier stages of the project.

References

[1] Sesar definiton phase d1- air transport framework : The current situation. Technical report, SESAR Consortium, 2006.

[2] Vivek Bharathan and John R. Josephson. An abductive framework for level one information fusion. In *Proceedings of the 9th International Conference on Information Fusion (IF-06)*, pages 1–7, Florence, Italy, July 2006. IEEE.

[3] Neil A. Bomberger, Bradley J. Rhodes, Michael Seibert, and Allen M. Waxman. Associative learning of vessel motion patterns for maritime situation awareness. In *Proceedings of the 9th International Conference on Information Fusion (IF-06)*, pages 1–8, Florence, Italy, July 2006. IEEE.

[4] John F. Buford, Gabriel Jakobson, and Lundy Lewis. Peer-to-peer coupled agent systems for distributed situation management. *Information Fusion*, 11:233–242, 2010.

[5] Gerard T. Capraro, Alfonso Farina, Hugh Griffiths, and Michael C. Wicks. Knowledge-based radar signal and data processing. *IEEE Signal Processing Magazine*, 23:18–29, January 2006.

[6] Thomas M. Cioppa, Thomas W. Lucas, and Susan M. Sanchez. Military applications of agent-based simulations. In *Proceedings of the 36th Conference on Winter Simulation(WSC-04)*, pages 171–180. Winter Simulation Conference, December 2004.

[7] Markus Dietl, Jens S. Gutmann, and Bernhard Nebel. Cooperative sensing in dynamic environments. In *Proceedings of the International Conference on Intelligent Robots and Systems (IROS-01)*, pages 1706–1713, Maui, HI, USA, November 2001. IEEE.

[8] Zhongli Ding. *Bayesowl: a Probabilistic Framework for Uncertainty in Semantic Web.* PhD thesis, University of Maryland at Baltimore County, Catonsville, MD, USA, 2005.

[9] T Eiter and G. Gottlob. On the complexity of propositional knowledge base revision, updates and counterfactuals. *Artificial Intelligence*, 57:227–270, 1992.

[10] Alessandro Farinelli, Luca Iocchi, Daniele Nardi, and Vittorio Ziparo. Assignment of dynamically perceived tasks by token passing in multi-robot systems. *Proceedings of the IEEE, Special issue on Multi-Robot Systems*, 94:1271 – 1288, 2006.

[11] Alessandro Farinelli, Luca Iocchi, Daniele Nardi, and Vittorio Amos Ziparo. Task assignment with dynamic perception and constrained tasks in a multi-robot system. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA-05)*, pages 1523–1528, Barcelona, Spain, April 2005. IEEE.

[12] Alessandro Farinelli, Daniele Nardi, Paul Scerri, and Alberto Ingenito. Dealing with perception errors in multi-robot system coordination. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI-07)*, pages 2091–2096. AAAI Press, January 2007.

[13] Sofia Giompapa, Alfonso Farina, Fulvio Gini, Antonio Graziano, and Riccardo Di Stefano. A model for a human decision-maker in a command and control radar system: Surveillance tracking of multiple targets. In *Proceedings of the 9th International Conference on Information Fusion (IF-06)*, pages 1–8, Florence, Italy, July 2006. IEEE.

[14] Eric Gregoire and Sbastien Konieczny. Logic-based approaches to information fusion. *Information Fusion*, 7:1–156, March 2006.

[15] Volker Haarslev and Ralf Möller. Racer: A core inference engine for the semantic web. In *Proceedings of the 2nd International Workshop on Evaluation of Ontology-based Tools (EON-03), located at the 2nd International Semantic Web Conference (ISWC-03)*, pages 27–36, Sanibel Island, FL, USA, October 2003. http://www.racer-systems.com/.

[16] David L. Hall and James Llinas. *Handbook of Multisensor Data Fusion.* CRC Press, 2001.

[17] Jochen Heinsohn. Probabilistic description logics. In *Proceedings of the 10th Annual Conference on Uncertainty in Artificial Intelligence (UAI-94)*, pages 311–318, San Francisco, CA, USA, July 1994. Morgan Kaufmann.

[18] Michael N. Huhns. Re-architected it systems for the u.s. navy. In *invited talk at International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS-09)*, Budapest, Hungary, May 2009.

[19] Manfred Jaeger. Probabilistic reasoning in terminological logics. In *Proceedings of the 4th International Conference on Principles of Knowledge Representation an Reasoning (KR-94)*, pages 305–316, Bonn, Germany, May 1994. Morgan Kaufmann.

[20] Aditya Kalyanpur. *Debugging and Repair of OWL Ontologies.* PhD thesis, University of Maryland, College Park, MD, USA, 2006.

[21] Holger Knublauch, Ray W. Fergerson, Natalya F. Noy, and Mark A. Musen. The protégé OWL plugin: an open development environment for semantic web applications. In *Poceedings of the International Semantic Web Conference (ISWC-04)*, volume 3298, pages 229–243, Hiroshima, Japan, 2004. Springer.

[22] Mieczyslaw M. Kokar, Christopher J. Matheus, and Kenneth Baclawski. Ontology-based situation awareness. *Information Fusion*, 10:83–98, 2009.

[23] Daphne Koller, Alon Levy, and Avi Pfeffer. P-classic: A tractable probabilistic description logic. In *Proceedings of the 14th National Conference on Artificial Intelligence(AAAI-97)*, pages 390–397, Providence, RI, USA, July 1997. AAAI Press.

[24] Claire Laudy, Juliette Mattioli, and Nicolas Museux. Cognitive situation awareness for information superiority. In *IST Panel on Information Fusion for Command Support*, pages 1–12, November 2005.

[25] Rikard Laxhammar. Anomaly detection for sea surveillance. In *Proceedings of the 11th International Conference on Information Fusion (IF-08)*, pages 1–8. ISIF, IEEE, June 2008.

[26] Eric G. Little and Galina L. Rogova. Designing ontologies for higher level fusion. *Information Fusion*, 10(1):70–82, January 2009.

[27] James Llinas, Christopher L. Bowman, Galina Rogova, Alan N. Steinberg, Ed Waltz, and Franklin E. White. Revisiting the jdl data fusion model ii. In *Proceedings of the 7th International Conference on Information Fusion (IF-04)*, pages 1218–1230, Stockholm, Sweden, June 2004. International Society of Information Fusion.

[28] Thomas Lukasiewicz. Expressive probabilistic description logics. *Artificial Intelligence*, 172(6-7):852–883, 2008.

[29] Alexei Makarenko and Hugh F. Durrant-Whyte. Decentralized data fusion and control algorithms in active sensor networks. In *Proceedings of the 7th International Conference on Information Fusion (IF-04)*, pages 479–486, Stockholm, Sweden, June 2004. ISIF.

[30] Fulvio Mastrogiovanni, Sgorbissa Antonio, and Renato Zaccaria. A distributed architecture for symbolic data fusion. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI-07)*, pages 2153–2158, Hyderabad, India, January 2007. AAAI Press.

[31] Christopher J. Matheus, Kenneth Baclawski, and Mieczyslaw M. Kokar. Derivation of ontological relations using formal methods in a situation awareness scenario. In *Proceedings of SPIE Conference on Multi-sensor, Multi-source Information Fusion: Architectures, Algorithms, and Applications*, pages 298–309, Orlando, FL, USA, April 2003. SPIE.

[32] Christopher J. Matheus, Mieczyslaw M. Kokar, and Kenneth Baclawski. A core ontology for situation awareness. In *Proceedings of the 6th International Conference on Information Fusion (IF-03)*, pages 545–552, Cairns, Australia, July 2003. ISIF.

[33] Christopher J. Matheus, Mieczyslaw M. Kokar, Kenneth Baclawski, John Salerno, et al. Lessons learned from developing sawa: A situation awareness assistant. In *Proceedings of the 8th International Conference on Information Fusion (IF-05*, pages 1–9, Philadelphia, PA, USA, July 2005. IEEE.

[34] Deborah L. McGuinness and Frank van Harmelen. OWL web ontology language overview. World Wide Web Consortium, Recommendation http://www.w3.org/TR/owl-features/, February 2004.

[35] Nicolas Museux, Juliette Mattioli, Claire Laudy, and Helene Soubaras. Complex event processing approach for strategic intelligence. In *Proceedings of the 9th International Conference on Information Fusion (IF-06)*, pages 1–8, Florence, Italy, July 2006. IEEE.

[36] Daniele Nardi and Ronald J. Brachman. An introduction to description logics. In *The Description Logic Handbook: Theory, Implementation and Applications*, pages 1–40. Cambridge University Press, 2003.

[37] Maria Nilsson, Joeri van Laere, Tom Ziemke, and Johan Edlund. Extracting rules from expert operators to support situation awareness in maritime surveillance. In *Proceedings of the 11th International Conference on Information Fusion (IF-08)*, pages 1–8. ISIF, IEEE, June 2008.

[38] C. A. Ntuen and A. R. Watson. Workload prediction as a function of system complexity. In *Proceedings of the 3rd Symposium on Human Interaction with Complex Systems (HICS-96)*, pages 96–100, Dayton, OH, USA, August 1996. IEEE Computer Society.

[39] David V. Pynadath and Milind Tambe. The communicative multiagent team decision problem:analyzing teamwork theories and models. *Journal of Artificial Intelligence Research (JAIR)*, 16:389–423, 2002.

[40] Branko Ristic, Barbara La Scala, Mark Morelande, and Neil Gordon. Statistical analysis of motion patterns in ais data: Anomaly detection and motion prediction. In *Proceedings of the 11th International Conference on Information Fusion (IF-08)*, pages 1–7. ISIF, IEEE, June 2008.

[41] Maria Riveiro, Goran Falkman, and Tom Ziemke. Improving maritime anomaly detection and situation awareness through interactive visualization. In *Proceedings of the 11th International Conference on Information Fusion (IF-08)*, pages 47–54. ISIF, IEEE, June-July 2008. Best Student Paper Award.

[42] Galina L. Rogova, Peter D. Scott, and Carlos Lollett. Reasoning about situations in the early post-disaster response environment. In *Proceedings of the 9th International Conference on Information Fusion (IF-06)*, pages 1–8, Florence, Italy, July 2006. IEEE.

[43] Matt Rosencrantz, Geoffrey Gordon, and Sebastian Thrun. Decentralized sensor fusion with distributed particle filters. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI-03)*, pages 493–500, Acapulco, Mexico, August 2003. Morgan Kaufmann.

[44] Maayan Roth, Reid Simmons, and Manuela Veloso. Reasoning about joint beliefs for execution-time communication decisions. In *Proceedings of 4th International Conference on Autonomous Agents and Multiagent Systems (AAMAS-05)*, pages 786 –793, Utrecht, The Netherlands, July 2005. ACM.

[45] Paul Scerri, Alessandro Farinelli, Steven Okamoto, and Milind Tambe. Allocating tasks in extreme teams. In *Proceedings of 4th International Conference on Autonomous Agents and Multiagent Systems (AAMAS-05)*, pages 727–734, Utrecht, The Netherlands, July 2005. ACM.

[46] Robert C. Schrag, Takikawa Masami, Paul Goger, and James Eilbert. Performance evaluation for automated threat detection. *Journal of Advances in Information Fusion*, pages 77–98, December 2007.

[47] Giuseppe Paolo Settembre, Alessandro Farinelli, Daniele Nardi, Roberta Pigliacampo, and Mirco Rossi. Solving disagreements in a multi-agent system performing situation assessment. In *Proceedings of the 12th International Conference on Information Fusion (IF-09)*, pages 1–8, Seattle, WA, USA, July 2009. International Society of Information Fusion.

[48] Giuseppe Paolo Settembre, Roberta Pigliacampo, and Daniele Nardi. Agent approach to situation assessment. In Filipe, Fred, and Sharp, editors, *Proceedings of the 1st International Conference on Agents and Artifical Intelligence (ICAART-09)*, pages 287–290, Porto, Portugal, January 2009. INSTICC Press.

[49] Giuseppe Paolo Settembre, Paul Scerri, Alessandro Farinelli, Katia Sycara, and Daniele Nardi. A decentralized approach to cooperative situation assessment in multi-robot systems. In *Proceedings of the International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS-08)*, pages 31–38, Estoril, Portugal, May 2008. International Foundation for Autonomous Agents and Multiagent Systems.

[50] Evren Sirin, Bijan Parsia, Bernardo Cuenca Grau, Aditya Kalyanpur, and Yarden Katz. Pellet: A practical owl-dl reasoner. *Web Semantics: Science, Services and Agents on the World Wide Web*, 5:51–53, June 2007.

[51] Ashley W. Stroupe, Martin C. Martin, and Tucker Balch. Distributed sensor fusion for object position estimation by multi-robot systems. In *Proceedings of the International Conference on Robotics and Automation (ICRA-01)*, volume 2, pages 1092–1098, Seoul, Korea, May 2001. IEEE.

[52] Katia Sycara, Robin Glinton, Bin Yu, Joseph Andrew Giampapa, Sean R. Owens, Michael Lewis, and Grindle LTC Charles. An integrated approach to high level information fusion. *Information Fusion*, 10:25–50, January 2009.

[53] David J. Walton, Eugene P. Paulo, Christopher J. McCarthy, and Ravi Vaidyanathan. Modeling force response to small boat attack against high value commercial ships. In *Proceedings of the 37th Conference on Winter simulation (WSC-05)*, pages 988–991. Winter Simulation Conference, December 2005.

[54] Bin Yu, Paul Scerri, Katia Sycara, Yan Xu, and Michael Lewis. Scalable and reliable data delivery in mobile ad hoc sensor networks. In *Proceedings of the 5th International Joint Conference on Autonomous Agents and Multi Agent Systems (AAMAS-06)*, pages 1071–1078, Hakodate, Japan, May 2006. ACM.