Multi-Agent Temporal Plan Monitoring Perspectives in Coalition Context Based on Dependence Networks

Mohamad K. Allouche, **Jean Berger**

Mohamad.Allouche@drdc-rddc.gc.ca, Jean.Berger@drdc-rddc.gc.ca

Defense Research Development Canada, Quebec, CANADA

Abstract—*In a coalition context, there are tasks synchronization issues since missions are carried out by different players (agents) distributed over the environment. In [1], a general framework has been proposed to monitor the execution of different sub-plans by a group of agents in a coalition context. The monitoring is performed by using a temporal constraint propagation mechanism in a coordinated plan obtained by the fusion of the different sub-plans. Although, the execution of the different sub-plans is decentralized, the monitoring of the coordinated plan is centralized. In this work, we attempt to identify the main requirements to decentralize the monitoring task to be performed by a multi-agent system in a coalition context.*

1. INTRODUCTION

Plan execution in a coalition context requires a close coordination of different players' (agents') activities to cope with the different constraints imposed by the global mission. Since the agents are distributed in the environment, we assume that each agent has partial knowledge of the other agents' activities. The success of the global mission depends on the result of the plan execution, i.e. the execution of all sub-plans by the different agents. It is then mandatory to have a coordination mechanism to coordinate the different agents' activities. This mechanism would allow the synchronisation of tasks and also an efficient agent communication. In previous work [1], we introduced a general framework that allows the monitoring of a plan, called coordinated plan, formed by the fusion of different sub-plans that need to be executed by a group of agents, members of a coalition. In such a context, each agent executes independently a sub-plan and communicates some execution results to the monitoring system, which integrates these results to the coordinated plan and alerts the commander whenever there is a risk of failure of the plan due to a violation of a temporal constraint. It is also important to mention that agents do not take part in the monitoring of their activities. They only communicate some execution results to the monitoring system. However, in some coalition contexts, it is hard to centralize the monitoring task due to political, tactical, technological or mission related reasons. Providing the agents with more autonomy means that they can participate to the monitoring process. In this work, we will identify the main requirements for a group of agents to monitor their own activities. The coordination mechanism is based on the notion of dependence networks.

This work should not be confused with previous reported contributions aimed at simultaneously combining scheduling, execution control or replanning tasks. In the current problem setting, it is assumed that planning and scheduling processes occur concurrently but separately from the execution monitoring process and that required information level on plans and scheduling can be realistically made available to agents (given bandwidth constraints, security and operations granularity level). Taking place during execution, monitoring mainly consists in detecting significant or unexpected deviations from a desired execution state. The proposed effort explicitly focuses on the temporal monitoring phase of plan execution, mitigating synchronization and latency problems arising from unexpected events which in turn could be further exploited to revisit plans and schedules at different stages.

2. TEMPORAL PLAN MONITORING

In this work, we are mainly interested in monitoring the synchronization between actions. In a simple plan where a task is merely a ranked list of actions, only precedence constraints need to be monitored and they do not change throughout the plan execution. Monitoring graph-structured plans is a more challenging task, since the order of execution actions may change. A graph-structure plan is a collection of actions with temporal constraints. Each action of the plan is represented by couple of nodes ($start_i$, end_i) and the arcs of the graph are the temporal constraints, expressed as intervals of integers, between the nodes ($T(e_i, e_j) = [a_{ij}, b_{ij}]$). If a temporal constraint is expressed on a couple of node belonging to the same

action, it constrains its duration (minimum and maximum authorized duration). If a temporal constraint is expressed between nodes belonging to two different actions, it has a role of synchronization between these two actions. The exact form of these temporal constraints is detailed in [1]. The ordering of actions is influenced by the decision-making process, which always tries to reach an optimal solution for the plan, or at least a feasible one.

The temporal monitoring of plan allows continual checking and updating of temporal constraints. Since the plan has a graph structure, each time an action starts of finishes, several actions may be potentially chosen to be executed in the next step, that is, different paths may be followed in the graph at each step. The choice of the most appropriate path should in our opinion be the result of a human decision-making process.

The duration of an action may be variable from one execution to another. The monitoring task should check and ensure that action durations do not violate the temporal constraints. It should also propagate the real durations of actions through the graph in order to manage synchronization with other actions. The duration of an action may impact the durations and synchronization of other actions that need to execute in the future.

We propose the following monitoring process including four possible states:

- *Initial state* S_0 (this state is visited once at the beginning of plan execution)

Initial execution time and first action to be executed are given by the decision-maker

- *Monitoring state* S_i (this state is visited after the execution of each action ($start_i$, end_i))

Compute the maximum authorized waiting time (*timeout* TO_i) for next action to execute

Compute potential actions to execute (*candidate list* CL_i)

Update the temporal constraint $T(start_i, end_i)$ with the real duration of action ($start_i$, end_i)

Minimize the current graph

- *Success state* S_+ (this state is reached if all actions are executed and no temporal constraint is violated)

- *Failure state* S_- (this state is reached if the current minimal graph is not coherent)

The transition diagram is shown in Figure 1.

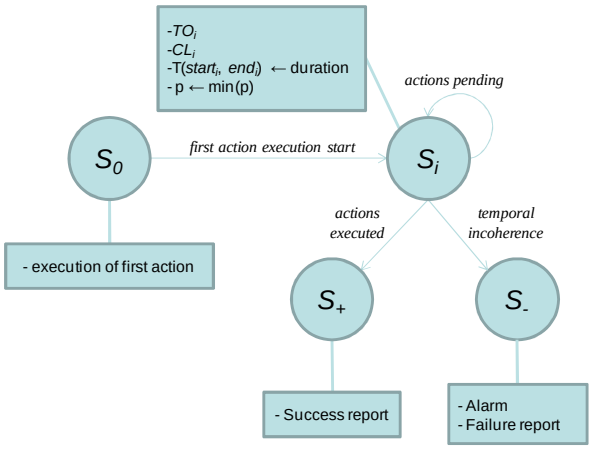

Figure 1: Transition diagram of the monitoring process

The process starts by the execution of the first action. The timeout TO_i and candidate list CL_i are computed. Each step of the process is executed at the end or the start of an action in the candidate list before the expiration of the timeout. If an action represented by the two nodes ($start_i$, end_i) finishes then the constraint $T(start_i, end_i)$ receives the real duration of the action execution. This new information is then propagated by minimizing the graph. The previous step is repeated until reaching a temporal incoherence in the graph or a successful execution of all actions. In the first case, an alarm and a failure report are sent to the user. In the second case, a success report is sent to the user.

It is important to mention that the expiration of the timeout or the occurrence of a node that does not belong to the candidate list is sufficient to cause a plan failure because they cause a temporal constraint violation. A plan will succeed when all its actions are executed successfully without violating any of the temporal constraints.

2.1 Timeout TO_i

The timeout is computed and armed after each step of the monitoring process (action start/end). Let e_i be the last occurred node in the graph (beginning or end of an action), the timeout TO_i is computed as the following: for all $T(e_i, e_j) = [a_{ij}, b_{ij}] \mid b_{ij} \geq 0$, $TO_i = \min_j(b_{ij})$. Since $b_{ij} = \max(T(e_i, e_j))$ then $TO_i = \min_j(\max(T(e_i, e_j)))$. The timeout is simply the minimum of maximum authorized times for all nodes e_j potentially to occur. In fact, past this time at least one node e_j will be considered as too late to occur after e_i.

2.2 Candidate list CL_i

The candidate list CL_i contains all the nodes that are authorized to occur after e_i. At least one node from this list is expected to occur before the timeout expiration. Suppose that $T(e_i, e_s)$ is the constraint that allowed computing the timeout after the occurrence of e_i. All $e_j \mid T(e_i, e_s) \cap T(e_i, e_j) \neq \emptyset$, may occur (candidates) after e_i. This is true because they may occur before the timeout expiration and without violating the corresponding temporal constraints.

3. COORDINATION CHALLENGES AND ISSUES

A decentralized monitoring system where each agent locally monitors a sub-plan requires direct communication between agents and a coordination mechanism in order to monitor the coordinated plan. Each agent will have to perform the monitoring process depicted in Figure 1 for a local sub-plan.

3.1 Dependence networks

In our opinion there are at least three situations where a direct communication between several agents becomes necessary:

- Need to execute the same action in different sub-plans monitored by several agents;
- Need to update a temporal constraint between two actions belonging to two different sub-plans monitored by two different agents.
- Need to choose a new action in the candidate list, which contains actions belonging to different sub-plans.

Agents' activities need to be coordinated by taking these interaction situations into account. A group of agents can build a set of dependence networks [2,3,5] that allow each agent to be related to the other agents. Usually, these dependence networks should be built based on dependence relations between tasks. More generally, they should relate a set of agents, a set of tasks and a set of resources. For instance, a dependence network can specify that an agent needs a specific resource to execute a specific task. If the resource belongs to another agent, then there is a dependence relation between the two agents. More recently, dependence networks were defined as relating agents regarding goals [6]. They allow determining for any set of goals, the set of agents an agent depends on. Making use of a dependence network means that each time an agent depends on another agent to execute a specific task or to reach a specific goal, there is need for direct communication between the corresponding agents. Agents may be cooperative or competitive [7]. A cooperative agent will always try to help other agents reach their goals. Competitive or selfish agents will always try to reach their own goals regardless of the other agent's goals. In some cases, they may even prevent others from reaching their goals if these goals conflict with their own goals. The dependence networks can be used in both cases; however, building such dependence networks can be an issue in the competitive context. In our context, we consider that agents are part of a coalition and they are cooperative.

3.2 Direct communication

In the centralized framework no direct communications were needed between agents. Each agent initially needed to communicate with the monitoring system, which implicitly played the role of agent coordinator. However, in a decentralized framework, agents need to coordinate their activities with one another resorting on peer-to-peer communication explicitly. Each agent need to know what information to communicate to the other agents and when. In our problem, this information is dictated by graph-structure plans and the agent's dependence network. Accordingly, an agent needs to communicate the following information content to another agent:

- A new timeout;
- A new candidate list;
- A new temporal constraint when an action starts/finishes.

Information-sharing is assumed to be supported by an appropriate communication language based upon a predefined ontology for agent communication, and then exploited by a hierarchy of suitable communication protocol layers to ensure proper and successful information transfer. The purpose of communication aims at synchronizing multiple and concurrent processes or agents to mitigate destructive interference that may result from an interaction (e.g. state consistency related to concurrency control issues). Such classical languages are based on speech act theory [8, 9] which perceives communication as base-level action in the physical world. The theory considers speech acts as actions (performatives) conveying intention and involve a variety of conditions [8] to successfully complete their

execution. KQML (Knowledge Query Manipulation Language), FIPA-ACL (Agent communication language) [10], and DAML (DARPA Agent Markup Language) [11] are agent communication languages based on a specified machine-understandable ontology.

4. APPLICATION TO COMBAT SEARCH AND RESCUE (CSAR)

In this section we illustrate the use of dependence networks in a CSAR (Combat Search and Rescue) mission in the context of the North Atlantis scenario. This fictitious scenario was used in 2000 as an exercise by the Canadian Forces Command and Staff College (CFCSC) to teach the Canadian Forces Operational Planning Process and allow the sharing of operational knowledge and expertise among the CFC staff and students. The choice of this type of application is motivated by three main reasons: First, the temporal constraints are key elements in planning and the execution of CASR missions. Second, a CASR mission requires different types of assets distributed over the environment. Finally, a close coordination of the activities of the different involved assets is a key factor for the mission success.

A crisis has developed over the past 10 days on the continent of Atlantis. It is the result of years of growing tensions since the fall of 1999, and has now erupted into armed conflict. Individual country studies are provided as well as a document entitled "The Manghalour Peninsula Crisis," to provide the detailed background to the crisis.

As a result of the critical situation between ORANGELAND/REDLAND and BLUELAND, the UN requested the Alliance Council to consider a military response to help resolve the crisis.

On 12 June, the second day following the commencement of the Alliance joint operations to secure Blueland territory and expel any Coalition invasion forces, a UK Royal Air Force (RAF) Tornado call-sign HAWK27, conducting an electronic countermeasures and reconnaissance (ECR) mission, was shot down over the Celtic Straits by a surface-to-air missile (SAM) at 1608 hours. Shortly after the location of the downed crew was known, a CH-124 Sea King helicopter from Wahhabe Airbase, with a crew of five, was sent to recover and evacuate the Tornado aircrew. At approximately 1800 hours, in the process of extraction of the downed Tornado crew, the Sea King crashed.

A CSAR mission represents many dynamic challenges for the mission planners to locate and extract lost crew members in a hostile environment. Various elements must be taken into account, which may be predictable such as the friendly elements of detect and rescue, or unpredictable such as the enemy elements of detect and destroy. Usually, mission planners use air and ground picture inputs to make their decisions.

4.1 Agents and sub-plans

The PC (Package Commander) designed a plan to meet two critical mission requirements: air superiority and CSAR extraction. This plan should also allow SEAD (Suppression of Enemy Air Defenses) and air superiority established at least 15 minutes ahead, ABR (Airborne Regiment) delay/harassment occurring approximately 10 minutes ahead and CAS (Close Air Support) in place five (5) minutes ahead of the CSAR helicopter extraction area TOT (Time On Target). The plan includes sub-plans assignment to allocated assets (agents) to counter the enemy threat for "efficiency and safety" and to gain and maintain local air superiority. In the following each group of aircraft is considered as a single agent for the sake of simplicity:

a) 4 x CF-18 – SIERRA 1-4 – sweep ingress and egress route and provide CAP (Combat Air Patrol) over CSAR pick-up area (above cloud);
b) 4 x CF-18 – ECHO 1-4 – escort CSAR helicopters inbound and outbound to the pick-up area (below cloud with assets);
c) 4 x CF-18 – BOMBER 1-4 – BAI (Battlefield Air Interdiction)(cluster munitions) pre-strike harass/delay of the LOC (Lines Of Communication) and ABR main body forward elements;
d) 4 x ECR Tornado – JAMMER 1-4 – SEAD of Eaglevista SAMs from five (5) minutes before to five (5) minutes after mission aircraft enter AOO (Area Of Operations);
e) 2 x ECR Tornado – ZAP 1-2 – SEAD of ABR SA 8 ahead of sweep aircraft and remain on station until all mission aircraft out of SA 8 range;
f) 2 x CH 53 – RESCUE 1-2 - each with maximum JTF2 (Joint Task Force)(less seven (7) for downed crews) such that each can carry out mission if other helicopter aborts;

g) 1 x AC-130 (Gunship) – GUNNER - for CAS in the target area;

h) 1 x Predator UAV – PREDATOR 1 – to locate and monitor pick-up area;

i) 1 x Predator UAV – PREDATOR 2 – to locate and assist in targeting ABR forward elements.

In this plan, the assets used such as the 4 x CF-18 in task c) for instance, correspond to an agent that is responsible to execute this task. BOMBER is the name of this agent.

4.2 Temporal constraints

From the description of the plan and related tasks, it is possible to deduce different temporal constraints between the different sub-plans.

- f) during b);
- f) during a);
- g) starts at least 5 min. before f) starts;
- c) starts at least 10 min. before f) starts;
- d) starts at least 15 min. before f) starts;
- e) starts at least 15 min. before f) starts;
- f) during h);
- e) during i);
- f) during e);
- e) overlaps a);
- d) starts at least 5 min. before and continues at least 5 min. after e) starts;
- f) must be performed between 10 and 20 min.

The global plan that must be executed by the coalition agents is shown in Figure 2. In practice, each agent will have a local representation of this global plan, which will be updated through the interaction with other agents.

It is worth mentioning that required global plan information to be shared by agents only represents partial high-level elements of the team's plan. Agents do not need to share detailed knowledge about respective tasks and sub-tasks. Sharing information on specific resource utilization about task execution is not desirable either. This is consistent with usually expected information-bounded coalition constraints, in which different nations may only/willingly share limited information about their operations. Initially having high-level global plan knowledge, agents progressively update task status through interactions with other agents during execution, and in so doing further capture revisited team plans.

4.3 Dependence networks for decentralized monitoring

In a centralized monitoring context, each agent can execute a sub-plan and report the execution results to a centralized monitoring system, which will coordinate the execution of the different sub-plans by the different agents. In a decentralized context where agents may have a certain degree of autonomy, each agent is responsible for the monitoring of the local sub-plan and needs to coordinate this execution with the other agents.

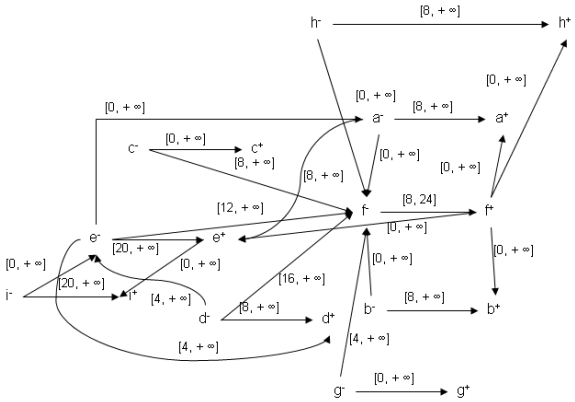

Figure 2: Global plan composed of several sub-plans with temporal constraints

Based on the temporal constraints between the sub-plans, each agent A_i with a sub-plan P_i can build a dependence network dep_i that includes all the agents A_j having sub-plans P_j where P_i and P_j are related with a temporal constraint. For the sake of simplicity, we will build the dependence network of agent RESCUE with sub-plan f) in charge of the extraction of the downed crews. This dependence network is depicted in Figure 3.

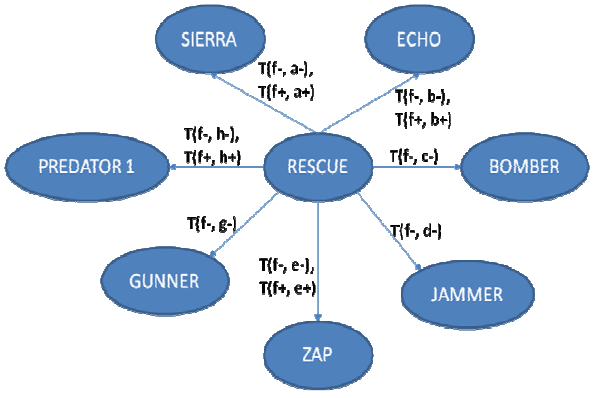

Figure 3: Dependence network of agent RESCUE

The execution of the mission is given by the following timeline. Each element of this timeline

includes the hour, the aircraft name and a summarized description of its activity.

0600

- Magic (1 x AWACS) on station north at 5800N/2600W

0700

- Predator 1 (1 x UAV) take-off from Bendeguz
- Exxon 1 (1 x KC-135 AAR (Air-to-Air Refuel)) on station Track A 5830N 2400W

0800

h- Predator 1 on station 5700N/2700W, detect and track downed aircrew, detect and track enemy forces

0900

- Predator 2 (1 x UAV) take-off from Bendeguz

0945

- Echo 1-2 (2 x CF-18) take-off from Bendeguz

i- Predator 2 on station 5730N/2630W, airborne backup

1000

- Spook (1 x JSTARS) on station 5720N/2400W, detect and track enemy forces (ABR and SA-8 TELs)
- Jammer 1-4 (4 x Tornado ECR) take-off from Bendeguz
- Bomber 1-4 (4 x CF-18) take-off from Bendeguz
- Exxon 2 (1 x KC-135 AAR) on station Track B 5830N 2400W

1015

- Echo join AAR Track B

1030

- Zap 1-2 (2 x Tornado ECR) take-off from Bendeguz
- Jammer 1-4 AAR Track A
- Echo 1-2 departs AAR Track B
- Bomber joins AAR Track B
- Gunner (1 x AC-130) take-off from Nitric

1035

- Rescue 1-2 (2 x CH-53) take-off from Nitric

1045

- Echo 3-4 (2 x CF-18) take-off Bendeguz

1100

- Jammer 1-4 departs AAR
- Zap 1-2 joins AAR Track A
- Sierra 1-4 (4 x CF-18) take-off from Bendeguz
- Bomber 1-4 departs AAR Track B

1110

- Predator 2 departs north hold to reposition to 5640N/2700W
- Predator 3 (1 x UAV) takes-off from Bendeguz

1115

- Rescue 1-2 turn south along coast

b- Echo 1-2 join CSAR for close escort

- Echo 3-4 join AAR Track B

Zap 1-2 departs AAR Track A1120

d- Jammer 1-4 push from 5730N/2400W

1125

- Jammer 1-4 (Low) ingress over Blueland, SEAD at Eaglevista
- Zap 1-2 push from 5750N/2500W

1130

- Sierra 1-4 Sweep, push from 5800N/2500W
- Echo 3-4 depart AAR Track B
- Predator 2 on station 5640N/2700W
- Predator 1 repositions to 5700N/2730W

1135

- Bomber 1-4 push from 5800N/2900W

1140

- Gunner push from 5800N/2800W

1145

e- Zap 1-2 engaged SEAD SA-8

j- Predator 3 on station 5730N/2630W, airborne backup

1150

a- Sierra 1-4 on CAP bullseye 5700N/2700W (southwest)

c- Bomber 1-4 TOT BAI LOC Cluster munitions

1155

g- Gunner TOT CAS

1200

f- Rescue 1-2 TOT extraction begins

- Echo 1-2 provide top cover in target area

- Echo 3-4 arrive to provide top cover with Echo 1-2

c+ Bomber 1-4 RTB (Return To Base) Bendeguz

1215

f+ Rescue 1-2 extraction complete

- Echo 1-2 RTB
- Echo 3-4 close escort CSAR egress

g+ Gunner RTB Nitric

1225

d+ Jammer 1-4 RTB Bendeguz

1230

a+ Sierra 1-4 RTB Bendeguz

e+ Zap 1-2 RTB Bendeguz

h+i+ Predator 1-2 RTB Bendeguz

1245

- Rescue 1-2 lands at Wahhabe

b+ Echo 3-4 RTB Bendeguz

In this timeline are represented the events related to the global plan shown in Figure 2, and also some other details that were not included in the global plan for the sake of simplicity. It is also worth mentioning that some details such as take off, landing and refueling of aircraft were not represented for the same reason.

4.4 Multi-agent coordination and execution monitoring

Once the execution of the global plan starts, each agent is responsible for the execution of a sub-plan and the monitoring of the temporal constraints of this sub-plan. An agent will also have to keep an up-to-date version of the global plan (temporal constraints).

We will illustrate the agent interaction mechanism based on their dependence networks at two different time points of the timeline (presented in Section 4.3), at 11:15 when ECHO 1-2 start Close Escort (b-) and at 12:15, when the extraction by RESCUE 1-2 of downed crews is complete (f+). The agents will behave in the same way at each time point of this timeline.

The updated version of the global plan and the positions of the different agents at 11:15 are shown in Figure 4. After starting the execution of sub-plan b), agent ECHO will update the temporal constraint T(*i*-, *b*-) = 90 (the sub-plan i) started executing at 9:45).

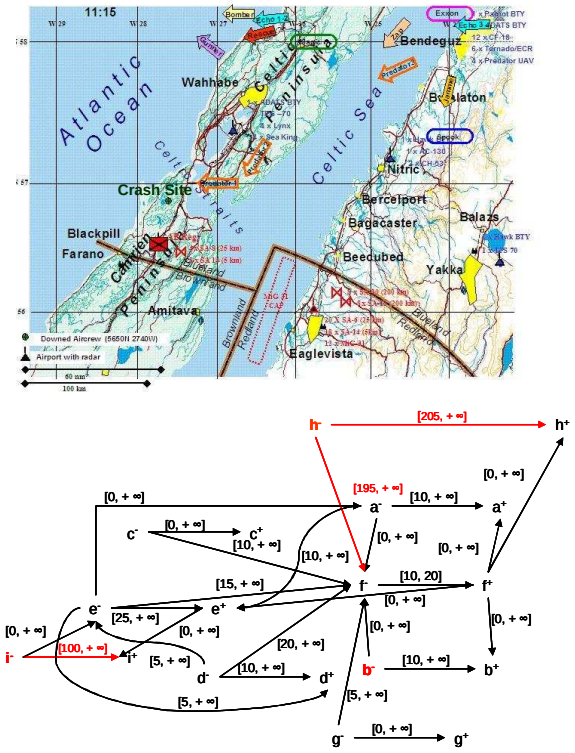

Figure 4: Local version of global plan for agent ECHO and positions of assets on the map at 11:15

The temporal constraints are then propagated into the graph by using a minimization operation described in [1]. The timeout and candidate list are computed. In this case the timeout T_{b-}= +∞. This is due to the fact that the main activity in this mission is the sub-plan f) (extraction). All other activities (sub-plans) will try to synchronize with this activity. In other terms, such a timeout value means that all the other sub-plans are temporally constrained by the sub-plan f) and not the inverse. The candidate list CL_{b-} = {g-, f-, a-, e-, d-}.

In this case, agent ECHO will have to communicate to agents: GUNNER, RESCUE, SIERRA, ZAP and JAMMER the following information:

$[T_{b-}$= +∞,

CL_{b-}= {g-, f-, a-, e-, d-},

T(*i*-, *b*-) = 90]

These agents are part of the dependence network of agent ECHO. This information will allow them to update their local version of the global plan.

The updated version of the global plan and the positions of the different agents at 12:15 are shown in Figure 5.

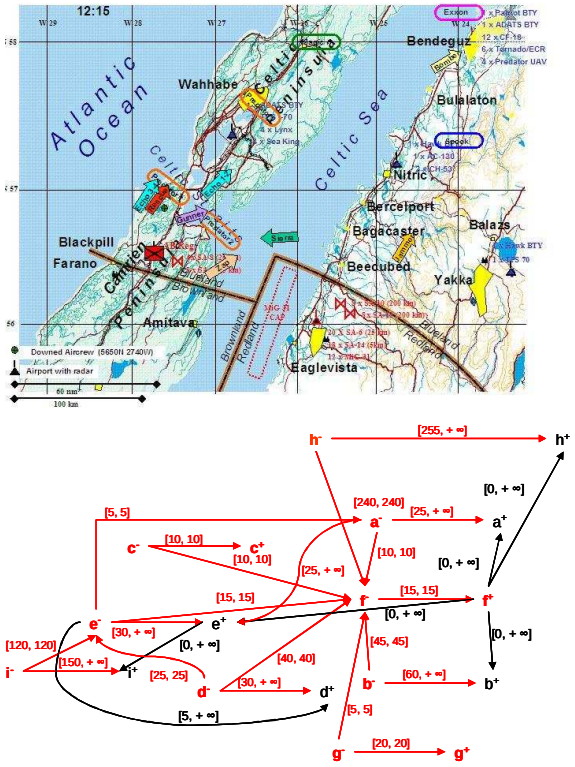

Figure 5: Local version of global plan for agent RESCUE and positions of assets on the map at 12:15

After the end of executing sub-plan f), agent RESCUE will update the temporal constraint T(f-, f+) = 15 (the extraction of the downed crews lasted exactly 15 minutes). This temporal constraint is then propagated through the graph by using the minimization operation as mentioned earlier. The timeout T_{f+}= +∞ since the remaining part of the mission is simply the Return To Base for all the assets. The new candidate list CL_{f+} = {e+, i+, d+, b+, a+, h+}. Agent RESCUE with send to agents: ZAP, PREDATOR 1, PREDATOR 2, JAMMER, ECHO and SIERRA the following information:

[T_{f+}= +∞,
CL_{f+} = {e+, i+, d+, b+, a+, h+},
T(f-, f+) = 15]

These agents belong to agent RESCUE's dependence network shown in Figure 3. This information will allow them to update their local version of the global plan.

It is worth mentioning that generally an agent is supposed to communicate different temporal constraints to each agent in the dependence network as shown in Figure 3. For instance, agent RESCUE will have to respectively send the following information to ZAP and ECHO:

[T_{f+}= +∞,
CL_{f+} = {e+, i+, d+, b+, a+, h+},
T(f-, e-),
T(f+, e+)],

[T_{f+}= +∞,
CL_{f+} = {e+, i+, d+, b+, a+, h+},
T(f-, b-),
T(f+, b+)]

In this particular case, it is sufficient to send the last temporal constraint and the other temporal constraints will be updated accordingly through propagation. This makes the communication simpler, since an agent will always communicate the same information to all agents in the dependence network.

5. DISCUSSION

This work is an attempt to indentify the main requirements for a multi-agent temporal plan monitoring in a coalition context. The motivation of a decentralized monitoring is that in many coalition contexts, members need to share duties, tasks, resources, etc. The participation of different members with their own constraints makes it difficult to centralize the monitoring of mission activities. For this reason, we think that agents need to have more autonomy in performing their tasks. However, they need to have well defined coordination and communication mechanisms. In this work, we claim that the use of dependence theory, where each agent builds a network including all dependence relations with other agents, is a good coordination mechanism. Direct communications among agents become then necessary to synchronize and coordinate their activities.

We illustrate the use of dependence networks to decentralize the temporal plan monitoring task by using a CSAR mission example. Each time an agent starts or finishes executing an action, communicates all necessary information to all agents that are part of this agent's dependence network. This information includes a timeout, a candidate list and a temporal constraint. As we mentioned earlier, this information means that at least one element of the candidate list

must be considered before the expiration of the timeout. Since the candidate list may contain several elements, it is worth considering the decision-making process in such a context. In the centralized context, a decision-maker had to choose an element in the candidate list. In the decentralized context, several options may apply. It is possible to use a centralized decision-making as in the centralized case. However, this limits the autonomy of the agents. It is also possible to have totally autonomous agents, where each agent is able to decide which action is the best candidate to be executed next. The last option, which is our opinion the most promising, is to keep a human decision-maker in the loop for each agent. In this case, it is possible to integrate heterogeneous decision-making processes in the same multi-agent system. Furthermore, these decision makers may use the multi-agent system as a cooperation and coordination tool, which crucial in any coalition context. More precisely, different decision-makers need to reach an agreement when it comes to choose an element of the candidate list, which contains elements belonging to different sub-plans. It is important to mention, that such an agreement must be reached before the expiration of the timeout. Thus, depending on the type of the mission and its temporal granularity, putting in place cooperation mechanisms between distributed decision-makers may interfere with the temporal constraints of the mission plan.

References

[1] Mohamad K. Allouche, Abdeslem Boukhtouta: Multi-agent coordination by temporal plan fusion: Application to combat search and rescue. Information Fusion 11(3): 220-232 (2010).

[2] Jaime Simão Sichman and Yves Demazeau
"On social reasoning in multi-agent systems" In: Revista Iberoamericana de Inteligencia Artificial, (13) Verano, pages 68-84, 2001.

[3] Jaime Simão Sichman, Rosaria Conte, Yves Demazeau and Cristiano Castelfranchi
"A social reasoning mechanism based on dependence networks"
In: Proc. 12th European Conference on Artificial Intelligence (ECAI'94) Amsterdam, The Netherlands, August 1994.

[4] Conte, Rosaria and Sichman, Jaime Simão. 1995. DEPNET: How to benefit from social dependence. Journal of Mathematical Sociology, 20(2-3), 161-177.

[5] Jaime Simão Sichman , Rosaria Conte, Multi-agent dependence by dependence graphs, Proceedings of the first international joint conference on Autonomous agents and multiagent systems: part 1, July 15-19, 2002, Bologna, Italy [doi>10.1145/544741.544855]

[6] Patrice Caire and Leendert van der Torre. Temporal dependence networks for the design of convivial multiagent systems. In Carles Sierra, Cristiano Castelfranchi, Keith S. Decker and Jaime Simão Sichman (Eds.), 8th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2009), Budapest, Hungary, May 10–15, 2009, Volume 2. pp. 1317–1318. IFAAMAS 2009.

[7] Makoto Yokoo, "Protocol/Mechanism Design for Cooperation/Competition," aamas, vol. 1, pp.3-7, Third International Joint Conference on Autonomous Agents and Multiagent Systems - Volume 1 (AAMAS'04), 2004

[8] Austin, J.L. How to Do Things With Words, Oxford University Press, Oxford (1962).

[9] Searle, J.R. Speech Acts: an Essay in the Philosophy of Language. Cambridge University Press, Cambridge (1969).

[10] Specification part 2 – agent communication language, Foundation for Intelligent Physical Agents (http://www.fipa.org/).

[11] DAML, The DARPA Agent Markup Language Homepage, http://www.daml.org/ (2006)