Multi-Agent Collaboration: A Satellite Constellation Case

Grégory BONNET \textsuperscript{a,1} Catherine TESSIER \textsuperscript{a}
\textsuperscript{a}Onera, DCSD/CD, France

Abstract. Physical agents such as robots are generally constrained in their communication capabilities. In a multi-agent system composed of physical agents, these constraints have a strong influence on the organization and the coordination mechanisms. Our multi-agent system is a satellite constellation, for which we propose a collaboration method based on incremental coalition formation in order to optimize individual plans and satisfy collective objectives. This involves a communication protocol and two coordination mechanisms: (1) an incentive to join coalitions and (2) coalition minimization. Results on a simulated satellite constellation are presented and discussed.

Keywords. Coalition formation, Multiagent systems, Teamwork and coordination, Satellite constellations

Introduction

In the multi-agent literature, most of the coordination mechanisms either based on norms \cite{6}, contracts \cite{14} or organizations \cite{3,8} involve \textit{software agents} or \textit{social agents}. In such contexts communications are generally assumed to be unconstrained. As far as \textit{physical agents} such as robots or satellites are concerned, physical and cost constraints have a major impact on communication and therefore on coordination. On the first hand an agent cannot always communicate with another agent or the communications are restricted to short time intervals; on the other hand an agent cannot always wait until the coordination process terminates before acting. Such constraints are present in space applications.

Let us consider satellite constellations i.e. 3 to 16 satellites placed in low orbit around the Earth to take pictures of the ground \cite{4}. Observation requests are generated asynchronously with various priorities by ground stations or the satellites themselves. As each satellite is equipped with a single observation instrument with use constraints, too close requests cannot be realized by the same satellite. Likewise, each satellite is constrained in memory resources and can realize only a limited number of requests before downloading, i.e. transferring the pictures taken to a ground station. Finally, the orbits of the satellites cross around the poles: two (or more) satellites that meet in the polar areas can communicate \textit{via} InterSatellite Links (ISL) without any ground intervention. So the satellites can communicate from time to time.

\textsuperscript{1}We would like to thank Marie-Claire Charmeau (CNES – The French Space Agency) and Serge Rainonneau (Thales Alenia Space) for their comments on this work.
Centralized planning [12,22] is not considered because (1) the aim of future space applications is to avoid using ground stations as much as possible (operating a ground station is expensive); (2) the asynchronous generation of new requests by each satellite prevents us from having a centralized view of the problem and therefore a centralized resolution.

Consequently the problem we focus on is a decentralized task allocation problem in a multi-agent system with new tasks arriving asynchronously and intermittent communications. Each satellite (each agent) builds and revises a task plan such that the number of tasks realized by the constellation is the highest possible, they are realized as soon as possible, the number of redundancies is the lowest possible (cf. Definition 5) and the number of high priority tasks that are not realized is the lowest possible. In order to address this problem, we propose an online incremental dynamic organization mechanism in three steps: (1) agents plan individually; (2) agents communicate in order to build a common knowledge; (3) agents build and revise coalitions that influence their plans.

1. A multiagent system

1.1. Public knowledge of the agents

The constellation is a multi-agent system where each satellite is represented by an agent:

Definition 1 (Constellation) The constellation $\mathcal{S}$ is a triplet $(\mathcal{A}, \mathcal{T}, \text{Vicinity})$ with $\mathcal{A} = \{a_1 \ldots a_n\}$ the set of $n$ agents representing the $n$ satellites, $\mathcal{T} \subset \mathbb{N}^+$ a set of dates defining a common clock and Vicinity : $\mathcal{A} \times \mathcal{T} \mapsto 2^\mathcal{A}$ a symmetric non transitive periodic relation specifying for a given agent and a given date the set of agents with which it can communicate at that date (acquaintance model). Vicinity represents the temporal windows when the satellites meet; it is calculated from the satellite orbits, which are periodic.

Definition 2 (Periodicity) Let $\mathcal{S}$ be a constellation and $\{p_1 \ldots p_n\}$ the set of the orbital cycle durations $p_i \in \mathcal{T}$ of agents $a_i \in \mathcal{A}$. The Vicinity period $\hat{p} \in \mathcal{T}$ is the lowest common multiple of set $\{p_1 \ldots p_n\}$.

Other agents, clock and Vicinity is knowledge that all the agents hold in common.

1.2. Private knowledge in terms of tasks and intentions

Each agent within the constellation knows some tasks to realize.

Definition 3 (Task) A task $t$ is an observation request associated with a priority $\text{prio}(t) \in \mathbb{N}^*$ and with a boolean $b_t$ that indicates whether $t$ has been realized or not.

Notice that in the space domain, 1 stands for the highest priority whereas 5 is the lowest. Consequently the lower $\text{prio}(t)$, the more important task $t$. The tasks may be constrained in two ways:

- **mutual exclusion**: it is an agent’s constraint meaning that it cannot realize several tasks at the same time $\tau$ ;
composition of $n$ tasks: all the $n$ tasks must be realized, it is useless to realize only a strict subset of them. Formally,

**Definition 4 (Compound task)** A compound task is a subset $T$ of tasks such that ($\exists t_i \in T, t_i$ is realized) $\Rightarrow$ ($\forall t_j \in T, t_j \neq t_i$ must be realized).

Moreover when a task is realized by an agent, it is redundant if it has already been realized by another agent:

**Definition 5 (Redundancy)** Let $a_i$ be an agent that realizes a task $t$ at time $\tau \in T$. There is a redundancy about $t$ if and only if $\exists a_j \in A$ and $\exists \tau' \in T (\tau' \leq \tau)$ such that $a_j$ has realized $t$ at time $\tau'$.

**Example 1** Let us suppose that an agent $a_1$ realized a task $t$ at time $\tau_1$. If an agent $a_2$ realizes the same task later, i.e. takes the same picture of the ground at time $\tau_2 (\tau_1 < \tau_2)$, there is a redundancy.

Let $T_{a_i}^{\tau}$ be the set of all tasks known by an agent $a_i$ at time $\tau$. Each agent $a_i$ has resources available to realize only a subset of $T_{a_i}^{\tau}$. These resources are the mass memory that allows to keep pictures in memory before downloading.

Each agent within the constellation knows some intentions about the tasks.

**Definition 6 (Intention)** Let $I_{a_i}^t$ be the intention of agent $a_i$ towards task $t$. $I_{a_i}^t$ is a modality of proposition ($a_i$ realizes $t$):

- $\Box$ (commitment): $a_i$ is committed to realize $t$
- $\Diamond$ (proposal): $a_i$ proposes to realize $t$
- $\Box \neg$ (strong withdrawal): $a_i$ will not realize $t$
- $\Diamond \neg$ (weak withdrawal): $a_i$ does not propose to realize $t$

A realization date $\text{rea}(I_{a_i}^t) \in T \cup \{\emptyset\}$ and a download date $\text{tel}(I_{a_i}^t) \in T \cup \{\emptyset\}$ are associated with each intention.

Let $I_{a_i} = (I_{a_i}^t)$ be the matrix of the intentions known by agent $a_i$ at time $\tau$. More precisely the set of an agent’s intentions corresponds to its current plan. We assume that each agent has an individual planner. Planning is a three-step process. (1) From the set of unrealized tasks known by $a_i$ at time $\tau$, $a_i$ computes an optimal local plan under two criteria: maximization of the number of planned tasks and minimization of the number of unplanned high priority tasks. (2) The intentions of agent $a_i$ about tasks $t$ at time $(\tau - 1)$ constrain the planning process (1): tasks linked to a commitment (\Box) are always planned and tasks linked to a strong withdrawal (\Box \neg) are never planned. (3) Agent $a_i$’s plan at time $\tau$ modifies its intentions as follows: each new planned task generates a proposal (\Diamond) and each new unplanned task is set aside (\Diamond \neg).

We can notice that the commitments (\Box) and strong withdrawals (\Box \neg) are not generated by the planning process. We will see in Section 3 that these intentions are generated by a collaboration process.

Finally tasks and intentions an agent knows are captured by knowledge:

---

2The individual planning process itself is beyond the scope of our work.
Definition 7 (Knowledge) A piece of knowledge $K_{a_i}^\tau$ of agent $a_i$ at time $\tau$ is a triplet

$$<D_{K_{a_i}^\tau}, A_{K_{a_i}^\tau}, \tau_{K_{a_i}^\tau}>$$

- $D_{K_{a_i}^\tau}$ is a task $t$ or an intention $I_{a_k}^t$ of $a_k$ about $t$, $a_k \in A$;
- $A_{K_{a_i}^\tau} \subseteq A$ is the subset of agents knowing $K_{a_i}^\tau$;
- $\tau_{K_{a_i}^\tau} \in \mathbb{T}$ is the date when $D_{K_{a_i}^\tau}$ was created or updated;

Let $K_{a_i}^\tau$ be the set of all pieces of knowledge of an agent $a_i$ at time $\tau$.

2. Communication

Communication is based on Vicinity: when two agents meet they can communicate. Consequently the Vicinity structure influences the communication capabilities. Two kinds of communications are defined:

Definition 8 (Communication) Let $S$ be a constellation and $a_i, a_j \in A$:

- $a_i$ communicate directly with $a_j$ iff $\exists \tau$ within $p$ such that $a_j \in \text{Vicinity}(a_i, \tau)$;
- $a_i$ communicate indirectly with $a_j$ iff $\exists \{a_k \in A, i \leq k < j\}$ and $\exists \{\tau_k\text{ within }p, i \leq k < j\}$ such that $a_{k+1} \in \text{Vicinity}(a_k, \tau_k)$.

In case of an indirect communication, $a_i$ and $a_j$ may communicate through several agents forming a daisy chain. As Vicinity is symmetric but not transitive, direct communication is symmetric whereas indirect communication is oriented from an agent to another one. Each communication from $a_i$ to $a_j$ is associated with a couple $(\tau_i, \tau_j) \in \mathbb{T}^2$ with $\tau_i$ the emitting date of $a_i$ and $\tau_j$ the receipt date of $a_j$. We will write: $a_i$ communicates with $a_j$ at $(\tau_i, \tau_j)$. In case of a direct communication, $\tau_i = \tau_j$.

2.1. An epidemic protocol

The agents have to reason on a common knowledge in terms of tasks and intentions. A epidemic protocol based on overhearing [11] has been proposed [2] to allow an agent to know what the other agents know. The agents use every opportunity to communicate information even if it does not concern themselves:

1. each agent $a_i$ considers its own knowledge changes;
2. $a_i$ communicates the changes to $a_j \in \text{Vicinity}(a_i, \tau)$;
3. $a_j$ updates its own knowledge thanks to the timestamp $\tau_{K_{a_i}^\tau}$;
4. $a_i$ and $a_j$ update the set of agents knowing the knowledge.

This last step allows us to define a common knowledge notion. Formally,

Definition 9 (Common knowledge) At time $\tau$, agent $a_i$ knows that agent $a_j$ knows the intention $I_{a_i}^t$ captured by $K_{a_i}^\tau$ iff $a_j \in A_{K_{a_i}^\tau}$ or $a_i$ communicated with $a_j$ at $(\tau_i, \tau_j)$ such that $\tau_{K_{a_i}^\tau} \leq \tau_i, \tau_j \leq \tau$. 
2.2. Last confirmation date

As the environment is dynamic, an agent may receive new tasks or new intentions and modify its plan, i.e. its own intentions, accordingly. Consequently an agent that receives a given proposal at time $\tau$ cannot be sure that this proposal will be the same at time $\tau'$ ($\tau' > \tau$). The more time between the generation of a given proposal and the realization date, the less an agent can trust it. However as the agents communicate every knowledge modification, an agent that does not communicate changes about its own intention confirms it. As the environment is dynamic, an agent may receive new tasks or new intentions and modify its plan, i.e. its own intentions, accordingly. Consequently an agent that receives a proposal known by $a$ from $a$'s point of view about $a$'s proposals is $\tau_2$.

2.3. Trust

Intuitively the trust associated with a proposal depends on the time between its last confirmation date and its realization. The agents cannot predict the arrival of new tasks. However as time passes, an agent meets other agents and each meeting is an opportunity to receive new tasks and revise its intentions. Consequently an agent’s trust about a given proposal at time $\tau$ if and only if $M_{\tau'}(I_t^{a_i}) = 0$.

Example 3 Let us resume Example 2. At $\tau_1$, $a_1$ and $a_2$ meet, $a_2$ will not trust $a_1$’s proposals for $a_1$'s proposals that would be issued after $\tau_2$ because $a_1$ will meet $a_3$. At $\tau_3$, due to the last confirmation date $\tau_2$. $a_2$ will trust $a_1$’s proposals that would be issued after $\tau_2$.

We can notice that the trust criterion of Definition 12 is hard: an agent is not trusted if it meets at least another agent before realizing its proposal ($M_{\tau'}(I_t^{a_i}) = 0$). This pessimistic assumption can be relaxed (e.g. $M_{\tau'}(I_t^{a_i}) \leq 1$).

Definition 10 (Last confirmation date) Let $a_i$ be an agent that known $I_t^{a_j}$ a proposal of an agent $a_j$ about a task $t$. The last confirmation date of $I_t^{a_j}$ for $a_i$ at time $\tau$ is:

$$\tau^* = \max_{\tau_{a_i} < \tau_j < \tau} \{ \tau_j : a_j \text{ communicates with } a_i \text{ at } (\tau_j, \tau_i) \}$$

Example 2 Let $a_1$, $a_2$ and $a_3$ be three agents. Suppose that $a_1$ communicate directly with $a_2$ at $\tau_1$, $a_3$ with $a_3$ at $\tau_2$, and $a_3$ with $a_2$ at $\tau_3$ ($\tau_1 < \tau_2 < \tau_3$). At $\tau_3$, the last confirmation date from $a_2$’s point of view about $a_1$’s proposals is $\tau_2$.

Definition 11 (Meetings) Let $a_i$ be an agent, $I_t^{a_j}$ a proposal known by $a_i$ and $\tau$ the current date. Let $\tau^*$ be the last confirmation date of $I_t^{a_j}$ for $a_i$ at time $\tau$. The number of agents $M_{\tau^*}^{a_i}(I_t^{a_i})$ agent $a_j$ will meet between $\tau^*$ and $\text{rea}(I_t^{a_j})$ is given by:

$$M_{\tau^*}^{a_i}(I_t^{a_j}) = | \bigcup_{\tau^* < \tau' < \text{rea}(I_t^{a_j})} \text{Vicinity}(a_j, \tau') |$$

Finally, the trust criterion is:

Definition 12 (Trust) Let $a_i$ be an agent, $I_t^{a_j}$ a proposal known by $a_i$ and $\tau$ the current date. Agent $a_i$ trusts agent $a_j$ about $I_t^{a_j}$ if and only if $M_{\tau^*}^{a_i}(I_t^{a_j}) = 0$.

Example 3 Let us resume Example 2. At $\tau_1$, $a_1$ and $a_2$ meet, $a_2$ will not trust $a_1$’s proposals that would be issued after $\tau_2$ because $a_1$ will meet $a_3$. At $\tau_3$, due to the last confirmation date $\tau_2$, $a_2$ will trust $a_1$’s proposals that would be issued after $\tau_2$.

We can notice that the trust criterion of Definition 12 is hard: an agent is not trusted if it meets at least another agent before realizing its proposal ($M_{\tau^*}^{a_i}(I_t^{a_i}) = 0$). This pessimistic assumption can be relaxed (e.g. $M_{\tau^*}^{a_i}(I_t^{a_i}) \leq 1$).
3. Collaboration via coalitions

3.1. Coalitions

A coalition is an agent organization with a short life cycle. It is formed in order to realize a given goal and is destroyed when the goal is achieved. Through a coalition, each agent tries to maximize its personal outcome. In the literature, the methods dedicated to coalition formation are based on the exploration of the lattice of the possible coalition structures [10,15,16,19]. However, these methods are often centralized or they use an auctioneer (or other kinds of hierarchy), assume that all tasks are known by all agents and are performed off-line [1,5,7,18,17]. The decentralized approach has been investigated by [9] but, in our application, agents cannot always exchange information and they may have to decide alone. Moreover some tasks cannot wait for the complete computation of the coalition structure and must be realized quickly.

Be that as it may, the coalition formation mechanisms are interesting for three reasons: (1) agents gather in order to realize a collective task; (2) the short life cycle of coalitions is adapted to dynamic environments; (3) agents search for efficient solutions under uncertain and (or) incomplete information. In our application, compound tasks require that some agents should realize some subsets of tasks jointly. However these joint realizations cannot be planned by the agents’ individual planners as an agent does not plan for the others. In order to dynamically organize the agents, we will consider a decentralized coalition formation mechanism taking into account the features of our problem, i.e. cooperative agents and constrained communications. The mechanism is as follows:

1. Agents build maximal-size coalitions from their own knowledge;
2. Coalitions are refined as the agents meet to remove useless agents.

Coalitions are defined as follows:

**Definition 13 (Coalition)** A coalition $C$ is a triplet $\langle A, O, P \rangle$:

- $A \subseteq A$ is a subset of agents that are the members of the coalition;
- $O$ is the set of tasks that are the goals of the coalition, i.e. that must be realized by the coalition;
- $P$ is the set of tasks that are in the power of the coalition, i.e. that are intended to be realized by the coalition.

A coalition $C$ can be in different states:

- $C$ is complete iff $O \subseteq P$;
- $C$ is minimal iff $C$ is complete and $A$ is minimal for inclusion ($\subseteq$).

Coalitions are build and managed locally by each agent, given the knowledge it has about the other agents through communication. Indeed each agent uses the coalition notion to reason and adapt its own intentions to the others’ intentions. Therefore, coalitions are formed implicitly through intentions but are not explicitly built by the multi-agent system. Each agent:

i . computes the current coalition structure from its point of view;
ii . checks whether it should join a coalition to increase its power;
iii . checks whether it can withdraw from a coalition to minimize it;
iv . modifies its intentions accordingly.
3.2. Computation of the coalition structure

Each agent \( a_i \) generates the current coalition structure as follows:

1. \( a_i \) organizes the set of tasks \( T_{a_i}^\tau \) as a partition \( \{T_1, \ldots, T_h\} \) according to the compound tasks;

   **Example 4** Let \( T_{a_i}^\tau \) be \( \{t_1, t_2, t_3, t_4, t_5\} \). Let us suppose that tasks \( t_1 \) and \( t_2 \) form a compound task as well as \( t_4 \) and \( t_5 \). Then \( T_{a_i}^\tau \) is organized as \( \{\{t_1, t_2\}, \{t_3\}, \{t_4, t_5\}\} \).

2. each \( T_i \) is the goal of a single potential coalition; as subsets \( T_i \) are disjoint\(^3\), the number of potential coalitions generated by agent \( a_i \) is given by the number of compound tasks \( a_i \) knows;

3. from agent \( a_i \)'s point of view, the potential coalition members for subset \( T_i \) are defined as:

   \[ \{a_k \in A : \exists t \in T_i / \exists I_{a_k} t \in K_{a_i}^\tau \} \]

   **Example 5** Let us resume Example 4. Let us consider \( t_3 \) and suppose that \( I_{a_i}^t = 3 \) and \( I_{a_k}^t = 2 \). \( a_i \) can build coalition \( C = < \{a_i, a_k\}, \{t_3\}, \{t_3\}> \). This coalition is complete but not minimal because \( \{a_i, a_k\} \) is not minimal for inclusion. Notice that \( a_i \) plans \( t_3 \) even if it knows that \( a_k \) did the same. Indeed, the others’ intentions are not taken into account in the planning step: they are taken into account in the collaboration steps (ii., iii., iv.).

4. then the power of each potential coalition \( C \) with goal \( T_i \) is defined as:

   \[ P = \{ t \in O | \exists a_i \in A : I_{a_i}^t \in \{\Box, \Diamond\} \} \]

   Let us notice that this framework defines the current coalition structure from the agent’s point of view. Each potential coalition may be minimal (thus complete), complete and not minimal or incomplete. Consequently we define two mechanisms to enrich and refine the power of a coalition.

3.3. An incentive to join coalitions

An incomplete coalition means that at least one goal task is not within the coalition power. But the more tasks within the coalition power, the more important goal tasks become because a coalition must realize all its goal tasks. If the coalition remains incomplete, all its members waste their resources.

When agent \( a_i \) computes the current coalition structure according to its knowledge, it can detect incomplete coalitions. As \( a_i \) is cooperative, it should be incited to modify its intentions and complete these coalitions when planning. In order to do that, we propose to increase the priorities of the goal tasks of the incomplete coalitions. In the remainder, we will note \( prio(t)' \) the priority of task \( t \) \( a_i \) uses for its next planning step. Notice that \( prio(t)' \) is a local priority only used by \( a_i \). The initial priority \( prio(t) \) of task \( t \) remains the same.

**Protocol 1 (Join a coalition)** For each incomplete coalition \( C = < A, O, P > \), agent \( a_i \) computes:

\[ \forall t \in O, prio(t)' \leftarrow \frac{prio(t)}{1 + |P|} \]

\(^3\)The compound tasks are assumed disjoint but they can overlap without modifying the process.
The agent is encouraged to join a coalition if and only if the goal of the coalition is to realize a compound task that is partially planned. This mechanism is stable, i.e. two successive incentive steps are consistent. For instance, an agent is not encouraged to give up a given task in order to realize another one, then ceteris paribus is not encouraged to give up the latter to realize the former.

Example 6 Let us resume Example 4. Let us consider \( \{t_1, t_2\} \) and suppose that \( I_{t_1}^a = \bigtriangleup \neg, I_{t_2}^a = \bigtriangleup \neg, I_{t_3}^a = \bigtriangleup \neg \) and \( I_{t_4}^a = \bigcirc. \) \( a_i \) can build coalition \( C = \langle \{a_k\}, \{t_1, t_2\}, \{t_2\} \rangle. \) This coalition is incomplete. So \( a_i \) applies Protocol 1. As \( a_k \) is already a member of the coalition, the priorities of \( t_1 \) and \( t_2 \) are halved for \( a_i. \) Therefore at its next planning step, \( a_i \) is more likely to plan \( t_1 \) or \( t_2 \) instead of other tasks.

3.4. Minimizing coalitions

A complete and non minimal coalition has the power to realize its goals with useless agents, i.e. agents that have redundant intentions. Within a coalition, an agent has to consider the agents that have planned the same tasks as it has, then to make a decision about modifying or not its own intentions. There is a conflict between two agents within a coalition if they have planned the same task(s). Formally:

**Definition 14 (Conflict)** Let \( a_i, a_j \) be two agents and \( C \) a coalition \( \langle A, O, P \rangle \) such that \( \{a_i, a_j\} \subseteq A. \) There is a conflict between \( a_i \) and \( a_j \) iff \( \exists t \in P \) such that \( I_t^{a_i} \in \{\bigtriangleup, \bigtriangleup\} \) and \( I_t^{a_j} \in \{\bigcirc, \bigtriangleup\}. \) It is a soft conflict iff either \( a_i \) communicates with \( a_j \) at \( (\tau_i, \tau_j) \) such that \( \tau_i < \tau_j \) and \( \tau_j < \min(rea(I_t^{a_i}), rea(I_t^{a_j})) \) or \( a_j \) knows agent \( a_i \)'s intention about \( t. \) Else it is a hard conflict.

A soft conflict means that involved agents have (or may have) a common knowledge of it. Consequently they can coordinate. A hard conflict means that only one agent is aware (and will be aware) of it because there is no common knowledge. In the remainder, given an agent \( a_i \) and a task \( t, \) we denote \( A^+ \) the set of agents with which it is in conflict about task \( t, \) \( A^+ \subseteq A^+ \) the set of agents in soft conflict and \( A^- \subseteq A^- \) the set of agents in hard conflict.

Example 7 Let us resume Example 5. The coalition is not minimal: there is a conflict about task \( t_3 \) between agents \( a_i \) and \( a_k. \) So \( a_i \) has to make a decision in order to withdraw (\( \bigtriangleup\neg \)), to keep its intention (\( \bigtriangleup\)) or to commit (\( \bigcirc\)).

As we are seeking to optimize the system swiftness, it is better that the agents realize the tasks as soon as possible and use the fewest resources possible. This is meaning keeping the pictures in the satellite memory for the shortest time possible, i.e. downloading them as soon as possible. Let us aggregate both criteria in a single expertise criterion. Formally:

**Definition 15 (Expertise)** Let \( A^+ \subseteq A \) be a set of agents in conflict about a task \( t. \) Let us note \( rea^* = \min_{a_i \in A^+} rea(I_t^{a_i}) \) the earliest realization date for task \( t. \) The expert agent for \( t \) is defined thanks to the following distance (see Figure 1):

\[
\begin{align*}
    a^* &= \arg \min_{a_i \in A^+} ||(rea(I_t^{a_i}) - rea^*, tel(I_t^{a_i}) - rea^*)||
\end{align*}
\]
Figure 1. This figure is a representation of the expertise criterion for a task $t$ in the plan $(rea(I^*_t), tel(I^*_t))$, $a_i \in A^*$. The origin $rea^*$ is the earliest realization date for $t$ and intention $(rea^*, rea^*)$ is the ideal intention corresponding to an agent being able to realize $t$ at time $rea^*$ and download the corresponding picture immediately. $tel^*$ is the latest download date for $t$, if $t$ is realized at time $rea^*$. Obviously $tel(I^*_t) > rea(I^*_t)$ therefore only the hatched part is meaningful.

The distance between a potential intention and an ideal intention (the earliest realization and download date) represents time criteria. The expert agent for $t$ is the one that minimizes this distance.

Both soft and hard conflicts are dealt with through protocols based three strategies:

1. an insurance strategy where $a_i$ maintains its proposal ($\Box$) if it does not trust the other agents therefore maintaining redundancies to make sure that the task will be realized.
2. a competitive strategy where $a_i$ commits ($\square$) if it is the expert agent therefore deciding on a part of the current coalition structure.
3. a opportunist strategy where $a_i$ strongly withdraws ($\square\neg$) if the expert agent is trusted thus minimizing the size of the coalition.

Protocol 2 (Hard conflict) Let $A^+$ be the set of the coalition members with which agent $a_i$ is in conflict about task $t$ such that $A^- \neq \emptyset$. $a_i$ is aware of the conflict and applies:

1. if $\min_{a_k \in A^-} M_{a_k}^i(I^*_t) > 0$ then $I^i_t \leftarrow \bigcirc$
2. else $I^i_t \leftarrow \square \neg$

In case of a hard conflict, the agent who is aware of the conflict (1) maintains its proposal if it does not trust the agents within the conflict; else (2) withdraws.

Protocol 3 (Soft conflict) Let $A^*$ be the set of the coalition members with which agent $a_i$ is in conflict about task $t$ such that $A^+ \neq \emptyset$. Let $rea^*$ be $\min_{a_i \in A^+} rea(I^*_t)$:

1. if $a_i = \arg \min_{a_i \in A^+} ||(rea(I^*_t) - rea^*, tel(I^*_t) - rea^*)||$ then $I^i_t \leftarrow \bigcirc$
2. else let $a^*$ be the expert agent:

(a) if $M_{a^*}^i(I^*_t) > 0$ then $I^i_t \leftarrow \bigcirc$
(b) else $I^i_t \leftarrow \square \neg$

For soft conflicts, each agent computes the expert agent. (1) If it is the expert agent, it commits. (2.a) If not, it maintains its proposal if it does not trust the expert. (2.b) If it trusts the expert, it withdraws.
4. Simulations and results

Simulations have been conducted on three kinds of constellations: (1) isolated constellations with no communication; (2) informed constellations where agents communicate only about tasks and coordinate \textit{a posteriori} by withdrawing already realized tasks from their plans; (3) coordinated constellations where agents communicate about tasks and intentions and coordinate \textit{a priori} thanks to coalition formation.

4.1. Performance

The first simulation round is based on a dynamic scenario with 3 agents. Every 6th hour, the ground stations send 40 new compound tasks (including at least 2 atomic tasks) to the agents. Two metrics are considered: the number of realized tasks (Figure 2) and the number of realized tasks without redundancy (Figure 3).

Informed and coordinated constellations outperform isolated ones. However we can notice that the benefits increase as time passes. Indeed incremental coordination allows coordinated constellations to realize more tasks than the other kinds of constellations. And as time passes the difference between informed and coordinated constellations increases: incremental coordination allows coordinated constellations to efficiently save and reallocate resources.

4.2. Scalability

In order to experiment the scalability of our system, we have considered a scenario with 500 atomic tasks and Walker satellite constellations [21] of different sizes (1, 4, 6, 8, 9, 12 and 16 satellites dispatched regularly on a finite number of orbital plans). The agents must realize all the tasks and the constellation swiftness and efficiency are then compared.

**Definition 16 (Performance)** Let $T_n$ the time of $n$ agents to realize all the tasks, $K$ the set of realized observations (i.e. the realized tasks and their redundancies) and $R$ the set of realized tasks. The constellation swiftness is given by $T_n$ and the constellation efficiency is given by $\frac{|R|}{|K|}$.
We can notice on Figure 4 that the swiftness of isolated constellations is approximated by a logarithmic function whereas the swiftness of informed and coordinated constellation are not regular. This is due to the heterogeneous structure of the satellite interactions. Indeed isolated satellites have no interactions but, for informed and coordinated constellations, interactions exist only between satellites belonging to different orbital plans (see Figure 6).

Consequently 2 satellites situated on 4 plans can have more interactions than 4 satellites situated on 3 plans: the topology of the interactions matters. More precisely the number of satellites is not the major parameter but their orbits: few satellites may communicate often whereas many satellites may only communicate from time to time. This phenomenon can be observed between the 8- and 12-satellite constellations. We can notice on Figure 5 that coordinated constellations are in average 5% more efficient than informed constellations. They are also 19% more efficient than isolated constellations. The constellations are scalable according to Turner [20]: a system is scalable if the resource consumption can be bounded by a polynomial function. In our application, the number of realized observations divided by the number of realized tasks \( \frac{K}{R} \) represents the resource overconsumption: it is the inverse of efficiency.

5. Conclusion

We have proposed a collaboration method for physical agents that communicate from time to time in a dynamic environment. This method has been applied to a constellation of satellites. A communication protocol has been proposed in order to build common knowledge (in terms of tasks and intentions) as the agents meet. The collaboration process is an online incremental coalition formation that proceeds through a planning - communication - collaboration loop within each agent. Each agent builds an initial plan;
from its knowledge, it builds the potential coalitions that can realize the tasks it knows; afterwards these coalitions are refined thanks both to an incentive mechanism and an optimization mechanism. The agents’ communication capabilities and the conflict definitions allow us to define protocols that refine the coalition structure dynamically and adapt it to new knowledge. The experimental results show that the coalition formation mechanism allows the resource consumption to be minimized; then the saved resources are reallocated in an incremental way and the number of realized tasks is increased. Future work will deal with the possible failures of the agents (communication or coordination).

References