Collective Decision-Theoretic Planning for Planet Exploration

Arnaud Canu*, Abdel-Illah Mouaddib*
*Université de Caen Basse-Normandie
UMR 6072 GREYC/CNRS, F-14032 Caen, France
Email: {arnaud.canu,abdel-illah.mouaddib}@unicaen.fr

Abstract—In this paper, we describe an approach of platoon formation for planet exploration like problems. In such problems, the agents have partial observability about their environment (using on-board sensors), must deal with uncertain outcomes of their actions (a wheel could slip) and can not communicate (it uses too much energy). In order to address those limitations, we use 2V-DEC-MDP: this model deals with such problems and uses local interactions between agents instead of explicit coordinations.

First, we describe how to use specific flocking rules in a 2V-DEC-MDP to build a platoon shape. Second, we use the stochastic games theory to analyze the optimality of our approach. Then, we show how we can effectively compute good policies with a low complexity. We finally give a practical usage example of our approach, on a platooning problem.

Keywords—Multiagent Planning; Markov Decisions Processes; Game Theory; Multi-Robot Systems

I. INTRODUCTION

The exploration of a planet such as Mars or the Moon with multiple robots is a challenging task. One important issue is the ability of robots to navigate in a coordinated way and sometimes evolve in a formation such as a platoon where the most robust robots should lead the formation and secure the path for the rest of the robots. We address this specific problem of the platoon formation by computing a plan for a group of rovers, so they can form a platoon and reach a given destination.

Robots platooning [1] is a concrete problem where the goal is to build and to maintain a formation for a group of mobile robots. Robotics community addressed this problem with different techniques where the most popular are the bio-inspired, evolutionist and explicit cooperative approaches. Most of these techniques are based on some strong assumptions such as communication, full observability and deterministic actions, or assuming the global desired behavior using top-down technique as ant algorithms [2]. Our approach relax most of these assumptions. In real world problems, robots must deal with uncertainty, so we use stochastic planning: effectors are not perfect (a wheel could slip), nor are sensors (noisy observations). Moreover, using communication can be very costly or not possible when the robots explore a distant planet, so we use decentralized decision methods with no communication. Finally, our approach is a bottom-up technique where the global behavior is emerged from local behaviors.

We will look at the particular problem of agents trying to organize themselves according to a line shape but other shapes could be considered. In order to build this line shape, we needed a way to coordinate robots together, without using any communication (which is important for robotic space applications). Those kinds of problems have been studied with flocking approaches, where agents have to maintain a global shape thanks to few local basic rules. So, we adapt the flocking techniques in this context where the outcomes of robot actions are uncertain. Reynolds proposed 3 rules, known as “flocking rules”, such that if each agent in a problem respects those rules, then the group moves like a group of birds would do: each agent stays near the other agents, but far enough to not collide. In our work, we describe a formal framework for robot platooning, based on the adaptation of those flocking rules in the Vector-Valued Decentralized Markov Decision Process (2V-DEC-MDP) framework. Using this framework, a robot only has to compute its own individual plan, but considers its neighbors while moving (local interaction). We show how the global platoon shape is built, only using such local interactions.

For those kind of problems, and for a single agent, the Markovian Decision process (MDP) [3] framework allows the agent to compute an optimal policy. DEC-POMDP framework [4] has been designed for the same purpose in multiagent settings. Although DEC-POMDPs can find optimal policies, their complexity generally is so high that it is hard to apply on real applications. Moreover, the objective function is considered to be mono-criterion. Then, 2V-DEC-MDPs have been introduced to coordinate a large number of agents in multi-criteria problems, extending the MDP framework, by considering local interactions with local full observability (which is a specific Partial Observability) [5]. We consider 2V-DEC-MDPs as stochastic games to provide a mathematical foundation to improve results presented in [6]. This theory [7] is a formalism for situations where rewards do not only depend on the agent’s actions but on the actions of all the agents. The Stackelberg Equilibrium (SE) is the theoretic concept of optimality to use with a leader.

In this paper, we adapt flocking rules to platooning, we express them as criteria in a 2V-DEC-MDP and we prove the

---

1This work is supported by the french DGA (Direction Générale de l’Armement, FRANCE).
conditions under which the leader follows the SE. Finally, we demonstrate that the initial 2V-DEC-MDP leads to a near SE with weak complexity while reaching SE requires a very high complexity. Then, we present several experiments to validate our approach, first on a simulator and second on a flock of robots.

II. BACKGROUND

In this section, we introduce the flocking and stochastic games concepts and we show how we use these concepts with a 2V-DEC-MDP, in order to form a platoon.

A. Flocking

The flocking [8] is the behavior of a group of birds. This behavior can be easily reproduced by giving three rules to each agent: cohesion (staying near the neighbors), separation (not being too close to any other agent) and alignment (moving in the average direction of the group). If each agent follows these three rules, then the group will move as a flock without any explicit coordination. In this approach, we describe platooning as a particular form of flocking, where the agents try to maintain a line shape and to move toward the platoon’s objective. This can be done with particular flocking rules:

1) Cohesion : the agent waits for agents behind it,
2) Separation : the agent avoids collisions with agents in front of it.
3) Alignment : the agent moves toward the near agent in front of it, or toward the objective if no one is in front of it.

In the following, we show how we formalize these rules with an extended 2V-MDP [9] to multiagent settings using 2V-DEC-MDP [5].

B. Stochastic games

2V-DEC-MDPs are a sub-class of stochastic games. A stochastic game [7] is defined by a tuple \( \langle N, S, A, T, R \rangle \). We describe this tuple as follows:

- \( N \) is the number of agents taking part in the game,
- \( S \) is the set of states in which the game can be (a state describes the world and every player/agent),
- \( A = \{ A_1, A_2, \ldots, A_N \} \) is the set of possible actions for every agent \( i \) with \( A_i = \{ a_1^i, \ldots, a_{|A_i|}^i \} \),
- \( T : S \times A_1 \times \ldots \times A_N \times S \rightarrow [0,1] \) is the transitions model between states (it gives the probability, when the agents apply a joint action, to move from a state \( s \) to a state \( s' \)),
- \( R = \{ R_1, R_2, \ldots, R_N \} \) are the reward functions of every agent with \( R_i : S \times A_1 \times \ldots \times A_N \rightarrow \mathbb{R} \) a function giving a reward to the agent when is does a transition.

The policy of an agent \( i \) (noted \( \pi_i \)) is a function which gives, for each state \( s \), the action \( a \) the agent has to execute. So, at each step, each player chooses an action based on its actual state and its policy: the game then moves to a new state \( s' \). We define the joint policy \( \pi = (\pi_1, \ldots, \pi_N) \). To estimate a strategy’s value, it is necessary to know the utility for a given player to follow a given strategy. Let \( \pi_i(s) \) be the chosen action by applying the policy \( \pi_i \) on the state \( s \). We can then write \( \pi(s) = (\pi_1(s), \ldots, \pi_N(s)) \) as the joint action for this state. In this game, every agent \( i \) has (by definition) an immediate reward \( R_i(s, \pi_i(s)) \) and we can compute \( U_i^\pi(s) \) the expected utility for an agent \( i \) if, in a state \( s \), all the agents apply \( \pi \). We compute this utility using the Bellman equation (with \( 0 \leq \gamma < 1 \)):

\[
U_i^\pi(s) = R_i(s, \pi_i(s)) + \gamma \sum_{s' \in S} T(s, \pi(s), s') \cdot U_i^\pi(s')
\]

In a game, a Stackelberg Equilibrium [10] (SE) is a situation where the leader of a group knows that it is the leader. It makes decisions, and its followers apply \( BR \) the best response (the best decision) according to this decision. The leader can then estimate the reactions of the other agents, and makes the decision which will bring it the best reward (or the best group reward), according to those reactions. SE is the theoretic concept of optimality to be used in such a situation. The optimal policy of an agent \( i \) is written \( \pi_i^* \).

Let us define the SE for a stochastic game. First, let \( BR_j(\pi_i) \) be the set of best policies \( \pi_j \) for agent \( j \), knowing that agent \( i \) apply the policy \( \pi_i \), and \( BR(\pi_i) = \{(\pi_1, \ldots, \pi_N)|\pi_1 \in BR_1(\pi_i), \ldots, \pi_N \in BR_N(\pi_i)\} \). If, for every state \( s \in S \), the leader’s policy \( \pi_i^* \) respects Eq.1, then we have SE.

\[
\min_{\pi_i \in BR(\pi_i)} \max_{(\pi_1, \ldots, \pi_N)} U_i^{\pi_1, \ldots, \pi_N}(s) = \min_{\pi_i \in BR(\pi_i)} \max_{(\pi_1, \ldots, \pi_N)} U_i^{\pi_1, \ldots, \pi_N}(s) \quad (1)
\]

C. 2V-DEC-MDP

The 2V-DEC-MDPs framework has been proposed to locally coordinate the actions of a group of agents. Assuming without loss of generality that all agents are identical, a 2V-DEC-MDP is a set of 2V-MDPs, one per agent. A 2V-MDP is composed by an off-line part (an MDP) and an on-line part to adapt its actions with the other agents.

The off-line part: MDP

An MDP [3] is a tuple \( \langle S, A, T, R \rangle \) similar to a stochastic game with one agent. This MDP describes the set of states in which the agent can be, the actions it can do, the transition probabilities between those states and the associated rewards. Using this MDP, the agent computes an individual policy, allowing it to achieve its objectives without considering the other agents.
The on-line part

For a given agent, the on-line part of its 2V-MDP is used to describe the local impact of its actions on the other agents, according to local observations. The functions for computing the value of the social impact are: ER for the individual reward (is this action good for me), JER for the group interest (is this action good for the others) and JEP for the negative impact on the group. Deriving a policy consists of solving a multi-criteria Bellman equation based on an Augmented Reward \( AR = (ER, JER, JEP) \). To solve this equation, a regret based value iteration using \textit{LexDiff} operator [5] (a Max-Min operator) has been designed.

To use flocking rules in a 2V-DEC-MDP, we translate the three rules into three formulæ to parameterize each 2V-MDP. We consider ER as the alignment criterion, JER as the cohesion criterion and JEP as the separation criterion.

III. PLATOONING AS A 2V-DEC-MDP

In this section, we describe how the flocking rules can be expressed in a 2V-MDP, in order to build a platoon with the agents. We use notations from sec.II-B and we write \( s_i^j \) the state \( j \) of agent \( i \).

A. Alignment

\( ER \), the individual reward, is associated to the alignment criterion (the agent receives a reward when it moves toward its objective):

\[
ER(s,a) = \sum_{s \in S} T(s,a,s^\prime) \times ER_i \quad i = 1,2,3
\]  

(2)

Depending on the situation, \( ER_i \) are defined by:

\[
ER_1 = V^*(s^\prime)
\]

\[
ER_2 = - \min_{s_j \in \text{face}(s^\prime)} \text{distance}(s^\prime, s_{b1})
\]

\[
ER_3 = \text{distance}(s^\prime, s_{b2})
\]

And we use \textit{face}(s) the agents closer to the objective than \( s \), \textit{back}(s) the next place available behind \( s \) and \( V^*(s) = \max \pi U^\pi(s) \). Moreover, \( s_{b1} = \text{back}(s_j) \) and \( s_{b2} = \text{back(leader)} \). In \( ER_2 \) and \( ER_3 \), we use \textit{back(target)} instead of \textit{target}, because the agent wants to go behind its target.

An agent does not have the same objectives whether it is a leader or a follower. Indeed, a leader will move in the direction of its objective, while a follower will follow the agent in front of it, so an agent have to choose which equation to follow before resolving its 2V-MDP. If the agent is a leader, or if it is out of range of any platoon, it chooses \( ER_1 \). If it is inside a platoon but it knows that the leader is behind it, it chooses \( ER_3 \). Otherwise, \( ER_2 \).

B. Separation

The \( JEP \) criterion gives the group penalties. Here, it describes the separation criterion (the group receives a penalty each time two agents collide):

\[
JEP(s,a) = \sum_{s^\prime \in S} [T(s,a,s^\prime) \cdot \sum_{s_j \in D} (T(s_j,a_j,s^\prime) \cdot C)]
\]  

(3)

We define the neighborhood for an agent \( i \) as the set of states of (detected) agents who can interact with \( i \). Then, \( D \) is the set of states of detected agents in the neighborhood and \( C \) is a constant equal to the cost of a collision between two agents.

C. Cohesion

Finally, the \( JEP \) criterion gives the group rewards, so it describes the cohesion criterion (the group receives a reward while cohesion is maintained):

\[
JER(s,a) = \sum_{s^\prime \in S} (T(s,a,s^\prime) \cdot K(s^\prime))
\]  

(4)

With \( K(s) \) a function which estimates the group gain at state \( s \) and gives a reward if at least one agent is behind \( s \).

D. Computation of a policy

Solving the platooning problem consists in deriving a policy from a 2V-DEC-MDP. We solve the MDP with standard algorithms (such as Value-Iteration) and we use it to compute \( ER \), \( JER \) and \( JEP \) (to describe local interactions). Then, we build the augmented reward and we solve the associated modified Bellman equation with the near-optimization operator \textit{LexDiff}. We are then able to derive a policy.

IV. STACKELBERG EQUILIBRIUM

In order to compare our formalism to the SE optimality concept, we aim to find when a 2V-DEC-MDP leads the leader to follow such an equilibrium.

A. Detecting Stackelberg equilibrium

First, we introduce two theorems used to detect if a group of agents is following an SE. Using these theorems, we deduce that in our approach, the agents do not follow such an equilibrium.

\textbf{Theorem 1.} An agent using a 2V-MDP is following an SE if and only if each criterion leads to an SE.

\textit{Proof:} We aim to show that each criterion leads to an SE if and only if applying \textit{LexDiff} on those criteria leads to an SE.

Here, the agent’s policy is built on the fly by \textit{LexDiff}: for each state, it chooses an action to apply. Because the SE definition says that a leader is in equilibrium if and only if
it follows Eq.1 for each state s, we just have to show that the LexDiff equation is equivalent to Eq.1 for every state.

Let c be a criterion and \( Q_c(s, \pi(s)) \) be the utility for an agent i for applying its policy \( \pi \) when it is in the state s, according to c. If we assume that each criterion c leads to an SE, we have:

\[
\min_{BR(\pi^*_L)} Q_c(s, \pi^*_L(s)) = \max_{\pi_L} \left( \min_{BR(\pi)} Q_c(s, \pi_L(s)) \right) \quad (5)
\]

\[
= \max_{\pi_L} \left( \min_{BR(\pi)} Q_c(s, \pi_L(s)) \right) = \max_{\pi_L} \left( \min_{\pi, \in \overline{Q}_{criteria}} Q_c(s, \pi_L(s)) \right) \quad (6)
\]

\[
\iff \min_{BR(\pi^*_L)} \left[ \min_{\pi, \in \overline{Q}_{criteria}} Q_c(s, \pi_L(s)) \right] = \max_{\pi_L} \left( \min_{\pi, \in \overline{Q}_{criteria}} Q_c(s, \pi_L(s)) \right) \quad (7)
\]

\[
\iff \max_{\pi_L} \left( \min_{\pi, \in \overline{Q}_{criteria}} Q_c(s, \pi_L(s)) \right) = \max_{\pi_L} \left( \min_{\pi, \in \overline{Q}_{criteria}} Q_c(s, \pi_L(s)) \right) \quad (8)
\]

And, we know that LexDiff seeks the action which minimizes the biggest regret. Minimizing regret relative to a criterion is equivalent to maximizing utility relative to this criterion. So LexDiff seeks the lowest utility maximizing action. In other words, because it is a local SE, we have:

\[
U_{\pi^*_L, \cdots, \pi^*_N}(s) = \max_{\pi_L} \left( \min_{\pi, \in \overline{Q}_{criteria}} Q_c(s, \pi_L(s)) \right) \quad (9)
\]

So, Eq.7=Eq.8 is equivalent to:

\[
\min_{BR(\pi^*_L)} U_{\pi^*_L, \cdots, \pi^*_N}(s) = \max_{\pi_L} \left( \min_{\pi, \in \overline{Q}_{criteria}} U_{\pi^*_L, \cdots, \pi^*_N}(s) \right) \quad (10)
\]

So the LexDiff operator leads to an SE. 

**Theorem 2.** The criteria of the 2V-DEC-MDP described in Eq.2,Eq.3 and Eq.4 do not lead to a Stackelberg equilibrium.

**Proof:** According to theorem 1 we aim to show that ER, JER or JEP are not an SE. By Eq.1, we have:

\[
\min_{BR(\pi^*_L)} ER(s, \pi^*_L(s)) = \max_{\pi_L} \left( \min_{BR(\pi)} ER(s, \pi_L(s)) \right) \quad (11)
\]

\[
\min_{BR(\pi^*_L)} JER(s, \pi^*_L(s)) = \max_{\pi_L} \left( \min_{BR(\pi)} JER(s, \pi_L(s)) \right) \quad (12)
\]

\[
\min_{BR(\pi^*_L)} JEP(s, \pi^*_L(s)) = \max_{\pi_L} \left( \min_{BR(\pi)} JEP(s, \pi_L(s)) \right) \quad (13)
\]

So, according to ER only, the objective will be to maximize the ER value. We then have, with \( \pi^*_L \) the leader’s policy resulting from the ER criterion:

\[
ER(s_L, \pi^*_L(s_L)) = \max_{\pi_L} ER(s_L, \pi_L(s_L))
\]

However, the equation:

\[
ER(s_L, \pi^*_L(s_L)) = \sum_{s'_L \in S_L} (P(s_L, \pi^*_L(s_L), s'_L) \cdot V^*(s'_L))
\]

supposes that only the best response policy \( \pi_i \) is assumed for each agent in the neighborhood when computing the value of ER. Indeed, \( V^*(s'_L) \) is the reward of the leader when it gets closer to its objective (ie. when \( s'_L \) is closer to the objective than \( s_L \)). However, even if an agent gets closer to a point according to a pure geographic point of view, it can in reality move away from this point because of the other agents: if they go between the leader and its objective, it will have to avoid them, what will imply additional movements. Another decision could make the leader to get closer to its objective without being constrained by the other agents. So we have:

\[
ER_{V-MDP} = \max_{\pi_L} \left( \min_{\pi, \in \overline{Q}_{criteria}} ER(s, \pi_L(s)) \right)
\]

while the equation for an SE will be:

\[
ER_{SE} = \max( \min_{\pi, \in \overline{Q}_{criteria}} ER(s, \pi_L(s)))
\]

But we are not sure that the agents will follow the assumed policies. We then have \( ER_{V-MDP} \leq ER_{SE} \) and thus:

\[
\min_{BR(\pi^*_L)} ER(s, \pi^*_L(s)) \leq \max_{\pi_L} \left( \min_{\pi, \in \overline{Q}_{criteria}} ER(s, \pi_L(s)) \right)
\]

So, ER does not lead to an SE, nor a 2V-DEC-MDP parametrized by ER, JER and JEP. 

**B. Rewriting criteria**

According to theorem 1 and 2, the leader does not follow an SE. We show how to adapt the criteria and the effects on the complexity in order to reach an SE.

**ER criterion:**

With \( d(s, g) \) the distance between \( s \) and the goal \( g \) if we don’t know what the other agents do, and \( d(x, g) \) the distance between \( s \) and \( g \) knowing the 1 to N agents’ policies, we have:

\[
ER(s_L, \pi^*_L(s_L)) = \sum_{s'_L \in S_L} T(s_L, \pi^*_L(s_L), s'_L) \cdot (-d(s'_L, g))
\]

Taking the other agents’ decisions into account, we can write:

\[
ER(s_L, \pi^*_L(s_L)) = \sum_{s'_L \in S_L} T(s_L, \pi^*_L(s_L), s'_L) \cdot (-d(s'_L, g))
\]
\[
\min_{BR(\pi_L)} \sum_{s'_L \in S_L} (T(s_L, \pi_L^s(s_L), s'_L) \cdot (-d_x(s'_L, g)))
\]

With \(d_x(s'_L, g) = \sum_s T(s, \pi(s), s') \cdot d_x(s'_L, g)\)

\[
= \sum_{s'} \left[ \prod_{i=1}^N T(s_i, \pi_i(s_i), s'_i) \right] \cdot d_x(s'_L, g)
\]

Where \(d_{s'_1, ..., s'_N}(s, g)\) is the distance from \(s\) to \(g\) without crossing \(s'_1\), nor \(s'_2\), nor ..., nor \(s'_N\).

Because we want to maximize this criterion, we will have:

\[
ER(s, \pi_L^s(s_L)) = \max_{\pi_L \in BR(\pi_L)} ER(s, \pi_L(s))
\]

**JER criterion:**

We said that JER was the following, with \(K(s) = 1\) if there are agents behind \(s\) and 0 if not:

\[
JER(s_L, \pi_L^s(s_L)) = \sum_{s'_L \in S_L} [T(s_L, \pi_L^s(s_L), s'_L) \cdot K(s'_L)]
\]

Taking the other agents’ policies into account, we write:

\[
JER(s_L, \pi_L^s(s_L)) = \min_{BR(\pi_L)} \left( \sum_{s'_L \in S_L} [T(s_L, \pi_L^s(s_L), s'_L) \cdot K(s'_L)] \right)
\]

Where \(K_s(s)\) estimates the probability that at least one agent stays behind \(s\) (the objective being not to break the platoon). \(K\) is defined by:

\[
s_b(s) = \{ s' \in S[S] \mid \exists s_i \in s' \text{ with isBack}(s_i, s) = \text{true} \}
\]

\[
K_s(s) = \sum_{s' \in s_b(s)} T(s, \pi(s_s), s')
\]

Where \(isBack(s^1, s^2)\) is a function which returns true if \(s^1\) is behind \(s^2\). To maximize the criterion, we will have:

\[
JER(s, \pi_L^s(s)) = \max_{\pi_L} JER(s, \pi_L(s))
\]

\[
\iff \min_{BR(\pi_L)} JER(s, \pi_L(s)) = \max_{\pi_L} \left( \min_{BR(\pi_L)} JER(s, \pi_L(s)) \right)
\]

**JEP criterion:**

The JEP criterion is the following, with \(C\) a negative constant which represents the cost of a collision between two agents:

\[
JEP(s_L, \pi_L^s(s_L)) = \sum_{s'_L \in S_L} T(s_L, \pi_L^s(s_L), s'_L) \lambda
\]

where \(\lambda = \sum_{s_j,j=1}^N T(s_j, \pi_j(s_j), s'_j) \cdot C\)

We can rewrite JEP to consider the other agents policies. We write, with \(p_L = T(s_L, \pi_L^s(s_L), s'_L)\) and with \(p_j = T(s_j, \pi_j(s_j), s'_j)\):

\[
JEP(s_L, \pi_L^s(s_L)) = \sum_{s'_L \in S_L} \left[ \prod_{j=1}^N \left( \min_{s_j \in BR(\pi_L)} p_j \cdot C \right) \right]
\]

Because we aim to maximize the criterion, we will have:

\[
JEP(s, \pi_L^s(s)) = \max_{\pi_L} \left( \min_{BR(\pi_L)} JEP(s, \pi_L(s)) \right)
\]

So, we rewrote the 3 criteria so that they all lead, independently of each other, to an SE. Thus, if the leader follows those criteria, it will be in an SE.

**V. Complexity**

We will now compare the complexity of our initial formalism to the one with an SE. We will make this comparison on the ER criterion (those results are the same for JER and JEP).

**A. Initial criteria**

The value of \(ER(s, a)\) depends on the distance. So, to find the best action, we will have to compute \(A\) times a distance (or \(K\), \(C\), \(A\) being the number of different actions an agent can do. Thus, if we write \(d\) the time to compute a distance, the complexity for this criterion will be \(O(A \cdot d)\), but \(g\) (the goal) is not a data of the problem: the agent has to estimate the distance between him and every neighbor to estimate which one is the nearest so, with \(N\) the number of agents in the neighborhood, the global complexity is \(O(N \cdot A \cdot d)\).

**B. Stackelberg adapted criteria**

A leader first have to estimate the different policies for all the agents. The complexity to estimate the policies of an agent is \(O(A \cdot d)\), because those agents apply the “normal” criteria. So the global complexity for this step is \(O(N \cdot A \cdot d)\).

We then compute \(d_x(s, g)\) \(X\) times, \(X\) being the number of different \(BR(a)\). With \(D\) the complexity for computing \(d_x(s, g)\), global complexity for this step is \(O(X \cdot D)\). In the worst case, the value of \(X\) is \(A^N\). The \(D\) value depends on how many times we have to compute a distance, i.e. the number of different possible tuples \((s'_1, ..., s'_N)\). Because an agent can end up in 3 different states after applying an action, we have \(3^N\) possible tuples. Complexity for \(D\) is \(O(3^N \cdot d)\).

Global complexity for computing the value of an action according to \(ER\) is then \(O((N \cdot A \cdot d) + (X \cdot D)) = O((N \cdot A \cdot d))\).
\(A \cdot d) + (A^N \cdot 3^N \cdot d))\). We can then estimate the complexity for computing the best action according to \(ER\), which is

\[
O(A \cdot \left[ (N \cdot A \cdot d) + (A^N \cdot 3^N \cdot d) \right]) = O((N \cdot A^2 \cdot d) + (A^{N+1} \cdot 3^N \cdot d))
\]

Thus, according to this criterion, complexity is much higher with Stackelberg.

VI. EXPERIMENTAL AND ROBOTICS RESULTS

In this section, we describe different tests we made to validate our approach. We first made several tests on a simulator, to analyze the computation times and the quality of the policies. Then, we tested our approach on a flock of robots and successfully had them to build a platoon.

A. Experimental results: with and without SE

We made a simulator in order to test our formalism, where agents’ behavior is directed by a 2V-DEC-MDP parametrized by the way we described in the previous sections. Actions are stochastic (each time an agent moves, it can slip on its side) and we use no communication at all. An agent can observe its neighbors, but do not know anything about the remaining agents. Fig.1 represents the situation on which we made our tests: circles are agents and polygons are locations (dark patterns are obstacles). The objective is, for the agents, to move from their initial position to a given objective, while building a platoon. We made several tests:

- with 7 agents, running the simulator 10 times with and without SE, to compare complexity and quality,
- with a chosen leader, running 5 simulations with 1, 2, . . . , 7 agents in its neighborhood, to analyze complexity.

We summarize results of our tests in the following figures. They show results from the environment presented before, as an example, but we did some tests with other initial configurations and other environments.

Fig.2 shows platoon’s distance (according to its objective) evolution over time. Distance is a good mean to estimate the platoon’s behavior quality, because it shows how fast the group is able to move. We can see that a platoon which considers a SE moves exactly at the same speed as a platoon which does not consider it. Thus, quality of the behavior following SE or not are almost the same.

There is an interesting point here: at the end of its evolution, the platoon moves a little faster without SE than with it. Why this difference? With SE, the platoon’s leader is more prudent: it chooses to move slowly, to be sure not to break the platoon. Although, this difference is not representative of the global platoon behavior: during most of the time, there is no difference at all.

So, it seems that a platoon using our formalism acts as well as a platoon using a SE. Now, what about the complexity?

![Platoon movement](image)

**Figure 2.** Distance to the objective

![Scalability](image)

**Figure 3.** Computation time

Fig.3 shows the complexity according to the number of agents in the neighborhood. When we don’t use an SE, the complexity seems to be proportional to the number of agents. This is trivial according to the equations of \(ER\), \(JER\) and \(JEP\): they depend on the agents in the neighborhood. Time needed grows slowly with the number of agents. We made some tests with agents starting scattered in the environment: computation time then stay under 0.01 second even with 50 agents.

Complexity is exponential with an SE. When more than 7 agents are in the neighborhood of the leader, time for computing an action becomes too high to be tractable by our simulation. Complexity is then much better with our formalism which can deal with problems up to 50 agents.

During tests on other situations, with other environments, results were the same than the ones depicted in Fig.2 and Fig.3. Thus, a platoon using our formalism acts as well as
a platoon following SE, with a clearly better complexity.

B. Implementation on a flock of robots (Koalas)

We made tests on 3 robots running a 2V-DEC-MDP. They know the “map” of their environment and have local observability. We placed them on a line and put an objective in front of them. In Fig.4 are captions from a video of those tests.

![Figure 4. Platoon formation](image)

When the test starts, the closest robot to the objective chooses the $ER_1$ function and go toward its objective. Because of the $JER$ function, it moves slowly enough to let the other agents time to follow it. In the same time, according to the $ER_2$ function, one chooses to follow the first agent, while the other chooses to take the third place.

The platoon then emerges from those interactions: in Fig.4 we can see the platoon formation in a short time (37s) while some greedy multiagent methods take more than 1mn.

Many other initial configurations were considered and we can see that, for each configuration, robots fully form a platoon after some moves.

VII. EXTENSIONS TO GENERAL FORMATIONS

In this section, we describe how to extend this work, in order to deal with more complex interactions between agents (such as non-static shapes).

A. Extended model

We are currently working on an extended model, based on the one presented above but able to deal with different kind of interactions. Using a 2V-DEC-MDP, we describe if an action helps or penalizes the other agents, but what about complex interactions? Our idea is to enumerate all the possible interactions, and to associate a reward function, such as each action would bring an individual reward and an “interactions” reward. Figure 5 describes how we enumerate the interactions.

![Figure 5. Space compartments](image)

Here, the space around the agent is divided in $8l$ areas, with $l$ the line of sight of the agent. Then for each area, the agent observes if it contains neighbors or not. The interaction reward of an action will be based on the set of occupied areas. Using such an approach, we can give a reward to each possible shape, and solve the decision problem using these rewards.

It is then possible to represent complex scenarios, with non-static interactions. For example, we could describe a problem where the agents are moving in a line shape (by giving it a high reward) and where the situation evolves and a new shape becomes preferred (simply by changing the associated rewards).

B. Practical example

We are currently working on a practical example, in collaboration with the Thales group (TOSA). In this example, an autonomous platoon is moving from a start position to an objective. The shape of the platoon depends on the situation:

- a straight line, in dangerous places where a “strong” robot opens a way for the other agents,
- agents scattered in the environment (moving to their objective without getting close to each other but staying far enough to see each other), in situations with no danger where the robots are exploring the environment,
- specific shapes, in situations where the agents are specialized and choose their position in the platoon according to their task,
- ...

Such a problem could be useful for planet exploration problems, with robots such as Mars Rovers. Figure 6 shows an example of a group exploring their environment, using a scattered shape.

Here, the rover at the center of the map observes three agents around it, and chooses its actions according to them (in order to optimize the exploration). It does not need to observe the fourth agent, which is too far to be considered. If these agents encounter a dangerous area, the rewards (associated to the possible shapes) change, and the agents move to a new formation (Figure 7).
In such a problem, we expect using several robots, each of them with specific capacities, and making them collaborate. Computing explicit collaboration in such situations is a very hard problem, especially when dealing with uncertainty and no communication. In those settings, our approach is able to “break” the combinatorial complexity and to build a good group behavior.

VIII. CONCLUSION

In this paper we addressed the platoon formation problem with no communication, partial observability and stochastic actions. We shown how to transform this problem into a flocking problem and we formalized it using 2V-DEC-MDP which is the most suitable decision model in this context. To evaluate the performance of our approach, we related 2V-DEC-MDP to stochastic game and we used SE as a theoretic concept for optimality. We shown that the initial 2V-DEC-MDP leads to a near SE with a weak complexity while its adapted version can lead to SE with a very high complexity.

Such an approach could apply, for example, on Mars Rovers problems. We are currently working on more general models able to deal with such problems and to describe complex and non-static interactions. We made experiments on a simulator and on a flock of robots and we were able to compute good policies for platoon formation problems.

In future works, we intend to solve complex scenarios, with a group of agents working on a joint task. Another direction consists of considering a unit composed with robots and operators interacting to accomplish complex missions such as exploration, recognition and mapping.

REFERENCES


