A MIXED INTEGER LINEAR PROGRAMMING APPROACH FOR STRATEGIC ISSUES IN ROBOFLAG

Nicolas Foirien * **Richard M. Murray** **

* *Ecole Nationale Supérieure des Mines de Paris*
nicolas.foirien@ensmp.fr
** *Control and dynamical systems, California Institute of Technology, Pasadena, CA 91125*
murray@cds.caltech.edu

Abstract: Many robotics applications involve teams of agents that have to act in a cooperative manner, often corresponding to several tasks to accomplish. For dynamic vehicles, with an adverse and thus strategic context, the nature of the optimization problem is hybrid and gathers allocation and optimal trajectories problems. In this paper a mixed integer linear programming approach is developed to bring a solution to these types of strategic issues, facing the challenge of finding a real-time solution. The problem is presented with the context of RoboFlag, motivated by the hardware implementation at Caltech.

Keywords: co-operative control, autonomous vehicles, decision making, linear programming, dicrete event systems, real-time

1. INTRODUCTION

In many decision problems, several kinds of choices have to be done. For instance, in air traffic control (Bayen and Tomlin, 2003), decisions correspond to discrete and continuous variables (order and times of landing for a set of airplanes). Similar issues arise in the field of cooperative control where several agents have to act together optimally to accomplish several tasks, in particularly when a strategic dimension appears.

In order to validate experimentally theoretical advances in this domain, RoboFlag, a "Capture the Flag"-like game has been developed at Caltech, using the multi-vehicle Laboratory. In a few words, the game involves two teams whose goal is for each one to protect its own flag located in its "home zone" and to try to seize the flag of the other team without being "tagged" or intercepted by an opponent. Precise rules can be found in (D'Andrea and Murray, 2003). As an adversary game, each team has to take strategic decisions that depends on the situation of its opponents. Due to the number of vehicles, a basic solution is quite hard to build, therefore, the autonomous control system has to have a high level control procedure able to take strategic decisions. This is precisely the problem addressed here by focusing on a cooperative defense strategy in the context of the RoboFlag application.

To make the game closer to real-world applications, the agents considered here have highly dynamic properties and only a local vision of the field. This has deep consequences in terms of strategy because decisions have to be taken in a dynamic and only partially known environment. A hybrid nature for the problem arises because the defense team has to decide, at a same time, how to explore the environment and how to intercept the attackers previously observed. Here, instead

of decomposing the problem, a unique description would be prefered. The mixed integer linear programming (MILP) can handle this type of hybrid problem. This technique has already been used in the area of control to deal with problems of collisions in multi-vehicles applications (Richards and How, 2002). Here, the nature of the problem can often change since new attackers can be detected. The originality of this approach is to apply a MILP formulation for a strategic problem whose nature is subject to frequent changes.

This paper begins by a description of the problem in RoboFlag context, followed in section 3 by a presentation of a first algorithm whose purpose is to serve as a reference to compare performances of algorithms. The MILP formulation is exposed next. Section 5 gathers some implementations issues and summarizes the results. Finally, some conclusions and perspectives are discussed.

2. PROBLEM FORMULATION

This section describes the problem in the context of RoboFlag, the application that motivates this study. First, the field on which the agents can move is modeled by a rectangular shape of width a and height b whose middle of the bottom edge will be the origine. Thus, the coordinates (x,y) for a given object on this field satisfy: $-a/2 \leq x \leq a/2$ and $0 \leq y \leq b$. Two set of agents can act, the defense team, which will be referred to the blue team, tries to protect a zone called the "Defense zone", whereas, the goal of the other team, the attack team (also called the red team), is to penetrate into this zone. This corresponds to the adversary nature of the game. Here, the "Defense zone" will be represented by the area below the bottom edge of the rectangle. Thus, the goal of the defense team is simply to prevent any attacker to cross the bottom line Z or defense line from now on, where $Z = \{(x, y)| - a/2 \leq x \leq a/2, y = 0\}$.

In this model, the defense team considered is composed of K agents with particular properties whose purpose is to reflect the main characteristics of RoboFlag from a strategic point of view. First, the vision capacity is restricted to a rectangular shape: for a given agent k in (x_k, y_k), an opponent situated inside $\{(x, y)|x_k - w_k/2 \leq x \leq x_k + w_k/2, y_k \leq y \leq y_k + h_k\}$ is detected by this agent (with $h_k < b$). A defender can also intercept or tag, in RoboFlag terminology, an opponent if they are located in a close neighborhood from each other. This can be parameterized, for a given agent k, by a "tagging range" g_k: a defender can intercept the attacker located in (x, y) if its position (x_k, y_k) satisfies $x_k - g_k/2 \leq x \leq x_k + g_k/2$ and $y_k = y$. Furthermore, it is assumed that there is no restriction and delay in terms of communication: any information is immediately shared within the team. A synthetic description can be viewed on figure 1.

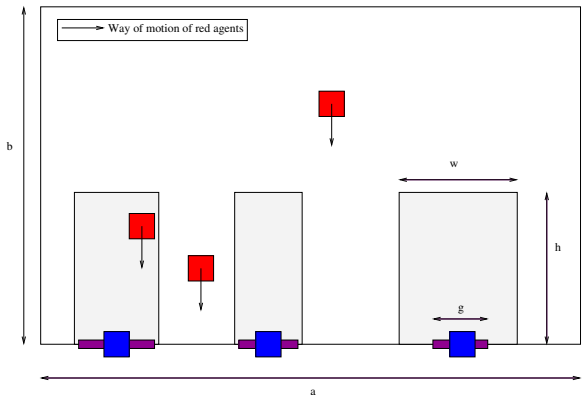

Fig. 1. Model of the RoboFlag defense problem (In blue the defense team, in red the attackers).

In RoboFlag application, the agents are vehicles with nonlinear dynamics, however a linearization of the problem gives a good approximation. A restriction to only one dimension is also a reasonable hypothesis since in the application, the robots have to defend a circular zone by patrolling around it. Thus, the defenders are restricted here to move on the x-axis ($y_k = 0, \forall k$). Here, to simplify the presentation without losing the essence of the problem, a simple and discretized over time dynamic will be considered: $\forall t,\ \forall k \in [1, K]$

$$x_k^{t+1} = x_k^t + u_k^{t+1} \text{ with } u_k^{t+1} \in [-u_{max}; u_{max}] \quad (1)$$

Given this presentation, the goal for the defense team is to intercept any potential attacker that would reach the defense line. Generally speaking, in any strategic game, a player has to have an idea on how his adversary takes its decisions to choose his optimally. Here, it is not claimed to design a strategy of defense that can cope with the cleverest ways of attacking. The goal is rather to show, given a simple model for the attack, how to find an optimized strategic behavior to respond to this threat. Thus, before going further, a model for the attack has to be defined.

2.1 A model of attack

As a first model, the attack team considered is composed of M agents whose motion is dictated by precise and simple rules. Any attacker m, whose position at time t is repered by (X_m^t, Y_m^t), starts at a random time t_m^0 from the top of the field ($Y_m^{t_m^0} = b$). $X_m^{t_m^0}$ is randomly chosen on $[-a/2; a/2]$ according to a uniform distribution. The random value t_m^0 is built with the following principle: at time t if t_m^0 is still not defined, either $t_m^0 = t + 1$ or it remains undefined with

probability p and $1-p$ respectively. It is also assumed that the attackers cannot move across the x-axis ($X_m^t = X_m^{t_m^0}, \forall t \geq t_m^0$). On the y-axis, the dynamic is similar to the one of defense and the attackers move at maximum speed in direction of the defense zone: $Y_m^{t+1} = Y_m^t - u_{max}$. As soon as an attacker has reached the defense line, it can begin a new attempt, according to the same independent laws described above. Thus all agents act independently from each other. Roughly speaking, the model of attack considered consists of independent agents starting at random times, uniformly distributed over the x-axis that move straight in the direction of the defense line until they have reached it.

2.2 Evaluation of performances

Since here the goal is rather to get the best behavior on average, a natural way to measure the performances of strategies is to have a statistical approach. This corresponds to observe a great number of sequences of the game and simply record the number of opponents the defense team was able to intercept. A rate of interception can be easily used to assess the various strategies.

3. A NAIVE SOLUTION

Generally speaking, given a problem or a task to accomplish for which the level of optimality is difficult to assess, the way to make an evaluation is to compare, or at least to have a reference so that the performance become relative. Here as well, a reference, a basis to assess optimized strategies, has to be built. The idea is to find a naive algorithm based on basic principles that can provide a first solution. The challenge is then to find a better solution and compare the difference, knowing the level of complexity introduced in the naive algorithm. In this problem, due to the limited sensing capacity, the team has to explore the environment to detect its opponents, and then to intercept them. By assuming that it is always better for the defense team to intercept a known target rather than to discover other opponents, the following rules can be applied as an algorithm for any agent k of the defense team:

- if the defender k has detected an attacker denoted by m (interception mode), it moves in a constant direction until its position x_k satisfies $x_k - g_k/2 \leq X_m \leq x_k + g_k/2$ (condition for interception). Then, it waits here until the interception ($y_k = Y_m = 0$)
- else (exploration mode), given an initial way of motion, it moves in this direction until it has reached its neighbor corresponding to that direction (more precisely, the definition of "reaching its right neighbor", for instance, corresponds to the condition $x_k + w_k/2 \geq x_{k+1} - w_{k+1}/2$). Then, it starts moving in the opposite direction.

For $k = 1$ and $k = K$ respectively, the natural extension is made by adding the extremities of the field as left and right neighbors respectively ($x_0 = -a/2$ and $x_{K+1} = a/2$). Obviously, this algorithm is suboptimal and even with extra basic principles it is clear that a better algorithm can be designed; it has to be viewed as a reference for comparing more powerful algorithms.

4. FORMULATION AS A MIXED INTEGER LINEAR PROGRAM

4.1 Stakes and advantages for a strategic problem

To find a reliable strategy for this problem, a natural way is to consider it as an optimization formulation. Two difficulties arise with that approach: first, because of the limited sensing capacity, the structure of the problem to solve is dynamic since at each step of time, new opponents can be observed leading to a real-time optimization problem. Secondly, the nature of the optimization problem comprises actually two components:

- a resource allocation problem: which defender is going to intercept a given target?
- a path planning problem: what is the best trajectory for each agent to optimize a given objective and respect an allocation?

Considering these problems separately suffers from one main point: it is indeed not clear which is the best allocation a priori, before computing optimal trajectories. To avoid these suboptimal solutions, the whole problem can be considered as a global one that can handle the decision variables for the allocation problem and the continuous ones for the control design. The mixed integer linear programming (MILP) is a powerful tool that can deal with problems of this nature, mixing two kinds of variables. It is an extension of linear programming to the class of problems in which some of the variables are constrained to be integer (Floudas, 1995). The MILP approach is particularly interesting here except that it requires expressing the problem, the objective and constraints, linearly with the variables selected. Moreover, the NP-completeness nature of a MILP formulation implies that the size of the problem, correlated to the number of variables, has to remain rather small, especially here because it is used for a real-time application.

The MILP problem, the core of the algorithm, takes indeed place in the main loop of the algorithm: first, at time t, each defender observes

the portion of the environment it can sense. This defines for the whole team a set, that may be empty, composed of M' targets ($0 \leq M' \leq M$). Based on that, an optimization problem (MILP formulation) is solved. Then, for each agent k, the control u_k^{t+1} is applied, meanwhile each attacker moves according to the dynamic defined previously.

4.2 Variables

To design a strategy of defense, the first thing to do is to choose on what the decisions are going to be based. As any strategy, a long-term vision has to be considered which corresponds to a classic receding horizon approach. The horizon will be fixed and equals to H. Thus, the continuous control variables involved here are at time t are represented by the matrix $u^t = (u_k^{t+t'})_{1 \leq k \leq K, 1 \leq t' \leq H}$. The allocation variables can be represented by binary variables d. For a given agent k and a target m', $d_{k,m'} = 1$ if this agent is allocated to m' and $d_{k,m'} = 0$ if not. This provides a complete description for discrete decisions, convenient to express constraints and objectives.

4.3 Expression of linear constraints

There are two main kinds of constraints in this problem, first the agents have to respect their own dynamics and second the team has to intercept the selected targets. With the dynamic introduced previously, the constraints at time t are easily expressed by: $\forall k \in [1, K]$, $\forall t' \in [1, H]$,

$$u_k^{t+t'} \leq u_{max} \text{ and } u_k^{t+t'} \geq -u_{max} \qquad (2)$$

Here, due to the simple form of the dynamic, the problems of collisions can be ignored. However, for more general dynamics, some other constraints have to be added: $\forall t' \in [1..H]$,

$$x_k^{t+t'} \leq x_{k+1}^{t+t'} \quad \forall k \in [1..K-1] \qquad (3)$$

$$x_1^{t+t'} \geq -a/2 \text{ and } x_K^{t+t'} \leq a/2 \qquad (4)$$

To formulate the interception problem with linear constraints, the basic idea is to use an arbitrary large constant C whose combination with binary variables will relax or not the constraint, depending on the decision for the allocation. Let us denote I_t the set of the M' agents known at time t, then the constraints can be expressed by $2.K.M'$ inequalities: $\forall k \in [1, K]$, $\forall m' \in I_t$

$$x_k^t + \sum_{t'=1}^{t^f_{m'}} u_k^{t+t'} + g_k/2 \leq X_{m'} + C(1 - d_{k,m'}) \qquad (5)$$

$$x_k^t + \sum_{t'=1}^{t^f_{m'}} u_k^{t+t'} - g_k/2 \geq X_{m'} - C(1 - d_{k,m'}) \qquad (6)$$

Therefore, if the allocation $(k - m')$ is chosen, the previous inequalities correspond to the interception constraint and otherwise, the constant C implies that the inequalities are always satisfied whatever the values of control variables. At this point, based on the simple observation that only one agent is required to intercept a target, a principle of parsimony can be expressed by the inequality:

$$\sum_{k=1}^{K} d_{k,m'} \leq 1 \quad \forall m' \in I_t \qquad (7)$$

4.4 Objectives

Finding a linear criterion whose optimization would lead to the goal desired is not an easy task here especially because of the limited vision properties of agents. As noticed previously, the agents have to have conjointly two types of behaviors: intercepting the observed targets and exploring the environment to detect new ones. Therefore two criterions need to be established to reflect these two behaviors but also the balance between them that will be expressed here by a linear combination.

Thanks to the allocation variables, the objective relative to the largest number of targets to intercept is direct to express:

$$max \sum_{m' \in I_t, k \in [1..K]} \gamma^{\sigma(m')} d_{k,m'} \qquad (8)$$

The γ parameter ($1/2 < \gamma < 1$) is aimed at producing a preference among the targets. For that purpose, the bijection σ over I_t on $[1..M']$ is used ranking each target according to its proximity to the defense line ($\sigma(m') = 1$ if m' is the closest target to intercept). The goal of this order is to deal with situations in which the team cannot intercept all the targets. The simple principle applied here is that, for a same number of targets to be intercepted, the best decision is to choose the closest targets so that the team is available for other tasks as soon as possible.

A linear criterion for an exploration purpose is much more delicate to find. Here, the approach is first to build a good one of any kind and then to proceed to a linearization. Exploring means here to have a good global knowledge of the environment. Observing always or too often the same region does not bring much information since any agent k can sense what happens at

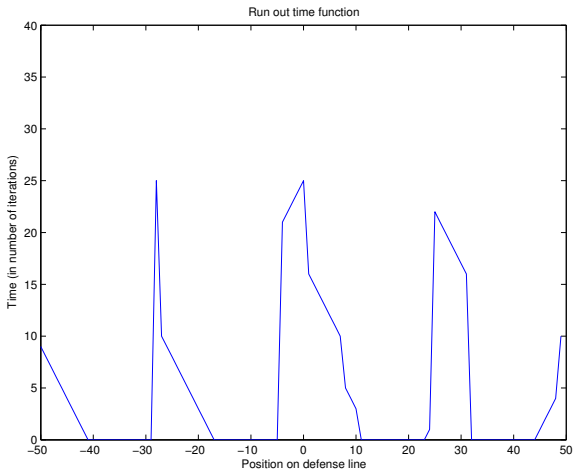

Fig. 2. T-function for $K = 4$ defenders ($w_k = 0.1.a, \forall k$)

distance h_k in front of it. Therefore, to have an effective exploring behavior requires recording the past actions of the team. A simple and convenient way to do so is to define a function T_t over the space $[-a/2; a/2]$ that associates for every x in this interval the time run out between the current time t and the time when a member of the team has observed x for the last time. This function synthesizes the past actions of the whole team and is also a useful representation to decide which regions need to be observed in priority. With this data, the goal of the team is to reduce globally this function. One way to capture synthetically the information contained in T is to consider an integral expression of it. Moreover, to introduce a weighting to differentiate the positions recently visited of the others (where the probability to detect an attacker is higher since it has not been sensed for a long time), to consider the function T^α where $\alpha > 1$ instead of T is better. Given the horizon of time H, the goal is then to minimize:

$$\Phi_t = \sum_{t'=1}^{H} \int_{-a/2}^{a/2} [T_{t+t'}(x)]^\alpha dx \qquad (9)$$

Φ_t can clearly be viewed as a function of future vectors of controls for the team $(u^{t+1}, ..., u^{t+h})$. An operator Θ can be readily built expressing the iterative dependency of T_{t+1} in T_t and u^{t+1}: if x is observed by any agent (i.e. $x \in \bigcup [x_k^t + u_k^{t+1} - w_k/2; x_k^t + u_k^{t+1} + w_k/2]$)

$$T_{t+1}(x) = \Theta(T_t, u^{t+1})(x) = 0 \qquad (10)$$
$$= T_t(x) + 1 \text{ otherwise} \qquad (11)$$

It is then immediate that T_t is a piecewise linear function. An example, of T_t is shown on figure 2.

To provide a linear criterion c_{Φ_t} of Φ_t depending on $(u^{t+1}, ..., u^{t+h})$ a first approach is to linearize by a least square (LSQ) approximation using an iterative technique based on the observation that the T-function does not change a lot between two steps of time. Knowing that K can be large, a decentralized approach has to be used by considering T as a set of K local functions $\{T_t^k,\ k \in [1..K]\}$ that depend only on the vector u_k and restricted over the neighborhood of k (the space between its two neighbors). Precisely, $\forall k \in [1..K]$, $T_t^k(u_k) = I_{[x_{k-1}; x_{k+1}]}.T_t(u)$, where I is the indicator function. With all the previous elements, the objective function of u^t and d to minimize at time t is:

$$\sum_{k=1,t'=1}^{K,H} c_{\Phi_t}^{k,t'}.u_k^{t+t'} - \beta \sum_{k=1,m' \in I_t}^{K} \gamma^{\sigma(m')} d_{k,m'} \qquad (12)$$

under the constraints (5), (6) and (7). β is a strictly positive parameter experimentally adjusted.

5. IMPLEMENTATION AND RESULTS

5.1 Software and implementation issues

The MILP problem was solved by using CPLEX which is a powerful solver that can handle this type of formulation. The modeling language AMPL (Fourer *et al.*, 1993) was used to avoid expressing the problem directly into the requested format. One main advantage of AMPL is that it is really easy to translate the mathematical form of the optimization problem into AMPL language. The problem is decomposed into two files, a model file that gathers the form and architecture of the optimization formulation and a data file that provides the parameters for the problem. AMPL translates into the standard required by CPLEX (MPS-file). AMPL/CPLEX were called by a Matlab script before it has generated the data for the MILP problem at time t. This requires writing through Matlab a data file at each iteration. The simulations were run under Linux on a 0.7GHz PC with 256MB RAM.

As said previously, a MILP problem is NP-complete (see (Till *et al.*, 2002) for MILP and complexity issues) and since it is used here for a real-time application, some precautions have to be taken assuming that the number of defenders and attackers at a same time can be important. One example used here to reduce the size of the problem is to discard allocations that are clearly suboptimal. Thus, the CPU time to interpret the AMPL model and solve it with CPLEX requires for $K = 10$ and $M = 40$ 0.1 second in the worst cases and 0.05 second on average. However, since the model almost changes at each iteration, this requires writing frequently a new data file. This slows down the actual execution time that goes

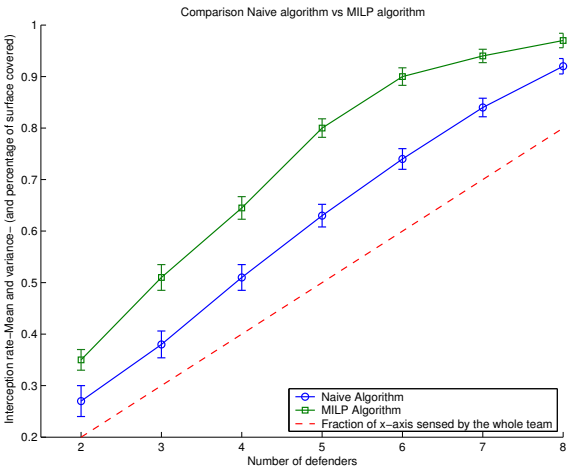

Fig. 3. Comparative results between the naive and MILP approaches ($w_k = 0.1.a$, $h_k = 15$ and $g_k = 0.01.a$, $u_{max} = 1$).

from 0.5 to 1 second per iteration. One way to deal with that problem will be to express directly the problem in CPLEX language.

5.2 Results and comparisons

The results got with the simulations show that in every case tested, the MILP approach performs better than the naive algorithm. The graph on figure 3 is an example that summarizes the rates of interception for both algorithms measured on 80 experiments of 1000 iterations each. In both simulations, the number of attackers M was chosen equal to 10, and the probability parameter p (roughly speaking, p set the frequency of apparition of attackers) was 0.1. The graph requires a few comments. First, the performance of the naive algorithm is approximately linear on the number of defenders whereas the MILP provides a concave curve. The gap between the algorithms reaches its maximum for a team composed of 5 defenders which corresponds to a coverage of 50% of the x-axis at a same time. Intuitively, the cooperation is more efficient in this situation because the distance between each defender is sufficiently small so that defenders can switch targets between each other; this cannot be often realized for small numbers K of defenders. On the contrary, when K gets bigger, the number of targets per agent is too small to take advantage of an allocation procedure.

6. CONCLUSION AND FUTURE WORK

This work showed how to use a MILP method for strategic problems with the framework of RoboFlag application. It has also been exposed the difficulties that arise with the dynamic nature of a strategic problem in the context of a real-time

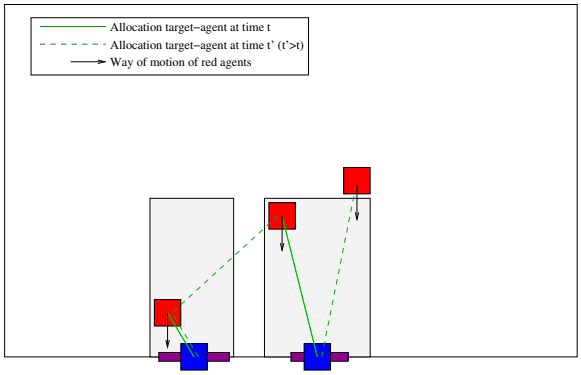

Fig. 4. Illustration of a dynamic of allocations

application and the way to cope with them. The results got with the MILP approach are encouraging to extend these ideas to much more complex model of strategies implying a probability approach. In the future, the algorithm can be implemented on the hardware structure of RoboFlag at Caltech to validate these results.

Another direction currently studied is to understand the performance of a team of heterogeneous agents in a strategic and cooperative context depending on their respective competences and properties. A coarse analysis approach, based on averages of relevant characteristics of behavior (like positions), is developed hoping to get an estimation of performances depending on the various parameters. Thus, the best defense structure can be optimally chosen a priori for a given problem.

REFERENCES

Bayen, A.M. and C.J. Tomlin (2003). Real-time discrete control law synthesis for hybrid systems using MILP: application to congested airspace. In: *Proc. American Control Conference.*

D'Andrea, R. and R. Murray. (2003). A behavioral architecture in strategy execution in RoboFlag game. In: *Proc. American Control Conference.*

Floudas, C.A. (1995). *Nonlinear and mixed-integer programming - Fundamentals and applications.* Oxford University Press.

Fourer, R., D.M. Gay and B.W. Kernighar (1993). *AMPL, a modeling language for mathematical programming.* The Scientific Press.

Richards, A., and J.P. How (2002). Aircraft trajectory planning using mixed integer linear programming. In: *Proc. American Control Conference.*

Till, J., S. Engell, S. Panek and O. Stursberg (2002). Empirical complexity analysis of a MILP-approach for optimization of hybrid systems. In: *Proc. IFAC Conference on Analysis and Design of Hybrid Systems.*