22

Reinforcement Learning to Support Meta-Level Control in Air Traffic Management

Daniela P. Alves[1], Li Weigang[2] and Bueno B. Souza[2]
[1] *Brazilian Institute of Information in Science and Technology - IBICT,*
[2] *Department of Computer Science, University of Brasilia - UnB, Brazil*

1. Introduction

In a complex environment where the messages exchange intensively among the agents, a difficulty task is to decide the best action for new arriving messages during on-line control. A typical form of communication in Air Traffic Flow Management (ATFM) is verbal language to exchange the information and there is almost no digital recording of this communication between pilot and air controller. As the development of distributed computation system [1], the digital message communication is proposed especially for processing the immense amount of messages.

In a real case such as the First Integrated Center of Air Defense and Air Traffic Control - CINDACTA I, in Brasilia, the system monitors 70% of the traffic flow in Brazil. According to the air traffic control procedure in CINDACTA I, Flight Information Region of Brasilia - FIR-BS is divided into 14 sectors and managed by 3 regional supervisors (Sao Paulo, Rio de Janeiro and Brasilia). Every air controller monitors his sector and is responsible to his supervisor. Only supervisor can make a decision to manage the air traffic flow. Any action is realized by air controller according to the decision of supervisor. For example, a decision may be to hold a flight in an airport for more 10 minutes, or assign the priority to another flight in the landing processing.

In CINDACTA I, the monitor system (equipment and operation software) is suitable, but there is no a general system to manage and synchronize the traffic dynamically and to support the decision for adequate traffic management. Supervisor makes the decision just by his experience without quantity analyses of the impact of the action. To resolve this kind of problem, some researches were developed using the solution of Artificial Intelligence and others according to the new conception of ATFM.

A distributed ATM system has been studied in Australia [2]. The advantages of that approach are inherent distribution, autonomy, communication and reliability. Prevôt, from NASA Ames Research Center, has studied a distributed approach for operator interfaces and intelligent flight guidance, management and decision support [3]. An application of multi-agent coordination techniques in ATM, which sets up a methodological framework using multi-agent coordination techniques that supports the collaborative work in ATM has also been presented recently by Eurocontrol [4]. It should be mentioned that the multi-agent

Source: Reinforcement Learning: Theory and Applications, Book edited by Cornelius Weber, Mark Elshaw and Norbert Michael Mayer
ISBN 978-3-902613-14-1, pp.424, January 2008, I-Tech Education and Publishing, Vienna, Austria

coordination technique is a useful methodological framework, however the research in [4] is limited to a software shell. Due to the huge amount of information traffic in ATFM, its implementation may not be straightforward when brought into practical fields.
A multi-agent system (MAS) for ATFM in grid computing - ATFMGC, was developed recently [5]. The research proposed an approach of cooperation and negotiation among agents using grid computing in a real time traffic synchronization problem. In some aspects such as the agent functions, their knowledge representation and inference processes [6] were developed. Standard of Balancing among Airports (SBA) as a criterion was also used to balance and measure the amount of communication among agents (i.e. airports) and the tolerated delay of the flights [5].
On the other hand, some problems have appeared for the fact of the intense exchange of messages in ATFMGC related with more than 10 airports. For a fixed SBA, it is impossible to efficiently equilibrate the communication. It is necessary to adapt a suitable mechanism to assist the decision process. The System of the Application and Management of the Decision Support Air Traffic Flow Control - SISCONFLUX is still in development [16]. Within this system, the idea is to introduce Meta-Level Control approach [7, 8] in ATFM. During information process, the reinforcement learning [9] is inserted to acquire experience to make Markov decision process more efficiency.
Meta-Level Control approach was developed by Raja and Lesser since 2003 [7, 8] for Multi-Agent System. This research proposed a Module of Evaluation and Decision Support (MAAD) in SISCONFLUX for ATFM with the combination of the Meta-Level Control approach and reinforcement learning algorithms (Q-leanring and SARSA [10]). With the developed system, four strategies of the modifications of the parameters in the Q-learning and SARSA algorithms are studied during reinforcement learning simulation, such as the cases of initial heuristic (IH), epsilon adaptative (EA), performance heuristic (PH) and the combination of above three (IH + EA + PH). The results from simulation and analyses show satisfactory in the traffic management process.
The chapter is organized in five parts: soon after this introduction, section II presents a general view of Meta-Control and algorithms of reinforcement learning. In the third section, the proposed system SISCONFLUX and principal model of this research - MAAD are described including control process and learning algorithms. A case study is illustrated and discussed in section IV. Last section presents the conclusions and future research.

2. Meta-level control and reinforcement learning algorithms

2.1 Meta-level control

Meta-level control is defined as the ability of complex agents operating in open environments to sequence domain and deliberative actions to optimize expected performance [7, 8]. It is an important topic in automatic control and some related fields. Russel and Wefald [11] presented a general approach of meta-reasoning and named as Principles of Meta-Reasoning. Some applications and related works were intensely studied by Raja and Lesser such as [7, 8, 11]. In these researches, an additional layer of meta-control that acts above of the component of control in the system to help the decision process. Reinforcement learning is also introduced to make the agents to learn knowledge from experience for better to make the decision.
The actions of the intelligent agents in the classic architecture are classified in two types: action and action to recover property of control, as the research in [10].

Considering the Meta-Reasoning, Raja and Lesser [7, 8] mentioned another action: the action of Meta-Level Control. It is a process of optimization of the performance of agents for choosing the sequence of actions to recover property and the deliberative of the activities [8].
Reactive controls may be suitable for situations with high restrictions such as: real time, limited resources, among others. There are five types of event triggers that require Meta-Level decision making [8]:

- Arrival of a new task from the environment;
- Presence of a task in the current task set library (current) that it requires negotiation with a non-local agent;
- Failure of a negotiation to reach a commitment;
- Decision to schedule a new set of tasks or to reschedule existing tasks;
- Significant deviation of online schedule performance from expected performance.

According [7, 8], Meta-Level Control is a process of deciding among the following choices: to drop the goal and not do any analysis; to delay goal analysis; reason about the amount of effort to go into goal analysis; and to determine the context of the goal analysis – whether to analyze it a single goal or multiple goals within a single agent perspective; or to analyze single or multiple goals in the context of a facilitating agent's goals.
Meta-Level Control is useful in situations where options for goal analysis are expensive, in other words, the costs or accumulated costs affect agent performance detrimentally. It is useful when the cost of goal analysis is significantly more expensive than cost of meta-level control actions. It is also useful where a choice has to be made about the type of goal analysis and the options for goal analysis have significantly different costs and produce results with significantly different utilities.

2.2 Reinforcement learning

Reinforcement learning is learning what to do and how to map situations to actions - so as to maximize a numerical reward signal [12]. The learner is not told which actions to take, as in most forms of machine learning, but instead must discover which actions yield the most reward by trying them. In the most interesting and challenging cases, actions may affect not only the immediate reward but also the next situation and, through that, all subsequent rewards. These two characteristics - trial-and-error search and delayed reward - are the two most important distinguishing features of reinforcement learning. One of the challenges that arise in reinforcement learning and not in other kinds of learning is the tradeoff between exploration and exploitation.
To obtain a lot of reward, a reinforcement learning agent must prefer actions that it has tried in the past and found to be effective in producing reward. But to discover which actions these are it has to select actions that it has not tried before. The agent has to exploit what it already knows in order to obtain reward, but it also has to explore in order to make better action selections in the future. The dilemma is that neither exploitation nor exploration can be pursued exclusively without failing at the task.
For control problems such as ATFM, reinforcement learning agents can be left to learn in a simulated environment and eventually they will come up with good controlling policies. Some advantages of using reinforcement learning for control problems is that an agent can

be retrained easily to adapt to environment changes, and trained continuously while the system is online, improving performance all the time [12].
The typical mathematical model witch bases Reinforcement Learning is known as Markov Decision Process [10]. This model considers two main conditions:

1. The Markov Property - Situation in which the probability of transition of a state s for a next state s ' depends only on state s and of the action the adopted in s. This means that the current state supplies the enough's information to the learning system decide which action must be taken.
2. Markov Decision Process - is a process in which a set of states S, $s \in S$, $a \in$ set of A(s) action, T set of transitions between states associates with the actions and a set of probabilities P on the joint S that represents a modeling of the transitions between the states.

Two algorithms of reinforcement learning are used in this research, such as: Q-learning and SARSA [12], as mentioned in the following sub-sections.

2.2.1 Q-leanring algorithm

A Q-learning agent learns an action-value function, or Q-function, giving the expected utility of taking a given action in a given state. The essence of the Q-learning algorithm is defined as follows [10 - 15]. At the time *t*, the agent:

1. Visit state s_t and select an action a_t.
2. Receive *r*, reward observed in the following state, from the process of the reinforcement $r(s_t, a_t)$ and observe the next state s_{t+1}.
3. Update the action value :

$$Q_{t+1}(s_t, a_t) \leftarrow Q_t(s_t, a_t)+\alpha[r(s_t, a_t) + \gamma \max_a, Q_t((s_{t+1},a_{t+1}) - Q_t(s_t, a_t)] \quad (1)$$

4. Repeat above steps until stopping criterion is satisfied.

Where [12],

s, a are the original state and action, r is the reward observed in the following state.

α : the learning rate, set between 0 and 1. Setting it to 0 means that the Q-values are never updated, hence nothing is learned. Setting a high value such as 0.9 means that learning can occur quickly.

γ: discount factor, also set between 0 and 1. This models the fact that future rewards are worth less than immediate rewards. Mathematically, the discount factor needs to be set less than 0 for the algorithm to converge.

$\max_a$: the maximum reward that is attainable in the state following the current one. i.e. the reward for taking the optimal action thereafter.

2.2.2 SARSA algorithm

The SARSA algorithm is a temporal difference (TD) method that learns action-value functions by a bootstrapping mechanism, that is, by making estimations based on previous estimations. It is an On-Policy algorithm for reinforcement learning and defined as follows [10 - 15]. At the time t, the agent:

1. Visit a state s_t and select an action a_t.
2. Receive *r*, reward observed in the following state, from the process of the

reinforcement $r(s_t, a_t)$ and observe the next state s_{t+1}.

3. Assign $e_t(s_t, a_t) = e_t(s_t, a_t) + 1$
4. Update the action value :

$$Q_{t+1}(s_t, a_t) \leftarrow Q_t(s_t, a_t) + \alpha[r(s_t, a_t) + \gamma\, Q_t((s_{t+1}, a_{t+1}) - Q_t(s_t, a_t)]\, e_t(s_t, a_t) \quad (2)$$

5. For all s, a do $e_t(s_t, a_t) = \gamma\lambda e_t(s_t, a_t) + 1$
6. Repeat above steps until stopping criterion is satisfied.

The major difference between SARSA algorithm and Q-learning, is that the maximum value for some action not taken for the next state is not used for updating the Q-values. Instead, a new action and a reward are selected using the same policy that determined the original action.

3. Proposed model: evaluation and decision support

3.1 Propose of SISCONFLUX

The System of the Application and Management of the Decision Support Air Traffic Flow Control - SISCONFLUX is in development. The system is proposed to make the suitable decisions for the supervisors of controllers of an air traffic control region, the decision restrictive the traffic flow control more effective ahead of determined scenario. Such decision action will represent the solution most adequate for the maintenance of the best condition of flow of traffic in the Flight Information Region of Brasilia - FIR-BS as a whole.

The SISCONFLUX also is proposed to support the decision according to the determined flights, routes or airports condition, to prioritize the traffic flow management. Such prioritized action may be established in agreement the emanated orientation of the responsible agencies for the air traffic flow management in Brazilian airspace.

As a part of this system, the Module of Evaluation and Decision Support (MAAD) is with the function as a module of analysis and learning. Based on the experience of the air traffic controllers and supervisors, MAAD learns the experience and suggest the decision to adjusted traffic flow in the sectors of traffic control do not reach the condition of congestion or saturation. This suggestion is defined by the other subsystem: Module of Balancing of Flow (MBF), which is general manager of the traffic flow to balance the delay (Sky or Grand Holding) in the minimum or accepted condition.

The definition of the control or adjustment of the flow is from series actions which are specified by some parameters in computation system. These actions are defined by air traffic controller and supervisors, for example, to delay a flight in an airport or give priority for a landing flight, etc. In such a way, the subsystem can get the input action from supervisor, according to pre-established parameters. That input will be sent to other integrant modules of the system for the projection of new scenery and restart of the operation cycle.

MAAD is also developed as the interface among the human agents (air controllers and supervisors) and with the other modules of the system. Basically it has the function to catch the experience of the controllers and supervisors, including these variants to the set of results analyzed from the previous modules, besides functioning as origin of data for the system as well.

Once the decisions are accepted by the supervisors, the same ones will be stored and will start to be part of the system as an accepted decision for the data base. Among All the tasks of this module can be mentioned:

- Presentation of the results to the supervisors;
- Collect of data for learning purpose;
- Storage of the set of pairs (state; action);
- Log of events as action of the operators;
- Activities that are considered useful administratively (item in discussion with the command of CINDACTA I).

To support the decision, the technique of Meta-Level Control is modified and applied to the system to improve the performance of the communication between the agents. During the control process, the decision is taken being based on the experience and the knowledge acquired for the agents with reinforcement learning.

MAAD as a learning module inserted in the architecture of the SISCONFLUX (Figure 1) that guarantees the capacity of adaptation in random situations, or either, adaptation to the alterations of the decision environment. The use of reinforcement learning speeds up heuristically the decision procedure, where Q-learning and SARSA algorithms are introduced.

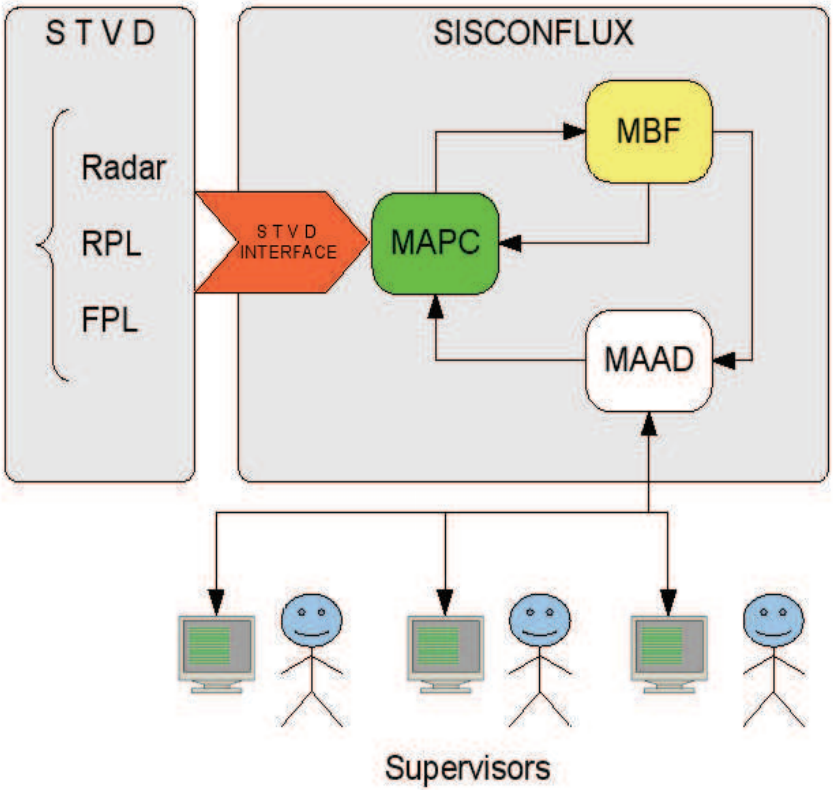

Fig. 1. Architecture of SISCONFLUX

Where:
SISCONFLUX: The System of the Application and Management of the Decision Support Air Traffic Flow Control.
MAPC: Module of Accompaniment and Forecast of Scenery.
MBF: Module of Balancing of Flow.
MAAD: Module of Evaluation and Decision Support.
STVD: System of Treatment and Visualization of Data.
RPL: Plans of Repetitive Flights.
FPL: Plans of Eventual Flights.

Figure 2 shows to a project of functioning of the subsystem and its relationship with the other modules of the SISCONFLUX.

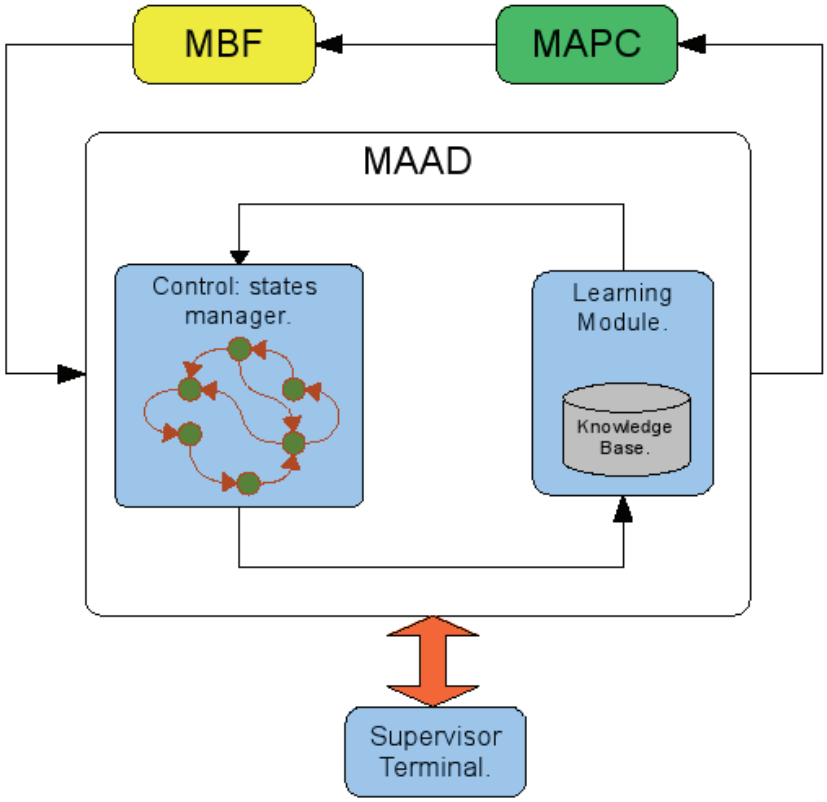

Fig. 2. Relationship among sub-models in SISCONFLUX

3.2 Architecture of module of evaluation and decision support (MAAD)

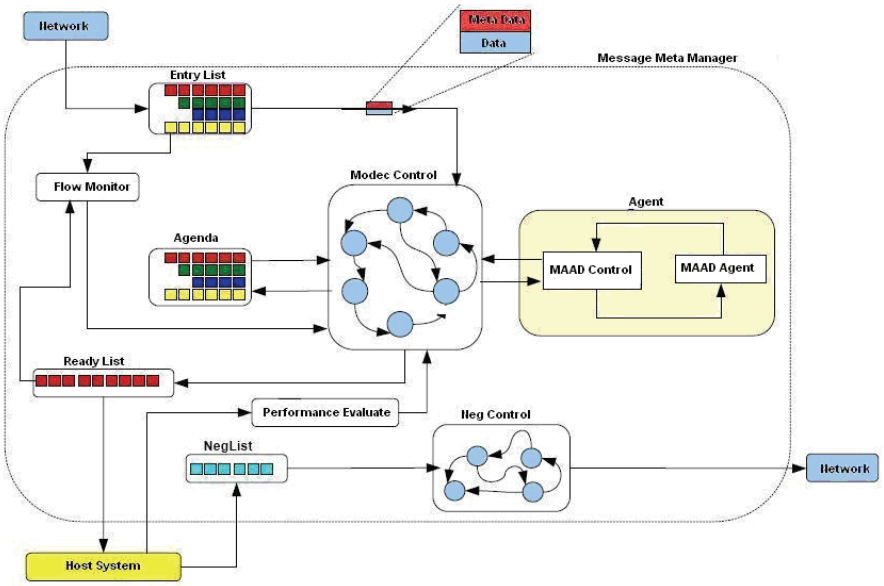

Fig. 3. Architecture of MAAD

The model of Module of Evaluation and Decision Support (MAAD) is developed as a Meta-Level Control layer that a system host receives the messages in a more appropriate sequence, set appointments determined messages and prioritizing others. MAAD consists of two main modules: Module of Decision and Control – MDC and Module Reinforcement Learning - MRL. In such a way, Meta-Level Control uses a set of parameters to associate each message (good utility of the message and maximum state period for the execution) and generate a series of other parameters: the probability of arrive message in the entrance list with high utility, the utility of the messages to set appointments in the agenda list, the period of execution for the message that is in the beginning of the agenda and reason of flow that measures the messages entering to and leaving from MDC. Figure 3 shows the architecture of MAAD.

Exchanging the messages in a distributed system with a manner of communication needs to establish a hierarchy according to aspects of this system. The attributes defined for the system are attached to the message of the Meta-Level Controller that the message encapsulates and sends it. The destination of the message is also processed by a Meta-Controller within the Multi-Agent System, which receives the message and analyzes the enclosed attributes to make the decision in the most appropriate. The manager in Meta-Level Control can decide among three actions: to set appointments of the message for posterior act of receiving; to transfer the message in the system or still to discard it.

The approach of intelligent agent makes use of some aspects of Meta-Level Control as to use the parameters in Meta-Level of the messages for taking efficient decision and without overload the performance of the system in general situation.

3.2 Decision and control approach

The decision process is carried through by MDC which is supported by a Knowledge Base. This base is defined as a part of MRL. The heuristic is used initially in MRL by ad hoc form, where for each state of the environment, it is suggested with an action that in accordance with the previous analysis. However, if dealing with a random environment, it is not always suggested a best action initially. This action may be indicated during a long time. Thus the suggestions are modified in the manner that the agent knows better the environment that it is acting.

The defined heuristic initially is necessary in the situation that the agent does not operate so bad form in the beginning state because it knows very little on the environment.

The state changes are shared by two modules. MDC is developed for controlling the state changes, managing the entrance and the exit of messages and carrying through the communication among the diverse agents. And MRL carries through reinforcement learning for better decision.

Six parameters are used to define a state: p_1, p_2, p_3, p_4, p_5, p_6. They are standardized respectively in the order of the parameters as follows:

- p_1 represents priority of the message that arrives;
- p_2 represents the existing stated period so that the messages can be processed;
- p_3 represents good utility of the set of appointment of the messages;
- p_4 represents the stated period of execution of the messages that are set appointments;
- p_5 represents probability of a arrived message with high priority;
- p_6 represents the reason of flow of the messages in MAAD.

Each parameter will receive a value: high, average or low (see figure 4). When a state has one or more of the six parameters with value not informing, such as 00_2, for project decision, that state will not be studied. In this form, valid states with the six informed parameters will only be dealt by MDC.

For example, the state $(100111011101)_2$, in binary, would present the following values for each parameter: $p_1 = 10$, $p_2 = 01$, $p_3 = 11$, $p_4 = 01$, $p_5 = 11$, $p_6 = 01$, that correspondents the state in decimal form of 2525.

$$\underbrace{11}_{P_1}\ \underbrace{11}_{P_2}\ \underbrace{01}_{P_3}\ \underbrace{01}_{P_4}\ \underbrace{10}_{P_5}\ \underbrace{11}_{P_6}$$

Fig. 4. The parameters for state presentation where: $(11)_2$ = High, $(10)_2$ = Medium, $(01)_2$ = Low and $(00)_2$ = Undefined

Given a state of the environment from MDC, it can be transferred to MRL and then an action will be indicated for the environment. The selected action is executed by the environment and then reinforcement behind this action. This action is returned for the agent of learning that influences the impact of the suggested action. This process of interaction enters the learning agent and the environment will continue and allow the accumulation of experience of the Agent for the long time.

If the learning of the agent to consider only reinforcement learning, the general process will

be very slow. Thus, a alternative approach to speed up the learning may be interesting. In this form, an adaptation to select randomly between exploitation and exploring may be influenced by the performance of the agent in the environment and the external factors to the environment.
To evaluate the performance of the current state of the system - MAAD, five parameters are considered and to get the final value of AvD:

$$AvD = \frac{(3 \times SDS + 2 \times RF + SEL + SAL + SPL)}{8} \tag{3}$$

where,
SDS (Situation of the Host System): this parameter presents situation of the system which the MAAD serves. It is with weight three in the general evaluation due to the necessity to increase the influence of the system host on the evaluation of performance of the MAAD.
RF (Reason of the Flow): this parameter is used to measure the flow of the messages in MAAD. It is with weight two due to the necessity to show more efficiency of the communication of the messages in MAAD.
SEL (Situation of the Entrance Lists): this parameter is used to measure the queue of entrance messages and can be reduced or increased when the system is more congested or less congested respectively.
SAL (Situation of the Agenda): this parameter is used to measure the Agenda of messages and is reduced or increased when the Agenda in the system is more congested or less congested respectively.
SPL (Situation of the Ready List): this parameter is used to measure the processed messages in the queue and can be reduced or increased when the queue is more congested or less congested respectively.

3.3 Module reinforcement learning – MRL

The agent of reinforcement learning uses two algorithms that are well defined in the literature of reinforcement learning: Q-learning and SARSA [5]. Four strategies of the modifications of the parameters in the Q-learning and SARSA algorithms are studied during reinforcement learning simulation, such as the cases of initial heuristic (IH), epsilon adaptative (EA), performance heuristic (PH) and the combination of above three (IH + EA + PH). In totally, eight modifications have been implemented: four modifications for Q-learning and four modifications for SARSA.
The first modification is the original implementation of Q-learning and SARSA with an initial heuristic (IH). This initial heuristic guides the agent in the start when it still does not have experience of the environment that it is acting. For long time, the rules go being substituted for the successful actions that the agent goes taking.
The second modification is a proposal epsilon adaptative (EA) that it reduces the percentage of exploration while the actions are successful or increases this percentage while the actions negatively influence the evaluation of the performance of the system. There is a version for Q-learning and another one for SARSA.
The third modification uses a definition of heuristic (PH) based on the performance of the MAAD, see Bianchi's work [14]. The policy of this variable considers a heuristic function

based on performance. So, the learning algorithm considers these actions which give the best reinforcement associated with the best performance by the heuristic function. An ideal performance is defined as 100% of the evaluation space and a value of performance is measured in the instant of analysis between 0% and 100%. The bigger the performance of the agent, the best is its behavior. The desired situation is to find the politics that allows the agent in the distance to minimize between the current performance and the considered ideal performance.
The fourth and last modification is in the implementation of Q-learning and SARSA when heuristic initial, adapt epsilon and heuristic (IH + EA + PH) is based on the performance.

4. Case study

The case study shows the results from the simulation of the message exchange within the developed system. Each knot in the distributed system presents a layer managed in Meta-Level where the meta-parameters are analyzed for each message. With this consideration, we can see that the classification of the messages is based on the importance of the message without losing: more importance, higher priority.

4.1 Simulation condition

Each message generating source is considered as a knot in the distributed system. The message from this source can be processed by other knots. In a determined moment, if there is less traffic, fewer messages are needed to be processed, in case no congestion arises in the airport system.
In basic simulation, four knots, i.e. four airports, form the study environment. Three of these generating sources send messages to one other knot for analysis. Generally, there is no logical difference among each generating source. The intention of the basic simulation is to identify the adversities of each airport and the important characteristics which need to be treated.
The adversities from each airport are simulated through generation intervals between the messages. To represent a process of intense message exchange, the messages are generated with a short interval. On the other hand, a process of small intensity message exchange is represented with a long interval.
To evaluate the quality of learning, an evaluation module is projected to attribute a note to the agent. Based on a default table of possible good actions for each case, the evaluation module generates the note. This note is generated from the action taken in relation to what it is suggested by the default table and also considering the degree of congestion of the external path. Here, the evaluation of learning quality is done within the mentioned periods. The interval of evaluation between the messages is varied throughout the four simulations. The configuration chosen included 32 periods and intervals between the messages of 1 second for Q-learning and 5 seconds for SARSA.

4.2 Evaluation of the performance of MAAD

Each algorithm is tested with the combination of proposed strategies (as mentioned in last section). In summary, the strategies include the use of heuristics and adapted parameters by

performance that enables the agent to learn flexibly and rapidly.
Figure 5 shows the result of the evaluation of the performance of messages meta-controlling. The curves of performance of Q-learning are worse, in all four evaluations, compared with the curves from SARSA algorithm. The literature justifies this result due to the nature of the two algorithms. While Q-learning makes a search among the possibility actions, SARSA makes a random choice that considers the probability distribution of the actions until the action is selected.

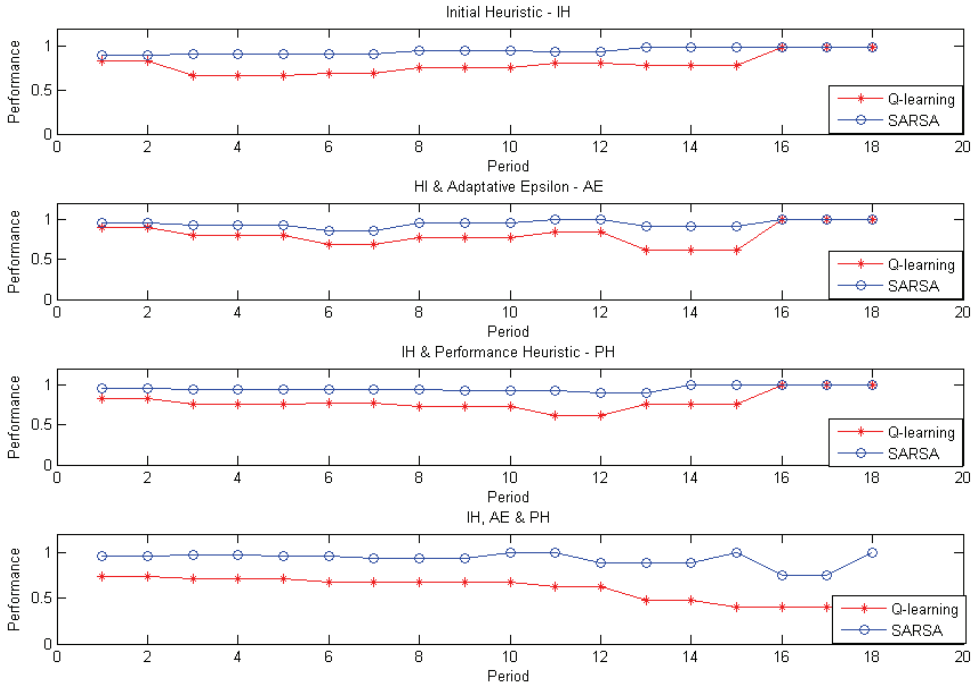

Fig. 5. Evaluation of the Performance of MAAD

As a result, the use of initial heuristic (IH) and epsilon adaptative (EA) present better results with SARSA algorithm. In case of Q-learning, the initial heuristic (IH) and performance heuristic (PH) present better results according to the observed learning performance. However, it is observed that the results of agents learning with Q-learning are worse in comparison with SARSA.

4.3 Evaluation of the quality of the learning and alteration of the *alpha* parameter

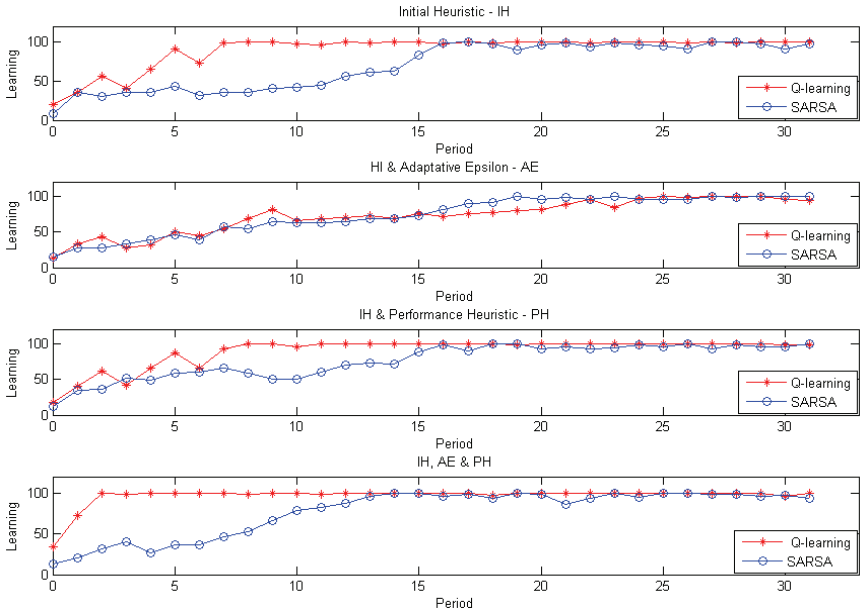

Fig. 6. Evaluation of the Learning Quality for α = 0.02

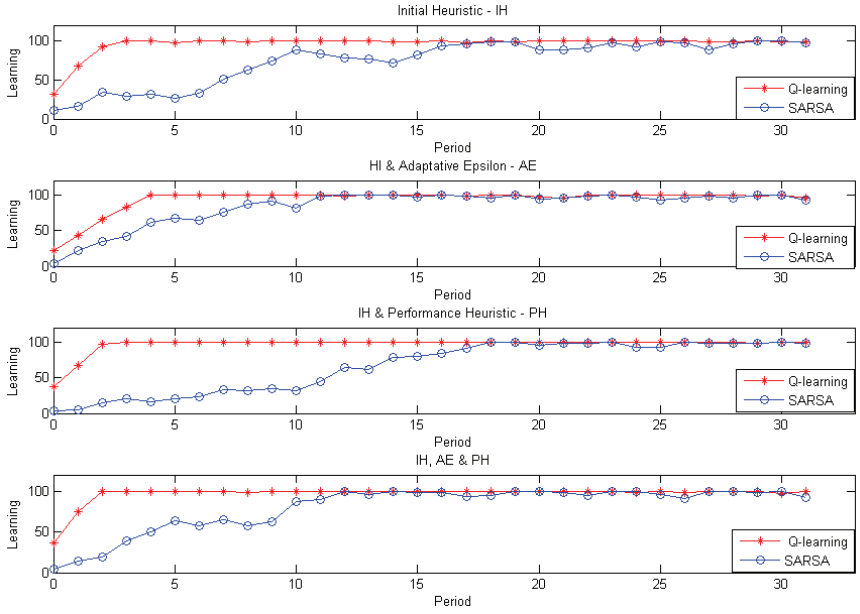

Fig. 7. Evaluation of the Learning Quality for α = 0.04

In this case, a vector of uplas (state, action, reward) is defined, considering the acceptable value depending on the environment conditions. When the decision is taken, it is compared with a defined acceptable standard which maximum value is 100%. If a decision is within the acceptable standards, a parameter note is developed and stored in a note vector. After the complete fulfilling of the note vector, with initially one hundred positions, the arithmetic mean of notes is calculated and note value is altered. This guarantees that a good decision will lead to the growth of the note and a bad decision to its decrease.
In figure 6, the system tries to make a decision following the expected standard. In the initial points, some decisions have bad performance. Initially, a note zero is attributed, and to the measure that the module takes good decisions, this note is increased. Some bad decisions should expected. This is one of the characteristics of reinforcement learning. SARSA algorithm has faster execution time than Q-learning (see figures 6 and 7). However, the experiments show that it learns slower to reach maximum note. On the other hand, the Q-learning algorithm takes a lesser amount of periods to reach a maximum note (see figures 6 and 7).
Concerning the results obtained with the change of alpha parameter, it was adjusted for values of 0.02 and 0.04, respectively. With alpha of 0.04, the simulation achieved better results.

5. Conclusions

This research presented a solution for Meta-Level Control application and reinforcement learning in the decision process to improve the efficiency for the exchange of messages within a distributed system in ATFM. As a part of the research of the Evaluation and Decision Support Module (MAAD), the reinforcement learning approach was applied and developed as a sub-module (MRL) with adaptation of two algorithms: Q-learning and SARSA. The Meta-Level Control was developed as another sub-module (MDC) for making decision regarding information process.
One of the advantages of the use of reinforcement learning is that the agents acquire experience during the iteration process with the environment. As a similar form of learning, it utilizes system performance as criteria to verify the performance of the agents in the environment. After analyzing the activities of controllers and supervisors, the hierarquization of these activities in computation module was established, allowing treating more critical situations faster and more effective. At the same time, the controllers and supervisors can get the quantitatively impact from the system for their decisions.
The simulation results from four cases by two reinforcement learning algorithms (Q-learning and SARSA) have shown the correctness and efficiency of the combination of Meta-Level Control and reinforcement learning in a special problem of ATFM. As a stage research report, the simulation is just a part of the internal message process. Other interesting aspect is the evaluation of changes in the value of alpha parameter. The learning quality is influenced by a suitable choice of alpha value.
In further research, it will be pursued the integration of support decision procedures with all modules in SISCONFLUX, to effectively assist the activities of controllers and supervisors. It is intend to consider a heuristic procedure in MAAD to improve the performance, reflecting quantitative criteria of the real life performance of the

controllers and supervisors in CINDACTA I. It is also interesting to use other reinforcement learning algorithms such as R-learning or Dyna in the system.

6. Acknowledgements

This research is partially supported by Brazilian National Council for Scientific and Technological Development – CNPq (Proc. 306065/2004-5) and FINATEC - Foundation of Scientific and Technological Enterprises of the University of Brasilia –UnB, Brazil.

7. References

A. S. Tanenbaum and M. V. Steen, Distributed Systems: Principles and Paradigms. Prentice-Hall, 2002.

G. Tidhar, A. Rao, M. Ljunberg, Distributed Air Traffic Management System, Tech. Rep. 25, Australian Artificial Intelligence Institute, 1992.

T. Prevôt, Exploring the Many Perspectives of Distributed Air Traffic Management: The Multi Aircraft Control System MACS, Proc. of the Int. Conference on Human-Computer Interaction in Aeronautics S. Chatty, J. Hansman, and G. Boy (Eds.), pp. 149-154, 2002.

M. Nguyen-Duc, J. P. Briot, A. Drogoul, V. Duong. An application of Multi-Agent Coordination Techniques in Air Traffic Management, Proc. of the IEEE Int. Conference on Intelligent Agent Technology, 2003.

L. Weigang, M. V. P. Dib, A. C. M. A. Melo, C. J. P. Alves, "Multi-Agents System by Grid Computing for Real Time Traffic Synchronization", Journal of the Brazilian Air Transportation Research Society, Vol. 2, pp. 63-82, 2006.

L. Weigang, C. J. P. Alves, N. Omar, "An Expert System for Air Traffic Flow Management". Journal of Advanced Transportation. Vol. 31, N. 3, pp. 343-361, 1997.

A. Raja and V. Lesser, Efficient Meta-Level Control in Bounded-Rational. In Proceedings of Autonomous Agents and Multi-Agent System, pp.1104-1105, Melbourne, Australia, 2003.

A. Raja and V. Lesser, "A Framework for Meta-level Control in Multi-Agent Systems", Autonomous Agents and Multi-Agent Systems, Springer, 2007.

R. S. Sutton and A. G. Barto, Reinforcement Learning: An Introduction. The MIT Press, England, 1998.

S. Russel and P. Norvig, Artificial Intelligence – A modern Approach. Pearson Education, Inc., Second Edition, New Jersey, 2003.

S. Russel, E. Wefald, Principles of Metareasoning, Artificial Intelligence, Vol. 49, pp. 361-395, 1991.

R. S. Sutton, A. G. Barto, Reinforcement Learning: An Introduction. The MIT Press, England, 1998.

P. D. Watkins, Technical note Q-learning. Machine Learning, Vol. 8, pp. 279-292, 1992.

R. A. C. Bianchi and A. H. R. Costa, Uso de Heurísticas para a Aceleração do Aprendizado por Reforço. No XXV Congresso da Sociedade Brasileira de Computação. São Leopoldo, Brasil, 2005.

S. Zhang and F. Makedon, Privacy Preserving Learning in Negotiation. In: Proceedings of the ACM symposium on Applied computing – SAC´05, ACM Press, Santa Fe,

pp. 821-825, 2005.

Crespo, A. M. F., Aquino, C. V., Souza, B. B., Weigang, L., Melo, A. C. M. A., Alves, D. P., Sistema Distribuído de Apoio a Decisão Aplicado Ao Gerenciamento Tático do Fluxo de Tráfego: Caso CINDACTA I, em Anais do VI Simpósio de Transporte Aéreo - SITRAER, pp. 317-327, Maringar, 2007.

Reinforcement Learning

Edited by Cornelius Weber, Mark Elshaw and Norbert Michael Mayer

ISBN 978-3-902613-14-1
Hard cover, 424 pages
Publisher I-Tech Education and Publishing
Published online 01, January, 2008
Published in print edition January, 2008

Brains rule the world, and brain-like computation is increasingly used in computers and electronic devices. Brain-like computation is about processing and interpreting data or directly putting forward and performing actions. Learning is a very important aspect. This book is on reinforcement learning which involves performing actions to achieve a goal. The first 11 chapters of this book describe and extend the scope of reinforcement learning. The remaining 11 chapters show that there is already wide usage in numerous fields. Reinforcement learning can tackle control tasks that are too complex for traditional, hand-designed, non-learning controllers. As learning computers can deal with technical complexities, the tasks of human operators remain to specify goals on increasingly higher levels. This book shows that reinforcement learning is a very dynamic area in terms of theory and applications and it shall stimulate and encourage new research in this field.

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821