Exploitation of User's Preferences in Reinforcement Learning Decision Support Systems

Micheline Bélanger[1], Jean Berger[1], Jimmy Perron[2], Jimmy Hogan[2], Bernard Moulin[3]

[1]Decision Support Systems Section, DRDC Valcartier, 2459 Pie-XI Blvd North, Québec, Québec, Canada, G3J 1X5
Emails: micheline.belanger@drdc-rddc.gc.ca; jean.berger@drdc-rddc.gc.ca
[2]NSim Technology, 3015 Laroche Street #10, Quebec, Qc, G1X 1K1
Emails: jimmy.perron@nsimtech.com; jimmy.hogan@nsimtech.com
[3] Department of Computer Science and Software Engineering, Laval University, Ste Foy, G1K 7P4, Canada.
Email: bernard.moulin@ift.ulaval.ca

Abstract

A system called COLMAS (COordination Learning in Multi-Agent System) has been developed to investigate how the integration of realistic geosimulation and reinforcement learning might support a decision-maker in the context of cooperative patrolling. COLMAS is a model-driven automated decision support system combining geosimulation and reinforcement learning to compute near optimal solutions. Building upon this hybrid approach, this paper proposes an extended framework to constructively incorporate user preferences providing mixed-initiative generation of further trusted and validated solutions. The proposed approach integrates the user's preferences in COLMAS by automatically extracting user's preferred solution.

Introduction

Researchers have proposed a variety of multiagent reinforcement learning approaches to tackle the problem of cooperative planning with unknown state-transition model, but few efforts have investigated the concurrent use of multiple methods. In this paper we propose a hybrid approach combining distributed reinforcement learning (RL) and geosimulation (geospatial information reasoning) while capturing the user's preferences.

Consider a cooperative surveillance/patrol scenario in which a team of unmanned aerial vehicles (UAVs) is given the task of navigating a region to continuously monitor a set of unpredictably moving land targets. Realistic planning for task allocation and navigation calls for suitable geographical modeling and simulation. A simple team's objective consists in jointly planning UAV paths, typically minimizing the *global idleness*, that is the sum of the time lag between successive visits of a target, over all targets.

Recently, Santana and his colleagues (Santana et al. 2004) proposed to use reinforcement learning to handle the cooperative patrolling problem. However, the authors' simplifying assumptions about low-level execution planning (e.g. geographical navigation) made the proposed solution rather unrealistic.

In order to further investigate this problem, DRDC Valcartier developed an automated advisory decision support called COLMAS (COordination Learning in Multi-Agent System) (NSIM Technology 2006). COLMAS is a hybrid system which couples a reinforcement learning approach (Q-Learning) with geospatial reasoning (using an agent-based geosimulator) in order to enable coordination learning in a more realistic spatial environment. The COLMAS geosimulator enables a team of UAVs to autonomously navigate in a simulated geographic territory, and to coordinate their actions, each agent is performing a plan (i.e. a course of action). Each agent has a perception capability that enables it to collect data about its environment. The agent can also reason about the collected data and choose its next action (or goal). Hence, the agent autonomously navigates and avoids collisions in the simulated environment.

In the COLMAS architecture, the learning process generates solutions for the patrolling problem. In this case, a solution is represented by a course of actions for each agent (UAV) in the scenario. To do so, the COLMAS learning process follows these steps:

1-The learning agent uses the geosimulator's perception capability to gather all available information from the spatial environment.

2- The agent's perception module formats data input from the environment and creates the current state "***s***". The state "s" corresponds to information needed to run the Reinforcement Learning (RL) algorithm. In the surveillance task, the only needed information corresponds to the 5 next potential targets to visit, coupled with the last visited target stored in the agent's MemoryState.

3-The agent's exploration policy (e.g. a greedy method based on (Sutton 2004)) chooses the next action "***a***" to execute, based on the learned information available in a Q-Function.

4-The chosen high-level action is then executed (low-level navigation planning) by the geosimulator.

The learner is executed to update the Q-Function:

a. Get current state "***s***".

b. Get the last executed action "***a***" in Policy.

c. Perceive next state "***s'***".

d. Get reward r = RewardFunction(***s, s', a***).

e. Compute Q = Q-Function(***s, s', a,*** r).

In the COLMAS system, the simulation is continuously running, learning and improving solution policy to solve the problem. An important feature developed in the simulation is a technique to detect and extract patterns from the last computed solution. Considering that a greedy exploration policy is used in the RL, we ideally search for a stable pattern executed by each UAV which minimizes idleness. Navigation patterns are identified using an history of the 100 latest executed actions for each UAV. A pattern is defined by a recurrent trajectory in target visits.

Building upon the baseline COLMAS approach, we propose an extended framework to constructively incorporate the user's preferences providing mixed-initiative generation of further trusted and validated solutions. The proposed approach integrates the user's preferences to COLMAS by automatically extracting the user's preferred solution during the solution generation process. The following sections explore a possible integration of such user's preferences.

Possible Exploitation of User's Preferences

COLMAS should automatically adjust problem-solving parameters to compute solutions constrained by user's preferences. User's preferences may be known (explicitly revealed) or not. In the former case, they relate explicitly to prior desirable solution properties whereas in the latter case they emerge naturally while the solution is generated dynamically.

Explicitly known user's preferences characterize desirable or intrinsic properties of a solution. For example:

- *Preferential path segment*: the user can provide part of a solution that he would like to be included into the global solution (called a preferred path). A preferred path segment might for instance require an agent to first visit a target x and then a target y.
- *Constrained path segment*: the user may prevent the emergence of a specific path in the global solution (called a path interdiction). For example, he may determine that a specific UAV must not go from target x to target y.
- *Agent-task assignment*: the user wants to prevent a specific UAV to visit certain targets in the global solution (called target interdiction).

These preferences can be considered as initial but non-exhaustive biases or constraints that candidate system solutions must satisfy. A way to consider these user's constraints is to develop a new COLMAS action selection procedure for the exploration/search algorithm. A MixedPolicy exploration algorithm, composed of 4 steps, handles this task:

- Step 1 - choose the next Greedy action to execute (in the Greedy approach, we choose the best action learned so far).
- Step 2 - verify if the selected action meet the constraints defined by the user.
- Step 3 - if the action satisfies these constraints, execute it; otherwise return to Step 1.

Even if the action selection method constrains a part of the solution, it is not possible to guarantee that the UAVs will find a pattern which necessarily includes the partial pattern specified by the user. For example, if the UAV doesn't visit the desired path of the solution, the action selection method will not be able to force the UAV to do so. Consequently, the UAV will learn another pattern which does not include the specified part.

When the user's preferences are not explicitly known, relevant information can be extracted from the user's interaction to identify his preferences (Rossi and Sperduti 2004). This is called a calibration process. In our context, user's acceptance or rejection of a solution provides information about how he appreciates this solution. One way of characterizing a solution to exploit this type of information is to evaluate the solution based on some critical dimensions. These evaluations can be aggregated into a global evaluation, considering the importance the user associates to each of these dimensions. This global evaluation can then be used 1) to order the solutions that would be presented to the user and; 2) as a reward function in the COLMAS system.

We proposed a number of criteria representing some critical dimensions for the surveillance problem:

- *Average global idleness*: an admitted evaluation criterion used to appreciate the quality of a patrolling solution. In the literature, the idleness is attached to a target and represents the time elapsed since the last visit.
- *Pattern exclusivity*: it indicates the level of exclusivity that a UAV has over targets. For example, if two different patterns share a given target, these patterns are not exclusive.
- *Target separation level*: this dimension represents the relative target sharing among all UAVs. The objective of this evaluation is to compute the metric which indicates if all UAVs visit the same number of targets in the problem.
- *Danger zone level*: this criterion computes how many times the solution crosses a danger zone.

These criteria are used to produce a general evaluation of the solution, which can be computed using a linear combination of all criteria. This global evaluation is the

sum of various metric values multiplied by their coefficients of relative importance (CRI). These respective coefficients of relative importance should reflect the user's preferences related to a particular solution. Then, the challenge consists in automatically determining the CRIs representing the user's preference for each criterion. Weight adjustment may rely upon different techniques such as linear progression, inverse reinforcement learning, neural network, etc. (Perron and Hogan 2006). We chose to use inverse reinforcement learning since it requires limited explicit user feedback to ensure convergence.

While reinforcement learning is a technique used to derive optimal action policies that maximize the expected reward collected while moving through some state-space, inverse reinforcement learning (IRL) (Abbel and Ng 2004) takes some optimal action policy and seeks to infer which reward function was optimized to generate this policy. The idea of inverse reinforcement learning is to learn the reward function from a policy provided by an external expert. In this work, we assume an expert trying (without necessarily succeeding) to optimize an unknown reward function that can be expressed as a linear combination of known "features". These known features are the critical dimensions used to evaluate a solution Thus, a reward function can be expressed by $R(s) = w \bullet \phi(s)$ where $w \in \Re^k$ is an unknown vector specifying relative weighting for each feature, and $\phi(s)$ is the feature vector related to state s. This is useful in scenarios where providing good solutions is far easier from the expert's perspective than formally defining the reward function.

The example pictured in Figure 1 shows that IRL was able to adjust the CRI values to map the user's preferences over 5 or 6 iterations. In this case, the calibration process determined that the user wanted COLMAS to find solutions minimizing all criteria. Moreover, we can see that the idleness criterion is the most important one, while other criteria have the same relative importance (weights) in the solution evaluation.

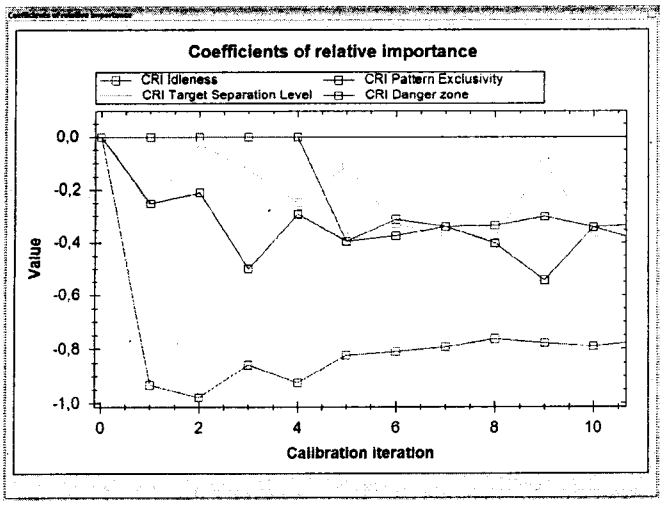

Figure 1 – Results of the Calibration Process

If the user does not accept any solution for a certain period of time, we may consider that the set of solutions presented by the system up to now does not satisfy his preferences. In this case, adjusting the control parameters of the search algorithm is likely to provide alternate guidance to explore a more suitable region of the solution space. For example, in COLMAS, the following parameters can be adjusted: the learning rate, the learning algorithm, the perception radius, the input state definition, the reward function, the navigation algorithm used, etc. The objective here is to get out local optima and to generate solutions which might provide a significant gain to the user. In this case, the user's preferences are directly related to COLMAS' parameters.

Global Approach

Based on the elements mentioned in the previous section, an approach has been developed to support the interactions between a user and the COLMAS system. This interaction aims at capturing human inputs and feedback into COLMAS. The proposed six-step procedure works as follows:

1. Initially the problem is presented to the user through a customized interface.
2. At this step, the user begins to specify constraints related to the scenario. The specified constraints may include, for each UAV part of the scenario:
   - Desired paths.
   - No-flight paths.
   - No-flight targets.
3. When the user begins to define the constraints at step 2, the COLMAS system begins immediately, in the background, to compute a solution with the currently available information.
4. When COLMAS finds a solution, it stores it in a working solution list. The user can now consult the generated solutions.
5. When visualizing a generated solution, the user can decide to calibrate the COLMAS system. The calibration process attempts to extract the user's preferences (trying to find the CRI using the IRL) according to the selected solution in order to determine which kind of solution the user really likes or favorably support. Based on this information, COLMAS tries to respect these preferences, as follows:
   - The system reorders the working solution list according to the new preferences. An existing solution with a low score can become a good solution under the new preferences;
   - The next solutions generated by the system must respect the user's new preferences.
6. The process iteratively returns to step 2 until a satisfactory computed solution is found by the user.

Interface Supporting the Approach

A suitable interface has been designed to mediate automated problem-solving and user interaction. In COLMAS, the problem is represented by a scenario which contains the following objects as pictured in Figure 2:

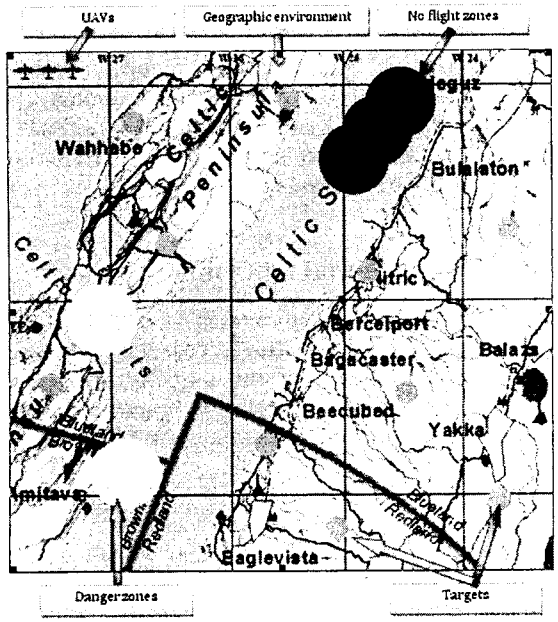

Figure 2 – A Scenario in COLMAS

- *Geographical environment*: a map reflecting the area of operations.
- *Targets*: object to be visited (monitored). It is represented by a colored circle positioned in the geographic environment.
- *UAV*: patrolling agent represented by a colored aircraft positioned in the geographic environment.
- *No flight zone*: area characterized by UAV flight interdiction (red circle). The UAV navigation algorithm imposes travel constraints on these zones. These regions are typically, considered as physical obstacles in the environment. Two navigation algorithms are implemented in the simulation:
  - *Simple obstacle avoidance*: This algorithm tries to avoid no-flight zones in an efficient way (reactive algorithm). However, it may happen, with a particular no-flight zones layout, that the algorithm is unable to find a valid path between two targets.
  - *A* planner*: This algorithm plans a path between two targets and always finds the optimal solution to avoid obstacles. This algorithm is computationally more expensive.
- *Danger zones*: A danger zone (yellow circle) can be positioned in the environment. Danger zones are only informative and represent potential threats (as opposed to pure physical obstacles) to be avoided by UAVs. These zones are used by a metric in the simulation (danger zone metric) to compute how many times UAVs cross the danger zone for each solution. This information will be used to specify preferences to cross danger zones or not, for each user.

In addition to scenario elements, the proposed user interface captures specific constraints subjectively and explicitly imposed by the user. Hence, the user may specify constraints in the considered scenario or the main problem to be solved via a special editor. As illustrated in Figure 3, the editor allows the user to visually specify three types of constraints for each UAV, namely:

- *Desired paths*: A desired path is represented by a colored arrow (a different color is used for each UAV) starting from an initial target to a final target. The user can specify a set of desired paths for each UAV in the scenario.
- *No-flight paths*: A no-flight path is represented by a black arrow starting from an initial target to a final target. The user can specify a set of no-flight paths for each UAV in the scenario.
- *No-flight targets*: A no-flight target path is represented by a black target and specifies a target that must not be visited by a particular UAV. The user can specify a set of no-flight targets for each UAV in the scenario.

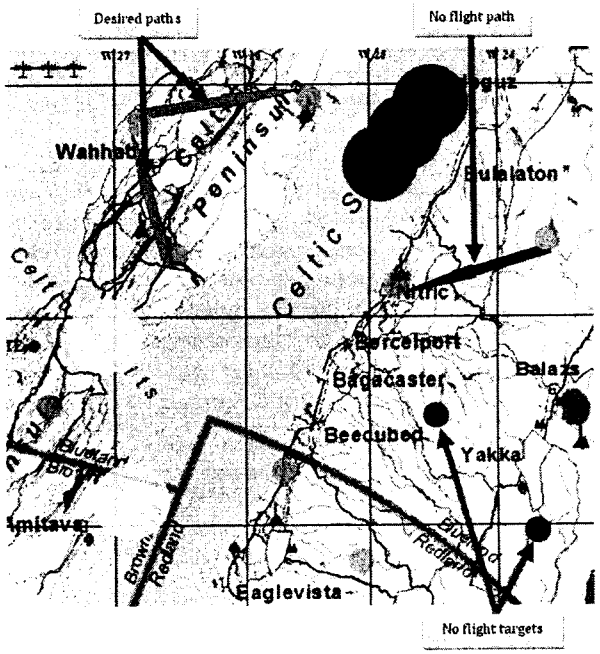

Figure 3 – Constraints defined for a UAV

Once the problem and constraints are defined, COLMAS may compute a solution for the patrolling problem as shown in Figure 4. When a solution is completed, it is evaluated by the evaluation module. The learning module uses the evaluation module as well as a background process to find solutions (using it as the reward function). The obtained solutions are then

presented to the user as cyclic patterns using different colors, one for each UAV. As shown in Figure 5, the global evaluation of each solution, as well as the evaluation value and the CRI associated with each criterion, can be visualized to help the decision-maker assess and compare the solutions. The integration of this information into the user interface can be presented as shown in Figure 6.

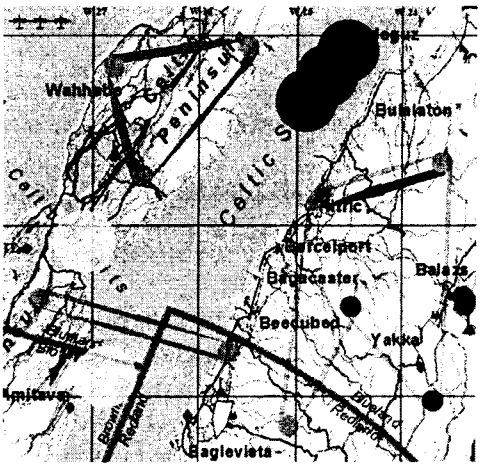

Figure 4 – A solution represented by 3 sets of arrows

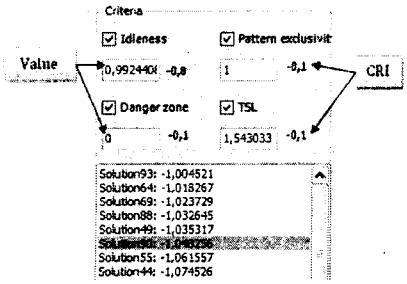

Figure 5 – Solution evaluation

Illustration

The functionality of the proposed decision support system can be described as follows. Consider the initial situation presented in Figure 6 from which the user has provided a solution. This solution, representing his preferred way to solve the problem, is used by the calibration process to identify the CRI as presented in Figure 7. In this case, the system inferred the user's desire to preferentially minimize the *Pattern exclusivity* and *DangerZone* criteria.

These preferences are now used as input to produce a new solution as shown in Figure 8. These results show that the system is able to respect the user's preferences in producing a different solution with *DangerZone* criterion at zero and *PatternExclusivity* criterion close to one (which are the minimum values.)

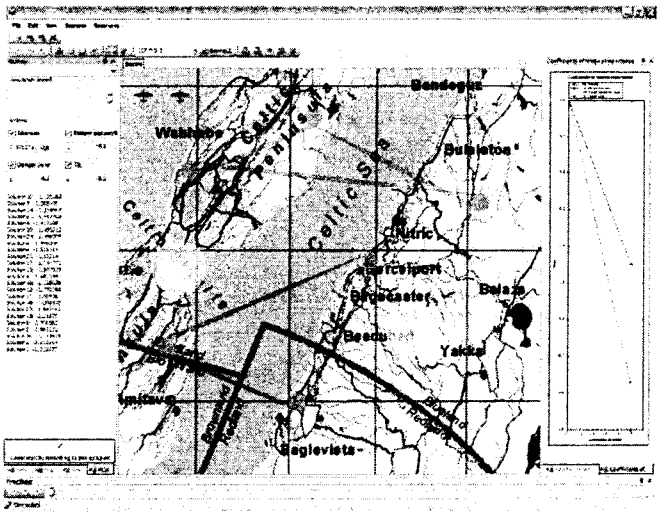

Figure 6 – Scenario1 - User's solution

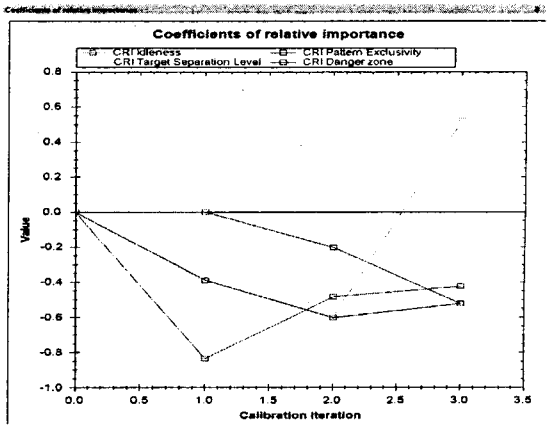

Figure 7 – Scenario 1 -Results of the calibration process

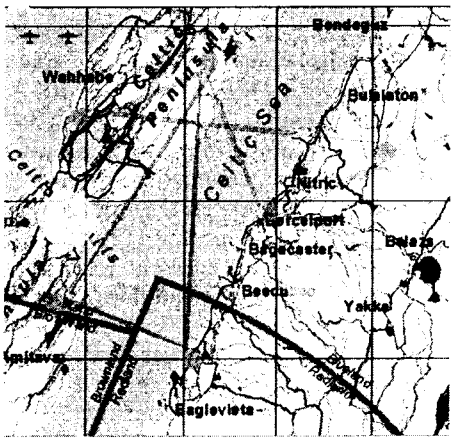

Figure 8 – Solution 1 after the calibration process

These results may be reutilized for alternate scenarios or future situations. In that case, COLMAS exploits the knowledge about preferences obtained from the previous exercise in order to generate new solutions which try to

respect the user's preferences, as pictured in Figure 9 and 10.

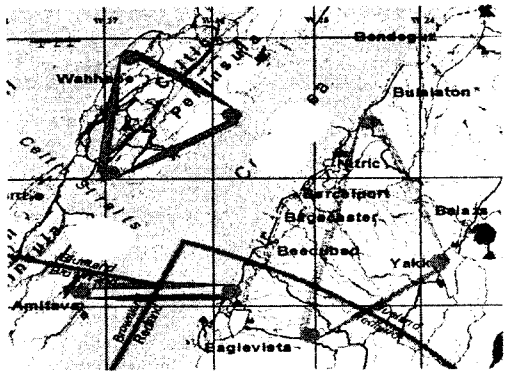

Figure 9 – S1- A new solution for a different scenario

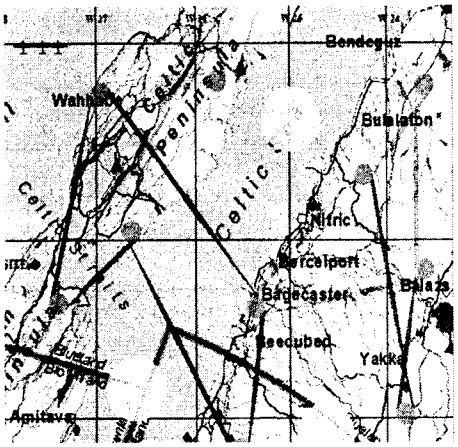

Figure 10 – S2- A new solution for a different scenario

CONCLUSION

The objective of this work was to develop an approach to integrate the user's preferences to the reinforcement learning solving process implemented in the COLMAS system, which generates realistic solutions for the patrolling problem. The hybrid approach allows to extract the user's preferences and calibrate the system. It tends to significantly reduce computational complexity, providing additional user guidance in exploring the solution space. The proposed inverse reinforcement learning technique relies only on a single expert solution, to automatically extract CRI which map the user's preferences. The adapted COLMAS solving process then uses these weights in the reward function in order to find solutions reflecting user's preferences. This feature turns out to be particularly convenient when the user doesn't know how to precisely specify his preferences for a given solution problem. As a result, the synergistic coupling procedure can extract, from a single solution provided by the user, the coefficients of relative importance from a large set of available criteria. This new kind of "preference mining" (Holland et al. 2003) can be exploited to find the users' preferences in order to personalize a particular decision support system. These preferences can then be reused to solve problems in other scenarios.

These concepts were implemented in COLMAS v2.0 providing the following system properties:

- Automated solution generation for the patrolling/ surveillance problem;
- Evaluation of generated solutions, using some specified criteria;
- User's constraints integration to produce adapted solutions;
- User's preferences extraction to guide problem-solving;
- Adaptive internal problem-solving process to capture user's preferences;
- User's preferences generalization for reutilization with alternate scenarios.

Another important aspect that has been developed is the user interface, which allows to:

- Modify an existing solution from the solutions set;
- Specify constraints for a scenario and let COLMAS finds the solution which satisfies these constraints;
- Accept/Reject complete solution provided by the COLMAS system.

Future work should allow the user to specify his preferences on partial proposed solution as well as on specific critical dimensions, i.e. idleness.

References

Abbeel, P. and Ng., A., Y. 2004. Apprenticeship Learning via Inverse Reinforcement Learning. In Proceedings of Twenty-First International Conference on Machine Learning (ICML). Banff, Alberta, Canada: ACM.

Holland, S., Ester, M., and Kießling, W. 2003. Preference Mining: A novel approach on mining user preferences for personalized applications. Universität Augsburg Technical Report, Institute of Computer Science, University of Augsburg.

NSIM Technology, 2006. COLMAS Project - Technical survey and COLMAS architecture. CR 2006-601, DRDC Valcartier.

Perron, J. and Hogan, J. 2006. Investigation of critiquing systems into the continual planning tool COLMAS. CR 2006-594, DRDC Valcartier.

Rossi, F. and Sperduti. 2004. Acquiring both constraint and solution preferences in interactive constraints systems. *Constraints* 9(4):311-332.

Santana, H., Ramalho, G., Corruble, V., Ratitch, B. 2004. Multi-Agent Patrolling with Reinforcement Learning. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagents Systems (AAMAS)*, 1122-1129. New York, New York: IEEE.

Sutton, R. and Barto. A. 2004. *Reinforcement Learning: An Introduction*. MIT press.