2006 International Joint Conference on Neural Networks
Sheraton Vancouver Wall Centre Hotel, Vancouver, BC, Canada
July 16-21, 2006

Training Coordination Proxy Agents

Myriam Abramson
Naval Research Laboratory
Washington, DC 20375
myriam.abramson@nrl.navy.mil

William Chao
Naval Research Laboratory
Washington, DC 20375
chao@itd.nrl.navy.mil

Ranjeev Mittu
Naval Research Laboratory
Washington, DC 20375
ranjeev.mittu@nrl.navy.mil

***Abstract*— Delegating the coordination role to proxy agents can improve the overall outcome of the task at the expense of cognitive overload due to switching subtasks. Stability and commitment are characteristics of human teamwork but must not prevent the detection of better opportunities. In addition, coordination proxy agents must be trained from examples as a single agent but must interact with multiple agents. We apply machine learning techniques to the task of learning team preferences from mixed-initiative interactions and compare the outcome results of different simulated user patterns. This paper introduces a novel approach for the adjustable autonomy of coordination proxies based on the reinforcement learning of abstract actions.**

INTRODUCTION

Advances in communication technologies has lead to increased agent interactions and increased complexity in the decision-making process. To deal with this added burden, the coordination role is delegated to a proxy agent. Coordination proxy agents [1] are personal agents that take on the coordination role on behalf of a human user (Fig. 1). While the optimization of the global task can be better achieved by the self-organization of proxy agents in dynamic environments, switching roles or teams involves preferences, such as loyalty, boredom and persistence thresholds, in addition to interpretations that might need to be elicited from the human in the loop. This paper addresses issues in determining when switching teams is appropriate to satisfy both the urgency of the subtask relative to the global task, the preferences of the user, and when input from the user is warranted. We claim that through result-driven reinforcement learning, the human can train coordination proxies in a task with examples biasing the way the task is achieved with respect to the outcome of the task.

Similarly, in mixed-initiative planning involving goal selection, directives from the user are obtained interactively in case of plan conflict or provided apriori in the form of plan constraints. Mixed-initiative interactions in multi-agent systems provide a flexible way to harness the cognitive capabilities of the human in the loop in solving a problem while delegating more mundane tasks to the proxy agents. As in the turn-taking problem found in dialog management [2], the key decisions for mixed-initiative interactions, as applied to the adjustable autonomy of proxy agents, include knowing *when* to ask for help, *when* to ask for more information, and *when* to inform the user of a decision. This paper claims that learning user preferences is not sufficient for training coordination proxies if those preferences conflict with other agents' preferences and affect the outcome of the task. As long as preferences are inconsistent with each other as evidenced by the outcome of the task, a proxy agent must keep training and continue interacting while suggesting alternatives.

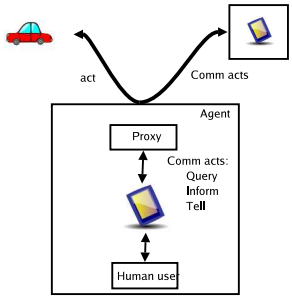

Fig. 1. Example of coordination proxies helping in traffic by negotiating the road.

This paper is organized as follows. A learning approach for training coordination proxies in making decisions is first introduced. We then motivate experiments in the prey/predator canonical coordination domain and present empirical results and analysis of our evaluation. Finally, we conclude with a summary of related work and recommendations for future work. The key contribution of this work is a mixed-initiative approach based on the reinforcement learning of abstract actions and its algorithm scalable to large state space for the adjustable autonomy problem of coordination proxy agents.

LEARNING APPROACH

A proxy agent can operate in three modes characterizing its dialogue to the user: (a) automatic or no interaction, (b) warn-user and (c) wait-for-user. As the proxy agent learns the preferences of its user, it should evolve to the automatic mode of interaction. "Warn-user" is an asynchronous mode of interaction where the opportunity to change roles is given to the human without interrupting the task of the proxy or the human. The "wait-for-user" mode is a synchronous mode of interaction that will interrupt the task (subject to timeout). Deciding autonomously which mode to be in for a specific situation constitutes adjustable autonomy [1]. This paper claims that the urgency to interrupt the human should be based on the ambiguity of a decision reflecting the uncertainty of a situation given the preferences of the user and the expected outcome of the task.

Our learning approach consists of (1) clustering examples of conflicting goal states to expedite the case-based retrieval of past examples, and (2) learning which goal to follow through the reinforcement of user interaction preferences and the outcome of the task. As a result, preferences will be learned only when they help coordination in some way. Inconsistencies in the user preferences will prolong the training mode and situational ambiguities will trigger mixed-initiative interactions. Those steps can be combined to learn online in an incremental way to adjust to novel situations.

Clustering of Conflicting Goal States

This clustering step quantizes a large continuous state space into a discrete representation that is amenable to tabular reinforcement learning techniques. Clustering as a preprocessing step ensures that distinct states are kept apart and prevents oscillations in state values when scaling up reinforcement learning to large state space.

A myopic agent with limited perception will not have goal conflicts since only one goal at a time will be perceived. Through communication and shared knowledge an agent might be aware of other goals, increasing the occurrence of decisions or "choice points." To scale up, patterns of observation are generalized and compressed through the competitive Hebbian learning process of self-organizing feature maps (SOFM)[3]. A SOFM maps an input data space $\Re^n$ to a lower dimensional space (usually two or one) of *prototype* vectors, the neurons, where each vector, $m_i = (x_1, x_2, ..., x_n) \in \Re^n$, is relatively ordered with respect to its "neighbors" preserving the structure of the input space in a lower dimension suitable for visual representation. The update equation of a prototype vector m given a temporally decreasing rate α $(0 < \alpha < 1)$ is as follows:

$$m(t+1) = m(t) + \alpha(t)[x(t) - m(t)] \quad (1)$$

The granularity of clustering needs to be based on the capability of recognizing goal conflicts. A distance function alone does not guarantee that important distinctions will be recognized. The initial set of prototype vectors can also affect the effectiveness of clustering. A semi-supervised approach is presented here where the "least used" prototype vector wins if a conflicting goal, $g \in G$, at time t fails to be recognized as distinct. This is in accordance to the general principle in using feature maps for pattern recognition that prototype vectors be placed at the class borders to avoid misclassifications [3].

Algorithm 1 Clustering of conflicting goals

```
Input: prototype vectors m_i
Output: updated prototype vectors m_i
Initialize: conflict set W ←{}
REPEAT
   Generate input signals {s,g}
      from P(s,g), s∈S, g∈G
   if new choice point then
     foreach w∈W
        update m_w closer to s_w
      W ←{}
     m ← argmin_i (distance(s,m_i))
     while (conflict←{m,g'}∈W and g'≠g)
       m ←argmin_i(hits(m_i))
     hits(m)++
     W ←W∪{s,m,g}
   endif
UNTIL (stopping criterion met)
```

Reinforcement Learning

From Markov Decision Processes (MDP) to Reinforcement Learning: Formally, an MDP is a 4-tuple $\{S, A, T, R\}$ where S is the set of states, A the set of actions, T the transition model specifying the probabilities mapping $S \times A \times S$ to $[0, 1]$ and R, the reward function, mapping $S \times A \times S$ to $\Re$. Algorithms in dynamic programming such as value and policy iterations solve MDPs provided T and R. The complete search space of an MDP is exponential in the number of steps required to solve the problem, $\{S \times A\}^n$. Reinforcement learning (RL) approximately solves MDPs without a model of transition probabilities T by directing its search of the state space based on sample return estimates obtained by interacting with the environment [4]. Those estimates are encapsulated in the value function $V(s)$ for state s

or the action-value function $Q(s,a)$ associating state s to action a. Monte-Carlo methods applies to RL when estimating sample returns based on the outcome of an episode while temporal-difference methods [5] are based on the next temporal step.

Reinforcement Learning of Abstract Actions: Abstract actions are high-level actions, for example a planning decision, that are implemented by several primitive actions but are temporally abstract. How can credit assignment be given to a high-level action since the external terminal reward might also depend on the successful execution of lower-level actions? High-level actions such as conflict resolution decisions occur "offline" and are not temporally part of the execution of a discrete sequence of steps. Similarly, in mixed-initiative strategies involving a human, the high-level decision from the human is not under the control of the coordination proxy agent. The decisions depend only on the current state at certain synchronization points that occur at random time intervals. Consequently, the high-level actions are not completely Markovian since they depend on past temporally selected high-level decisions. In the theory of semi-Markov decision processes (SMDPs), the high-level reward obtained is the mean reward accrued during the time taken to accomplish the goal weighted by the probability of reaching the goal in t time steps [6].

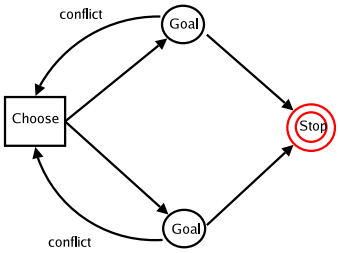

Fig. 2. Non-deterministic HAM controller for goal conflict resolution

Based on the theory of SMDPs, hierarchical abstract machines (HAMs) [7] address the issue of combining high-level actions with primitive actions in a Markov decision process (MDP). A HAM is a non-deterministic finite state machine (FSM) specifying valid transitions constraining the underlying temporal MDP. The "HAM-induced" MDP can then be solved more efficiently. It is however possible to learn in the reduced state space of HAMs directly. Figure 2 shows a general HAM for goal conflict resolution. Machine states superimpose to environmental states to identify behavioral states (e.g. explore, hunt, stop, etc.) and choice points. Given an environment state s, a machine state m, a reward r in the environment state, a past choice point c in a HAM, the accumulated reward r_c since the previous choice point c, and the accumulated discount γ_c ($0 \leq \gamma < 1$) since the previous choice point c, HAMQ-learning (Eq. 2) proceeds as follows.

$$Q([s_c,m_c],a) = Q([s_c,m_c]) + \alpha[r_c + \gamma_c V([s,m]) - Q([s_c,m_c],a)] \quad (2)$$

where $r_c = r_c + \gamma_c r$ and $\gamma_c = \gamma\gamma_c$. The value of the current state $V([s,m])$ is obtained from the underlying temporal MDP as the expected sum of discounted reward $E[\sum_{t=0}^{\infty} \gamma^{t-1} r^t]$.

Reinforcement Learning of User Preferences: The compressed patterns of goal states learned in the SOFM preprocessing step described above constitute a proxy agent's internal representation of choice points augmented with HAMQ-learned action values for the two possible actions of selecting or not selecting the goal state. This function approximation approach separates learning the action value function from learning the state representation (but see [8] for a combined approach). The intermediate reward r_c is obtained from the user decision at the choice point while the discounted terminal rewards upon reaching the goal states are obtained from the underlying temporal MDP. Figure 3 describes the architecture of a coordination proxy that learns from reinforcement.

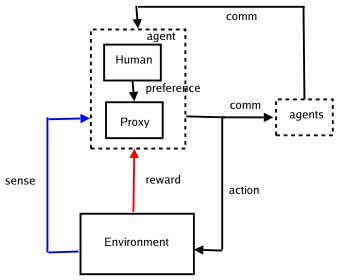

Fig. 3. Coordination proxy system architecture with interactions from the environment and with other agents

By decomposing the state space into machine states and temporal states, HAMs avoid the looping problem [9] in introducing intermediate rewards based on user preferences, however choice-point rewards can override the outcome if $r_c \gg \gamma_c V([s,m])$, misleading the agent in learning non-optimal preferences for the coordination task. The novel mixed-initiative HAMQ algorithm (Algorithm 2), based on the credit assignment of choices and

alternatives, enables coordination proxies to dynamically adjust their interactions depending on the preferences of their users and the outcome of the task. It is a Monte-Carlo algorithm based on the smooth average of episodic returns of environmental states. Eligibility traces [5] are used here as uncertainty variables modulated by the strength of the pattern matching association of a goal state to a machine state. The complementary credit assignment to alternatives is the key characteristic of this algorithm. The sign of the action value at a choice point determines whether the choice is selectable. Ambiguity arises when more than one option at a choice point is selectable or if none are selectable. The constraints for the fairness and rationality of reward r_a for action $a \in A$ at choice point c and actions $a' \in A$, $a' \neq a$, are as follows:

$$r_a \leq \gamma_c V([s,m]) \tag{3}$$

$$r_{a'} = -\frac{1}{|A|-1} r_a \tag{4}$$

$$\sum_{a \in A} r_a = 0 \tag{5}$$

A proof of convergence of this algorithm in reducing the ambiguity rate while increasing user precision would be based on the convergence of online learning of linear functions such as the Winnow algorithm [10].

EXPERIMENTAL EVALUATION

Experiments with different simulated patterns of user preferences were conducted: (1) autonomous, (2) conservative, (3) risky, (4) heuristic, and (5) mixed. In an autonomous pattern, the proxy agent learns how and when to switch team independently of its user. In a conservative pattern, the users initially select a team at random and never switch teams afterwards; in a risky pattern, the users initially select a team at random and always switch teams afterwards; in a heuristic pattern, users have a principled way of selecting a team. In our experiments for a heuristic pattern, agents select the team with the highest sum of preferences and switch teams accordingly. In a mixed pattern, a heuristic, risky or conservative pattern is selected randomly. This last pattern reflects best the heterogeneity of human users. We show corresponding learning performance results of coordination proxies in the prey/predator domain (introduced below) in terms of autonomously resolving conflicts based on user preferences and outcome of the task.

Algorithm 2 Mixed-initiative training (binary choice points)

```
Input: prototype vectors m_c
       eligibility traces e{[m_c]}
Output: a mixed-initiative policy
Parameters: α,γ
Initialize:
      forall m_c
         Q([m_c]) = 0
         e([m_c]) = 0
      W←{}
      user mode
REPEAT
   when choice point c
     foreach goal state s∈S
       [s,m_c]←map(s)
        inform user of Q([s,m_c])
     γ_c ← γ
     if (mode == WAIT or Q([s,m_c]) is AMBIGUOUS) then
       [s,m_c] ← askuser
     else
       [s,m_c]← argmax Q([s,m_c])
     endif
    foreach s'∈S, s'≠s
        W ← W ∪ {[s',m_c]}
     obtain reward r_s
      Q([s,m_c]) ← (1 - α)Q([s,m_c]) + αr_s
      e([s,m_c]) ← similarity(s,m_c)
      foreach w∈ W
        Q([s',m_c]) ← (1 - α)Q([s',m_c]) - αr_s'
        e([s',m_c]) ← -similarity(s',m_c)
    when [s,m], obtain V([s,m])
      forall [s,m_c]
         δ = γ_c e([s,m_c])V([s,m])
         Q([s,m_c]) ← Q([s,m_c]) + α_t δ
  forall c
    γ_c ← γγ_c
UNTIL end of episode
```

Each goal state s at a choice point c is mapped to a prototype vector m_c.

Prey/Predator

The prey/predator pursuit game is a canonical example in the teamwork literature [11] because one individual predator alone cannot accomplish the task of capturing a prey. Practical applications of the prey/predator pursuit game include, for example, unmanned ground/air vehicles target acquisition and search and rescue operations. Due to the decomposability of the global reward as a sum of local rewards, the original problem can be extended to multiple teams by including additional preys. Prey/predators can sense each other if they are in proximity p but do not otherwise communicate. Predators communicate with other predators by broadcasting messages to their neighbors according to a communication range h. Four predators are needed to capture a prey by filling out four different roles: surround the prey to the north, south, east and west. Those roles are independent of each other and can be started at any time obviating the need for scheduling. The only requirement is that they

have to terminate at the same time either successfully when a capture occurs or unsuccessfully if no team can be formed. The predator agents are homogeneous and can assume any role but heterogeneity can be introduced by restricting the role(s) an agent can assume. The prey and predators move concurrently and asynchronously at different time steps. In addition to the four orthogonal navigational steps, the agents can opt to stay in place. Non-determinism is introduced with the modeling of collisions. In case of collision, the agents are held back to their previous position. The preference u_{ij} of predator agent i for a role j is inversely proportional to the Manhattan distance d required to achieve the role.

The predators move in the direction of their target when assigned a role or explore the space according to a memory-based scheme on the last few steps. The decision space for the role allocation of P predators and p preys is $O(p^T)$ where T is the number of teams of size t. This problem belongs to the most difficult class of problems for constraint satisfaction in multi-agent systems due to the dynamic nature of the environment and the mutually-exclusive property of role allocation. An optimization algorithm can be used in parallel fashion by each agent based on sensed and communicated information from the other agents in the group to autonomously determine which role to assume (see Algorithm 3). It is assumed that the other agents reach the same conclusions because they use the same optimization algorithm [12] and the same payoff function. This type of algorithm degrades gracefully when communication is completely impaired since it does not rely on the communication of intent or preference from the other agents and can rely solely on sensory information. Information necessary to determine the payoff of each role needs to be communicated. Therefore, it is the current local state within the perception range, or augmented with second hand information, that is communicated to the neighbors instead of the intended role. What is being communicated is a location on the grid. The "Hungarian" algorithm [13] based on weighted graph bipartite matching was found to outperform other types of distributed role allocation in dynamic and uncertain environments [14], albeit with the assumption of a homogeneous cost function, and is used here to determine the optimal role for the agent in a team characterized by the prey to pursue as the common goal.

Which team to join when multiple preys are present requires a commitment for teamwork beyond role allocation optimization if not enough agents are available to accomplish the overall task. Human users of coordination proxy agents might have vested interest in selecting one team over another such as friendships, trust, loyalty, etc.

Algorithm 3 Distributed Role Allocation

```
Initialize:
  set initial role to explore
  active ←true
while (no termination condition) do
  if (active) then
    act according to role
    sense environment
    broadcast local state to neighbors
   active ←false
  endif
  collect neighbors' new information
  estimate possible roles with
    allocation algorithm
  select role
  active ←true
end while
```

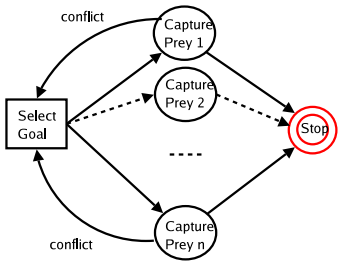

Fig. 4. Prey/Predator HAM

Figure 4 describes the HAM for selecting a team with the common goal of capturing a prey. Our state representation of goal conflict consists of 5 features: the distance of the prey from the agent, the number of agents allocated to the prey, the total proximity of the agents allocated to the prey, and the direction of the prey from the agent (4 quadrant values based on the coordinate system: NW, NE, SW, SE) using 1-of-n encoding. Associated with the state representation are two high-level decisions – select the target or not – determined by the sign of the action value . An ambiguity will occur if the decisions do not uniquely select one goal or if no goals are selectable.

Empirical Results

The experiments were conducted with RePast [15], an agent-based simulation and modeling tool where agents act concurrently in a decentralized manner. Its powerful scheduling mechanism was used to model the asynchronous behavior of the agents in a discrete-event simulation. In addition, its neighborhood mechanism was used to model broadcast communication between neighbors.

Table I shows the number of clusters (number of winning neurons) obtained in the preprocessing step (see) of a random sample of 2400 choice points when varying the communication range h of 4 predators and 2 preys on a 30x30 grid and 5% message loss depending on distance. In our implementation only the winner prototype vectors were trained. The increase in granularity indicates the increased number of distinct recognizable situations in the training set with increasing h (t-test p-value of 0.003 over 10 runs between clusterings with h=5 and h=11). Results shown are obtained after convergence to a maximum norm correction distance less than 1.E-4 of 1000 initial prototype vectors or 5000 training epochs.

Comm Range h	Avg Clusters	Avg Err
5	240	0.0137
7	344	0.0182
9	377	0.0287
11	367	0.0268

TABLE I

CLUSTER GRANULARITY FOR VARYING COMMUNICATION RANGES BASED ON RANDOM SAMPLES OF 2400 CHOICE POINTS IN THE TRAINING SET AND AVERAGE ERROR FOR 400 CHOICE POINTS IN THE TESTING SET ($p = 2, \alpha = 0.07$)

The following experiments were done on a 15x15 grid with 5 predator agents, 2 random preys, and 5% message loss. The agents start at random locations on the grid and the predators are as likely to be slower or faster than the prey. A terminal reward of +1.0 is propagated after each capture or a penalty of -1.0 if no preys are captured after 200 cycles (episode). Learning occurs across 500 episodes. Figure 5 compare performance trend results for the different user patterns with refinement of those strategies by the coordination proxies and performance improvement by the proxy agent acting autonomously (with Boltzmann exploration) for catching the first prey. Duplicate consecutive goal states were eliminated. An intermediate reward is allocated when the proxy agent selects the same decision as the user pattern (user precision). Trend results are also shown in terms of ambiguities resulting from this refinement in Figure 6 and in terms of precision to those user patterns in Figure 7.

The results for the different user patterns compared with autonomous learning show clearly that mixed-initiative HAMQ learning can produce a more stable behavior while reducing interactions due to situational ambiguities with the human in the loop and increased user precision.

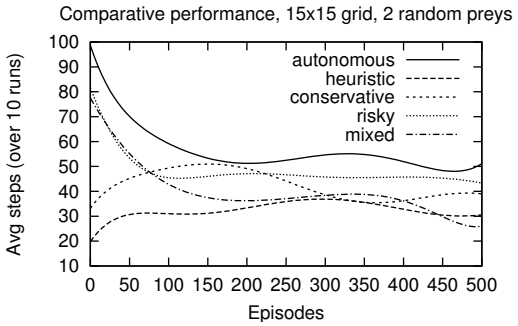

Fig. 5. Comparative performance of user pattern refinements and autonomous learning of 5 mixed-initiative predator agents in capturing the first prey
$h = 7, p = 2, \alpha = 0.07, \gamma = 0.99, r = 4E - 5$

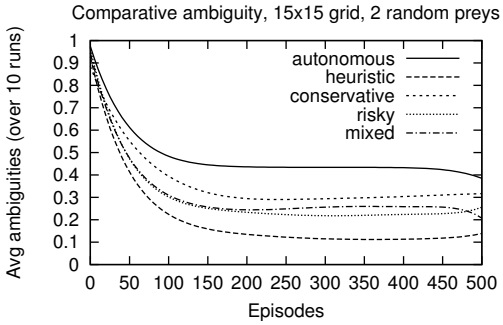

Fig. 6. Ambiguity rate

If an unforeseen situation arises resulting in no selectable goal, the coordination proxy agent will initiate an interaction with its user. The standard deviations of the learned behaviors over 100 episodes were as follows: (1) autonomous 47.54, (2) heuristic 25.42, (3) risky 43.42 , (4) conservative 34.91, and (5) mixed 35.25. This methodology can also validate user patterns according to their precision rate since successful behavior will most likely be followed. The key factor in reducing the ambiguity rate for all behaviors seems to be in the credit assignment to alternatives (Kolmogorov-Smirnov test p-values of average performance over 10 runs in ablation studies were (1) autonomous 2.2E-4, (2) heuristic 2.2E-4, (3) risky 2.2E-4 , (4) conservative 3.E-5 and (5) mixed 0.20).

RELATED WORK

Previous work on adjustable autonomy [16], [17] has concentrated on learning user preferences in isolation without relation to the outcome of the task. Learning co-adaptive predictive models of the exogenous outcome

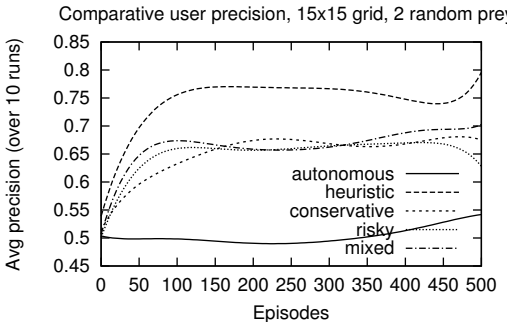

Fig. 7. User Precision rate

instead of co-adaptive behaviors has been suggested in the context of the El Farol coordination problem where the paradox was that agents could coordinate without communication by not going to the same place, the El Farol Bar, at the same time [18]. It was observed that the online adaptation of strategies to recent situations rather than convergence was the key to achieving coordination in a multi-agent system. Other work in this area found that coordination could be achieved with less variance if the agents relied on the accuracy of the same adaptive gradient algorithm [19].

Techniques from advice taking [20], [21] where the user preference is explicitly incorporated into the internal representation of the agent are complementary to this approach. Similar to apprenticeship learning[22], state-action trajectories are used to refine an existing policy through a reinforcement learning algorithm. In W-learning [23], each behavior in a flexible subsumption architecture compete with other behaviors as selfish agents indirectly collaborating through the communication of the W value and a winner-take-all scheme. Here, the user and the result-driven proxy agent interact directly as collaborating agents trying to learn from each other.

CONCLUSION

We have shown how coordination proxies can safely adjust their autonomy in switching teams based on user preferences and taking into account the outcome of the decision. Hierarchies of abstract machines are superimposed to the temporal behavioral of the proxy agent to specify the high-level behavior of the human in the loop. The representation of goal states instead of environmental states enables reinforcement learning to scale up and generalize to different situations. The complementary credit assignment to alternatives seems to be a key factor in reducing ambiguities while increasing user precision and this hypothesis will be further studied. In addition, future work should extend this approach to preferences of states as hints for exploration and compound goals where choice points determine a sequence of high-level goals. This work is part of on-going research in the coordination of intelligent agents in open environments.

REFERENCES

[1] P. Scerri, D. Pynadath, N. Schurr, A. Farinelli, S. Gandhe, and M. Tambe, "Team oriented programming and proxy agents: The next generation," Workshop on Programming MultiAgent Systems, AAMAS 2003.

[2] J. F. Allen, "Mixed-initiative interaction," *IEEE Intelligent Systems*, 1999.

[3] T. Kohonen, *Self-Organizing Maps*, 2nd ed. Springer, 1997.

[4] M. L. L. L. P. Kaebling and A. W. Moore, "Reinforcement learning: A survey," *Journal of Artificial Intelligence Research*, vol. 4, pp. 237–285, 1996.

[5] R. S. Sutton and A. Barto, *Reinforcement Learning: an Introduction*. Cambridge, MA: MIT Press, 1998.

[6] M. L. Putterman, *Markov Decision Processes*, 2nd ed. Wiley-Interscience, 2005.

[7] R. Parr and S. Russell, "Reinforcement learning wiht hierarchies of machines," in *Neural Information Processing Systems*, 1998.

[8] M. Abramson, P. Pachowicz, and H. Wechsler, "Competitive reinforcement learning for continuous control tasks," in *Proceedings of the International Neural Network Conference*, 2003.

[9] V. N. Papudesi and M. Huber, "Learning from reinforcement and advice using composite reward functions," in *Proceedings of the 16th International FLAIRS Conference*, 2003.

[10] A. Blum, "On-line algorithms in machine learning," in *Online Algorithms; the State of the Art*, Fiat and Woeginger, Eds. LNCS, 1998, no. 1442, ch. 14.

[11] M. Benda, V. Jagannathan, and R. Dodhiawalla, "On optimal cooperation of knowledge sources," Boeing AI Center, Boeing Computer Services, Tech. Rep. BCS-G2010-28, 1985.

[12] B. P. Gerkey and M. J. Mataric, *RobotCup 2003*. Springer-Verlag Heidelberg, 2004, vol. 3020, ch. On Role Allocation in RobotCup.

[13] H. W. Kuhn, "The hungarian method for the assignment problem," *Naval Research Logistics Quarterly*, vol. 2, no. 83, 1955.

[14] M. Abramson, W. Chao, and R. Mittu, "Design and evaluation of distributed role allocation algorithms in open environments," in *International Conference on Artificial Intelligence*, Las Vegas, NV, 2005.

[15] N. Collier, "Repast: the recursive porous agent simulation toolkit," Retrieved from http://repast.sourceforge.net, 2001.

[16] P. Scerri, M. Tambe, H. Lee, and D. Pynadath, "Don't cancel my barcelona trip: adjusting autonomy of agent proxies in human organizations," in *AAAI Fall Symposium on Socially Intelligent Agents – the Human in the Loop*, 2000.

[17] M. Tambe, P. Scerri, and D. Pynadath, "Adjustable autonomy for the real world," *Journal of Artificial Intelligence Research*, vol. 17, pp. 171–228, 2002.

[18] W. B. Arthur, "Inductive reasoning and bounded rationality," *American Economic Review*, 1994.

[19] A. M. Bell and W. A. Sethares, "Avoiding global congestion using decentralized adaptive agents," *IEEE Transactions on Signal Processing*, vol. 49, no. 11, 2001.

[20] R. Maclin and J. W. Shavlik, "Incorporating advice into agents that learns from reinforcements," in *Proceedings of the 1994 American Association of Artificial Intelligence*, 1994.
[21] M. Boicu, G. Tecuci, and D. Marcu, "Mixed-initiative assistant for modeling expert's reasoning," in *Proceedings of the AAAI-05 Fall Symposium on Mixed-Initiative Problem-Solving Assistants*, 2005.
[22] P. Abbeel and A. Y. Ng, "Exploration and apprenticeship learning in reinforcement learning," in *Proceedings of the 22nd International Conference on Machine Learning*, 2005.
[23] M. Humphrys, "W-learning: Competition among selfish q-learners," University of Cambridge, Tech. Rep. 362, 1995.