Real Time Learning of Behaviour Features for Personalised Interest Assessment

Sylvain Lemouzy, Valérie Camps and Pierre Glize

Abstract This paper deals with an adaptive and personalized algorithm to dynamically determine the interest of a user in a document from the observation of his behaviour during its consultation. Several existing works propose an implicit feedback of user's interest from observations of his behaviours when he is reading a document. Nevertheless, the used algorithms are *a priori* defined and then cannot be really adapted to each user who has personal habits when he is working with the Web. This paper focuses on the generalisation of this problem and proposes a multi-agent algorithm that, dynamically and according to each user's specific behaviours, computes the personal interest assessment of each of them. Several experimentations show the efficiency of this algorithm, able to self-adapt its functionality in real time when the user habits evolve, compared to existing methods.

1 Introduction

The incessant growth of the World Wide Web increases the number of responses provided by the search engines to a request. The user is then faced to an information overload from which it is difficult to distinguish information that is directly relevant from information that is secondary or even irrelevant. Furthermore, the answers to requests are determined regardless of the user that sent them or the context in which they are issued; in particular, the answers to a same request expressed by two different users are identical, even if these users do not have the same expectations, the same preferences and the same interests.

The use of personalisation is a way to reduce this informational overload. It enables to implicitly or explicitly limit the answers according to the needs and specificities of the user. Personalisation is based on the notion of user profile ([8], [1]

Sylvain Lemouzy, Valérie Camps and Pierre Glize
Université Paul Sabatier Toulouse 3 – IRIT, Toulouse, France e-mail: {lemouzy, camps, glize}@irit.fr

which represents his interests, his preferences and his needs. User's preferences usually relate to various aspects such as the access to the search system, the display of the returned answers, the criteria for limiting the desired information, the data related to his geolocalisation, etc. The interests of the user are related to fields which can be expressed through preferences. Finally the user needs can either be explicitly expressed by him or be discovered by a system that observes his actions, learns about his behaviour and makes a representation of his habits from which it can deduct his need(s).

This paper focuses on this last point: the implicit determination of the interest of a user facing document provided by a search engine in response to his request. Having highlighted the importance of the adaptive needs of the implicit user preferences assessment, a generalization and a formalization of this problem are proposed. The section 3 is devoted to the proposed algorithm which is based on the AMAS approach. Various experiments and results to evaluate its relevance are then presented before concluding and giving some perspectives to this work.

2 Personalized assessment of user's interests

2.1 Towards a personalized implicit feedback

Whatever the interest of a user may be, it changes over time. The user profile has to be updated accordingly, by taking into account and by incorporating these changes. Several methods exist to account for this evolution depending on whether the system appeals to the user explicitly (explicit feedback) or not (implicit feedback).

In an explicit feedback, the user has to indicate his assessment of the results returned by the system (a document in the context of information retrieval). Three main approaches exist to express this assessment: a binary approach (like / not like, interesting / not interesting or relevant / irrelevant) [3] a more incremental approach using discrete values expressed by a note [9] and a textual approach. Such an explicit evaluation is admittedly easy to integrate into the system; but if one takes the user's point of view, it is not quite easy to express an opinion using numerical scales and particularly, the user does not necessarily have the time or the inclination to devote too much energy to this task.

Implicit feedback tries to remedy these problems by attempting to automatically infer the preferences of the user, without appealing to him but by observing him interacting with the system. This observation relies on some *implicit interest indicators* such as the study of links followed by users when searching [7], the study of the history of purchases (Amazon), etc. Others considers the indicators commonly used [5], [4] such as the time spent on a document, the number of mouse clicks, the page bookmarking, the printing or saving a document, etc., these indicators needing to be later interpreted to extract an interest.

However, to be faithful to the user and contrary to what is usually done in such cases in the literature [12], interpretation of these indicators should not be identical for all users but should truly adapt to specificities of each of them. In particular, a user might prefer to print a document of interest while another might prefer to bookmark it. The particularity of each user has to be taken into account to extract as closely as possible his interests and to provide a truly personalized system because tailored for one specific individual. It is therefore necessary to learn and to find relevant indicators but also their involvement in the function enabling to determine the most interesting documents for a particular user, while bearing in mind that they can evolve for a same user but also from one user to another.

Nevertheless, the implementation of such implicit feedback can not be fully disconnected from the user and requires some minimal interaction with him. In particular it requires to construct and to take into account two highly interconnected aspects: (*i*) the first one, based on a user's feedback and on a range of indicators obtained from observations of the user's behaviour, consists in extracting discriminating indicators that contextually reflect his interests in the consulted documents; (*ii*) the second consists in defining and in implementing mechanisms which, as few as possible, appeal to the user for a feedback (for example only when mechanisms which implement the implicit feedback cannot deduct anything).

This paper focuses on the first point and supposes that the second issue is solved; we assume that the decision function of the user is known at every moment, even if it is evolutive.

2.2 Learning user behaviour features to asses its centres of interest

We aim at designing a system able to assess the interest of a user for a document from some observations on his behaviour when he is consulting it. Each observation is a potential criterion enabling the determination of a piece of the interest of a user for this document. Thus, the adaptation of the assessment function of the user's interest consists in dynamically defining the importance of each criterion. That is why we propose a real time learning algorithm based on behaviour features which aims at deducing the importance of each criterion for a personalized interest assessment (i.e. contextual and adapted to each user).

Two main classes of learning algorithms can be distinguished: the supervised one where a learning "teacher" giving input and output samples exists and the unsupervised one where a "teacher" does not exist, where the desired output is not used [6]. As our objective is to be the most independent as possible of the user, a supervised learning algorithm is not suited. That is why neuronal networks (NN) cannot be a solution to solve our problem. Such systems strongly depend on the intended solution that is, in our case, on the user's feedback. The reinforcement learning (RL), first introduced by [15] as $\mathscr{Q}$-learning and improved and implemented in many other works [13], is an intermediate class; "after choosing an action the agent is told the immediate reward and the subsequent state, but is not told which action would have been in its best long-term interests" [6]. Even if they have been extended to MAS [14], $\mathscr{Q}$-learning inspired algorithms are only efficient in Markovian environments but in our problem, the environment, *i.e.* the user, is known as not being Markovian at all. Furthermore, we also believe that faced with the diversity of observed behaviours to take into account and faced with the dynamic of these behaviours, the intended assessment function cannot be checked and managed by an external supervision. The system implementing this function has to be autonomous and to adapt itself locally to environmental changes, according to what it perceives and its internal state. Thus, the learning phase is a never-ending process.

Because of its intrinsic nature, human decision making cannot be reduced to a precise formal model. Moreover, this decision making function evolves during time. Thus, according to [11] the environment of the system to build – *i.e.* the user – is dynamic, inaccessible (its state cannot be totally known) and non deterministic (it is not possible to known the effect of a given action upon its behaviour). The AMAS (Adaptive Multi-Agent Systems) approach [2] is particularly adapted to solve such kind of problem; therefore, we used it to implement our real time learning algorithm.

In this approach, a system is said *functionally adequate* if it produces the function for which it was conceived, according to the viewpoint of an external observer who knows its finality. To reach this functional adequacy, it had been proven that each autonomous agent that composes an AMAS and pursues a cycle composed of three steps (perception/decision/action) must keep relations as cooperative as possible with its social (other agents) and physical environment. The definition of cooperation we use is not conventional (simple sharing of resources or common work). Our definition is based on three local meta-rules the designer has to instantiate according to the problem to be solved: **Meta-rule 1** (c_{per}): Every signal perceived by an agent has to be understood without ambiguity; **Meta-rule 2** (c_{dec}): Information coming from its perceptions has to lead the agent to produce a new decision; **Meta-rule 3** (c_{act}): This reasoning must lead the agent to make the actions which have to be useful to other agents and the environment. If one of this meta-rule is not checked, the agent is faced to a "Non Cooperative Situations" (NCS) that can be assimilated to an "exception" in traditional programming. It occurs when at least one of the three previous meta-rules is not locally verified by an agent. Different generic NCS have been highlighted: *incomprehension* and *ambiguity* if (c_{per}) is not checked, *incompetence* and *unproductiveness* if (c_{dec}) is not verified and finally *uselessness*, *competition* and *conflict* when (c_{act}) is not checked. This approach is proscriptive because each agent must first of all, anticipate, avoid and repair a NCS. This has strong methodological implications: designing an AMAS consists in defining and assigning cooperation rules to agents. In particular, the designer, according to the current problem to solve, (*i*) has to determine what an agent is, then (*ii*) he has to define the nominal behaviour of an agent then (*iii*) he has to deduce the NCSs the agent can be confronted with and (*iv*) finally he has to define the actions the agent has to perform to come back to a cooperative state. The designer has to keep in mind that agents do not have a view of the global system and do not base their reasoning on the expected collective function realized by the system.

2.3 Problem Generalisation

In order to formalise the problem, we set some hypothesis. First of all, we suppose that two types of criterion exist : *(i)* boolean criterion (*i.e.* document printing, saving, *etc.*) ; *(ii)* continuous criterion (*i.e.* document consultation time, *etc.*). Let $\mathscr{C}_b$ be the set of boolean criteria and $\mathscr{C}_c$ be the set of continuous criteria. The consultation of a document leads to a set of observations, each of them matching with a criterion. Let $c_{i,j}$ be the normalised value of the criterion c_i for the j^{th} document consultation.

If $c_i \in \mathscr{C}_b$ then $c_{i,j} = 1$ when the value of c_i is "`true`", otherwise $c_{i,j} = 0$.

If $c_i \in \mathscr{C}_c$ then $c_{i,j} \in [0;1]$. This standard range avoids taking into account the meaning of an observation. For example, if we assume that the consultation time doesn't relay any interest value before 5 seconds and reaches the highest interest after 2 minutes, then before 5 seconds $c_{i,j} = 0$, after 2 minutes $c_{i,j} = 1$ and inside this range $c_{i,j}$ evolves linearly.

Each criterion c_i is associated with an influence degree $\omega_i \in [0;1]$ that stands for the importance of the criterion during the user's interest calculus. When $\omega_i = 1$, c_i is very important for it, but when $\omega_i = 0$ c_i has no influence. Thus the local interest of c_i is equal to $c_{i,j} \cdot \omega_i$.

We suppose that higher the local interests are, higher the global interest is. Thus, the interest value of the user u for an observed document j, can be approximated by a weighted sum function $\mathscr{F}$ and the adaptation of the user interest evaluation function can be generalised to the search of the multi-criterion decision function $\mathscr{F}$ considering:

$$\mathscr{I}_u(j) = \mathscr{F}(j), \forall j \text{ with } \mathscr{F}(j) = \sum_{i=1}^{n} (\omega_i \cdot c_{i,j})$$

Where $\mathscr{I}_u$ is the multi-criterion decision function of the user u to approximate; n is the number of criteria; $c_{i,j}$ is the normalised value of c_i for the decision j; ω_i is the influence value of c_i.

Thus, solving this problem consists in searching the set of ω_i that verifies the previous equation. As already said in the §2.2, the decision function $\mathscr{I}_u$ can evolve. Therefore, in order to keep an adequate decision function, this equation has to be verified for only the few last decisions.

3 Real Time Learning of User's Interests

3.1 System Analysis

The AMAS that we have to conceive is a multi-criterion decision making function $\mathscr{F}$ defined by the weighted sum $\mathscr{F}(j)$ given in section 2.3. The goal of the system is to learn the set of ω_i in order to approximate the target decision making function $\mathscr{I}_u$. This function has to self-adapt in real time to its environment: the user. Therefore, we propose to incrementally improve it at each decision j, with the following life cycle: *(i)* the system perceives the values of the criteria; *(ii)* the decision value $\mathscr{F}(j)$ is calculated with these values and is sent to the environment; *(iii)* depending on this value, the environment returns a feedback $fb(j) \in \{\uparrow, \downarrow, \sim\}$ in order to inform the system if the searched $\mathscr{I}_u(j)$ is higher ($\uparrow$), lower ($\downarrow$), or equal ($\sim$) to the provided one; *(iv)* the system adjusts the values of ω_i in order to correct the possible errors.

Thus, at each new decision, the system adjusts its function in order to converge towards a more and more adequate decision making value. Because the decision making function is not *a priori* known, it is possible that this function can only be approximated. Thus, ε expresses a tolerance value such as: if $|\mathscr{F}(j) - \mathscr{I}_u(j)| \leqslant \varepsilon$ then the environment returns a "$\sim$" feedback.

3.2 Agents Identification

Here the only entities that have a local goal inside the system are the ones that are in charge of the local decision value of a criterion: so we can identify the *criterion agents*. In the rest of this paper, "agent" refers to "criterion agent". The goal of a criteria agent c_i is then to adjust its influence value ω_i so that its partial function – its local decision – enables the system to figure out an adequate global decision. Now, we have to define the local behaviour of our agents.

3.3 Identification of Non Cooperative Situations

During the perception phase, the perceived value of an agent is the current value of its criterion. This value is well defined and cannot generate an ambiguity, thus, agents cannot be faced to a NCS at this step. During the decision phase, each agent has to multiply two values, so no NCS can occur during this step. During the action phase that can generate three NCS (uselessness, concurrency and conflict), only the conflict situation is relevant for an agent. When an agent decides locally a value that contradicts the decision of the user, it is in conflict with the environment. This situation occurs when the agent provides a local decision value that is involved in the generation of a wrong global decision, that is to say, when the normalised value of the criteria is considered not null and when the environment returns a feedback different to "$\sim$". More formally, this cooperation failure is detected by the agent c_i for the decision j when $c_{i,j} > \varepsilon \wedge fb(j) \neq \ \sim$.

3.4 Cooperative Agent Behaviour

When a conflict is detected by an agent, he has to act in order to suppress this situation. Thus, when the feedback is "$\uparrow$" the agent c_i has to increase ω_i; when the feedback is "$\downarrow$" the agent has to decrease ω_i. Let be Δ_{ω_i} the increment or decrement step value of ω_i. In order to be really cooperative, each agent has to tune its ω_i according to its possible involvement in the conflict situation. If the value of $c_{i,j}$ is relatively small, an agent can assume that it was not really involved in the error, so it decides to tune its ω_i very slightly. In the opposite case, c_i must strongly tune ω_i. So, from a local point of view, ω_i values are tuned proportionally to the possible involvement of the corresponding agent c_i. More formally, the agent c_i increments or decrements ω_i with $\Delta_{\omega_i} \cdot c_{i,j}$.

An autonomous and cooperative agent must be able to decide by itself the modification strength of its ω_i. Therefore, each agent must be able to define the uncertainty of the current value of ω_i. Higher the uncertainty is, higher ω_i is modified. On the other hand, lower the uncertainty is, slighter ω_i is modified.

Thanks to the local observation of successive feedbacks, agents can consider the uncertainty of ω_i and then define its tuning step Δ_{ω_i} as follows. Let *(i)* $\Delta_{\omega_i}^{max}$ be the max value of the tuning step of ω_i; *(ii)* λ_{ω_i} be the maximum number of ω_i modifications that leads to the modification of Δ_{ω_i}. λ_{ω_i} stands for a kind of "tuning sensibility" for ω_i; *(iii)* $\Delta_{\omega_i}^{min}$ be the maximal value of the modification step of ω_i. It is defined as ❶. When the agent tunes successively ω_i towards opposite directions, the uncertainty decreases, thus Δ_{ω_i} is decreased as ❷. When the agent tunes successively ω_i towards the same direction, the uncertainty increases, thus Δ_{ω_i} is increased as ❸.

$$❶\ \Delta_{\omega_i}^{min} = \frac{\Delta_{\omega_i}^{max}}{2^{\lambda_{\omega_i}}}\ ;\ ❷\ \Delta_{\omega_i} \leftarrow \max\left(\frac{\Delta_{\omega_i}}{2}, \Delta_{\omega_i}^{min}\right)\ ;\ ❸\ \Delta_{\omega_i} \leftarrow \min(\Delta_{\omega_i} \cdot 2, \Delta_{\omega_i}^{max})$$

Thus, without any other interaction with other agents, this behaviour enables each agent to tune in real time its ω_i in order to contribute as well as it is possible to the global assessment function. Although agents are fine-grained, they have a real autonomous behaviour. This behaviour is led by the local estimation of the goodness or badness of their contribution to the collective activity. Each agent is able to estimate the uncertainty (Δ_{ω_i}) of its ω_i. The observation of each criterion does not explain on its own the result of the collective activity which is effectively an emergent one. Indeed, the use of the AMAS cooperation ensures the functional adequacy at the higher level. This adequacy is verified in the following section.

4 Experiments and analysis

In order to analyse the multi-agent system introduced in section 3, this algorithm has been implemented in Java. The system, which represents a $\mathscr{F}$ function, is initialised with n criteria c_i, each associated to a random ω_i. Its environment is the target function $\mathscr{I}_u$ which is initialised with other random ω_i values. At each decision phase, $\mathscr{F}$ receives feedback from $\mathscr{I}_u$ and then tries to adapt its function.

4.1 Checking of convergence

The purpose of the first experiments was to check if the system is able to find a solution whatever its internal state is, with a decision function $\mathscr{F}$ containing 5 boolean criteria and 5 continuous criteria. During the adaptation process, the system is regularly disturbed by random modifications of agents' states (*i.e.* each 1500 cycles) each ω_i and Δ_{ω_i} are randomly modified. The figure 1 shows the evolution of the average distance of ω_i between the adaptive function and the searched one. After each disturbance, the system converges towards the solution and gets set after approximately 600 cycles. Thus, we can conclude that whatever the state of the system, it is able to converge towards the solution.

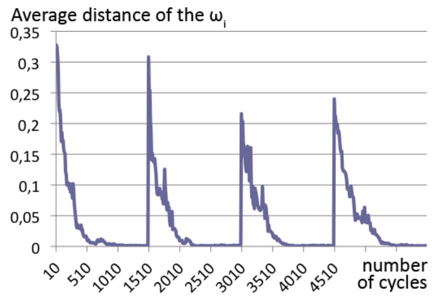

Fig. 1 Convergence of the system

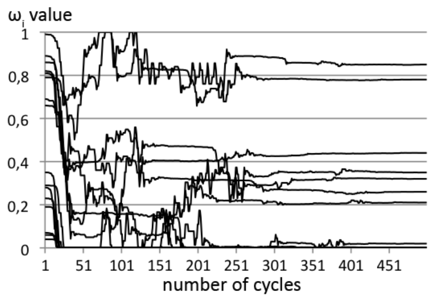

Fig. 2 The pruning of useless criteria

4.2 The pruning of useless criteria

Because the relevant criteria are not *a priori* known, the algorithm has to perceive as many criteria as possible even if some of them can be useless. Indeed, the adaptation process has to find by itself useless and useful criteria. The figure 2 illustrates an example of this criteria pruning where only 8 out of 16 criteria are relevant ($\omega_i > 0$). We notice that the system filters out theses criteria by setting their influence to zero. The influence of these useless criteria converges quickly towards zero; this does not really disturb the influence of relevant criteria. Thus, for practical purposes, it will be worthwhile to add more criteria than those that are expected to be relevant.

4.3 Convergence speed

We have studied the convergence speed of the algorithm when *(i)* the system is composed of boolean criteria and when *(ii)* it is composed of continuous criteria. After

100 searches, the average number of cycles required to find a solution is calculated for 4, 8, 16, 32, 64, 128 and 256 criteria. We consider that the system finds a solution if the environment returns a "$\sim$" feedback during 100 consecutive cycles.

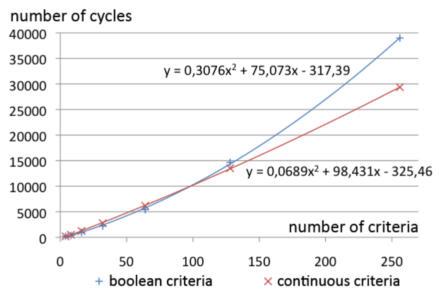

Fig. 3 Average convergence speed

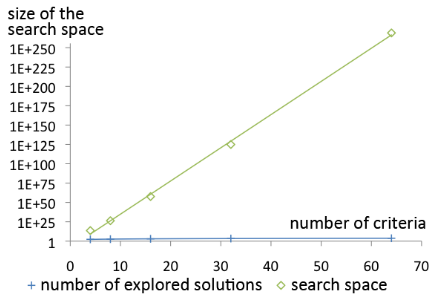

Fig. 4 Search space

The figure 3 shows the average number of cycles required to find a solution depending on the number of criteria belonging to the searched function. When the function is composed of boolean criteria the complexity is polynomial whereas when the function is composed of continuous criteria, the complexity, even though being actually polynomial, is very close to the linear complexity. This difference of complexity is due to the fact that boolean criteria have a value that is either 0 or 1 and then their ω_i tuning is more crude than the continuous criteria one.

In order to estimate more precisely the efficiency of the algorithm, we compared the number of explored states (before finding a solution) to the size of the solution space. Let $|C|$ be the number of criteria and $|\omega_i|$ be the cardinal of possible values of ω_i. This cardinal is obtained by dividing the range of ω_i by $\Delta_{\omega_i}^{min}$. The search space of $\mathscr{F}$ is then $S = \prod_{i=1}^{|C|} |\omega_i|$. As shown in figure 4, whereas the search space grows exponentially (the ordinate axis range is logarithmic), the number of explored solutions evolves quasi-linerally. We can then conclude that the algorithm converges quasi-linerally depending on the number of criteria and does not depend on the size of the search space. Thereby, this algorithm is extremely efficient, all the more so the processing time of each criterion (not introduced here) is linear.

5 Conclusion and Perspectives

This article proposes, implements and evaluates an adaptive model for the assessment of the interest of one particular user. This problem is generalised to the adaptive learning of a multi-criteria decision function which is a weighted sum of observed criteria values. This decision function is *a priori* unknown and can evolve. Solving this problem consists in searching the relevant criteria that are actually used by the user during the decision making process.

The proposed system is an Adaptive Multi-Agent System composed of criteria agents. Their local goal is to determine their relative influence in the decision function. We have defined and implemented a local behaviour that enables agents to correct the system errors. Thanks to a set of experiments, we have verified the convergence and the adaptiveness properties of the system. These experiments have also shown the efficiency of the proposed solution: it finds a solution with a polynomial complexity, even if the search space grows exponentially with the number of criteria.

Although this system works well with simulated users, we are now experimenting and implementing it in a real world problem. Therefore we are currently working to its integration in the iSAC project [10], an intelligent Citizens Information Service, in collaboration with the university of Girona.

References

1. Brusilovsky, P., Kobsa, A., Nejdl, W.: The Adaptive Web, Methods and Strategies of Web Personalization. In: The Adaptive Web, *LNCS*, vol. 4321. Springer (2007)
2. Camps, V., Gleizes, M.P., Glize, P.: A self-organization process based on cooperation theory for adaptive artificial systems. In: 1st Int. Conference on Philosophy and Computer Science "Processes of evolution in real and Virtual Systems", Krakow, Poland (1998)
3. Chen, L., Sycara, K.: Webmate: A personal agent for browsing and searching (1998)
4. Claypool, M., Claypool, M., Le, P., Le, P., Waseda, M., Waseda, M., Brown, D., Brown, D.: Implicit interest indicators. In: In Intelligent User Interfaces, pp. 33–40. ACM Press (2000)
5. Jude, J.G., Shavlik, J., Dept, C.S.: Learning users' interests by unobtrusively observing their normal behavior. In: In Proceedings of the 2000 International Conference on Intelligent User Interfaces, pp. 129–132. ACM Press (2000)
6. Kaelbling, L.P., Littman, M.L., Moore, A.W.: Reinforcement learning: A survey. Journal of Artificial Intelligence Research **4**, 237–285 (1996)
7. Lieberman, H.: Letizia: An Agent That Assists Web Browsing. In: C.S. Mellish (ed.) Proceedings of the 14^{th} International Joint Conference on Artificial Intelligence (IJCAI-95), pp. 924–929. M. Kaufmann Inc., Montreal, Quebec, Canada (1995)
8. Montaner, M., López, B., Lluís De La Rosa, J.: A Taxonomy of Recommender Agents on the Internet. Artif. Intell. Rev. **19**(4), 285–330 (2003)
9. Moukas, A.: User Modeling in a MultiAgent Evolving System. In: Workshop on Machine Learning for User Modeling, 6^{th} International Conference on User Modeling. Chia Laguna, Sardinia. (1997)
10. de la Rosa, J., Rovira, M., Beer, M., Montaner, M.: Reducing Administrative Burden by Online Information and Referral Services. In: Citizens and E-Government: Evaluating Policy and Management. Reddick, C.G., Austin, Texas (to appear in 2010)
11. Russell, S., Norvig, P.: Artificial Intelligence: A Modern Approach, 2nd edition edn. Prentice-Hall, Englewood Cliffs, NJ (2003)
12. Seo, Y.W., Zhang, B.T.: A reinforcement learning agent for personalized information filtering (2000)
13. Sutton, R., Barto, A.: Reinforcement Learning: An Introduction. MIT Press, Cambridge (1998)
14. Thomas, V., Bourjot, C., Chevrier, V.: Interac-DEC-MDP: Towards the use of interactions in DEC-MDP. In: Third Int. Joint Conference on Autonomous Agents and Multi-Agent Systems - AAMAS'04, pp. 1450–1451. New York (2004)
15. Watkins, C.: Learning from delayed rewards. Ph.D. thesis, University of Cambridge, England (1989)