INTERSPEECH 2009 BRIGHTON

Hybridisation of Expertise and Reinforcement Learning in Dialogue Systems

Romain Laroche [12], *Ghislain Putois* [1], *Philippe Bretier* [1], *Bernadette Bouchon-Meunier* [23]

[1] Orange Labs, Issy les Moulineaux, France
[2] Laboratoire d'Informatique de Paris VI, Paris, France
[3] Centre National de Recherche Scientifique, Paris, France
romain.laroche@polytechnique.org

10.21437/Interspeech.2009-660

Abstract

This paper addresses the problem of introducing learning capabilities in industrial handcrafted automata-based Spoken Dialogue Systems, in order to help the developer to cope with his dialogue strategies design tasks. While classical reinforcement learning algorithms position their learning at the dialogue move level, the fundamental idea behind our approach is to learn at a finer internal decision level (which question, which words, which prosody, ...). These internal decisions are made on the basis of different (distinct or overlapping) knowledge. This paper proposes a novel reinforcement learning algorithm that can be used to make a data-driven optimisation of such handcrafted systems. An experiment shows that the convergence can be up to 20 times faster than with Q-Learning.

Index Terms: SDS, reinforcement learning, adaptive dialogue

1. Introduction

Spoken Dialogue Systems (SDS) with learning capabilities have been deeply investigated these last ten years [1, 2, 3, 4]. The primary goal pursued by these studies was to reduce the development cost by automating the SDS design. They use Markov Decision Process (MDP) to learn the best dialogue move according to the current dialogue state. This approach has led to promising results but, as [5, 6] acknowledge, some obstacles are difficult to overcome. The high dimensionality of the dialogue states and actions is the main reason for all these troubles. Firstly, the MDP sets (the state set and the action set) require a tight tuning, which can only be performed by a reinforcement learning expert and which is quite contrary to the original automation objective. Secondly, the learning process requires to gather huge dialogue corpora which can be problematic depending on the application. Finally, even when all those conditions are fulfilled, the automatically generated SDS reported in the literature are usually simple and hardly reach the dialogue quality of the fine tuned handcrafted ones.

Releasing noticeably the automation goal, while [7, 8] focussed more specifically on the SDS robustness with POMDP, [9, 10] made several attempts to improve the learning convergence at the expense of even more MDP design. These papers propose to use various Hierarchical Reinforcement Learning algorithms (MAXQ [11] and HAM [12]) in order to help the learning with a defined hierarchy of strategies: macro-strategies (*i.e.* a strategy that is followed during several turns) and one-dialogue-turn strategy. The one-dialogue-turn strategies sequences are optimised in order to fulfil a macro-strategy and the transitions between macro-strategies are learnt too. While [9, 10] correctly stresses that in a dialogue, the system needs to handle macro-strategies, we rather insist on the fact that even a single dialogue move is already most of the time the composition of several strategic decisions (see subsection 2.1).

Another group of studies [13, 14] went even a bit further than [9, 10], and proposed to mix a handcrafted SDS with an MDP-based SDS. The goal here is completely different: to optimise the dialogue capabilities of a conventional SDS. In substance, the idea consists in handcrafting almost totally the SDS so that it provides a small collection of options among which the MDP-based SDS picks the action to generate to the user. Our paper completely agrees with this objective. We endeavour to improve these previous works by proposing a novel framework that works at the internal decision level (which question, which words, which prosody, ...), instead of the dialogue move level. This enables us to drastically reduce the dimensionality of the learning and to consequently speed up the convergence.

Section 2 gives a detailed description of the problem and enumerates the constraints implied by the environment under consideration, which leads us to propose an extension of MDP called Module-Variable Decision Process (MVDP). Section 3 provides an algorithm for reinforcement learning in the MVDP framework: the Compliance-Based Reinforcement Learning (EBRL). Using a generalised problem, section 4 shows that the EBRL alleviates the "curse of dimensionality" and outperforms the benchmarked algorithms that could have been applied within the constraints defined in section 2. Finally, section 5 concludes the paper with the foreseen improvements.

2. Problem Description: Towards a New Model

2.1. Problem Constraints

Figure 1 shows a part of the design of an automated hotline dedicated to ADSL box installation. It reveals that the welcome message can be split up into three internal decisions: nature of the greeting/presentation, the help messages insertion or not and finally some choices of questions. These internal decisions can be made with a very localised view of the system state. Assuming all the enumerated information is available, the choice of the greeting message is dependent on the area of the call and the age of the caller and the choice of the help message is dependent basically on the SDS expertise of the caller. All these dependency definitions are part of the designer's work and are not discussed in this article. This paper aims at providing a model and an algorithm for learning in such an environment. If we used a classical MDP at the dialogue move level, for the figure 1 example, the state set would be the crossproduct of all the dependency variables and the action set would also be the crossproduct of all the alternatives. The dimensionality would soar, compromising the convergence without an inordinate amount of dialogues. In

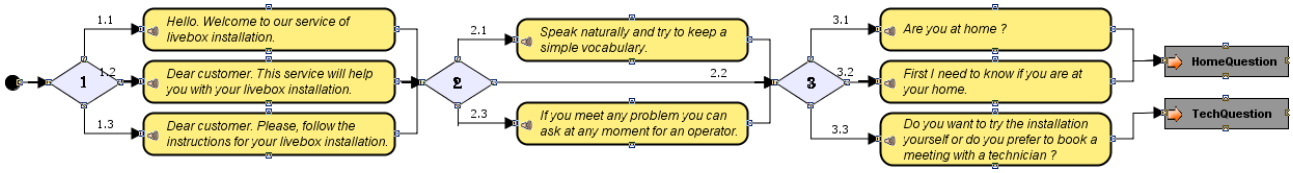


Figure 1: Example of an dialogue move constituted with three internal decisions.

order to solve this "curse of dimensionality", we consider the decision process at an internal decision level. Doing this breaks the Markovian assumptions. Indeed, the local state reached during the "help message insertion" internal decision is not solely dependent on the state-action pair of the decision concerning the greeting message. Figure 2 illustrates the complexity improvement provided by our approach.

As the decision process is non-markovian, in a given state, we cannot assume to be able to anticipate the next reached state. As a consequence, we cannot use any bootstrapping method [15], *i.e.* any method that updates expectation estimates on the basis of the estimates of the following states. The best reinforcement learning algorithms use bootstrapping, such as Dynamic Programming [16] or Temporal Difference Learning [17]. Solely Monte Carlo methods [15] are not bootstrapping. We propose an improved version of Monte Carlo method for reinforcement learning. But first, let us cope with the definition of a new decision process model that better reflects the problem: the Module-Variable Decision Process.

2.2. Module-Variable Decision Process

A *module* is a processing unit that can choose an internal *action* according to its local *variables*. This leads us to the definition of the Module-Variable Decision Process (MVDP) framework (M, V_M, A_M) where:

- M is the set of modules.
- $\forall m \in M$, V_m is the variable space used in module m.
- $\forall m \in M$, A_m is the set of possible actions for m.

In figure 1, the modules are the diamond-shaped boxes labelled 1, 2 and 3. For module 1, the possible internal actions are the 1.1, 1.2 and 1.3 transitions. The local states are not illustrated. In subsection 2.1, we told that the dependency of module 1 was the area of the call and the age of the caller. In such a design, $(Paris, 30) \in V_1$.

In a dialogue system, rewards are not synchronised with internal decisions. Although rewards are received at each dialogue turn, some dialogue moves are composed of several internal decisions and some other dialogue moves are completely handcrafted (*i.e.* without using the learning capabilities of the system). As a consequence, internal decisions and rewards must

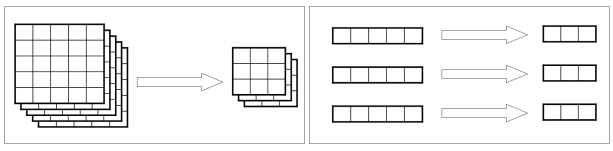


Figure 2: Comparison of complexity between learning $(y_1, y_2, y_3) = f(x_1, x_2, x_3)$ (on the left) and learning $y_1 = f_1(x_1)$, $y_2 = f_2(x_2)$ and $y_3 = f_3(x_3)$ (on the right).

be associated with an absolute timestamp in $\mathbb{R}$ (typically expressed in dialogue turns or in seconds).

Each module has a *policy*. The policy governs the choices that are made when the system needs to make a decision in the module. A policy is a function from the variable space into the action space $\pi_m : V_m \mapsto A_m$. In order to build its policy, the module may generate a *state-action value function* $Q_m : V_m \times A_m \mapsto \mathbb{R}$ which intends to predict the dialogue-term reward given the local state and the chosen action. The exploitation policy is the one that aims to maximise the dialogue-term reward expectations after a given decision d:

$$r_d = \sum_k \gamma^{t_k - t_d} R_k \tag{1}$$

Where $\gamma \in [0, 1]$ is the discount factor, used in order to encourage the shortest path to a dialogue success, t_d is the time when decision d has been made and $t_k > t_d$ is the time when reward R_k has been received.

3. Compliance-Based Reinforcement Learning

A (internal) *decision* d has the following features: the module m where d is made, the (local) state v reduced to the relevant information concerning d, chosen action a and timestamp t.

$$d = (m, v, a, t) \in M \times V_m \times A_m \times \mathbb{R} \tag{2}$$

An *episode* is the chain of decisions and rewards generated during a dialogue. The CBRL added value consists in avoiding to learn that an upfront decision is bad because the episode that tried it made further bad decisions. The CBRL rates an episode according to the system's current policy. Thus, the CBRL may consider that an experience for a decision is not meaningful because it regards its further decisions as irrelevant. Concretely, the algorithm checks if the current policy would trigger the same choices if it had to repeat a recorded episode. This rating $c_\pi(e) \in \mathbb{R}^-$ is called the *compliance* of an episode e with the policy π. It represents the deviation of the episode e decisions from the policy π. The local compliance $c_\pi(d_k)$ is the expected loss of performance implied by the chosen action a_k of decision d_k, compared to the π-optimal one. It is computed thanks to the $Q_{m_k}(v_k, a_k)$ state-action value functions:

$$c_\pi(d_k) = Q_{m_k}(v_k, a_k) - \sup_{a \in A_{m_k}} Q_{m_k}(v_k, a) \tag{3}$$

Then, the global compliance $c_\pi(e)$ of an episode $e = (d_1, ..., d_{|e|})$ after a decision d_0,is computed with a formula similar to that of the reward (equation (1)) is proposed:

$$c_\pi(e) = \sum_{k=1}^{|e|} \beta^{t_k - t_0} c_\pi(d_k) \tag{4}$$

Algorithm 1 Compliance-Based Reinforcement Learning

$E = \emptyset$, the set of episodes
$Q_m(v, a) = 0$, the state-action function
loop
  Generate a new episode e using $Q_m(v, a)$: $E \leftarrow E \cup \{e\}$
  for all $d_k \in e$ **do**
    Compute episode-term reward r_k {Equation 1}
  end for
  for all $e_i \in E$ **do**
    for all $d_{ij} = (m_{ij}, v_{ij}, a_{ij}, t_{ij}) \in e_i$ **do**
      Compute local compliance c_{ij} {Equation 3}
      Compute episode-term compliance {Equation 4}
    end for
  end for
  Compute state-action function $Q_m(v, a)$ {Equation 6}
end loop

Once corpus $C = \{m_k, v_k, a_k, r_k, c_k\}_{k\in[1,n]}$ is generated, the Monte Carlo method [15] is adapted to accept weighted average on the returns. Therefore, Q expectation is computed as follows:

$$C_{m,v,a} = \{r_k, c_k\} \text{ such that } \{m, v, a, r_k, c_k\} \in C \quad (5)$$

$$Q_m(v, a) = \frac{\sum_{\{r_k,c_k\}\in C_{m,v,a}} r_k e^{\tau c_k}}{\sum_{\{r_k,c_k\}\in C_{m,v,a}} e^{\tau c_k}} \quad (6)$$

Where τ is a parameter expressing the impact of the compliance on the weight for the averaging. The exponential reflects the intuition that several incompliances should multiply.

A CBRL implementation is provided by algorithm 1.

4. Experiment

4.1. The Generalisation Problem

In the SDS literature, most papers test their learning algorithms with user simulations. However, we prefer to make our study on a simple generalisation problem for several reasons: (1) the generalisation problem provides a universal evidence of the interest of the approach, the model and the algorithm, (2) user simulations are too simple to grant a tangible evaluation of a dialogue system and (3) dialogue applications can be very unalike and proving the effectiveness of a method on an application is barely a proof for its effectiveness on another application.

Therefore, we consider in this section a problem that generalises the example in figure 1. The system is given 27 alternative dialogue moves which correspond to the cross combinations of the three internal decision blocks proposing each 3 alternative local actions. Every dialogue turn, the same chain of decisions are made and the dialogue ends when:

- one of the three internal decision blocks is reached in a dialogue failure state. A negative reward -1 is received,
- all three internal decision blocks have reached their dialogue success states. A positive reward $+1$ is received.

Excluding the dialogue failure state (the dialogue ends as soon as they are accessed) but including the dialogue success state, each internal decision block has 5 possible states where it is required to take an internal decision. Therefore, the system has 124 non terminal global states ($5 \times 5 \times 5 = 125$ minus the dialogue success state). In order to reflect the dialogue domain variety, a strong Gaussian noise is applied to the system. Each episode starts in a random non terminal state among the 124.

The main goal of this problem is to study the gain obtained with the parallelisation of the problem thanks to the CBRL algorithm as illustrated in figure 2.

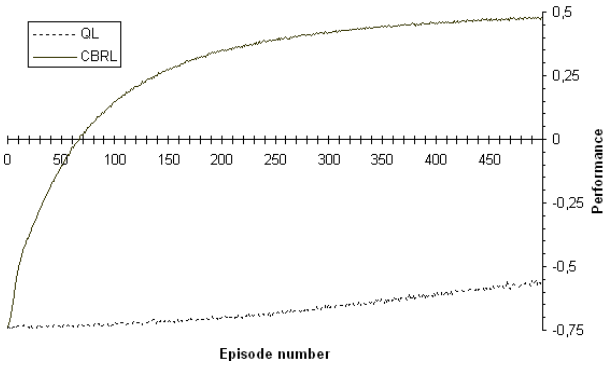


Figure 3: Comparison between the QL and CBRL algorithms.

4.2. Compared Algorithms

Three reinforcement learning methods are compared. For all of them, an ϵ-greedy exploration is used with $\epsilon = 0.5 \times 0.998^{|E|}$. The parameters used for each method were set in order to optimise each algorithm's performance:

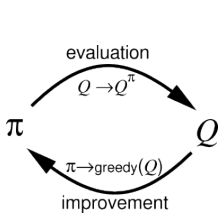
- Compliance-Based Reinforcement Learning (CBRL): algorithm 1, with $\tau \neq 0$. The three internal decision blocks constitutes the M module set. $\forall m \in M, V_m$ comprises the 5 non-terminal states and A_m the 3 alternative local actions.
- Q-Learning (QL): Q-Learning on the MDP of 124 states and 27 actions, as described in subsection 4.1.
- Monte Carlo (MC): Monte Carlo algorithm [15] on the factored states, which is equivalent to CBRL with $\tau = 0$.

4.3. Comparison with the Q-Learning Algorithm

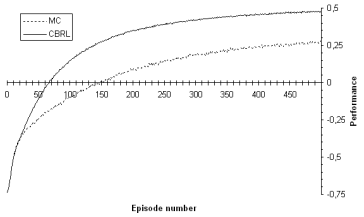
As shown in figure 2, the complexity of the system is reduced by $\frac{124\times27}{3\times(5\times3)} \approx 74$. The CBRL algorithm takes great advantage of the huge space reduction brought with the MVDP framework and figure 3 reveals that after making 500 episodes, which means basically 500 dialogue failures, QL algorithm only succeeded in surviving one turn and a half longer on average, while CBRL has usually reached the optimal policy. The QL algorithm actually requires 10000 episodes to reach the CBRL performance after 500 episodes.

4.4. Impact of τ

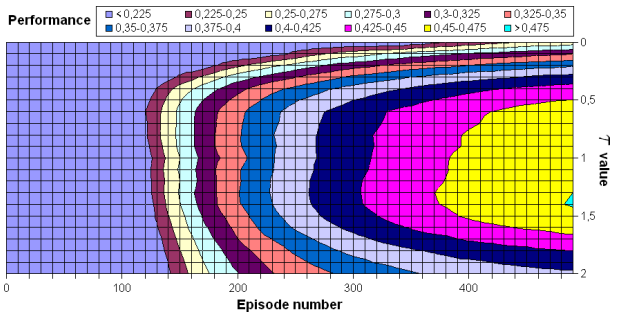
Figure 4(a) shows the Monte Carlo control loop. Our algorithm improves the evaluation process by evaluating as well the past episodes quality for learning. As shown in figure 4(b), CBRL with the optimal τ value: 1.4 performs significantly better than MC ($\tau = 0$). Figure 4(c) shows that for moderate τ values, CBRL reliably outperforms MC which does not guarantee to find the optimal policy. The fact that the compliance calculation is based on a sub-optimal policy in the early episodes explains the loss of performance when τ grows above 1.5.



(a) Monte Carlo control loop. See [15].



(b) Comparison between the MC and CBRL algorithms.



(c) Gray scale view of the impact of τ value on the learning speed.

Figure 4: Ratio impact on the learning performance.

5. Conclusion

This paper investigates reinforcement learning for SDS. It focuses more precisely on an expertise/learning hybridisation of the system. This approach leads to propose a novel decision process model: the Module-Variable Decision Process. Under this model, constraints prevent from using classical bootstrapping methods such as TD Learning. Then, a new algorithm is proposed: the Compliance-Based Reinforcement Learning. Eventually, an evaluation on a generalised problem shows that our algorithm outperforms all the well-known reinforcement learning algorithms.

The scope of this contribution is enhanced in three directions. First, figure 1 has for purpose to illustrate in an automaton design how internal decisions can combine and become a single dialogue move. But the MVDP framework does not rely on the automaton structure: the system gathers internal decisions independently of the internal structure, collects rewards, builds a corpus and uses it to learn a best practice policy. Second, CBRL can easily be adapted to accept any supervised learning algorithm instead of the weighted average computed with formula 6. Third, the application domain of CBRL can be extended to any handcrafted system requiring learning capabilities such as robotics or artificial intelligence design for video games.

As for perspectives, the algorithm has been integrated in a dialogue application for internet box installation help for applicative tests. Concerning the theoretical aspects, we plan to study the correspondence between the compliance and the average weights and to introduce uncertainty handling to the CBRL algorithm.

6. Acknowledgements

This research has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement number 216594 (CLASSIC project: www.classic-project.org).

7. References

[1] E. Levin, R. Pieraccini, and W. Eckert, "Using markov decision process for learning dialogue strategies," in *Proceedings of ICASSP1998*, 1998.

[2] S. Singh, M. Kearns, D. Litman, and M. Walker, "Reinforcement learning for spoken dialogue systems," 1999.

[3] S. J. Young, "Probabilistic methods in spoken-dialogue systems," in *Philosophical Transactions of the Royal Society*, ser. Royal Society of London Philosophical Transactions Series A, vol. 358, Apr. 2000, pp. 1389–1402.

[4] O. Lemon and O. Pietquin, "Machine learning for spoken dialogue systems," in *Proceedings of the European Conference on Speech Communication and Technologies (Interspeech'07)*, August 2007, pp. 2685–2688.

[5] T. Paek, "Reinforcement learning for spoken dialogue systems: Comparing strengths and weaknesses for practical deployment," 2006.

[6] T. Paek and R. Pieraccini, "Automating spoken dialogue management design using machine learning: An industry perspective," *Speech Communication*, vol. 50, 2008.

[7] J. Williams and S. Young, "Scaling up pomdps for dialog management: The summary pomdp method," in *Automatic Speech Recognition and Understanding, IEEE*, 2005, pp. 177–182.

[8] S. Young, "Using pomdps for dialog management," in *Spoken Language Technology Workshop, IEEE*, 2006.

[9] S. Cuayáhuitl, H.and Renals, O. Lemon, and H. Shimodaira, "Reinforcement learning of dialogue strategies with hierarchical abstract machines," in *Proceedings of IEEE/ACL Workshop on Spoken Language Technology (SLT)*, December 2006.

[10] H. Cuayáhuitl, S. Renals, O. Lemon, and H. Shimodaira, "Hierarchical dialogue optimization using semi-markov decision processes," in *Proceedings of the European Conference on Speech Communication and Technologies (Interspeech'07)*, August 2007.

[11] T. G. Dietterich, "The maxq method for hierarchical reinforcement learning," in *In Proceedings of the Fifteenth International Conference on Machine Learning*. Morgan Kaufmann, 1998, pp. 118–126.

[12] R. Parr and S. Russell, "Reinforcement learning with hierarchies of machines," in *Advances in Neural Information Processing Systems*, M. I. Jordan, M. J. Kearns, and S. A. Solla, Eds., vol. 10. The MIT Press, 1997.

[13] S. Singh, D. Litman, M. Kearns, and M. Walker, "Optimizing Dialogue Management with Reinforcement Learning: Experiments with the NJFun System." *Journal of Artificial Intelligence Research*, vol. 16, pp. 105–133, 2002.

[14] J. Williams, "The best of both worlds: Unifying conventional dialog systems and pomdps," in *International Conference on Speech and Language Processing*, 2008.

[15] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction (Adaptive Computation and Machine Learning)*. The MIT Press, March 1998.

[16] R. Bellman, *Dynamic Programming*. Princetown University Press, 1957.

[17] R. S. Sutton, "Learning to predict by the methods of temporal differences," in *Machine Learning*, 1988, pp. 9–44.