Naïve Reinforcement Learning As a Guide for Path Discovery

Elisha T. O. Opiyo, SCI, University of Nairobi, Nairobi, Kenya, opiyo@uonbi.ac.ke
Eric Ayienga, SCI, University of Nairobi, Nairobi, Kenya, ayienga@uonbi.ac.ke
William Okello-Odongo, SCI, University of Nairobi, Nairobi, Kenya, opiyo@uonbi.ac.ke
Bernard Manderick, COMO, VUB, Brussels, Belgium, bmanderi@vub.ac.be
Ann Nowé, COMO, VUB, Brussels, Belgium, ann.nowe@vub.ac.be

Abstract

Usually, a problem arises given that one is at a point – call it *A*, and needs to reach a destination, let's say point *G* through several other intermediate points in a large space. In the absence of a viable tool to use in getting to the destination, it becomes very hard to get to it especially if the search space is large. Sometimes one may resort to trial and error and this method is normally very inefficient in a large space. The exhaustive search is also inefficient in a large space, especially where the resources are limited. Reinforcement learning can play the role that provides the necessary guide for the search process to successfully discover the destination. We demonstrate, using a concrete example of a map representation, that by only indicating that the action taken is 'right' (1) or 'wrong'(-1) the search process proceeds successfully to the destination. The 'right' and 'wrong' indications come from the environment. This is naïve reinforcement learning since it neither takes into account the cumulative reinforcement values nor insists on discovering a policy. The contribution of this technique is that there are little overheads and resource inputs, while the search problem is successfully solved. The challenge with this method is the need to model the environment.

Keywords: Naïve Reinforcement Learning, Machine Learning, Learning Agent

Introduction

In this paper we consider the problem that arises when a path from a given point A to a goal point G is to be found. Limited knowledge is assumed in which the environment can only supply a limited guide. The environment can only tell whether the direction taken is right or wrong. Such a problem can be solved using search techniques in which the search space is examined, Russel & Novig (2003). The difficulty is that when the search space is large it is hard to apply the usual search methods such as systematic exhaustive search. The required time may, also, be too long and impractical to get. It is, further, more challenging when there is limited knowledge of the space such that it is not easy to enumerate the search space elements. In case of the path finding the search space would consist of a set of all possible paths. The issue is that the path finder has no idea about the future intermediate points, nor where to proceed. The path seeker knows the destination, G but not how to get to it.

The reinforcement learning approach is adopted in solving the problem of finding a path when the available information is limited. Reinforcement learning is a machine learning technique in which the learner has a goal with no knowledge of how to attain it but takes some actions and gets some reward from the environment. The learner interprets this reward and seeks the actions that will maximize the cumulative reward, as observed by Sutton & Barto (1998); Kaelbling & Littman (1996).

The naïve reinforcement learning is a process in which the learner has a goal without any knowledge of how to get to it but gets the reward from the environment. The learner interprets the reward and avoids some actions and prefers others. The learner remembers the incremental progress made and actions taken at local points or states. The naïve reinforcement learning is demonstrated using a learner that is situated at a geographical point A, and seeks to reach another geographical point G. The learner, at some local point, chooses a direction and gets to a destination. The environment indicates to the learner that she is 'right' (1) or 'wrong'(-1). The search process then proceeds successfully to the destination provided the learner is given enough chances to choose directions. This naïve reinforcement learning does not take into account the cumulative reinforcement values nor insist on discovering a policy. A policy is guide at every state that tells the learner how to act. The advantage of this technique is that there are fewer overheads and the required resource inputs, while at the same time the search problem is successfully solved.

The rest of this paper considers the reinforcement learning model, the naïve reinforcement learning model, the experiments, results, the discussion and the conclusion.

The Reinforcement Learning Model and the Related Work

Learning in human beings is the process that results in the changes of attitudes and behaviors. These changes occur due to the acquisition of knowledge, skills and values. Learning in machines, is handled under machine learning that is a subfield of artificial intelligence, Mitchel (2005). In machine learning the algorithms are developed to model the learning processes. The learning processes in machines are categorized in some of the groups that include the supervised, unsupervised, semi-supervised and reinforcement learning processes. In supervised learning the learner is required to learn input-output pairs that are used as examples to enable the learner to subsequently perform the matching. In unsupervised learning the learner has only a set of inputs; the learner must discover how to match the inputs to the outputs. In semi-supervised learning, the learner has some examples with complete input-output pairs and some examples without the pairs. In reinforcement learning the learner has a goal with no knowledge of how to attain it but takes some actions and gets some reward from the environment. The learner needs to discover a policy that guides actions in given states of the environment. Reinforcement learning is a machine learning technique in which the learner has a goal with no knowledge of how to attain it but takes some actions and gets some reward from the environment. The learner interprets this reward and seeks the actions that will maximize the cumulative reward, Sutton & Barto (1998).

Consider the learner as an agent that is in some environment. The agent interacts with her environment so that a goal is achieved. According to Russel & Novig (2003), agents are objects in

the environment that perceive and react to states in the environment. Examples of agents include anything for which an environment can be specified, and that acts and reacts such as humans, animals, ants, some Internet software or computational processes in operating systems context.

A model for reinforcement learning consists of the agent that is taken to be the learner. The agent interacts with the environment. The agent selects an action and the environment responds to the action by moving to a different state and gives a reward to the agent. The interactions of the agent and the environment occur in a sequence of discrete time steps, Sutton & Barto (1998); Kaelbling & Littman (1996). In the model there is a set of environment states, a set of actions that the agent can take, a set of rules of changing between the states, a set of rules that determine the reward of the transitions, and a set of rules that describe what the agent perceives.

Reinforcement learning has many applications. It has been used to solve problems in many areas including robot control, elevator scheduling, telecommunications, backgammon, chess, jobshop scheduling, project scheduling, robotic soccer, resource allocation, Sutton & Barto (1998); Wei and Dietterich (1995); Arai et al. (2000); Wauters et al. (2010); Kaelbling & Littman (1996); Maes et al. (2007); Maes (2003), Galstyan, et al. (2005).

The Naïve Reinforcement Learning Model

In the naïve reinforcement model, the agent is the learner. The agent has a locality, or her current state, in the environment. The agent has a set of allowed actions within her locality that is in her current state. The agent chooses one of the actions within the state at random. If there is only one allowed action, then it is the one that is taken. After the action, a new state occurs and the reward for the action is obtained. The agent remembers her actions and previous states and avoids the actions that resulted in poor reinforcement. The agent also remembers her progress towards the goal.

For the path finding example, an agent acts by moving in some directions that are possible at the given points or states. The agent can move to the left, the right, straight or reverse. Every state has one or more allowed moves. A terminus, for example, has only one allowed move that is ‘reverse’. In every state and for every allowed move the environment has some reinforcement. The reinforcement takes only two forms, ‘right’ or ‘wrong’. The right move is awarded 1 and the wrong move awarded -1. The agent knows what these rewarded mean. The agent drops all the immediate destinations that are rewarded -1 and maintains the immediate destinations that are rewarded 1. The agent keeps on trying the various actions at the local states. See figure 1 for the map that is used in the example. The agent needs to find a path from A to G.

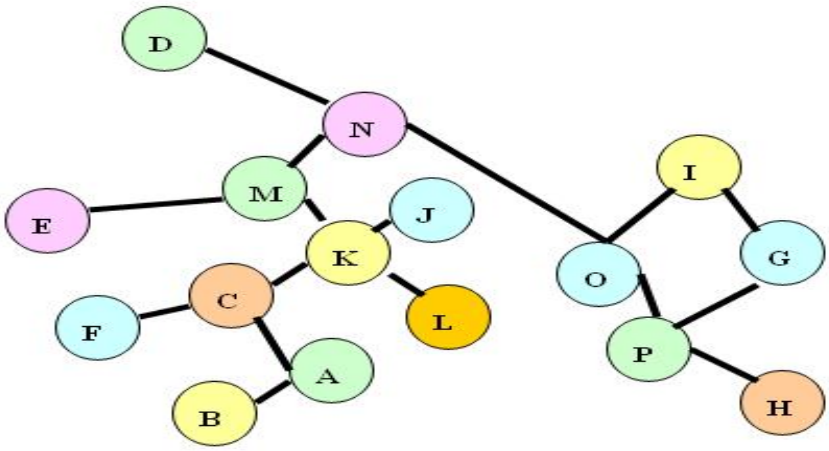

Figure 1. The Map showing the nodes (states)

The Naïve Reinforcement Learner's Contextual Architecture

The naïve reinforcement learner is implemented using the agent-based approach. The learner is the agent. The agent is an object that can perceive and react in the various environmental states, Russel & Novig (2003). The agent acts by choosing a direction and moving in the direction chosen. In this way the agent can turn left and proceed, turn right and proceed, move straight and proceed or reverse and return to earlier place. The initial place the agent is in is A and the agent wants to be in place G since it is the agent's goal. When the agent acts, the place where the agent is situated changes. The place changes from the current one to the next logical place according to the links of the places. These are depicted on Figure 1. For example, if the agent is situated at A, the agent can move straight or turn right. Suppose the agent chooses to move straight. The agent acts 'move straight'. Then the next logical place is B. Since B is a terminus, the only option available is to reverse; this means that at B this action takes the agent back to A.

The agent can do several things; she has a basic knowledge that enables her to recognize the various directions, the place where she is and the allowed actions at the place where she is. She remembers the minimal path that leads to the right direction. She recognizes when the goal destination, G, is reached. She can show the right minimal path at any time. She can show the state of affairs at any time such as action taken and reward obtained. She recognizes the allowed actions at any place. She can take action. She can get and interpret the reinforcement value.

The environment is modeled using the basic elements that are related to each local place or decision point. For each local place such as K, see figure 1 for the places labeled A .. O, one or more tuples, is maintained. A basic tuple is:- <place, direction, destination, reinforcement score>. In this case the place is any one of A .. O, the direction is any one of left, right, straight, or reverse. Destination is any one of A .. O, and reinforcement score is right (-1) and wrong (1). Enough tuples are kept for each place according to the possible allowed actions at that place. At any one point when the agent takes an action at one place, the reinforcement score can be given using the appropriate tuple. The contextual architecture is as shown in Figure 2.

The overall agent control loop consists only of the agent taking an action each time. This activity of the agent is only constrained by the allowed number of chances or the discovery of the path.

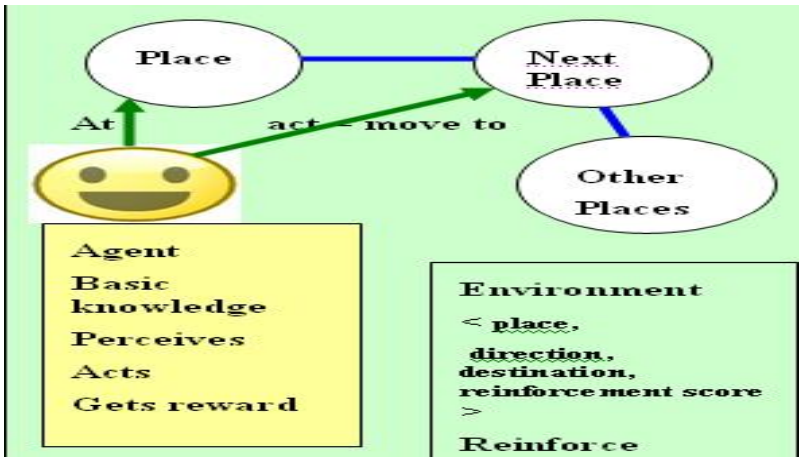

Figure 2. Basic Contextual Architecture

The Experiment, Results and Discussion

The naïve reinforcement learner was implemented using an agent approach. The learner was implemented as an agent through an object oriented development environment. A base agent with the functionalities was implemented and then instantiated. The functionalities are the capabilities of the agent such as the ability to recognize where she is, the available options and take action. Out of possible actions at any place, the agent selects options at random and relies on the reinforcement for subsequent actions. The main objective of the experiment is to find out the empirical efficiency of the naïve reinforcement learning process. The learner is given a number of chances to try to find a path from A to G. The learner or the agent may find the path or not. Out of the number of chances that are given it was noted at which trial stage the learner discovered the path. The learner operated in a context with 16 places or nodes as shown on figure 1. Every time the learner starts at node or place A and gets a number of chances to find a path to G. The chances given were 10, 15, 20, 25, 30, 35, 40, 45, 50, 55, 60 and 100. For every number of chances, say for example 15 chances, the experiment was repeated five (5) times and in each replication the following were noted: time taken in seconds, if the path was found and the number of trials before the path was found. In every case the mean value was computed and rounded. Efficiency of the search effort was computed to give an indication of the chances that were given and the number of chances used before the correct path is found.

The results that ware obtained are shown on Figures 3, 4 and 5.

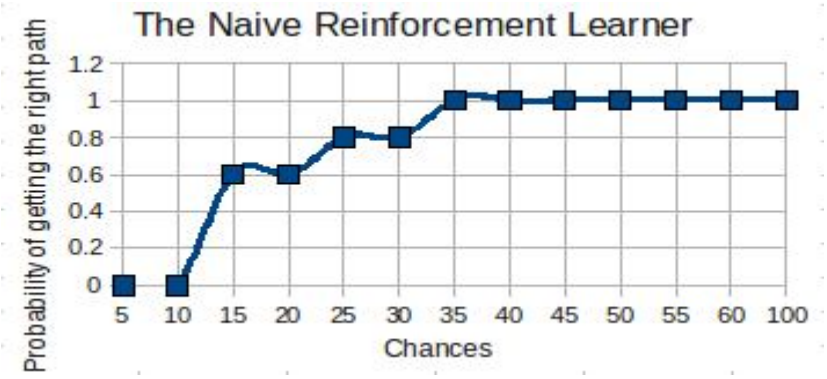

Figure 3. Getting the right path versus the chances given

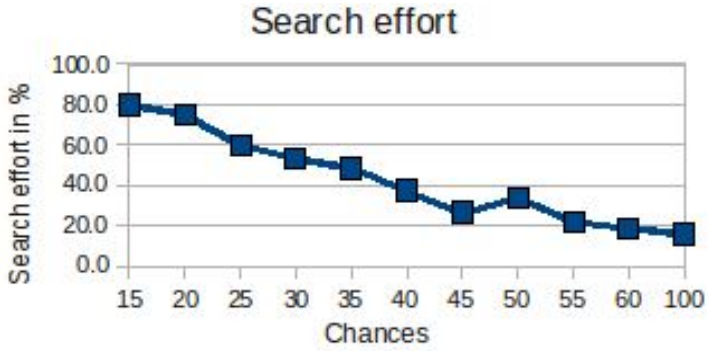

Figure 4. The search effort

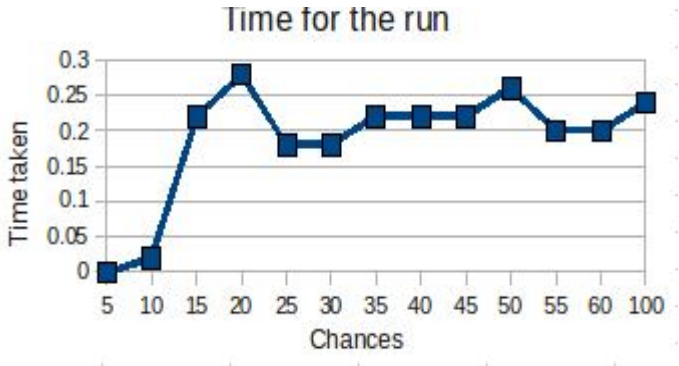

Figure 5. The time it takes for the run

We now discuss the results. In Figure 3, the results for getting the right path versus the chances that are given are shown. For all the replications when the chances were 5 and 10, no correct path was found. This can be explained by the fact that there were 16 places or nodes and the learner guesses most of the time. The chance of getting directions correctly and successively is low. For chances 15 to 30 the correct path is found most of the time with a probability ranging from 0.6 to

0.8. For the chances ranging from 35 and above, the correct path was found, every time. From figure 3 it is strongly indicated that beyond some number of chances, the correct path is certainly found. In this example the number is 35 changes. This works out to approximately 2 trials for every node or place. It may also be noted that under the same example, the possibility space is 20736 for exhaustive search. At 35 trials for a guaranteed correct path, the saving on the search effort seems significant. Figure 4 shows the search effort. This is a measure of the number of trials taken against the total number of changes that are given. In all cases the after 15 chances and beyond, the learner did not exhaust the number of chances given. The learner took fewer numbers of trials to find the correct path compared to the total number of chances given. The implication of this is that the naïve reinforcement learning process can also be effective. Figure 5 shows the time it takes to end the run. A run is the time it takes to exhaust the chances or find the correct path. Either way the search process stops, when the chances are exhausted or the correct path is found. The run time is generally the same, varying from 0.03 seconds to 0.28 seconds. The most notable observation is that the number of chances does not affect the length of the run. This is explained by the fact that the run stops when the correct path is found. These results, though very limited in variation, indicate that the naïve reinforcement learning process can be useful. A typical use may be in the embedded control for robots, where a robot may have a range of actions and some environmental indicators that show him that the actions are right or wrong. There are some notable limitations of the naïve reinforcement learner. First, like reinforcement learning, it needs the environmental feedback without which it cannot work. Secondly, the local environmental elements or tuples may be too many if the learner has to cope with varying start places and goal places. The naïve reinforcement learner however, both explores and exploits in the sense of Sutton & Barto (1998). She explores since the actions are selected at random and exploits since the reinforcement is useful information that forms the basis for avoiding some places.

The Conclusion

This paper has considered the problem of finding a path from one place to another using the naïve reinforcement learning method. The naïve reinforcement model, unlike the usual reinforcement model, only considers providing the learner with a right and wrong indicators on their actions from the environment. This alone has been found to be sufficient to guide the learner in discovering the correct path. The empirical results with a single context show that out of a total of 20736 possible trials needed for the exhaustive search only 35 trials are needed to guarantee finding the correct path. Whereas this is not indicative of the actual savings ratio in other contexts that may be different, substantial savings on the search effort is expected where the naïve reinforcement method is used to guide in path finding where local knowledge and information is very limited. The most important strength of the method is the little overhead that it has, and the ease of implementation. The usual reinforcement model has been extensively relaxed yet the learner still manages to find a correct path. Its drawback is the need for environment and the many knowledge elements that may be needed for a generalized application.

Acknowledgment

We are grateful to the University of Nairobi and Free University of Brussels for the various types of support that we have received for our research work.

References

Arai, S., K. Sycara ,T. R. Payne (2000). Multi-agent Reinforcement learning for Planning and Scheduling Multiple Goals. *Proceedings of The International Conference on Multi-Agent Systems (ICMAS2000)*, pp. 359-360.

Galstyan, Aram, Karl Czajkowski, Kristina Lerman (2005). Resource Allocation in the Grid with Learning Agents. *Journal of Grid Computing*, vol. 3, pp 91–100.

Kaelbling, Leslie Pack & Michael L. Littman (1996). Reinforcement Learning*: A* Survey. *In Journal of Artificial Intelligence Research*, Vol. 4, pp 237-285.

Maes Sam, Karl Tuyls, Bernard Manderick (2007). *Reinforcement Learning in Large state spaces: simulated robotic soccer as testbed*. Retrieved from http://citeseerx.ist.psu.edu/viewdoc/ summary? doi=?doi=10.1.1.4.1998 ; Quelle http://como.vub.ac.be/Members/karl/rl-robo02.ps, last visited July, 2011.

Maes, S. (2003). Reinforcement Learning in Large State Spaces-simulated Robotic Soccer as Testbed. *Robocup 2002 Robot Soccer World Cup VI*, 2752, p 319-326.

Mitchel ,Tom (2005). *Machine Learning*. McGraw Hill Publishers.

Russell, Stuart J., Peter Norvig (2003). *Artificial Intelligence: A Modern Approach*, (2nd ed.), Upper Saddle River, New Jersey: Prentice Hall.

Sutton, Richard S. *and* Andrew *G.* Barto (1998). *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA.

Wauters T., K. Verbeeck, G. Vanden Berghe, P. De Causmaecker (2010). Learning Agents for the Multi-mode Project Scheduling Problem. *Journal of the Operational Research Society*, Vol. 62, pp 281–290, 2010.

Wei Zhang and Thomas G. Dietterich (1995). A Reinforcement Learning Approach to Job-shop Scheduling. *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI 1995*), Vol.2, Morgan Kauffman Publishers.