MDQL A Reinforcement Learning Approach for the Solution of Multiple Objective Optimization Problems

Carlos E. Mariano[1] and Eduardo Morales[2]

[1] Instituto Mexicano de Tecnología del Agua, Paseo Cuauhnáhuac 8532, Jiutepec, Morelos, 62550, México, cmariano@tlaloc.imta.mx
[2] ITESM – Campus Morelos, Paseo de la Reforma 182 – A, Temixco, Morelos, 62589, México, emorales@campus.mor.itesm.mx

Abstract. We present the foundations of a new algorithm based in Reinforcement Learning (RL) for the solution of Multiple Objective Optimization Problems (MOOP). MDQL considers the decision variable space partitioned in a fixed number of states, each state there with a set of actions associated and used to traverse the decision variable space. A family of n agents for each objective function is also considered; every agent in a family learns the the best way to reach an optimal solution for its correponding objective function. Solutions proposed for one objective are negotiated with solutions for the rest of the objectives, obtaining trade off solutions, which are evaluated in the Pareto sense. Pareto solutions are keeped in memory, and are used by the agents in future episodes (an episode is a finite set of steps or agent transitions). The Pareto set is updated when an episode finalise. Results obtained in the solution of two test bed problems are also presented.

1 Multiobjective Distributed Q-Learning

In RL an autonomous agent, starting in an initial state, learns the best way to reach a previously defined. In other words, the agent learns an optimal policy π for the set, $\mathcal{S}$, of states in an unknown environment, the learned policy lets know which is the best action, $a \in \mathcal{A}_{s_t}$, to obtain the highest reward in the long run. An agent in an state selects one of the available actions considering the state-action value functions estimations, $Q(s,a)$, that takes the agent to another state, s'. Once in s' the state-action value function estimation is updated considering the reward obtained, $r \in \Re$, and applying: $Q(s,a) \leftarrow Q(s,a) + \alpha\left[r + \gamma max_{a'} Q(s',a') - Q(s,a)\right]$. To apply RL to the solution of MOOP, the decision variables space is partitioned to obtain $\mathcal{S}$, discretizing the range of the decision variables in a finite set of partitions, depending on the desired precision, see Figure 1 (left). So, for one decision variable the number of states will be equal to the number of partitions, for two or more, to the number of partition intersections (the product of the number of partitions for each variable), see Figure 1 (right). Furthermore, a set of actions is considered in every state,

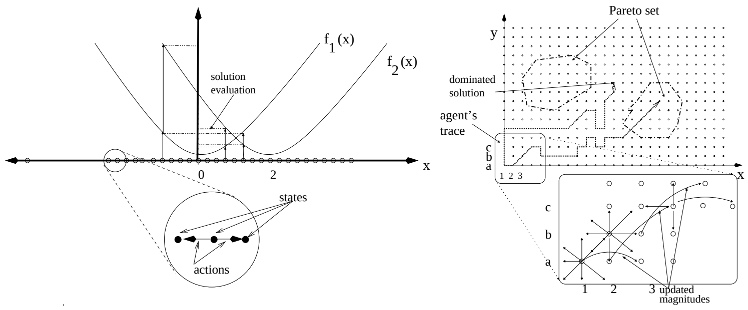

Fig. 1. a two objective functions problem with one decision variable (left) and a two decision variables (right)

representing the mechanism used by the agents to traverse the decision variable space. The configuration used in MDQL, considers for one dimension, two move directions (actions) (see Figure 1(left), for two, eigth (see Figure 1(right)), and for three, 26 see [1]; actions can be settled arbitrarilly, but they must be enough to let the agents explore the decision variable space.

A family of agents is considered for each objective function, all with the same number of agents. An "map" of the environment made with a copy of the state-action value functions, $Q_c(s, a)$, is available for the agents in each family. Every time an agent performs a transition, using the "map" information, it updates the "map" value function considering the following rule: $Q_c(s, a) \leftarrow Q_c(s, a) + \alpha[\gamma max_{a'} Q_c(s', a') - Q_c(s, a)]$, favoring those actions selected most often. Because final states (solutions) are unknown, a solution is obtained when: a) agents complete an episode, or b) agent reach an state previously defined as a solution. In this way, when agents in all families obtain a solution, a negotiation procedure initiates. The negotiation mechanism used compares the solutions (parameters for the decision variables) obtained by the kth agent in all families. If one of the compared solutions dominates the rest, it is selected, otherwise random selection is made. It is important to note that the negotiation criterion is not unique, other forms of negotiations has been tested, for example, *domination count*, where the solutions with the higher domination count are selected. Solutions clasified as non dominated are used to update the Pareto set, finally all solutions in Pareto set are rewarded and the state-action value functions used in its construction updated using the actualization rule for Q-Learning, the map is erased and a new episode is initiated.

An important issue in MDQL is that rewards, r, represent the magnitude, dynamically adjusted, for the moving vectors considered as actions. So, an state-action pair receive the double of the reward received in the past (rewards are initialized equal to the distance between partitions) if it contributes in the construction of a Pareto solution (see bottom of Figure 1 (right)). This is in order to help agents to reach Pareto solutions in fewer steps.

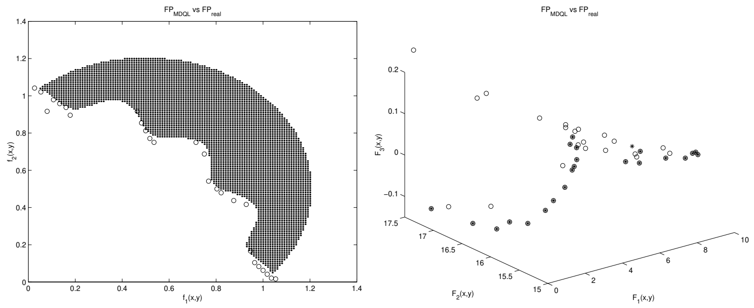

Fig. 2. $\mathcal{PF}_{MDQL}$ for MOOP1 (left) and MOOP2 (right) $\circ$, compared with $\mathcal{PF}_{real}$, $*$.

2 Experiments

Results for two benchmark problems are incuded in Figure 2.

$$\begin{aligned}
&\text{MOOP1:}\\
&f_1(x,y) = 0.5(x^2+y^2)+\sin(x^2+y^2),\\
&f_2(x,y) = \frac{(3x-2y+4)^2}{8}+\frac{(x-y+1)^2}{27}+15,\\
&f_3(x,y) = \frac{1}{(x^2+y^2+1)}-1.1e^{(-x^2-y^2)},\\
&\textit{subject to:}\quad -30 \le x,y \le 30
\end{aligned}$$

$$\begin{aligned}
&\text{MOOP2:}\\
&f_1(x,y) = x,\\
&f_2(x,y) = y,\\
&\textit{subject to:}\quad 0 < x,y < \pi\\
&\qquad 0 \ge -(x^2)-(y^2)+1+0.1\cos(16\arctan\frac{x}{y}),\\
&\qquad \frac{1}{2} \ge (x-\frac{1}{2})^2+(y-\frac{1}{2})^2.
\end{aligned}$$

References

Mariano, Carlos., Morales, Eduardo. A New Approach for the Solution of Multiple Objective Optimization Problems Based on Reinforcement Learning, In O. Cairo et al., eds. *Lecture Notes in Artificial Intelligence*, Vol 1793, Springer-Verlang, 2000, pp. 212-223.