Decomposing a Value Function into a Sum of Neural Networks

Nicolas Langlois
Rémi Coulom REMI.COULOM@UNIV-LILLE3.FR
Philippe Preux PHILIPPE.PREUX@UNIV-LILLE3.FR
GRAPPA, Université Charles de Gaulle – Lille 3, France

Abstract

When using tile coding as a function approximator for high-dimensional reinforcement-learning problems, it is common to decompose the value function into a sum of low-dimensional functions. In this paper, we investigate how such a decomposition may also help feedforward neural networks. Empirical results are presented on the problem of controlling artificial articulated swimmers with up to six joints. They show that decomposition improves learning speed.

1. Introduction

When solving reinforcement-learning problems with a high-dimensional state, approximating the value function with discretization-based methods suffers from the curse of dimensionality. Since the cost of discretization is exponential with the dimension, it becomes impractical to obtain a high resolution in every dimension.

An usual method to overcome this difficulty consists in decomposing the state space into a superposition of tilings, each of which having a high resolution for a few dimensions only, like in (Sutton, 1996). Another approach consists in using feedforward neural networks, since they have better theoretical properties to deal with high-dimensional input (Coulom, 2002).

Although they can work without it, one may wonder whether feedforward neural networks would also benefit from a decomposition into a sum of low-dimensional networks. This paper presents experiments on the swimmer problem that were run to explore this question. They indicate that decomposition brings a significant speed-up.

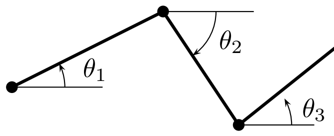

Figure 1. A 3-segment swimmer

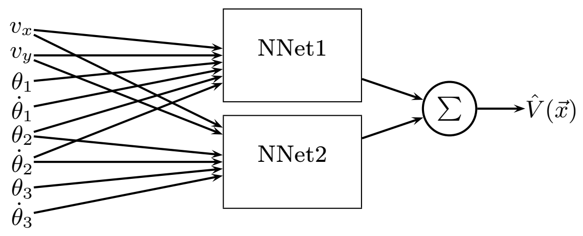

Figure 2. Decomposition for a 3-segment swimmer

2. Experimental Setting

The state $\vec{x}$ of an n-segment swimmer (Figure 1) is made of $2n + 2$ variables: $\vec{x} = (v_x, v_y, \theta_1, \dot\theta_1, \ldots, \dot\theta_n)$. v_x and v_y are the time derivatives of the two Euclidian coordinates of the center of mass. θ_n is the angle of the n-th segment, and $\dot\theta_n$ its derivative with respect to time. The reward is v_x. See (Coulom, 2002) for complete specifications of the problem.

Among all the possible decompositions, we have chosen to use one neural network for each joint (Figure 2). The input of each of these networks are the velocity of the center of mass, the angles of the two segments of the joint, and their derivatives with respect to time. Other decompositions have been tried, but this one provided the best results.

The learning algorithm was continuous TD(λ). Parameters where $s_\gamma = 0.2$, $s_\lambda = 0.8$, $\delta t = 0.02$s, $T = 25$s. T is episode length. See (Coulom, 2002)

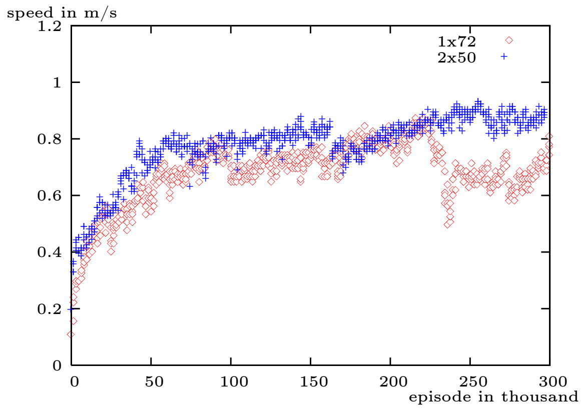

Figure 3. 3-segment Swimmer

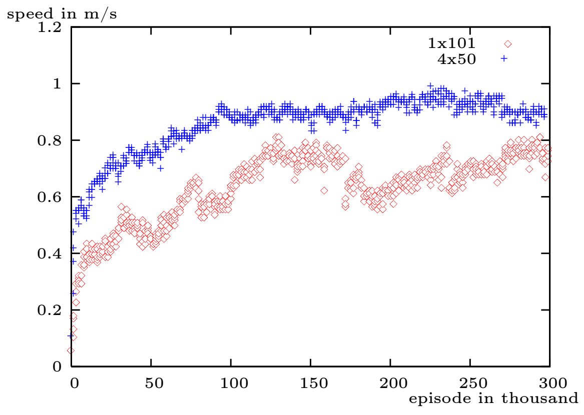

Figure 4. 5-segment Swimmer

for a precise description of the meanings of these parameters. Experiments were run, with and without decomposition, on 3-segment, 5-segment, and 7-segment swimmers, for 300,000 episodes each. In order to filter out the effect of random initialization of weights, every learning experiment was repeated ten times, and results were averaged. The sizes of neural networks were determined so that the decomposed and monolithic architectures have the same number of weights. This way, they used almost identical amounts of memory and processing time.

3. Results and Discussion

Results are plotted on Figures 3–5. These plots clearly indicate the superior performance of the decomposed architecture. This superiority is very small for the 3-segment swimmer, but it increases a lot for 5- and 7-segment swimmers.

In order to close the gap in performance between the monolithic and decomposed approaches, it is possible to increase the size of the monlithic network, while keeping the size of the decomposed network constant. This is done at the cost of requiring more CPU time and memory. Table 1 summarizes this cost for the three kinds of swimmers. Again, it clearly indicates the superior performance of the decomposed architecture.

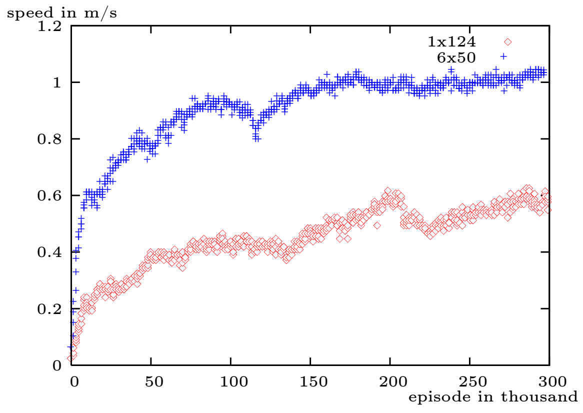

Figure 5. 7-segment Swimmer

	3 segments	5 segments	7 segments
Monolithic	17.4h	78.9h	≈ 20 days
Decomposed	10,4h	22.6h	34.9h

Table 1. Average calculation time

4. Conclusion

Experiments presented in this paper have demonstrated that decomposing a value function into a sum of neural networks is more efficient than using one single neural network on the swimmer task.

The decomposition used in this paper was obtained by trial and error, and is very specific to the particular problem that was studied. A potential direction for future research would be to study methods to obtain such a decomposition in a more automated manner.

References

Coulom, R. (2002). *Reinforcement learning using neural networks, with applications to motor control.* Doctoral dissertation, Institut National Polytechnique de Grenoble.

Sutton, R. S. (1996). Generalization in reinforcement learning: Successful examples using sparse coarse coding. *Advances in Neural Information Processing Systems 8* (pp. 1038–1044). MIT Press.