Pole-Balancing with different Evolved Neurocontrollers *

Frank Pasemann

Max-Planck-Institute for Mathematics in the Sciences,
D-04103 Leipzig, Germany
email: pasemann@mis.mpg.de

Abstract. The paper presents various evolved neurocontrollers for the pole-balancing problem with good benchmark performance. They are small neural networks with recurrent connectivity. The applied evolutionary algorithm, which is not based on genetic algorithms, was designed to evolve neural networks with arbitrary connectivity. It uses no quantization of inputs, outputs or internal parameters, and sets no constraints on the number of neurons. Network topology and parameters like weights and bias terms are developed simultaneously.

1 Introduction

The inverted pendulum is one of the simplest inherently unstable systems and provides a wide class of control problems. Pole-balancing usually serves as a benchmark problem for trainable controllers [9]. The task is given as follows: An inverted pendulum is mounted on a cart, which can move on a finite interval. A controller, providing a finite force to the cart, then has to balance the pole by avoiding the interval boundaries, and has to center the cart simultaneously. Besides conventional control techniques, there have been many successful applications of neural networks to this problem [2], [3], [4], [5]. Different from many other methods, our approach does not make use of quantization, neither of the physical phase space variables nor of internal network parameters, like weights and bias terms, and output values.

We used pole-balancing to test the capabilities of an evolutionary algorithm, which is not based on genetic algorithms (GA). It was mainly designed to evolve recurrent neural networks, which should be able to use internal dynamics for (higher) information processing, and to study the relation between behavior oriented function and structure of the evolved neural networks [6], [7]. Different from most GA algorithms, it sets no constraints neither on the number of neurons nor on the architecture of a network. Network topology and parameters like synaptic weights and bias terms are developed simultaneously.

* in: Gerstner, W., Germond, A., Hasler, M., and Nicoud, J.-D. (eds.), *Artificial Neural Networks - ICANN'97*, Lausanne, Switzerland, October 1997, Proceedings, LNCS 1327, Springer-Verlag, pp. 823 - 829.

Although the combined application of neural network techniques and genetic algorithms turned out to be a very effective tool for solving an interesting class of problems (for a review see e.g. [10], [11], [1]), we suggest, that the algorithm outlined in section 2 is better suited for the evolution of recurrent neural networks; especially in situations, where there is no good guess for an appropriate network architecture or where recurrent dynamic networks should be used for tasks like generation of temporal sequences, recognition, storage and reproduction of temporal patterns, or control problems, which require memory to compute derivatives or integrals.

2 The evolutionary algorithm

To start the evolutionary algorithm (for details see [6] or [7]), we first have to determine the type of neurons of the network. We use the same type of neurons for output and internal units, and input units are used as buffers (as in feedforward networks). Then the number of input and output units is chosen according to the definition of the problem in terms of incoming (sensor) and outgoing (motor) signals. Nothing else is determined, neither the number of internal units nor their connectivity, i.e. self-connections and every kind of recurrences are allowed as well as excitatory and inhibitory connections.

To evolve the desired controllers we consider a population $p(t)$ of $n(t)$ neuromodules undergoing a variation-evaluation-selection loop, i.e.

$$p(t+1) = \mathbf{S} \cdot \mathbf{E} \cdot \mathbf{V}\, p(t) \quad . \tag{1}$$

The variation operator $\mathbf{V}$ acts on each individual neuromodule separately and includes operators for insertion and deletion of neurons, insertion and deletion of connections, and for alterations of bias and weight terms. These operators are of stochastic character, and execute their respective action by fixed per-neuron and per-connection probabilities.

The evaluation $\mathbf{E}$ of individual neuromodules is defined problem-specific. As described in [7], for the pole-balancing problem the performance of a network is determined by balancing time until failure, costs per neuron and synapse (to obtain kind of minimal networks), and - for optimization - costs for the applied force and for deviations from the zero position of cart and pole.

Differential survival of the varied members of the population is defined by the selection operator $\mathbf{S}$. Here it is defined as probabilistic survival according to the evaluation results, i.e. the number of network copies passed from the old to the new population depends on the network's performance. In consequence of this selection process the average performance of the population will either stay the same or increase. Thus, after repeated passes through the variation-evaluation-selection loop, networks will emerge that solve the considered problem.

3 Evolved pole-balancing controllers

For simulation of the cart-pole system we use the standard dynamics and benchmark values defined in [9], i.e. the cart position is $|x| < 2.4$[m], the angle

is $|\theta| < 12[^\circ]$, and the force applied to the cart varies continuously between $-10 < F < 10$[N]. We use no damping terms for the cart-pole dynamics nad compute it by using Euler discretization with time step $\Delta = 0.02$s.

In the following we present and discuss different evolved neurocontrollers. There are two classes of controllers, one, called *t-class*, uses additive units with anti-symmetric transfer function $tanh(x)$, the other one, the *s-class*, uses the strictly positive transfer function $\sigma(x) = (1+e^{-x})^{-1}$. The first class of controllers needs only one output neuron providing a force $F = 10 \cdot tanh(a_i)$[N], where a_i denotes the activity of output unit i. The s-class needs two output units, i and $i+1$, giving a force $F = 10 \cdot (\sigma(a_i) - \sigma(a_{i+1}))$[N]. Using the relation $tanh(x) = (2\sigma(2x) - 1) = (\sigma(2x) - \sigma(-2x))$ an architecture from one class may be converted to an equivalent one in the other class, but here we wanted to study, if there will evolve specific architectures for each class of controllers.

Evolved controllers having the full access to phase space information of the cart-pole system appear to be very simple. Their four input signals are proportional to cart position x and velocity $\dot{x}$, pole angle θ and angular velocity $\dot{\theta}$. An evolved minimal solution in the t-class uses only the output unit and performs quite well on benchmark initial conditions (see [7]). One of the evolved s-class controllers (not shown here), having comparable performance, uses a self-excitatory and a self-inhibitory output unit and only five connections to the inputs.

In the following we concentrate on evolved controllers using only reduced information, i.e. they get only two input signals proportional to cart position x and pole angle θ, respectively. In this situation recurrent connectivity has to be expected, because the derivatives $\dot{x}$ and $\dot{\theta}$ now have to be computed. Satisfying all three objectives simultaneously - balancing, avoiding boundaries and centering the cart - will be a more difficult problem in this case.

3.1 t-class solutions

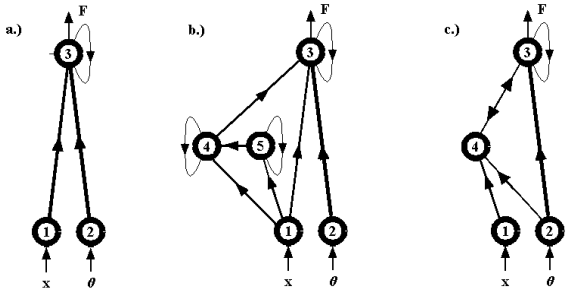

Fig. 1. Three examples (w^1, w^2, w^3) of evolved t-class neurocontrollers.

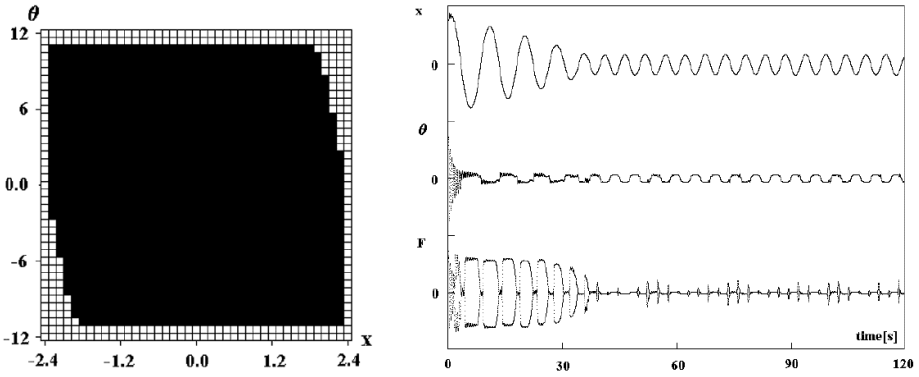

Fig. 2. a.) Initial conditions (black squares) for which the pole is balanced longer than 300 seconds by the controller (4), b.) cart position x, pole angle θ and applied force F as functions of time starting with initial condition $x = 2.0$, $\theta = 0.16$ for the same controller.

Among many other and even larger networks the evolutionary algorithm came up with the three architectures depicted in Fig. 1. Denoting weight vectors of configuration k by w_i^k, with w_{i0}^k the bias term of unit i, the evolved controllers are given by

$$w^1 = (\,0.0, 1.52, 18.36, -0.86\,)\ ; \tag{2}$$

$$w^2 = \begin{pmatrix} 0.0 & -0.76 & 11.6 & -0.89 & -1.02 & 0.0 \\ 0.0 & -16.63 & 0.0 & 0.0 & 0.54 & -9.88 \\ 0.0 & -0.08 & 0.0 & 0.0 & 0.0 & 0.95 \end{pmatrix}\ , \tag{3}$$

$$w^3 = \begin{pmatrix} 0.0 & 0.0 & 2.11 & -1.17 & 0.11 \\ 0.24 & 13.3 & 2.84 & -7.62 & 0.0 \end{pmatrix}\ . \tag{4}$$

They all balance the pole longer than 300 seconds on a large (x, θ)-domain of initial conditions with $\dot{x} = \dot{\theta} = 0$. We tested all configurations on 40 x 40 benchmark initial conditions (x, θ, $\dot{x} = \dot{\theta} = 0.0$) represented by squares as in Figs. 2a and 4a. Outer squares represent initial conditions which already produce a failure signal. A typical domain for successful control is given in Fig. 2a for the controller (4) of Fig. 1c. All three controllers finally keep the cart-pool system oscillating with a small amplitude around zero position and angle. We observed also configurations which finally brought the cart and pole at rest in the zero position, but they operated successfully only on smaller sets of initial conditions.

The simplest evolved t-class solution (Fig. 1a) uses only the output neuron, but with an inhibitory self-connection. This simple control module (2) does balancing and wall avoiding for a large domain of initial conditions (compare [7]); but it does not center the cart, i.e. the cart keeps oscillating around zero with an apparently constant amplitude corresponding to its initial position. The second solution (3) (Fig. 1b) is interesting because it represents two coupled modules: Neurons 1,2 and 3 reproduce the balancing module of Fig. 1a which does not

center the cart; the module composed of neurons 1, 4 and 5 appears to be a cart centering network, i.e. taken as a separate module acting on the cart with a force given by the output of unit 4, it finally brings the cart into the central position by damped oscillation; and this holds for any initial position on the benchmark interval $|x| < 2.4$ [m]. Apparently, it uses a delay line for solving the task and self-inhibition of units for smoothing the cart movement. With values given by (3) it has an unusual way of solving the problem: it keeps the cart oscillating with growing amplitude and, applying a short oscillatory force signal at a certain point, starts again with a smaller amplitude. The third solution (4) has two recurrent loops (one self-connection w_{33} and the loop (w_{34}, w_{43}). It uses a switchable period two oscillator to stabilize the system occasionally. This can be seen in Fig. 2b where cart position, pole angle and applied force are depicted as functions of time.

3.2 s-class solutions

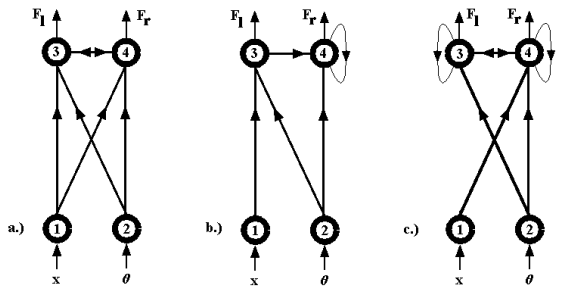

Fig. 3. Three examples (w^4, w^5, w^6) of evolved s-class neurocontrollers.

The three neurocontrollers shown in Fig. 3 are examples of evolved "minimal" configurations for the s-class. Again they solve the problem for a large domain of initial conditions as shown for the controller (7) in Fig. 4a. With a successful control the final state of the cart-pole system is again an oscillatory one, i.e. cart and pole are oscillating with a small amplitude around zero, and the amplitude is the same for all (tested) initial conditions. The configurations of Fig. 3 are given by the following weights:

$$w^4 = \begin{pmatrix} 0.0 & 23.34 & 38.58 & 0.0 & 2.69 \\ 2.98 & 0.92 & 11.92 & 6.72 & 0.0 \end{pmatrix} \quad ; \tag{5}$$

$$w^5 = \begin{pmatrix} -1.21 & 25.23 & 37.85 & 0.0 & 0.0 \\ -3.29 & 0.0 & -16.21 & 11.73 & -6.45 \end{pmatrix} \quad ; \tag{6}$$

$$w^6 = \begin{pmatrix} -2.76 & 0.0 & 22.81 & -9.79 & 14.3 \\ -1.89 & -15.5 & -39.85 & -10.71 & 11.01 \end{pmatrix} \quad . \tag{7}$$

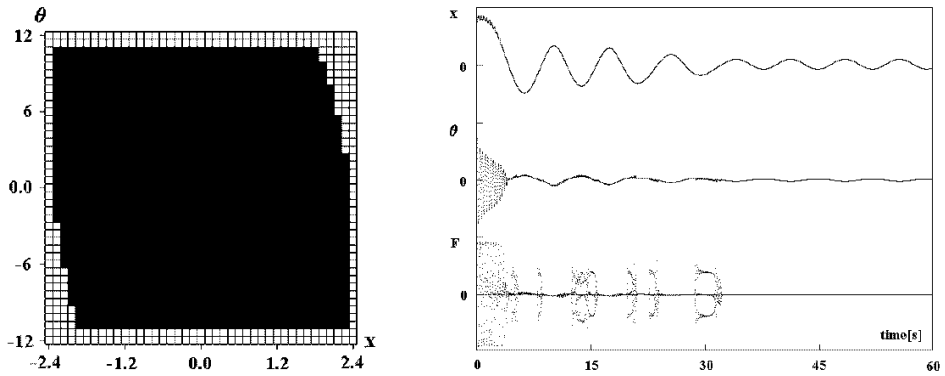

Fig. 4. a.) Initial conditions (black squares) for which the pole is balanced longer than 300 seconds by the controller (7), b.) cart position x, pole angle θ and applied force F as functions of time starting with initial condition $x = 2.0$, $\theta = 0.16$ for the same controller.

Although the three controllers have almost the same benchmark performance, they use different "techniques" to solve the problem. For instance controller (6) finally stabilizes the system with a "periodic burst" of applied force, and controller (7), which is the one with the best benchmark performance (see Fig. 4a), uses higher periods and even chaotic dynamics to stabilize the system in a comparatively short time (around 40 seconds for extreme initial conditions, compare Fig. 2b). In fact, the output units 3 and 4 form a "chaotic neuromodule" [8], and simulations for this configuration show, that the controller really uses also the chaotic domain of the module, i.e. for specific (stationary) control inputs the dynamics of the output module is determined by a chaotic attractor.

4 Discussion

We have demonstrated that our evolutionary algorithm can be applied successfully to control problems like pole balancing. The evolved solutions are remarkable small when compared with those obtained by classical control techniques or pure feedforward networks. We also observe a strong effect of "survival conditions" on the topology of evolved networks: If there is no condition to minimize the applied force for balancing, solutions tend to use internal oscillators. We have also observed solutions making use of periodic and even chaotic dynamics to stabilize the system if it enters critical phase space domains. Furthermore, the evolved controllers are quite robust in the sense, that a moderate noise on the input signals did not effect their performance. The same architectures still operated effectively when changing the time discretization to $\Delta = 0.01$ or $\Delta = 0.03$, or when adding conventional damping terms to the cart-pole system.

The algorithm still can be optimized. For instance, an appropriate learning procedure may be placed between the variation and the evaluation phase of the

evolution cycle (1). Besides the first tests presented here, further simulations on more difficult problems, like for instance balancing a rotator, indicate, that the evolutionary algorithm may be effectively applied to many nonlinear control problems.

References

1. Albrecht, R. F., Reeves, C. R., and Steele, N. C. (eds.), *Artificial Neural Nets and Genetic Algorithms*, Proceedings of the International Conference in Innsbruck, Austria, 1993, Springer, Wien 1993.
2. Anderson, C. W. (1989). Learning to control an inverted pendulum using neural networks. *IEEE Control Systems Magazine*, **9**, 31 - 37.
3. Barto, A. G., Sutton, R. S., and Anderson, C. W. (1983). Neuronlike adaptive elements that solve difficult learning control problems. *IEEE Transactions on Systems, Man, Cybernetics*, **13**, 834 - 846.
4. Bapi, R. S., D'Cruz, B., and Bugmann, G. (1996) Neuro-resistive grid approach to trainable controllers: A pole balancing example, submitted to *Neural Computing and Applications*.
5. Dasgupta, D., and McGregor, D. R. (1993). Evolving neurocontrollers for pole balancing. In S. Gielen and B. Kappen (eds.), *ICANN'93 Proceedings of the International Conference on Artificial Neural Networks*, Amsterdam 13.-16. Sept. 1993. Berlin: Springer-Verlag, 1993, pp. 834-837.
6. Dieckmann, U. (1995), Coevolution as an autonomous learning strategy for neuromodules, in: Herrmann, H., Pöppel, E., and Wolf, D. (eds.), *Supercomputing in Brain Research - From Tomography to Neural Networks*, Singapore: World Scientific, (pp. 331–347).
7. Pasemann, F., Dieckmann, U. (1997). Evolved Neurocontrollers for pole-balancing. In: Proceedings IWANN'97, Lanzarote, Spain, June 4-6, 1997, Lecture Notes in Computer Science. Berlin: Springer-Verlag.
8. Pasemann , F., Nelle, E. (1993). Elements of non-convergent neurodynamics, in: Andersson, S. I., Andersson, A.E., Ottoson, U.: *Dynamical Systems - Theory and Applications*. Singapore: World Scientific.
9. Geva, S., and Sitte, J. (1993). A cartpole experiment benchmark for trainable controllers, *IEEE Control Systems Magazin*, **13**, 40-51.
10. Schaffer, J. D., Whitley, D., and Eshelman, L. J. (1992). Combination of genetic algorithms and neural networks: A survey of the state of the art. In: *Proceedings International Workshop on combinations of genetic algorithms and neural networks (COGANN-92)*, Los Alamitos, CA, IEEE Computer Society Press.
11. Yao, X. (1993). A review of evolutionary artificial neural networks. *International Journal of Intelligent Systems*, **8**, 539 - 567.