Evolutionary Reinforcement Learning: General Models and Adaptation

Danilo Vasconcellos Vargas
Kyushu University
Japan

vargas@inf.kyushu-u.ac.jp
http://itslab.csce.kyushu-u.ac.jp/~vargas/index.php

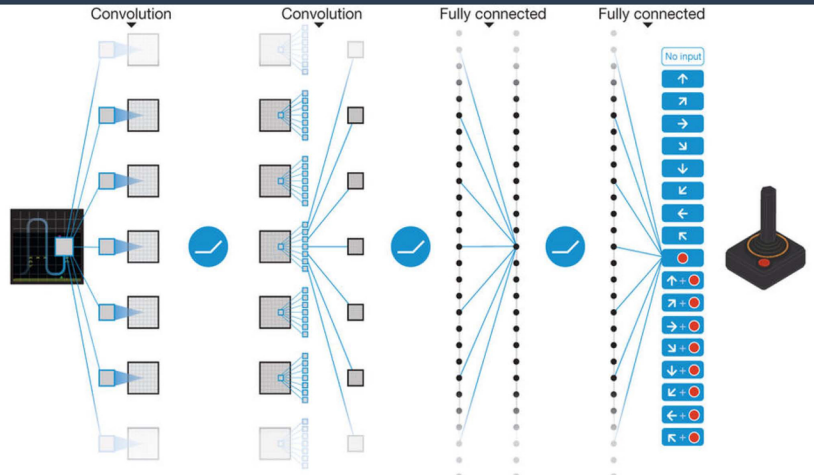
GECCO '18 Companion, July 15–19, 2018, Kyoto, Japan

ACM ISBN 978-1-4503-5764-7/18/07.

https://doi.org/10.1145/3205651.3207865

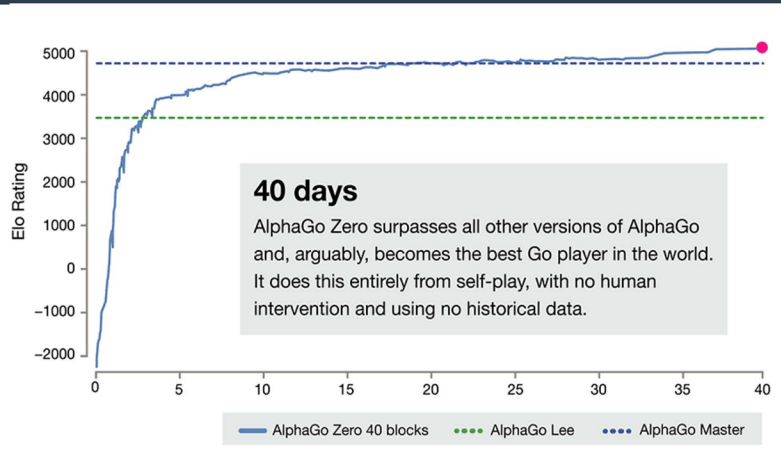
Why Reinforcement Learning?

Reinforcement Learning – Atari Games



Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., ... & Petersen, S. (2015). Human-level control through deep reinforcement learning. Nature, 518(7540), 529-533.

Alpha Go Zero

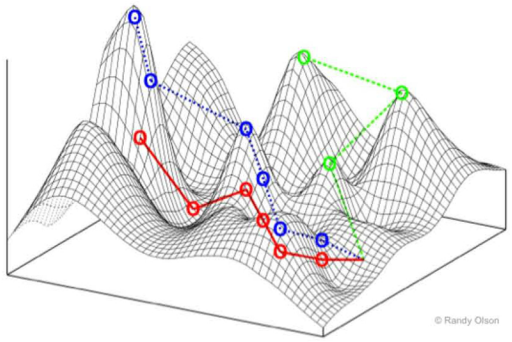


Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., ... & Chen, Y. (2017). Mastering the game of go without human knowledge. Nature, 550(7676), 354.

Why Evolutionary Reinforcement Learning?

5

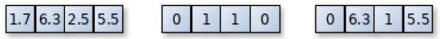
Global Search (Diversity + Population of Solutions)



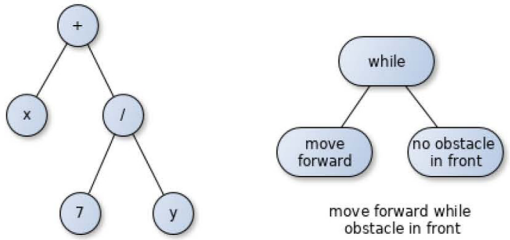
6

Flexible Design

Value Encoding (Array of Real, Binary or Mixed Types)



Tree Encoding



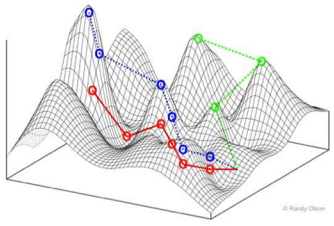
move forward while obstacle in front

x + 7/y

7

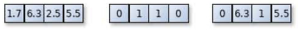
Looking deeper...

Global Search

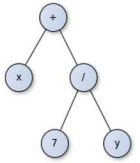


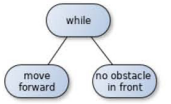
Flexible Design

Value Encoding (Array of Real, Binary or Mixed Types)



Tree Encoding



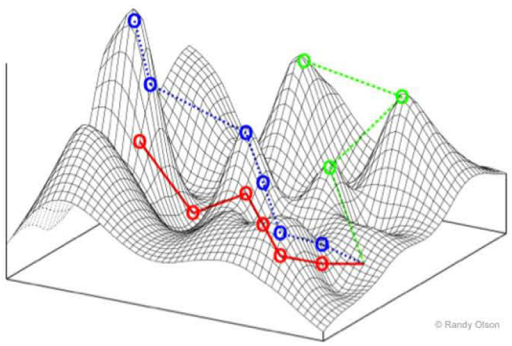


move forward while obstacle in front

x + 7/y

8

Global Search (Diversity + Population of Solutions)



Classical Methods 1: Fitness Sharing

$$f'_i = \frac{f_i}{m_i}$$

- **New fitness (f') is equal its fitness (f) divided by the number of individuals with similar fitness in the population (m)**
  - Requirement: A measure of similarity needs to be defined together with a threshold to define which individuals are close to each one (m)
  - Problem: If a niche grows, all its members have their fitness decreased which can impact evolution negatively

Sareni, B., & Krahenbuhl, L. (1998). Fitness sharing and niching methods revisited. IEEE transactions on Evolutionary Computation, 2(3), 97-106.

Classical Methods 2: Crowding Methods

- **New individuals in the population competes with (and possibly substitute) similar individuals.**
  - Requirement: Needs a measure of similarity between individuals.

Sareni, B., & Krahenbuhl, L. (1998). Fitness sharing and niching methods revisited. IEEE transactions on Evolutionary Computation, 2(3), 97-106.

Classical Methods 3: Clearing

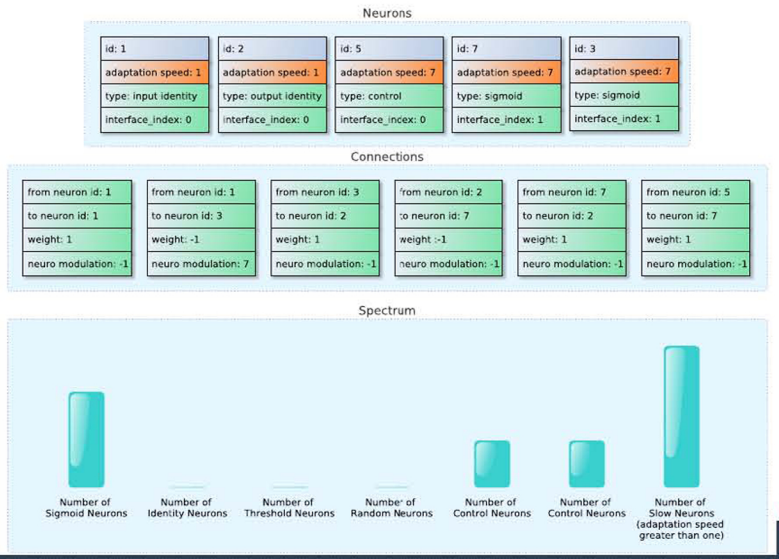
- **Similar to fitness sharing however it preserves the fitness of the best members in each subpopulation (dominant individuals).**
  - Problem: niche radius (how close individuals should be to be from the same niche) is hard to estimate.

Sareni, B., & Krahenbuhl, L. (1998). Fitness sharing and niching methods revisited. IEEE transactions on Evolutionary Computation, 2(3), 97-106.

Another Problem: Curse of Dimensionality

- Distance measures have little use for high dimensional chromosomes.

Spectrum-Diversity (General Distance based on DNA's histogram)



Spectrum-Diversity

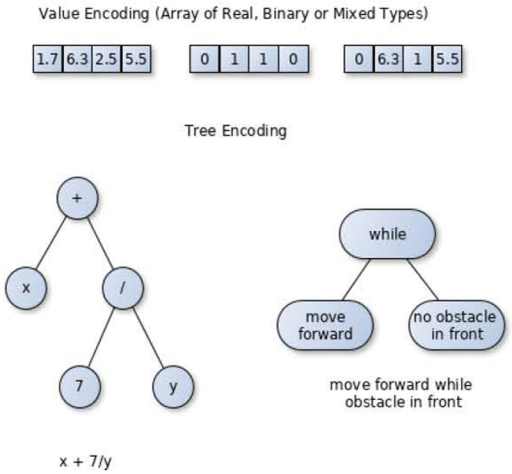
- Similarities to crowding: An individual compete with another individual.
- Difference from crowding: Instead of competing with the most similar individual it competes with the best individual of its niche.
- Similarities to clearing: The best individuals in each niche are preserved.
- Differences to clearing: Niches are automatically defined by using a novelty measure which automatically adapts its niche radius to the population diversity.

Spectrum-Diversity (Summary)

- In other words, it is clearing without the need of defining a niche radius and with a pre-made distance measure for high dimensional chromosome.

Vargas, D. V., & Murata, J. (2017). Spectrum-diverse neuroevolution with unified neural models. IEEE transactions on neural networks and learning systems, 28(8), 1759-1773.

Flexible Design

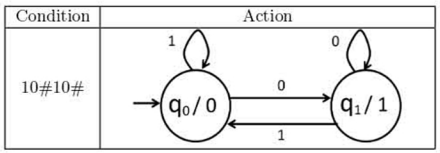


Example: Hoare logic-based Genetic Programming

Skip statement:	$\{P\}skip\{P\}$
Assignment:	$\{P[t/x]\}x := t\{P\}$
If-statement:	$\frac{\{P \wedge e\}S_1\{Q\}, \{P \wedge \neg e\}S_2\{Q\}}{\{P\} \text{ if } e \text{ then } S_1 \text{ else } S_2\{Q\}}$
Repetition:	$\frac{\{P \wedge e\}S\{P\}}{\{P\} \text{ while } e \text{ do } S\{P \wedge \neg e\}}$
Composition:	$\frac{\{P\}S_1\{R\}, \{R\}S_2\{Q\}}{\{P\}S_1; S_2\{Q\}}$
Rewriting:	$\frac{P \rightarrow P_1, \{P_1\}S\{Q_1\}, Q_1 \rightarrow Q}{\{P\}S\{Q\}}$

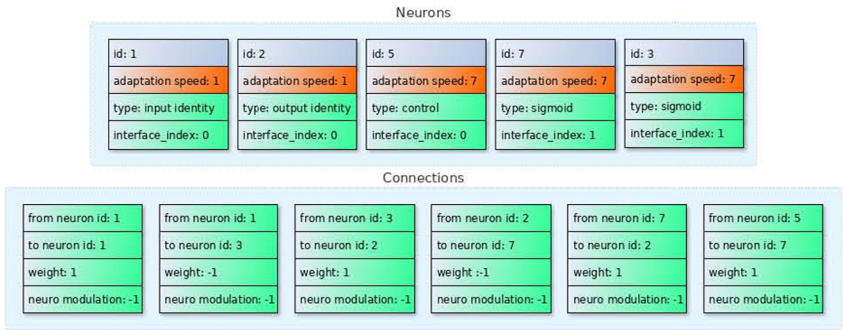
He, P., Kang, L., Johnson, C. G., & Ying, S. (2011). Hoare logic-based genetic programming. Science China Information Sciences, 54(3), 623-637.

Example: Conditional Rule + Cyclic Graphs



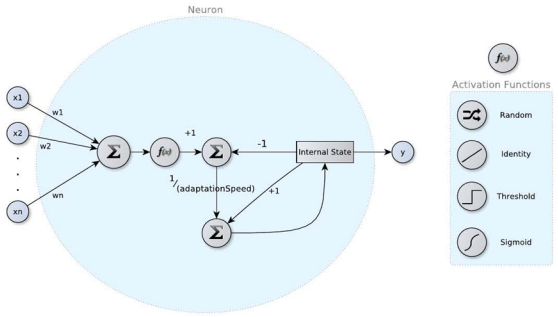
Iqbal, M., Browne, W. N., & Zhang, M. (2017). Extending xcs with cyclic graphs for scalability on complex boolean problems. Evolutionary computation, 25(2), 173-204.

Example: Unified Neural Model



Vargas, D. V., & Murata, J. (2017). Spectrum-diverse neuroevolution with unified neural models. IEEE transactions on neural networks and learning systems, 28(8), 1759-1773.

Example: Unified Neural Model

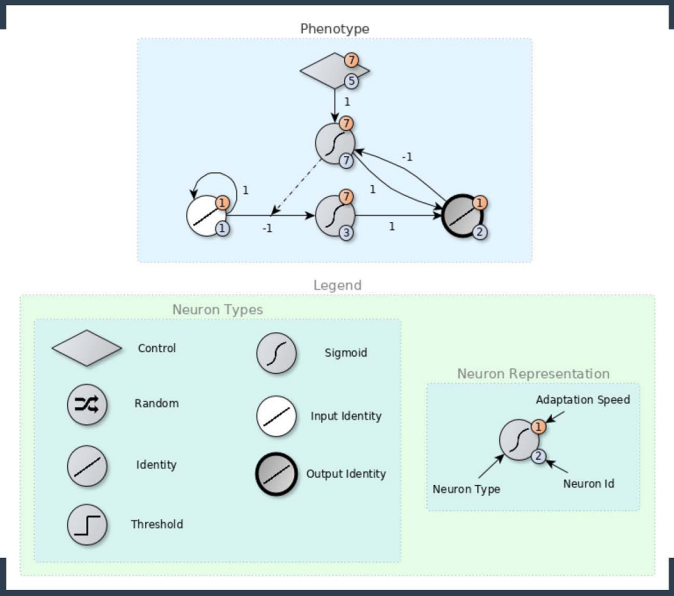


$$a = f(\sum_{i=1}^{n}(w_i x_i))$$

$$Ins_t = Ins_{t-1} + \frac{1}{adaptationSpeed}(a - Ins_{t-1})$$

$$y = Ins_t$$

Example: Unified Neural Model



How to make an Evolutionary Reinforcement Learning Method?

Approaches...

Learning Classifier Systems

Basic Idea: Divide & Conquer

Budzlife
Flickr

Kevin Hunter
@youtube

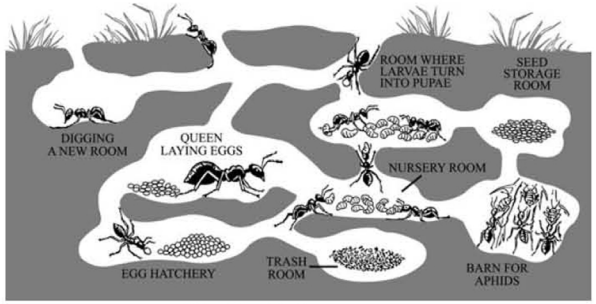
Basic Idea: Divide & Conquer

Budzlife
Flickr

Kevin Hunter
@youtube

Unity Strength

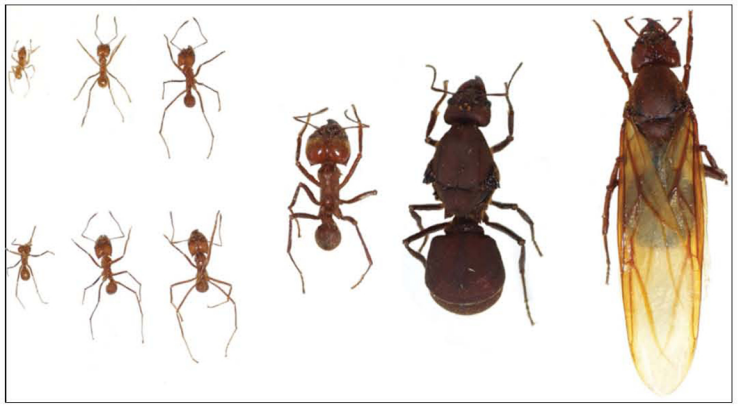
Breaking the Problem



The Team Approach by Steven Stowell

Dividing the problem

Specialization or Heterogeneous Agents (Polymorphism in Ants)

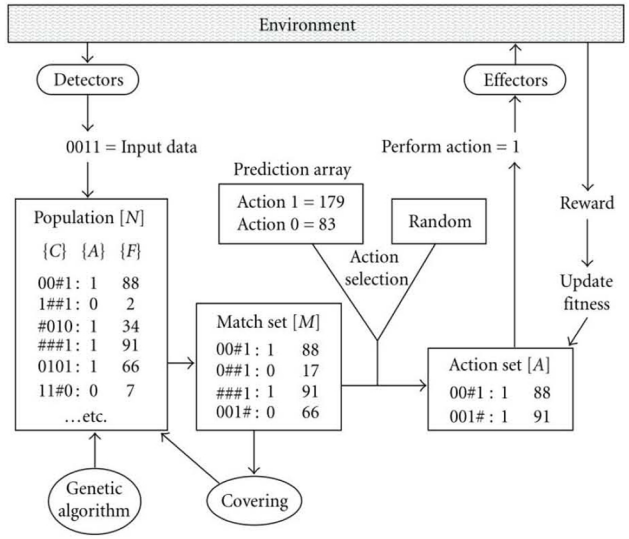


Caste morphology, used under the GNU Free Documentation License version 1.3

"Correlated to specific tasks within the colony. These include small workers that undertake garden management and brood care, medium workers that forage leaves, large workers that can serve as soldiers, and winged sexuals that lose their wings after mating."

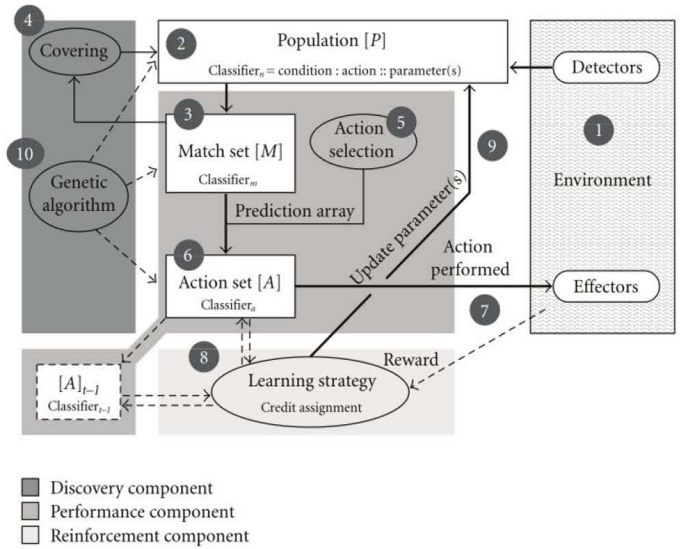
Suen, G., Teiling, C., Li, L., Holt, C., Abouheif, E., Bornberg-Bauer, E., ... & Denas, O. (2011). The genome sequence of the leaf-cutter ant Atta cephalotes reveals insights into its obligate symbiotic lifestyle. PLoS Genetics, 7(2), e1002007.

Learning Classifier Systems (Main Idea)



Urbanowicz, R. J., & Moore, J. H. (2009). Learning classifier systems: a complete introduction, review, and roadmap. Journal of Artificial Evolution and Applications, 2009, 1.

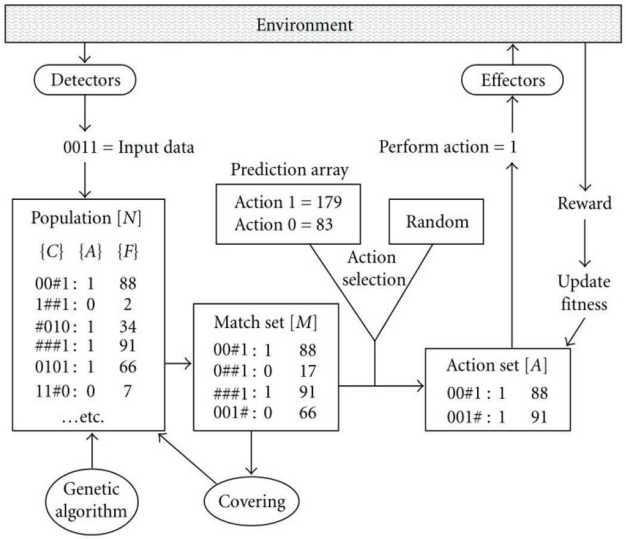
Learning Classifier Systems (Main Idea)



Urbanowicz, R. J., & Moore, J. H. (2009). Learning classifier systems: a complete introduction, review, and roadmap. Journal of Artificial Evolution and Applications, 2009, 1.

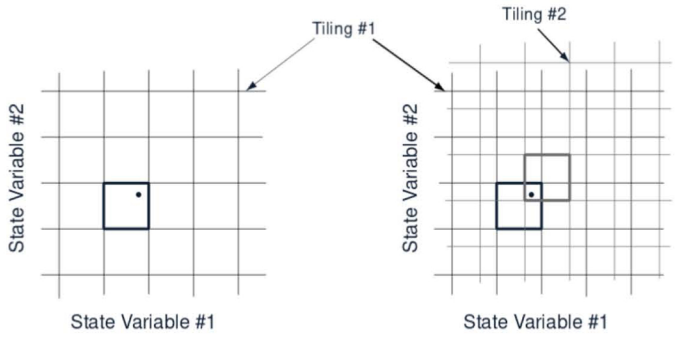
How to get continuous states and actions? (same problem as standard Q-learning)

Learning Classifier Systems (Main Idea)



Urbanowicz, R. J., & Moore, J. H. (2009). Learning classifier systems: a complete introduction, review, and roadmap. Journal of Artificial Evolution and Applications, 2009, 1.

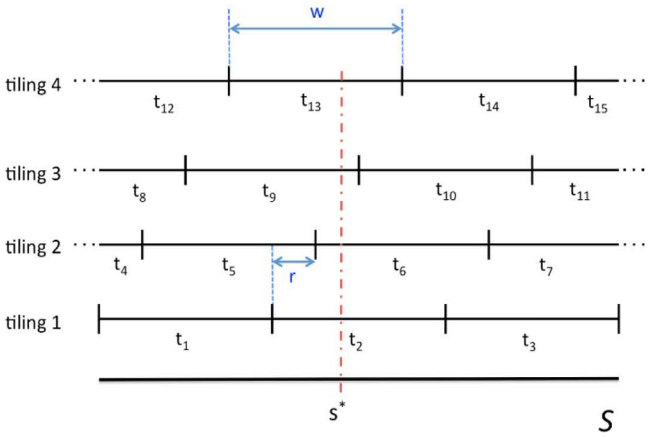
Tile Coding (1996) - higher precision with binary coding



Sherstov, A. A., & Stone, P. (2005, July). Function approximation via tile coding: Automating parameter choice. In International Symposium on Abstraction, Reformulation, and Approximation (pp. 194-205). Springer, Berlin, Heidelberg.

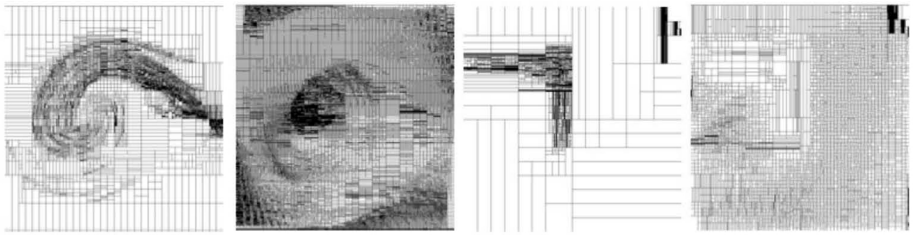
Sutton, R. S. (1996). Generalization in reinforcement learning: Successful examples using sparse coarse coding. In Advances in neural information processing systems (pp. 1038-1044).

Increase resolution by increasing tiles



Lanzi, P. L., & Loiacono, D. (2015). XCSF with tile coding in discontinuous action-value landscapes. Evolutionary Intelligence, 8(2-3), 117-132.

Adaptive Tile-coding (2007)

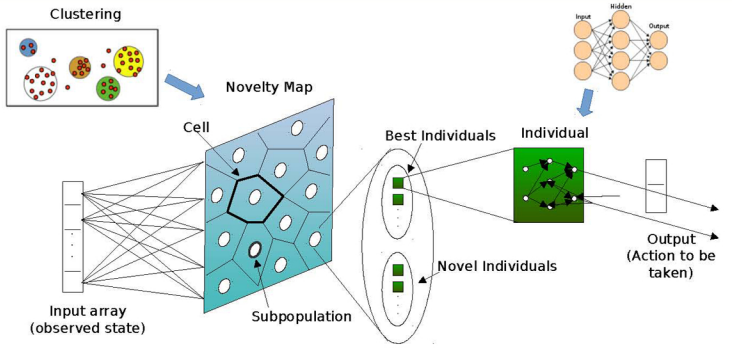


Whiteson, S., Taylor, M. E., & Stone, P. (2007). Adaptive tile coding for value function approximation. Computer Science Department, University of Texas at Austin.

Problems

- Learning becoming increasingly slow with the number of tilings (states are visited inequally)
- Trade-off between granulity and value prediction quality. In other words, many tilings approximate better the real function but is harder to learn.

Self-Organizing Classifiers (2013)



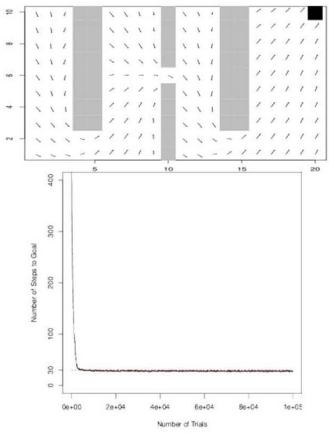
Algorithm Type	Model	Fitness
Self-Organizing Classifiers	Dynamic state-table of prediction values	Strength based
Learning Classifier Systems	Set of condition-action-prediction rules	Accuracy based
Standard Reinforcement Learning	Static state-action lookup tables	Strength based

Vargas, D. V., Takano, H., & Murata, J. (2013). Self organizing classifiers and niched fitness. In Proceedings of the 15th annual conference on Genetic and evolutionary computation (pp. 1109-1116). ACM.

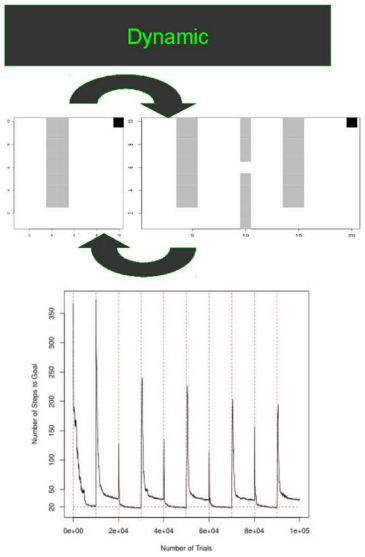
Results: Maze

One of the few methods that solve noisy mazes and the only one capable of solving dynamic mazes.

Noisy

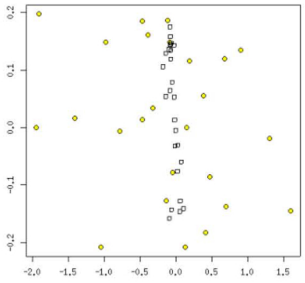


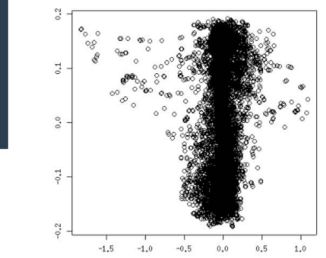
Dynamic



Vargas, D. V., Takano, H., & Murata, J. (2013). Self organizing classifiers: first steps in structured evolutionary machine learning. Evolutionary Intelligence, 6(2), 57-72.

SOM's issues





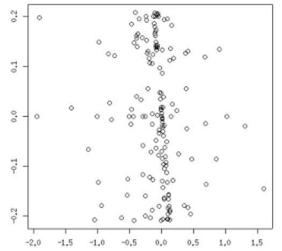


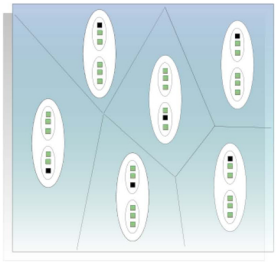
Fig. 2 Final superposed SOM's (squares) and Nmap's (circles) weight array positions after 10^4 episodes in the pole-balancing problem.

Fig. 1 Accumulated SOM's (above) and Nmap's (below) weight array positions during 10^4 episodes in the pole-balancing problem.

Vargas, D. V., Takano, H., & Murata, J. (2014, September). Novelty-organizing team of classifiers-a team-individual multi-objective approach to reinforcement learning. In SICE Annual Conference (SICE), 2014 Proceedings of the (pp. 1785-1792). IEEE.

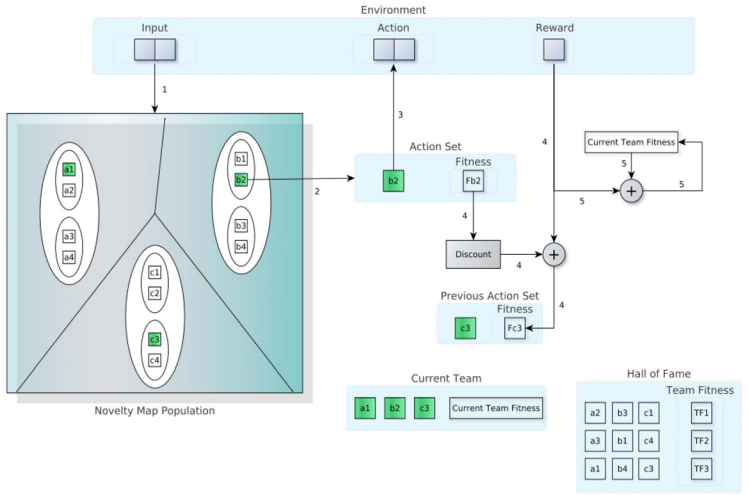
Novelty-Organizing Team of Classifiers (2015)

- First approach that joins the value-function and policy search paradigms into one framework.
- Introduces Novelty Map: keep top novel individuals. Avoid problems with high frequency states present in SOM.



Vargas, D. V., Takano, H., & Murata, J. (2015). Novelty-organizing team of classifiers in noisy and dynamic environments. In Evolutionary Computation (CEC), 2015 IEEE Congress on (pp. 2937-2944). IEEE.

Novelty-Organizing Team of Classifiers (2015)

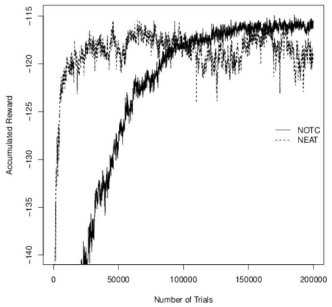


Vargas, D. V., Takano, H., & Murata, J. (2015). Novelty-organizing team of classifiers in noisy and dynamic environments. In Evolutionary Computation (CEC), 2015 IEEE Congress on (pp. 2937-2944). IEEE.

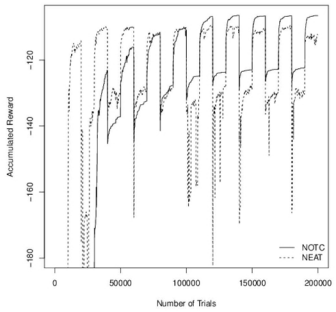
Results: Mountain Car

Illustration of Mountain Car

Noisy



Dynamic



Vargas, D. V., Takano, H., & Murata, J. (2015). Novelty-organizing team of classifiers in noisy and dynamic environments. In Evolutionary Computation (CEC), 2015 IEEE Congress on (pp. 2937-2944). IEEE.

Neuroevolution

Fixed Topology (1989)

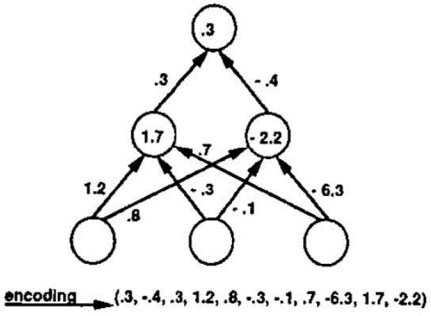


Figure 1: Encoding a Network on a Chromosome

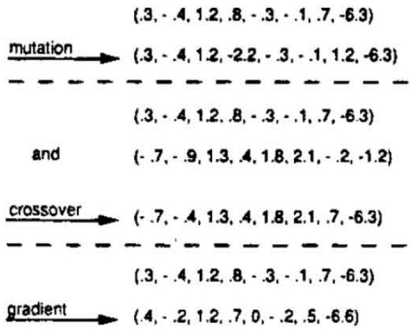


Figure 2: Operation of the Operators

Montana, D. J., & Davis, L. (1989, August). Training Feedforward Neural Networks Using Genetic Algorithms. In IJCAI (Vol. 89, pp. 762-767).

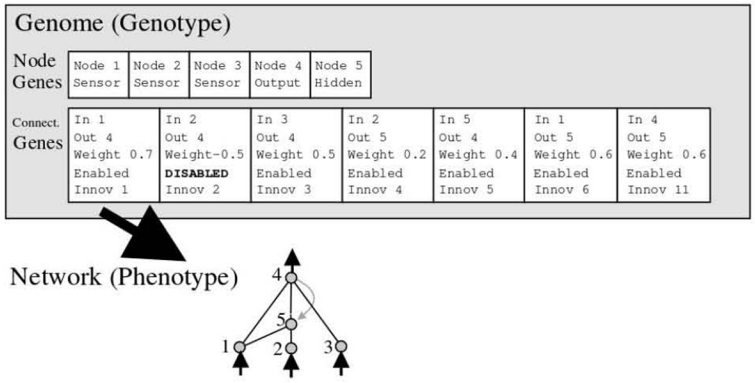
GNARL (1994)

- One of the first neuroevolution methods that evolves both the topology and the weights of the network.
- Departs from GA and gets close to evolutionary programming or GP.
- Removes crossover.
- Allow for recurrency to occur.

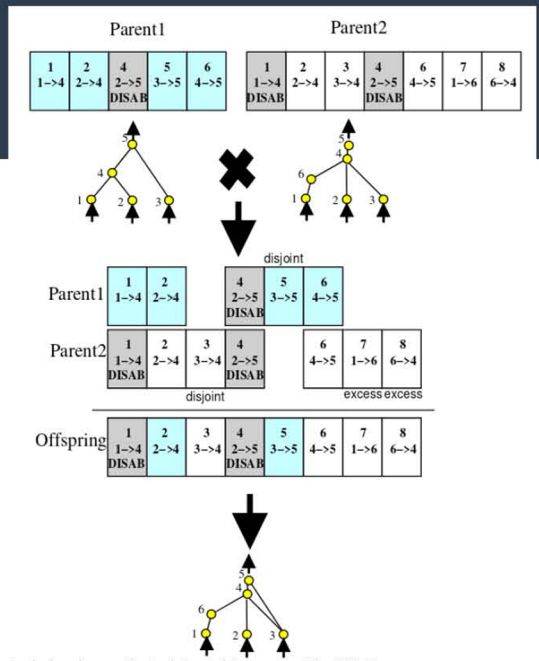
Angeline, P. J., Saunders, G. M., & Pollack, J. B. (1994). An evolutionary algorithm that constructs recurrent neural networks. IEEE transactions on Neural Networks, 5(1), 54-65.

NEAT (2002)

- Start from a network without hidden nodes.
- Mutation can only increase nodes



Crossover



Stanley, K. O., & Miikkulainen, R. (2002). Evolving neural networks through augmenting topologies. Evolutionary computation, 10(2), 99-127.

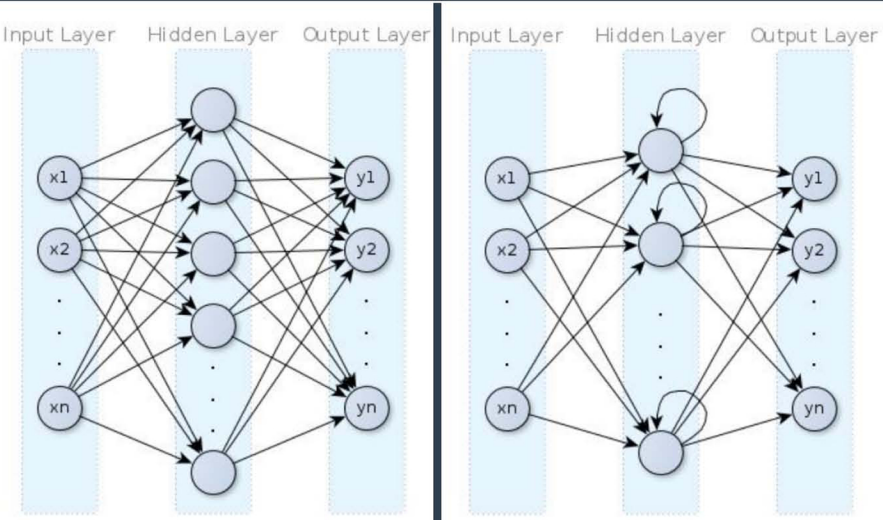
Crossover: to be or not to be?

- Crossover is a complex problem-dependent procedure which is often similar to a higher mutation rate.
- Removing crossover making them only slightly slower in the problem tested .

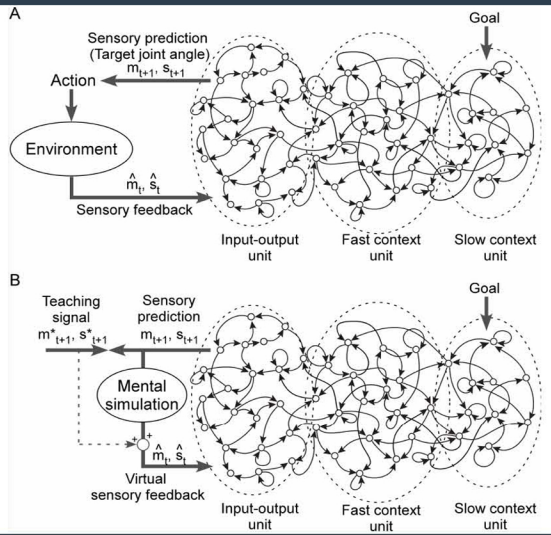
Method	Evaluations	Failure Rate
No-Growth NEAT (Fixed-Topologies)	**30,239**	80%
Non-speciated NEAT	**25,600**	25%
Initial Random NEAT	**23,033**	5%
Nonmating NEAT	**5,557**	0
Full NEAT	**3,600**	0

Stanley, K. O., & Miikkulainen, R. (2002). Evolving neural networks through augmenting topologies. Evolutionary computation, 10(2), 99-127.

Problems depend on the representation, but the representation was limited to:



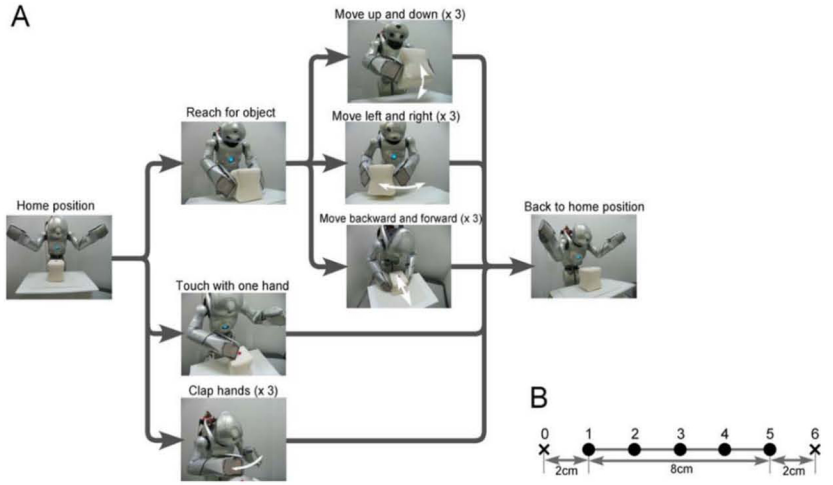
Higher level behavior need slower neurons (2008)



Note: this work does not use EAs

Yamashita, Y., & Tani, J. (2008). Emergence of functional hierarchy in a multiple timescale neural network model: a humanoid robot experiment. PLoS computational biology, 4(11), e1000220.

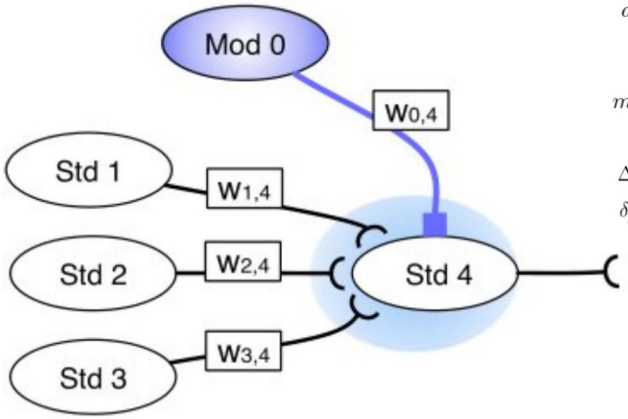
Higher level behavior need slower neurons (2008)



Yamashita, Y., & Tani, J. (2008). Emergence of functional hierarchy in a multiple timescale neural network model: a humanoid robot experiment. PLoS computational biology, 4(11), e1000220.

Neuromodulation (2008) adaptation: learning and memory

Plastic = all weights change
Modularity = weights' change depend on activation



$$a_i = \sum_{j \in Std} w_{ji} \cdot o_j \quad ,$$

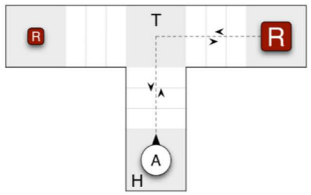
$$m_i = \sum_{j \in Mod} w_{ji} \cdot o_j \quad ,$$

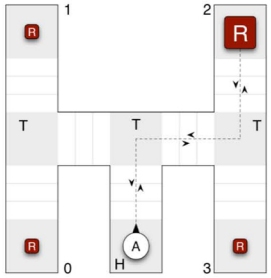
$$\Delta w_{ji} = tanh(m_i/2) \cdot \delta_{ji}$$

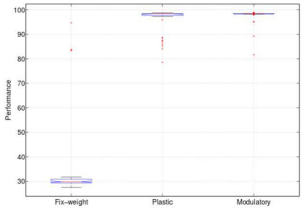
$$\delta_{ji} = \eta \cdot [Ao_j o_i + Bo_j + Co_i + D]$$

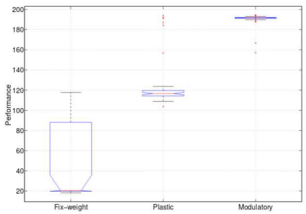
Connection change depends on previous and post neurons output.

ES is used to find the parameters A, B, C and D of all neurons.





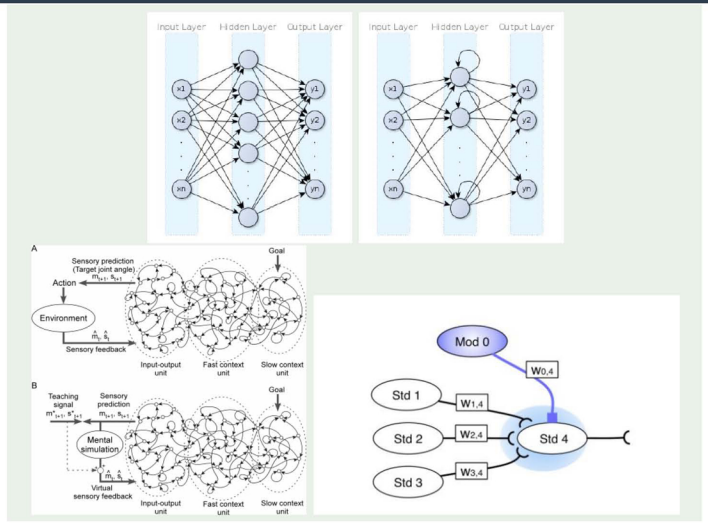




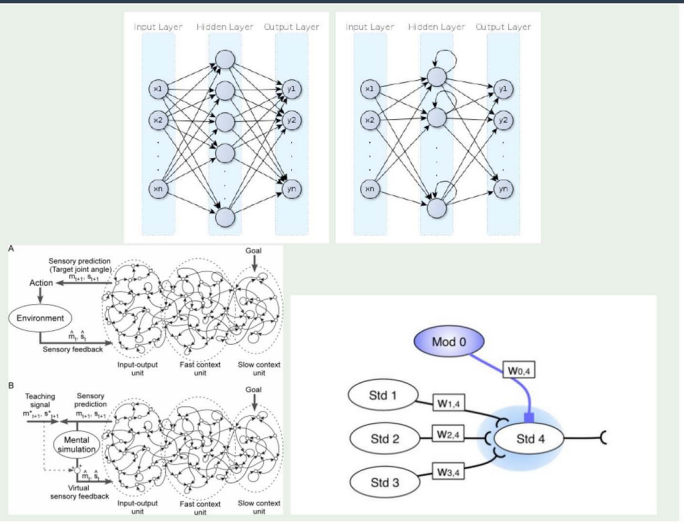
Soltoggio, A., Bullinaria, J. A., Mattiussi, C., Dürr, P., & Floreano, D. (2008). Evolutionary advantages of neuromodulated plasticity in dynamic, reward-based scenarios. In Proceedings of the 11th international conference on artificial life (Alife XI) (No. LIS-CONF-2008-012, pp. 569-576). MIT Press.

Soltoggio, A., Bullinaria, J. A., Mattiussi, C., Dürr, P., & Floreano, D. (2008). Evolutionary advantages of neuromodulated plasticity in dynamic, reward-based scenarios. In Proceedings of the 11th international conference on artificial life (Alife XI) (No. LIS-CONF-2008-012, pp. 569-576). MIT Press.

Different Models – Different Applications

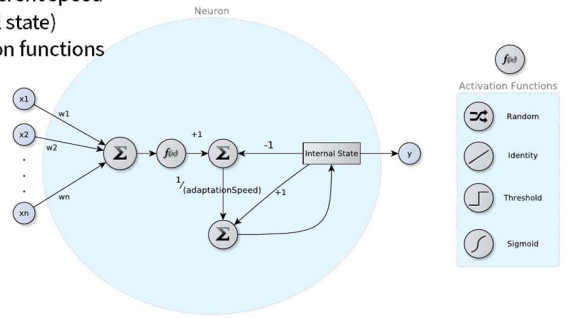


How to create a system that can generalize to any problem?



Spectrum-diverse Neuroevolution Algorithm
SUNA (2017) - Unified Neural Model

- neurons with different speed
- memory (internal state)
- different activation functions



$$a = f(\sum_{i=1}^{n}(w_i x_i))$$

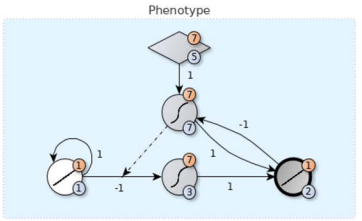
$$Ins_t = Ins_{t-1} + \frac{1}{adaptationSpeed}(a - Ins_{t-1})$$

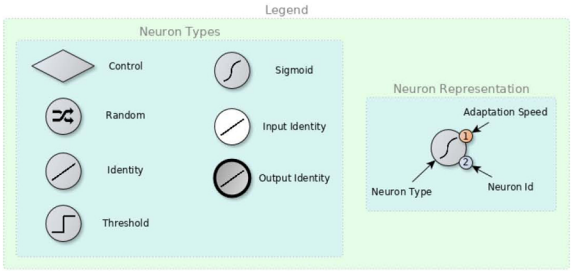
$$y = Ins_t$$

Vargas, D. V., & Murata, J. (2017). Spectrum-diverse neuroevolution with unified neural models. IEEE transactions on neural networks and learning systems, 28(8), 1759-1773.

SUNA (2017) - Unified Neural Model

- neuromodulation of connections
- neuromodulation of neurons
- loops allowed
- unbounded input





Vargas, D. V., & Murata, J. (2017). Spectrum-diverse neuroevolution with unified neural models. IEEE transactions on neural networks and learning systems, 28(8), 1759-1773.

SUNA (2017): Unified Neural Model

Neurons

id: 1	id: 2	id: 5	id: 7	id: 3
adaptation speed: 1	adaptation speed: 1	adaptation speed: 7	adaptation speed: 7	adaptation speed: 7
type: input identity	type: output identity	type: control	type: sigmoid	type: sigmoid
interface_index: 0	interface_index: 0	interface_index: 0	interface_index: 1	interface_index: 1

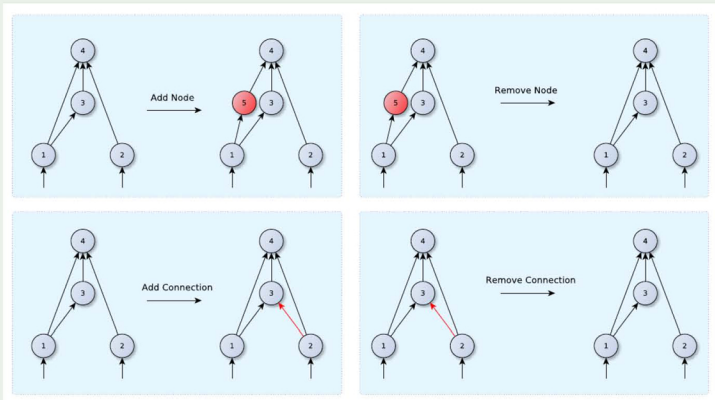
Connections

from neuron id: 1	from neuron id: 1	from neuron id: 3	from neuron id: 2	from neuron id: 7	from neuron id: 5
to neuron id: 1	to neuron id: 3	to neuron id: 2	to neuron id: 7	to neuron id: 2	to neuron id: 7
weight: 1	weight: -1	weight: 1	weight :-1	weight: 1	weight: 1
neuro modulation: -1	neuro modulation: -1	neuro modulation: -1	neuro modulation: -1	neuro modulation: -1	neuro modulation: -1

Vargas, D. V., & Murata, J. (2017). Spectrum-diverse neuroevolution with unified neural models. IEEE transactions on neural networks and learning systems, 28(8), 1759-1773.

SUNA (2017): Evolution

Structural Mutation:



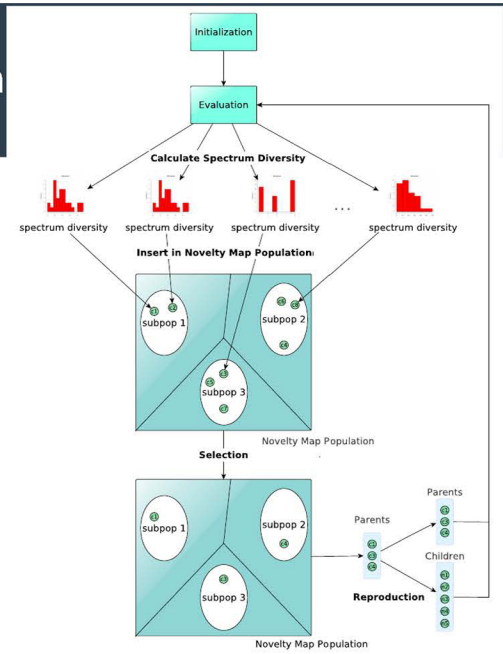
Ordinal Mutation: Weight Perturbations

Vargas, D. V., & Murata, J. (2017). Spectrum-diverse neuroevolution with unified neural models. IEEE transactions on neural networks and learning systems, 28(8), 1759-1773.

How to evolve such a complex model?

61

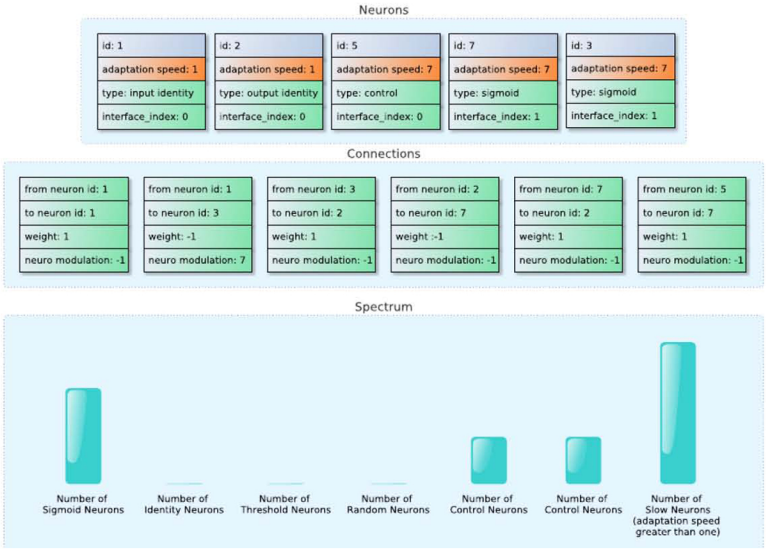
SUNA: Evolution



62

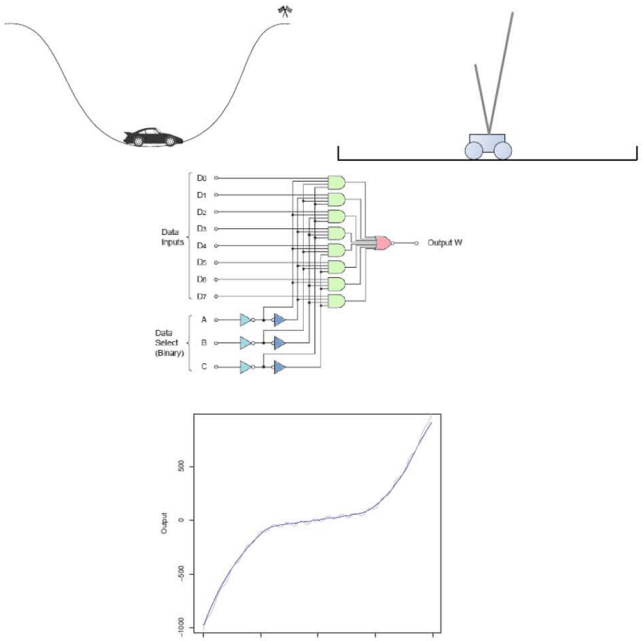
Vargas, D. V., & Murata, J. (2017). Spectrum-diverse neuroevolution with unified neural models. IEEE transactions on neural networks and learning systems, 28(8), 1759-1773.

Spectrum-Diversity (General Distance based on DNA's



63

Problems Tackled - The Algorithm Can Learn 5 Completely Different Control Problems Without Changing Any Parameters



SUNA – Tackling 5 completely different problems without any preprocessing

Notice that the range of input and output vary from problem to problem. This needed to be treated for NEAT but no preprocessing was done for SUNA.

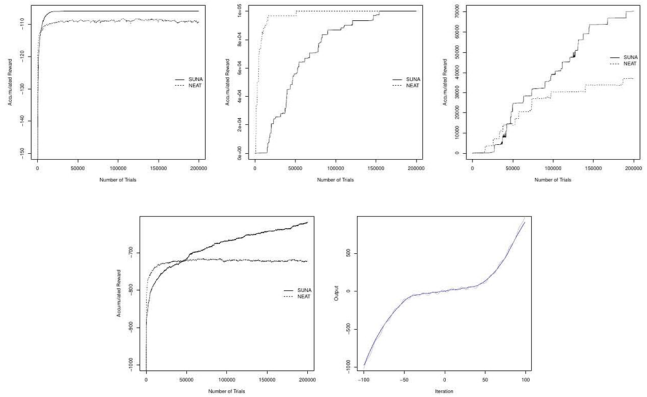


Figure: Results on Mountain Car, Double Pole, Non-Markov Double Pole, Multiplexer and Function Approximation

Vargas, D. V., & Murata, J. (2017). Spectrum-diverse neuroevolution with unified neural models. IEEE transactions on neural networks and learning systems, 28(8), 1759-1773.

SUNA: Are all the neuron types important?

Table: Percentage of improvement (positive) or worsening (negative) of the ablation results in relation to the original one. The problems are Double Pole (DP), Function Approximation (FA), Mountain Car (MA), Multiplexer (MU) and Non Markov Double Pole (NMDP).

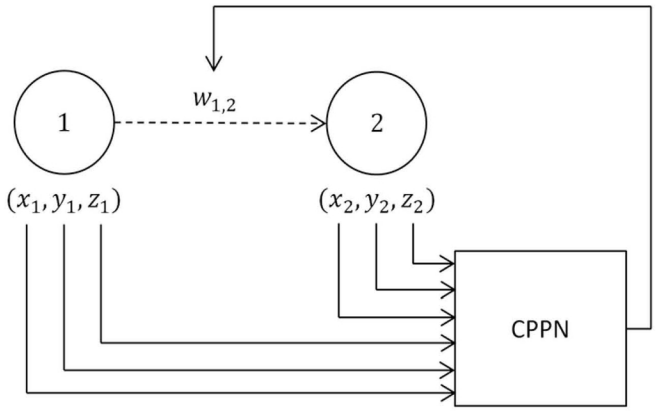
Test Type	DP	FA	MC	MU	NMDP
No control neuron	0%	5.87%	0%	−10.55%	15.76%
No linear neuron	0%	−22.38%	0%	−1.76%	11.18%
No neuromodulation	0%	6.92%	0%	−66.17%	7.88%
No random neuron	−6.28%	7.37%	0%	8.26%	−56.54%
No real weights	−3.29%	9.43%	0%	6.16%	−99.85%
No sigmoid neuron	0%	−3.73%	0%	−3.35%	1.97%
No slow neuron	−6.46%	−32.69%	0%	12.61%	−35.94%
No threshold neuron	−3.13%	6.43%	0%	1.56%	11.77%

- Threshold neuron is similar to sigmoid neuron outside of a the [0,1] range. Therefore, removing one makes no difference.
- Control neuron was not usefult, although having an interesting representation power. Are the problems still too easy? This remains as an open question.

Vargas, D. V., & Murata, J. (2017). Spectrum-diverse neuroevolution with unified neural models. IEEE transactions on neural networks and learning systems, 28(8), 1759-1773.

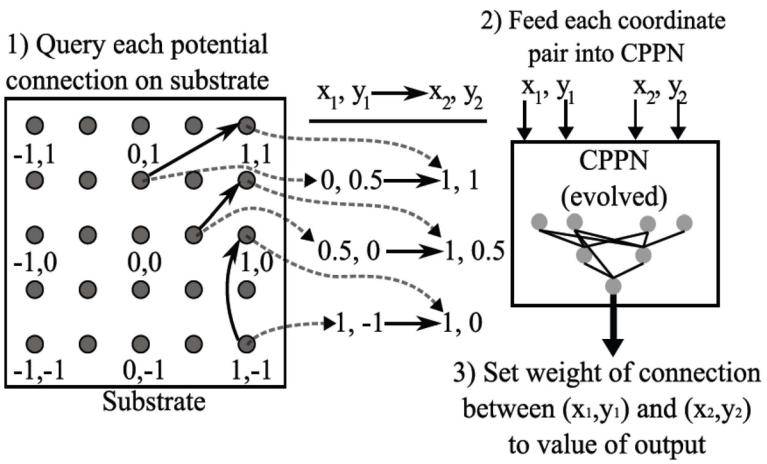
Vision Based Reinforcement Learning

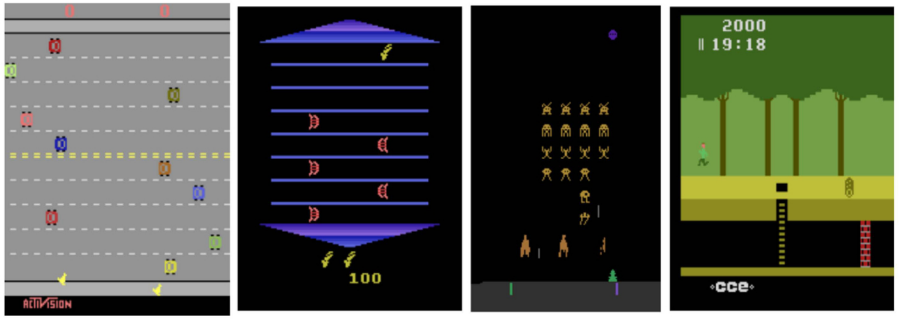
HyperNEAT (2009) – Indirect Encoding with NEAT



Wikipedia: EvilxFish

HyperNEAT – Atari Games (2014)



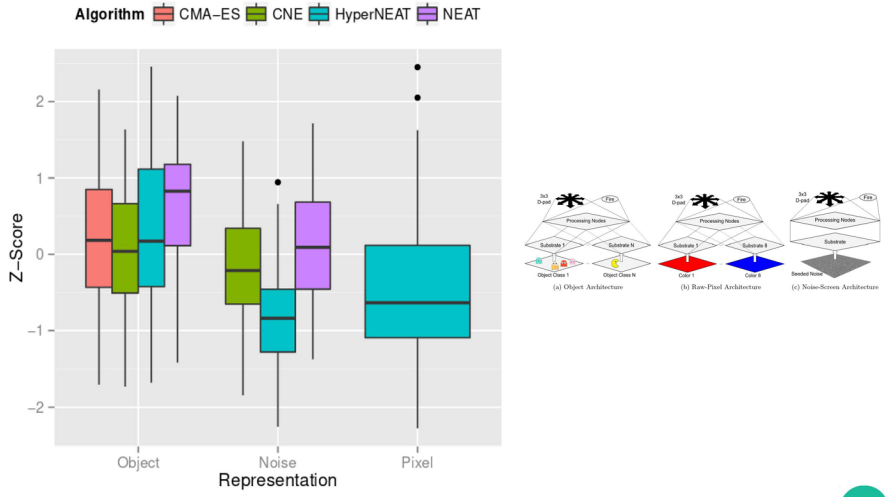


(a) Freeway (b) Asterix (c) Space Invaders (d) Pitfall

Stanley, K. O., D'Ambrosio, D. B., & Gauci, J. (2009). A hypercube-based encoding for evolving large-scale neural networks. Artificial life, 15(2), 185-212.

Hausknecht, M., Lehman, J., Miikkulainen, R., & Stone, P. (2014). A neuroevolution approach to general atari game playing. IEEE Transactions on Computational Intelligence and AI in Games, 6(4), 355-366.

HyperNEAT – Atari Games (2014)



Hausknecht, M., Lehman, J., Miikkulainen, R., & Stone, P. (2014). A neuroevolution approach to general atari game playing. IEEE Transactions on Computational Intelligence and AI in Games, 6(4), 355-366.

Visual RNN (2013)

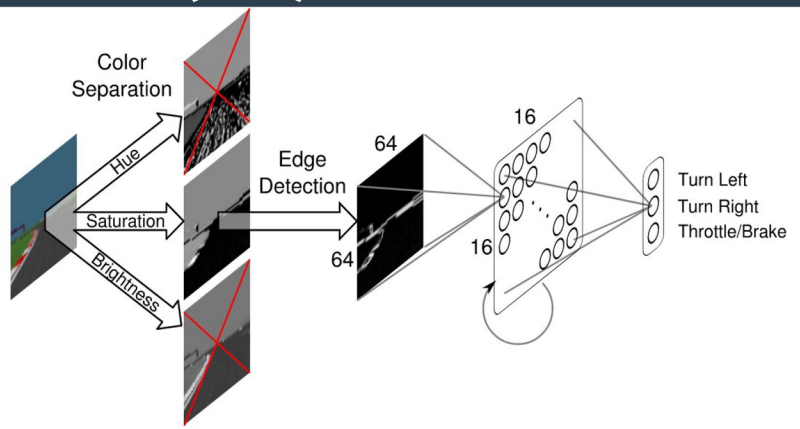


Figure 5: Visual TORCS network controller pipeline. At each time-step a raw 64×64 pixel image, taken from the driver's perspective, is split into three planes (hue, saturation and brightness). The saturation plane is then passed through Robert's edge detector [12] and then fed into the 16×16=256 recurrent neurons of the controller network, which then outputs the three driving commands.

Koutník, J., Cuccu, G., Schmidhuber, J., & Gomez, F. (2013, July). Evolving large-scale neural networks for vision-based reinforcement learning. In Proceedings of the 15th annual conference on Genetic and evolutionary computation (pp. 1061-1068). ACM.

Visual RNN

- Indirect encoding + image pre-processing + internal memory (RNNs)
- Huge Network - “With this architecture, the networks have a total of 1,115,139 weights, organized into 5 weight matrices. The weights are encoded indirectly by 200 DCT coefficients...”

Do networks need to be big?

Unsupervised MPCNN + 33-weight RNN (2014)

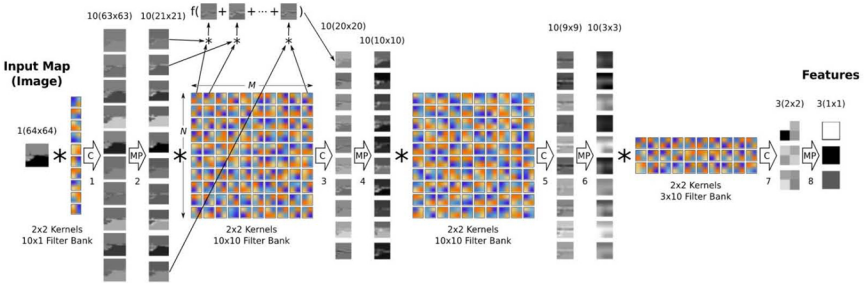


Figure 6: Max-Pooling Convolutional Neural Network (MPCNN) with 8 layers alternating between convolution (C) and downsampling (MP; using max-pooling). The first layer convolves the input 64×64 pixel image with a bank of 10×10 filters producing 10 maps of size 63×63, that are down-sampled to 21×21 by MP layer 2. Layer 3 convolves each of these 10 maps with a filter, sums the results and passes them through the nonlinear function f, producing 10 maps of 20×20 pixels each, and so on until the input image is transformed to just 3 features that are passed to the RNN controller, see Figure 3.

Koutník, J., Schmidhuber, J., & Gomez, F. (2014, July). Evolving deep unsupervised convolutional networks for vision-based reinforcement learning. In Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation (pp. 541-548). ACM.

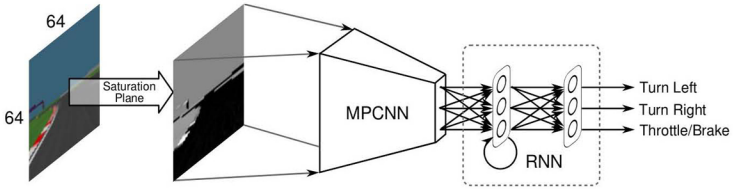


Figure 3: Visual TORCS network controller pipeline. At each time-step a raw 64×64 pixel image, taken from the driver's perspective, is split into three planes (hue, saturation and brightness). The saturation plane is fed into the max-pooling convolutional network (MPCNN), that generates features for the recurrent neural network (RNN) controller, that drives the car by controlling the steering, brakes, and accelerator.

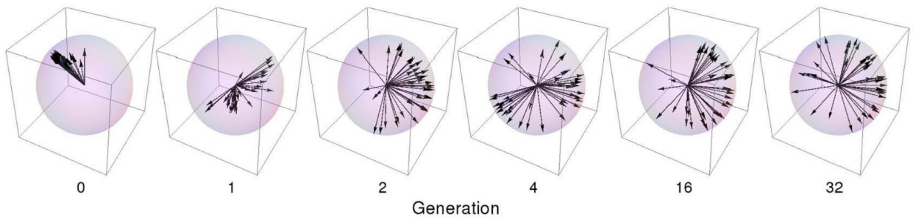
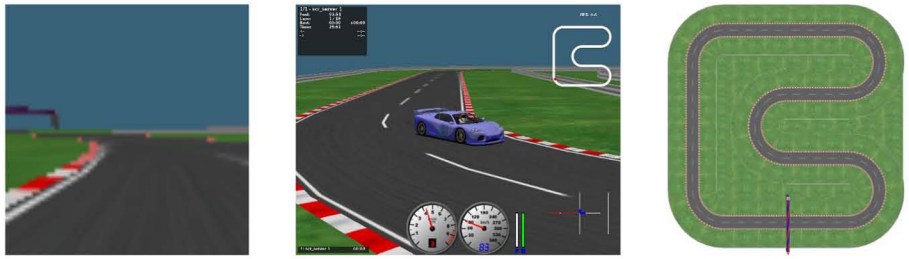


Figure 4: Evolving MPCNN features. Each plot shows the feature vectors for each of the 40 training images on the unit sphere. Initially (generation 0), the features are clustered together. After just a few generations spread out so that the MPCNN discriminates more clearly between the images.

Koutník, J., Schmidhuber, J., & Gomez, F. (2014, July). Evolving deep unsupervised convolutional networks for vision-based reinforcement learning. In Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation (pp. 541-548). ACM.

Problem Tackled: TORCS Racing Simulation



(a) (b) (c)

Figure 2: Visual TORCS environment. (a) The 1st-person perspective used as input to the RNN controllers (figure 3) to drive the car around the track. (b), a 3rd-person perspective of car. The controllers were evolved using a track (c) of length of 714.16 m and road width of 10 m, that consists of straight segments of length 50 and 100 m and curves with radius of 25 m. The car starts at the bottom (start line) and has to drive counter-clockwise. The track boundary has a width of 14 m.

MPCNN and Visual RNN's Results

controller	d [m]	v_{max} [km/h]
olethros	570	147
bt	613	141
berniw	624	149
tita	657	150
inferno	682	150
visual RNN[13]	625	144
MPC-RNN	547	97

Table 2: Maximum distance, d, in meters and maximum speed, v_{max}, in kilometers per hour achieved by hand-coded controllers that come with TORCS which enjoy access to the state variables (the five upper table entries), a million-weight RNN controller that drives using pre-processed 64×64 pixel images as input, evolved indirectly in the Fourier domain, and the MPC-RNN agent with just 33 weights in its RNN controller.

Exploration

Reinforcement Learning - Standard Exploration

- Є -greedy
  - Action A is:
    - A* with probability 1 - Є
    - Random Action with probability Є

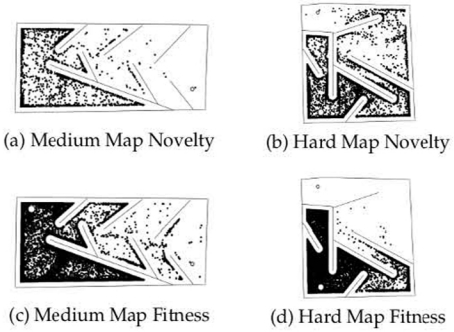
Dyna-Q+ (1990)

Not an evolutionary approach.

- Perhaps one of the first results where an agent is guided by novelty.
- State-action pairs have their interest increased if they are not frequently taken.

Sutton, R. S. (1990). First results with Dyna, an integrated architecture for learning, planning and reacting. Neural Networks for Control, 179-189.

Abandoning Objectives (2011)



(a) Medium Map Novelty (b) Hard Map Novelty

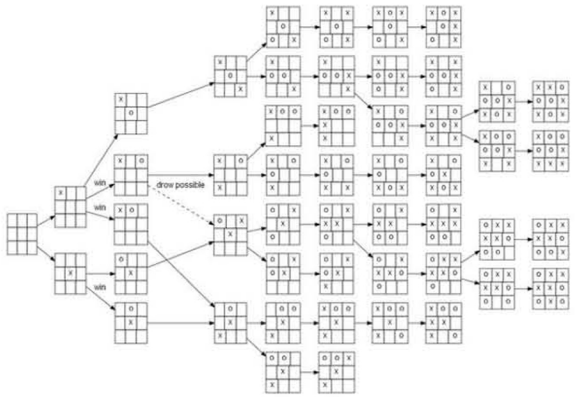
(c) Medium Map Fitness (d) Hard Map Fitness

Using the novelty of a behavior (user defined feature vector) as fitness.

Lehman, J., & Stanley, K. O. (2011). Abandoning objectives: Evolution through the search for novelty alone. Evolutionary computation, 19(2), 189-223.

Alpha Go Zero (2017): Monte Carlo Tree Search + DNN

Not an evolutionary approach.



MCTS is mapping and exploring the environment similar to a novelty search.

The Building Blocks

Building Blocks of ERL: Incomplete Summary

- **Model (Representation of State, Action, World)**
  - Model Generality: should be able to deal with any problem
  - Model Adaptability: should be able to adapt to problem dynamics (problems also change!).
  - Multi-Agent System: break the problem. Many but simple models.
- **Search for Good Policies (Policy Search Approach)**
  - Diversity Procedure: increase exploration and avoid deleterious competition.
  - Indirect Encoding: decrease search space.
- **Evaluate State-Action Pairs (Value Function Approach)**
- **Explore Environment**
  - Curiosity, Novelty detenction, Novelty search
  - Monte Carlo Tree Search