Indirect Encodings of Artificial Neural Networks

Jan Drchal
drchajan@fel.cvut.cz

COMPUTATIONAL
INTELLIGENCE
GROUP

Department of Computer Science and Engineering
Faculty of Electrical Engineering
Czech Technical University in Prague

2012

Overview

- Large-scale Artificial Neural Networks.
- Computational Development.
- Indirect Encodings of ANNs.
- Hyper-cube based encoding.
- Base algorithms.

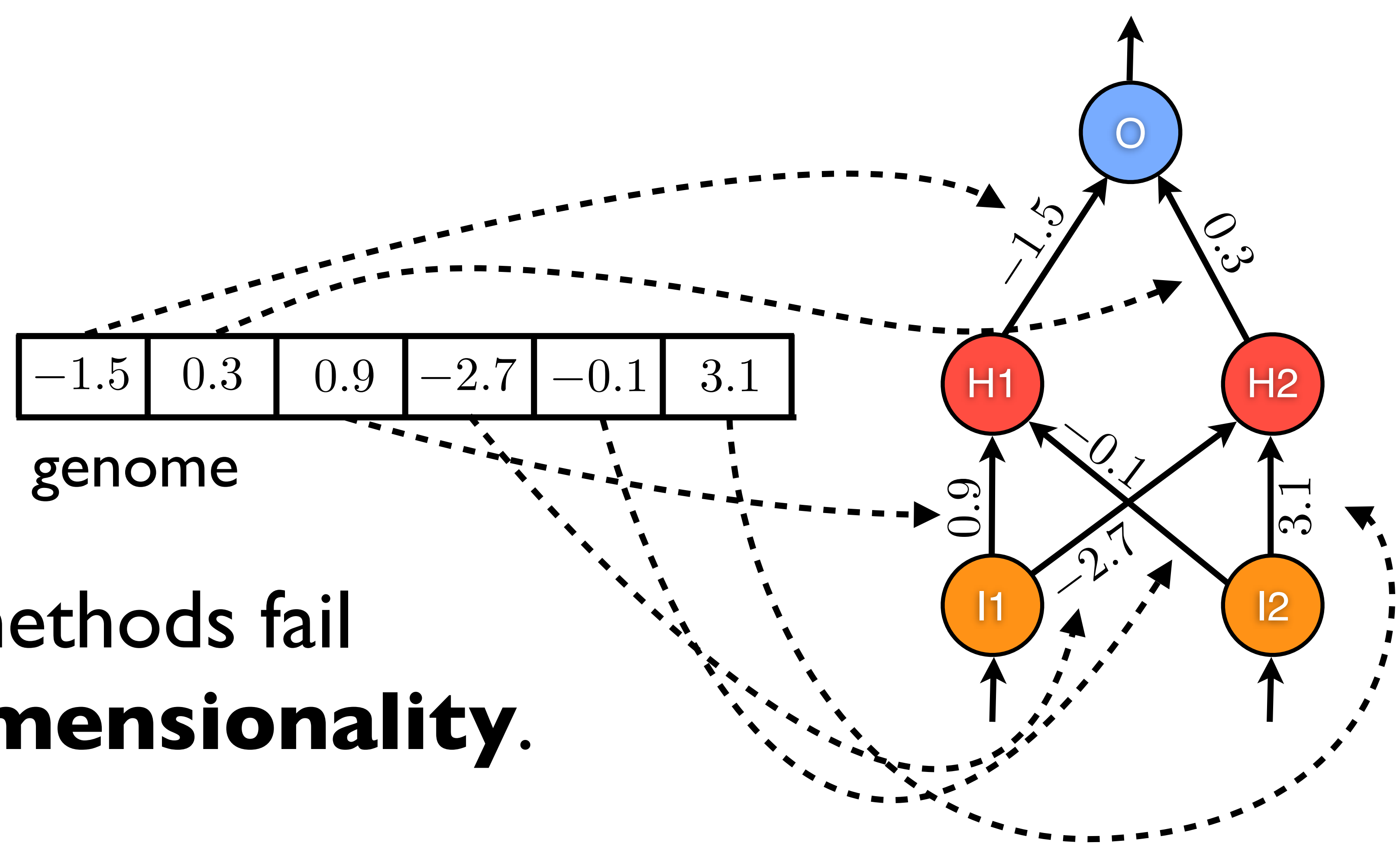
Note: additional material including implementation details, sources, exact parameter settings and detailed results can be found here: http://neuron.felk.cvut.cz/~drchaj1

Evolving Large-scale ANNs

- 1000+ neurons (& corresponding # of links).
- **Why to do that?**
  - Complex models,
  - ability to process huge amount of inputs/outputs without hand-coding features (i.e. pattern recognition)...

Direct Encoding

- **Direct encoding** → each structural part (neuron/link) is represented by a dedicated gene.
- Not suitable for Large-scale ANN's:

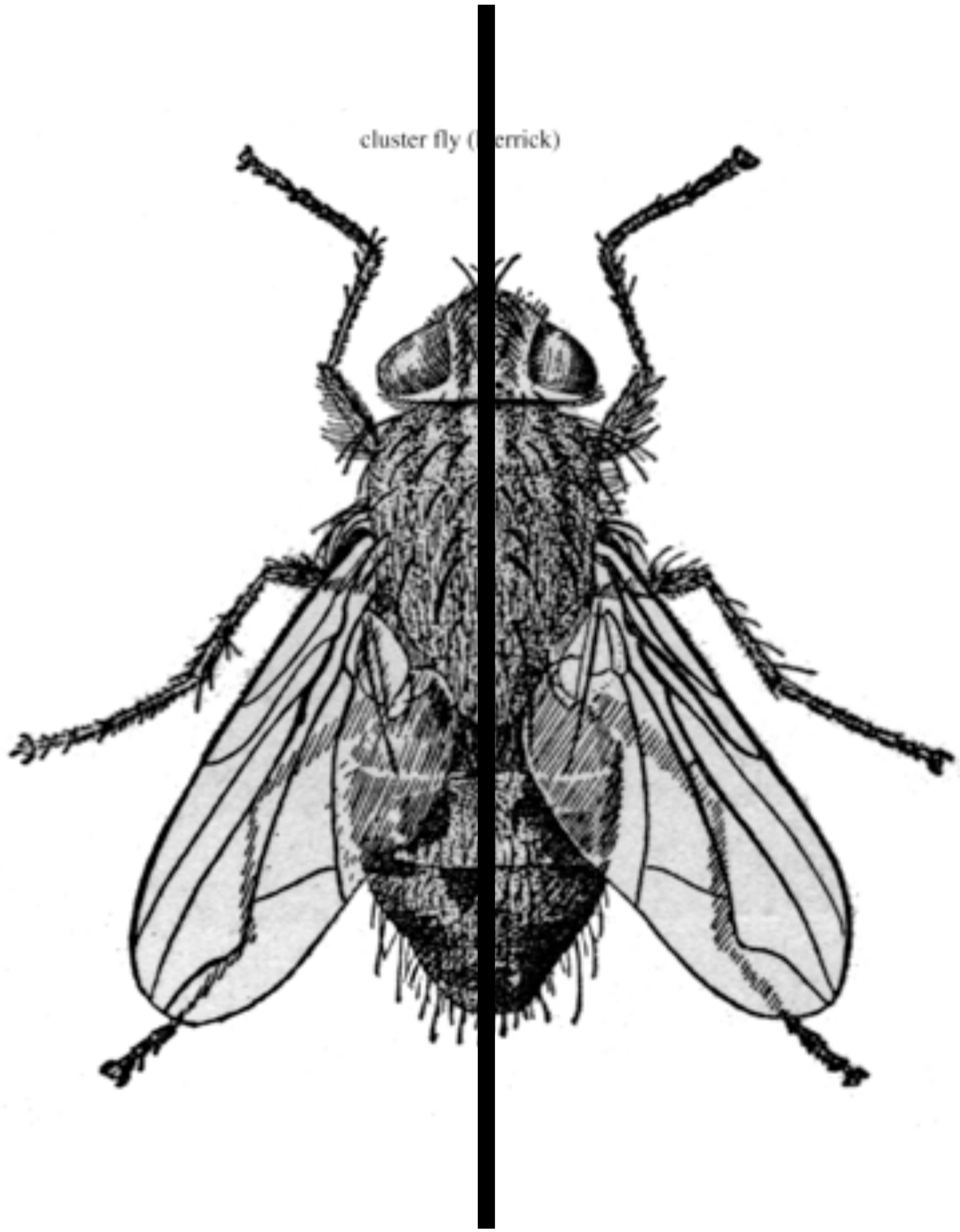


Direct optimization methods fail
→ **the curse of dimensionality**.

Indirect Encoding: the Way it Works in Nature

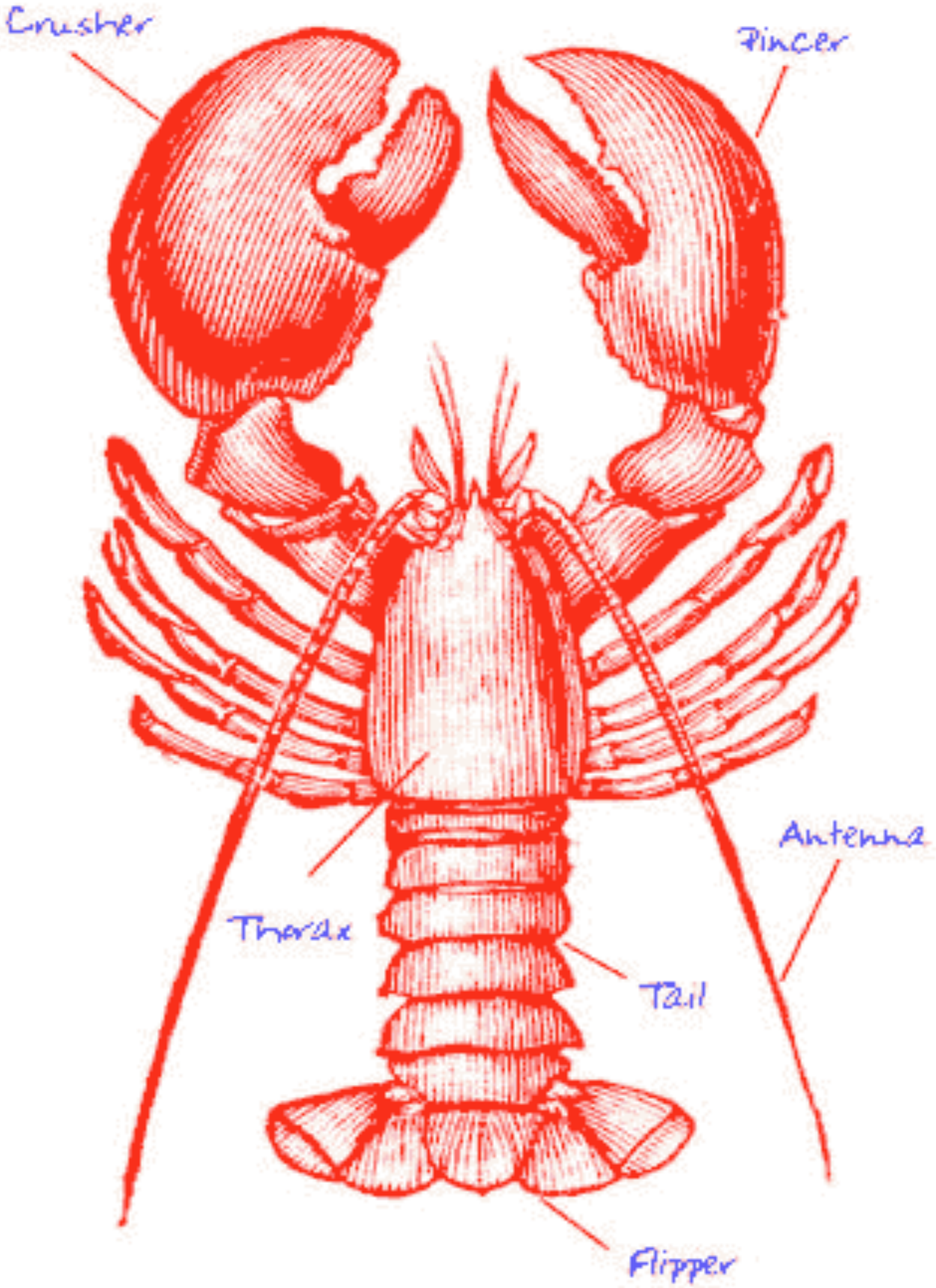
- Human genome → **20 000 - 25 000 genes describing almost 100 billion neurons each linked to as many as 7 000 others** (plus the rest of organism!).
- We need some kind of **compression**:
  → **indirect encoding.**
- But we also need a **regularity** in data being compressed.
- **Q:** What are the regularities found in living organisms?

Symmetry



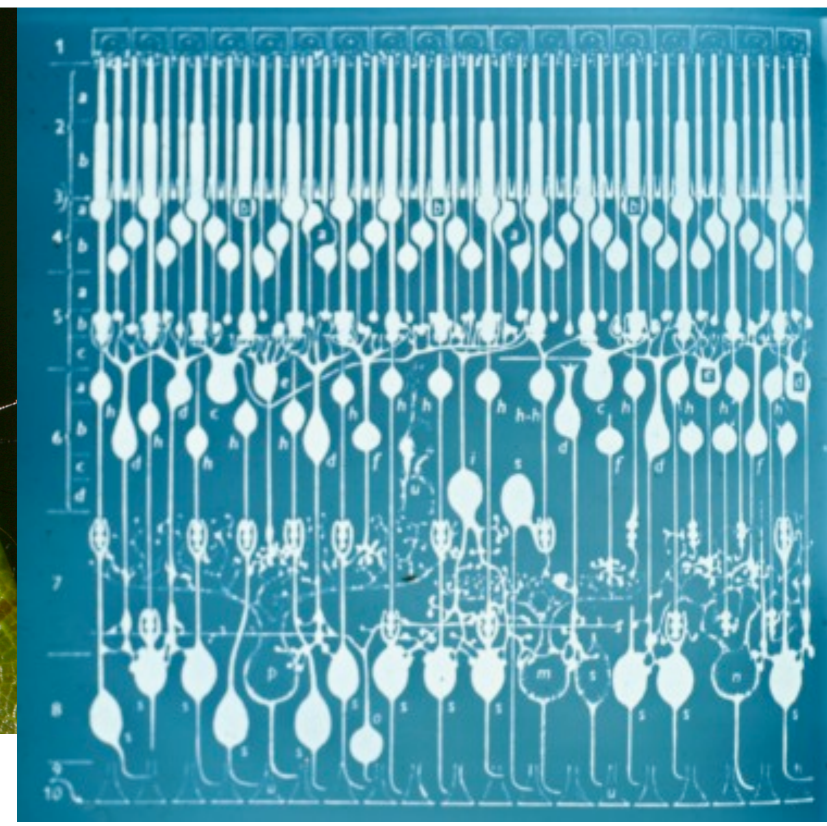
(wikimedia commons)

Imperfect Symmetry



(wikimedia commons)

Repetition with Variation

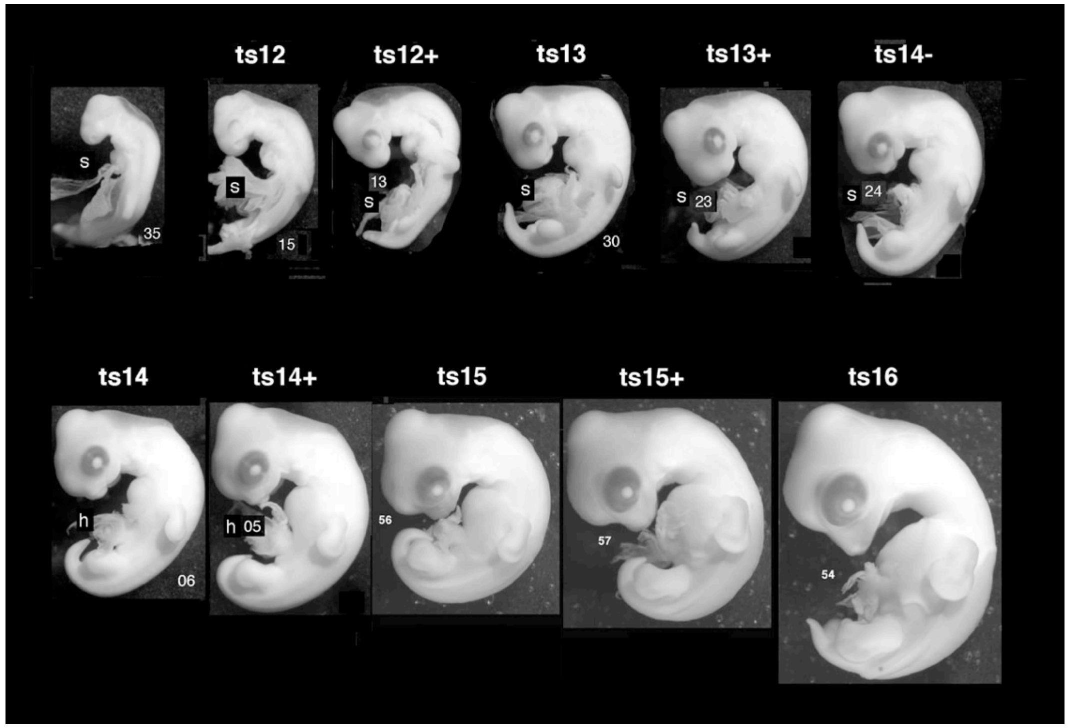


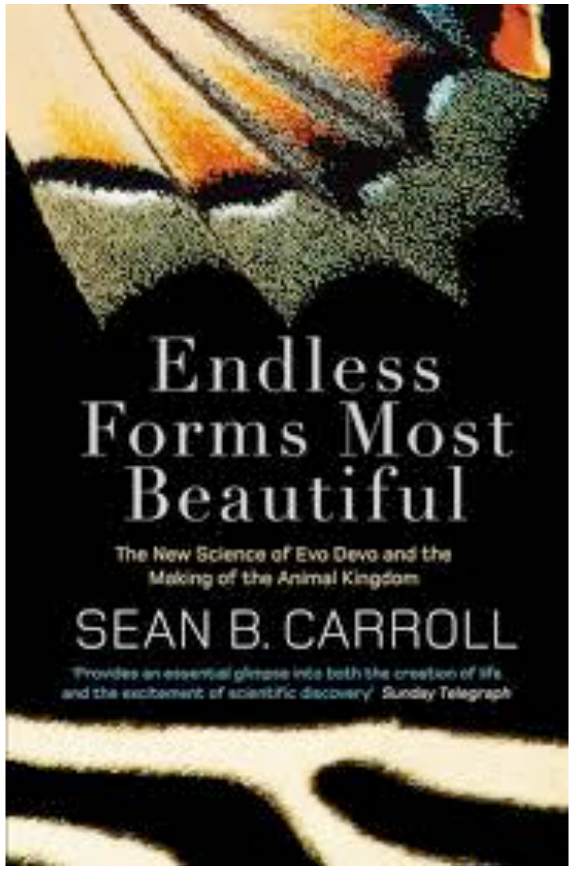
- Note that all these regularities happen at **all scales** of an organism.

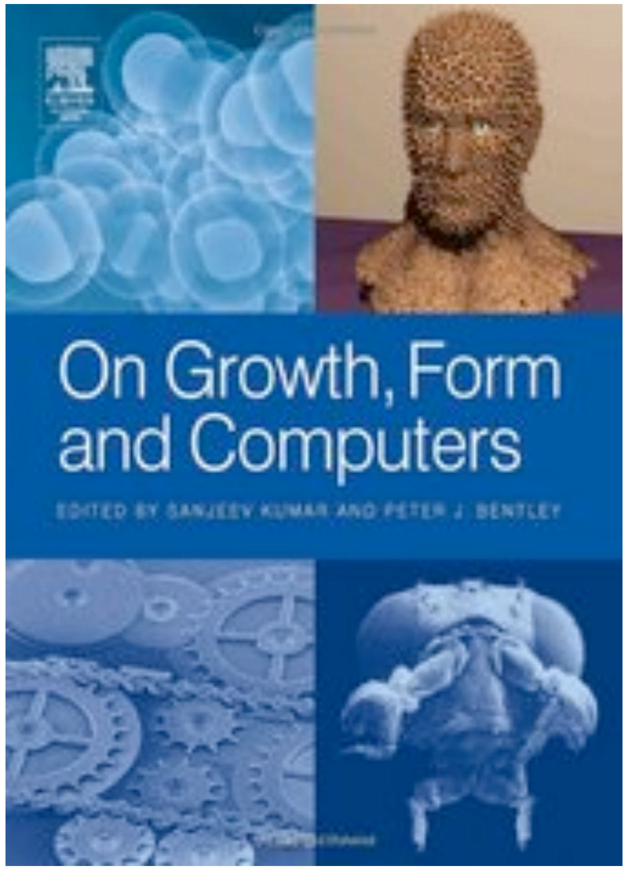
(wikimedia commons)

How Are Organisms Built?

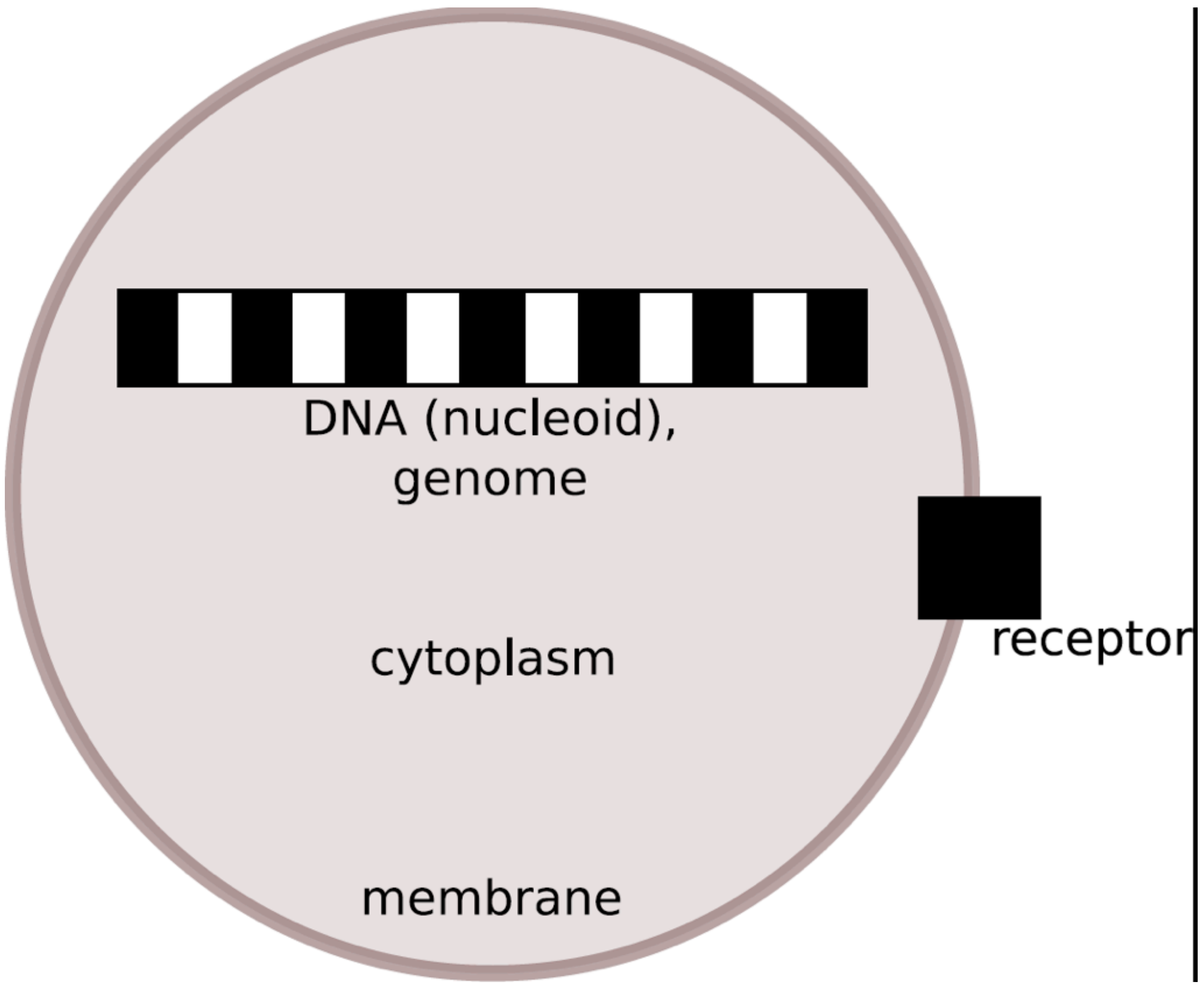
- Development from a single cell (zygote).
- Evolutionary Development "Evo-Devo".



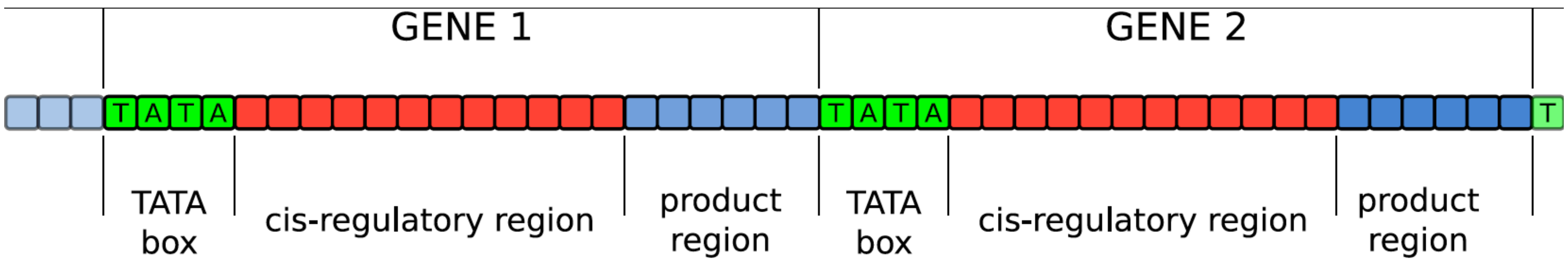




The Cell

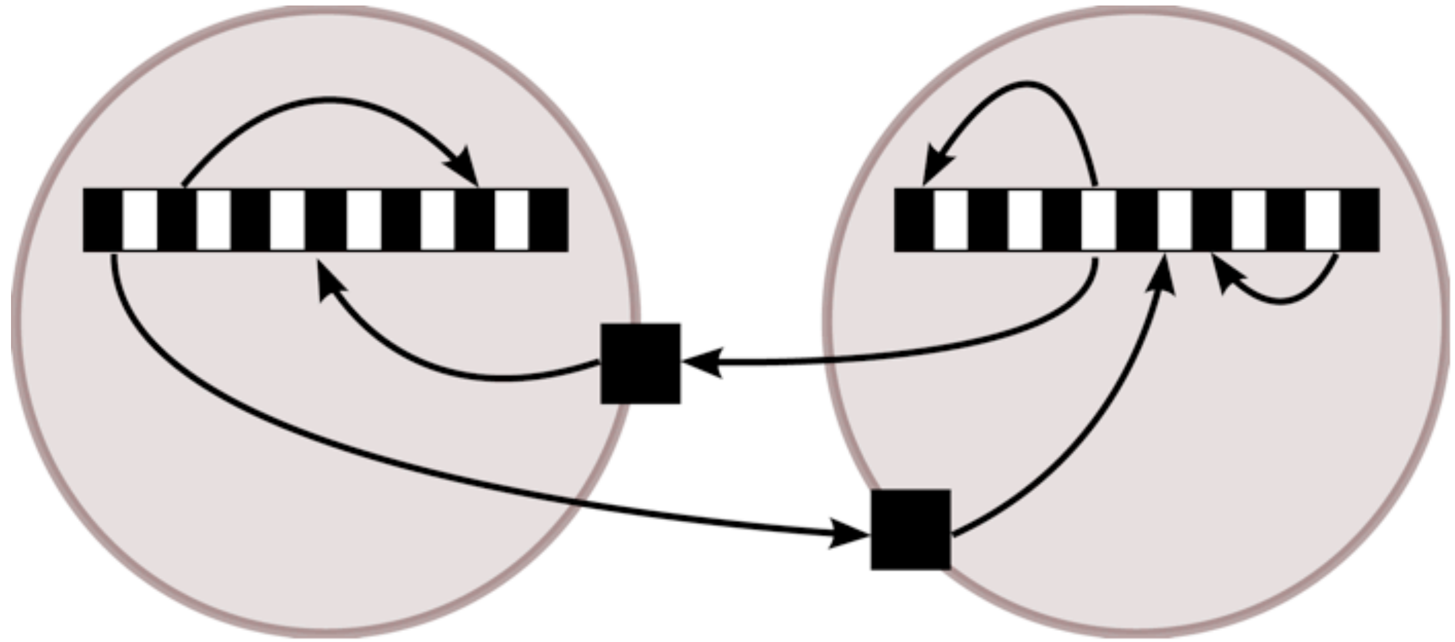


Genome: A Closer Look



- *TATA box* – marks the start of a gene
- *(cis-)regulatory region* – composed of binding sites.
- *binding site* – binds regulatory proteins → gene activation/ inhibition
- *product region* – when gene is active a protein is produced:
- *special*: cell division, differentiation,
- *regulatory*: can bind to binding sites of other genes,
- *structural*.

Cell Divisions



- Program same for all cells.
- What differs?
  - Regulatory protein *concentrations*.
- *Receptors* – selectively pass regulatory proteins from inter-cellular space.
- Diffusion, decay, cell differentiation.
- Gene Regulatory Networks (GRNs).

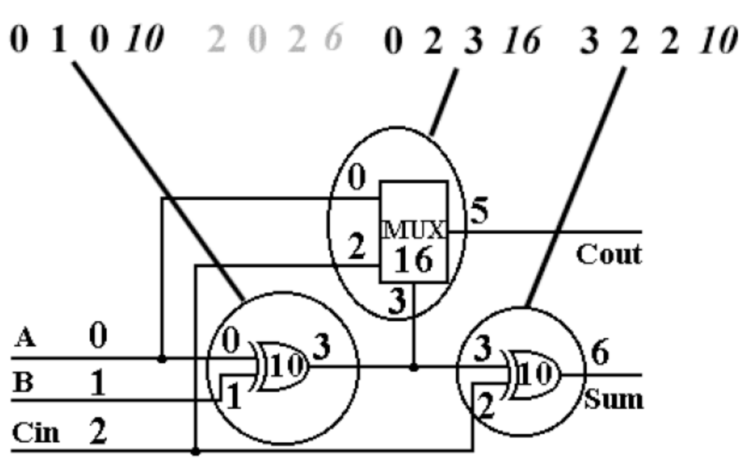
How to Simulate Development?

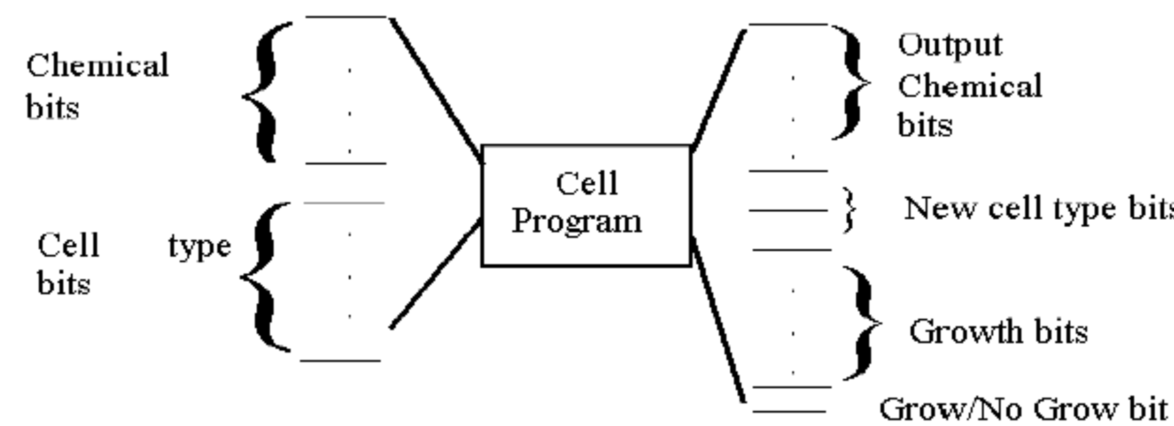
- Cell program – ANN, FSM or other controller:
  - *inputs*: binding sites,
  - *outputs*: one for each gene → gene activity.
- *Physical simulation*: diffusion, decay, receptors...
- Cell division:
  - *copy cell program* from mother → daughter cell,
  - *different concentrations* for mother/daughter.
- This is called: *Computational Development*.

"French Flag" Organism

- Cell program evolved using Cartesian Genetic Programming (CGP).

CGP encoded adder





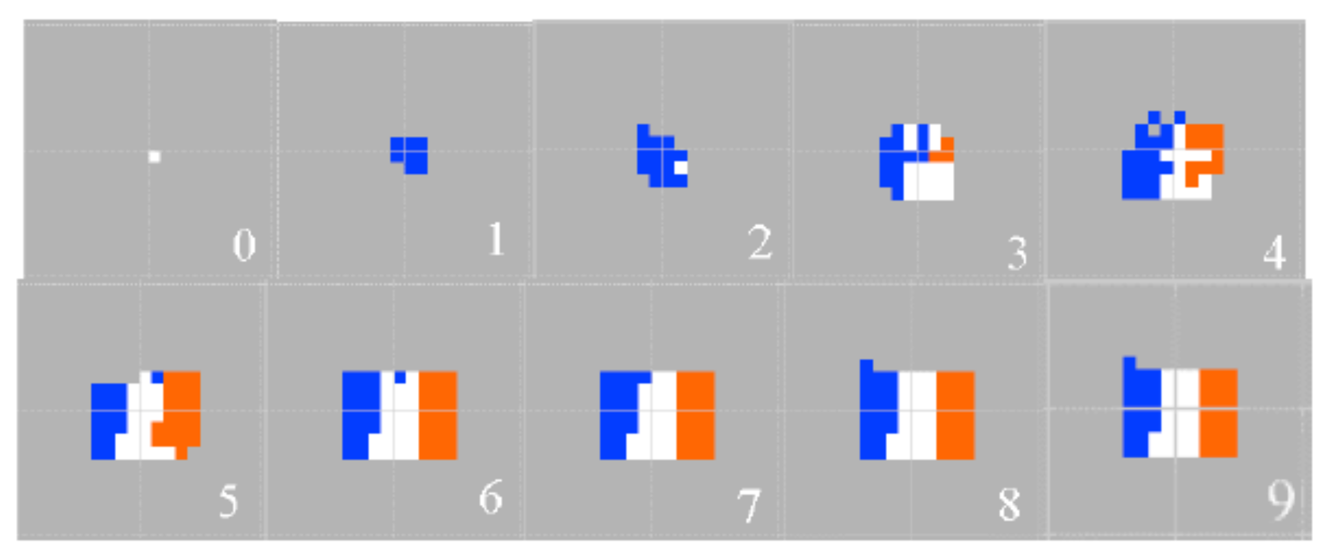


Fig. 4. Growth of fittest cell program from a white seed cell to a mature French flag (two chemicals)

Julian Francis Miller (2004):
Evolving a Self-Repairing, Self-Regulating, French Flag Organism

“French Flag” Organism II

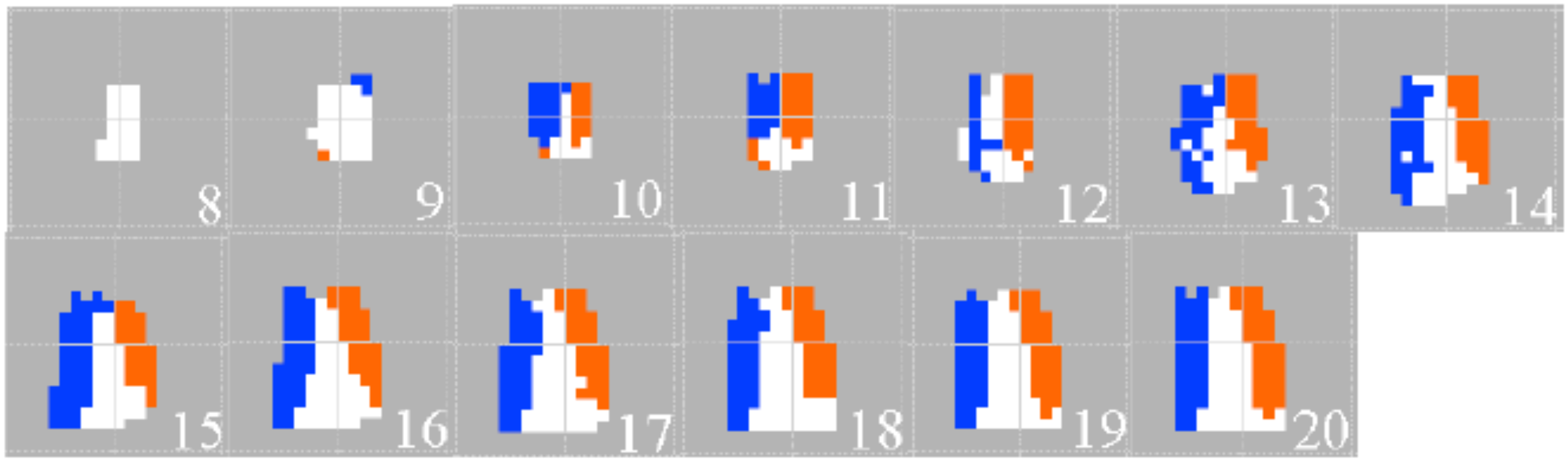


Fig. 7. Autonomous recovery of badly damaged French flag organism conditions (blue and red regions killed at iteration 8 - see Fig. 4). There is no further change after iteration 20

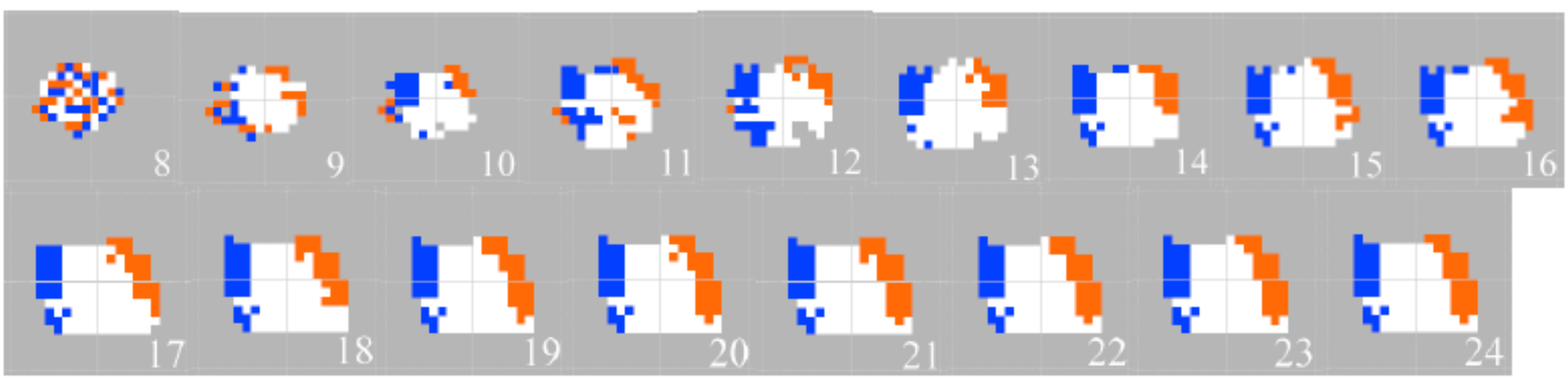
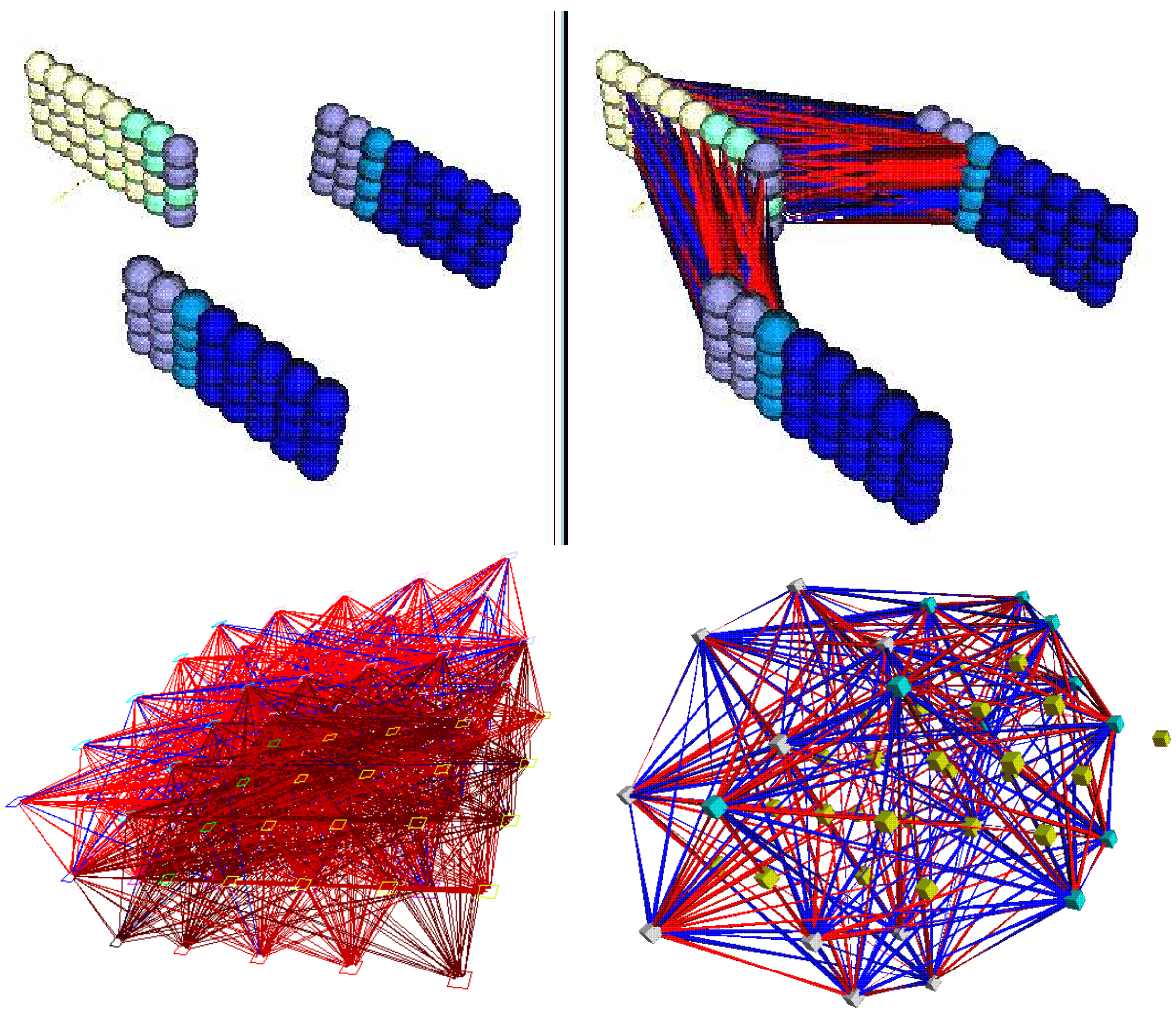


Fig. 8. Autonomous recovery of French flag from randomly rearranged cells (French flag at iteration 8 - see Fig. 4). There is no further change after iteration 24

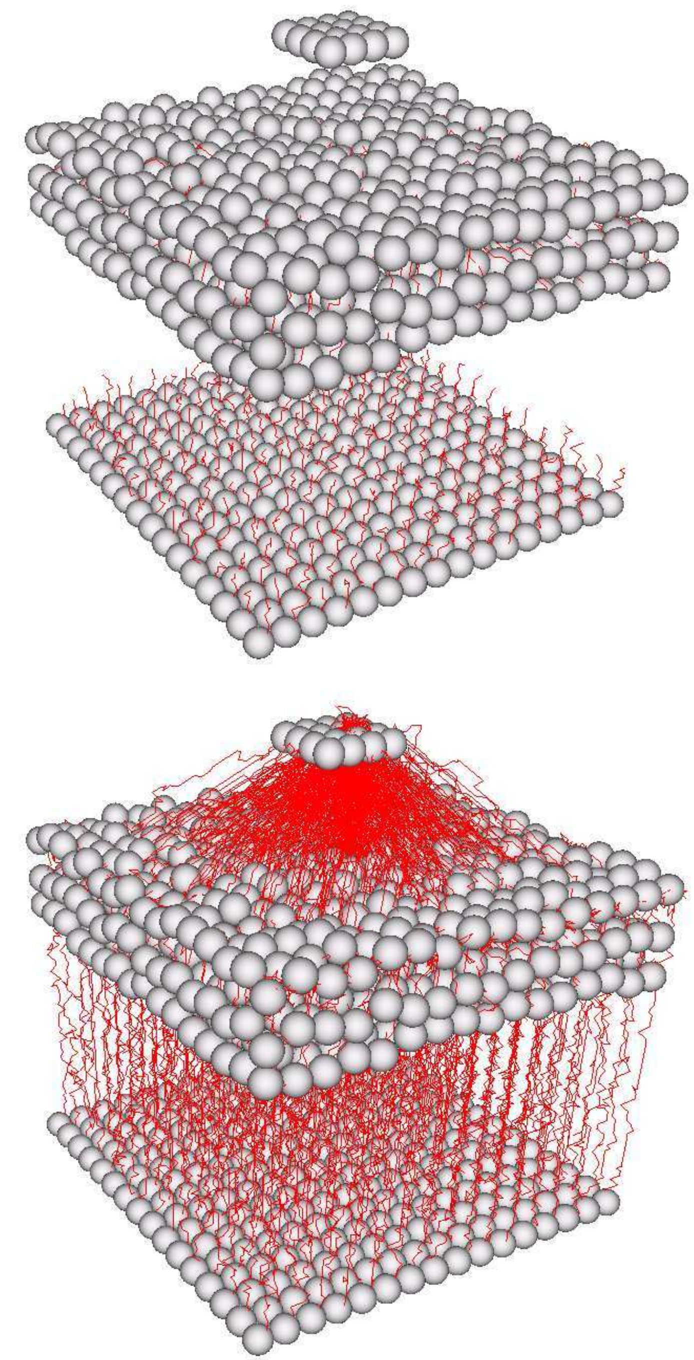
Indirect encodings of ANNs

- GRN-based
- Cellular Encoding
- Hypercube-based
- Other: rewriting rules, L-systems, ...

GRN-based



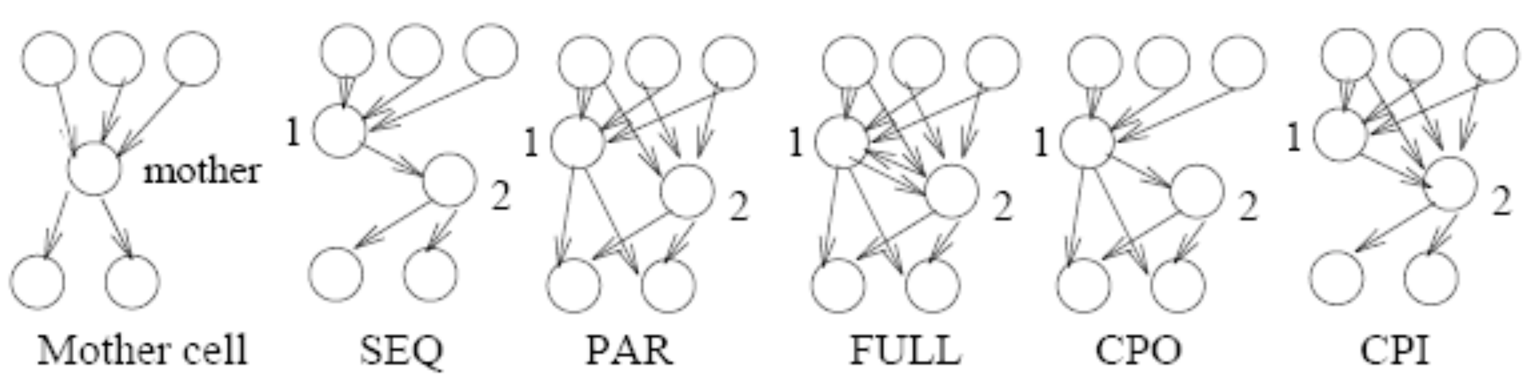
Peter Eggenberger-Hotz (1997):
Creation of Neural Networks Based on Developmental and Evolutionary Principles

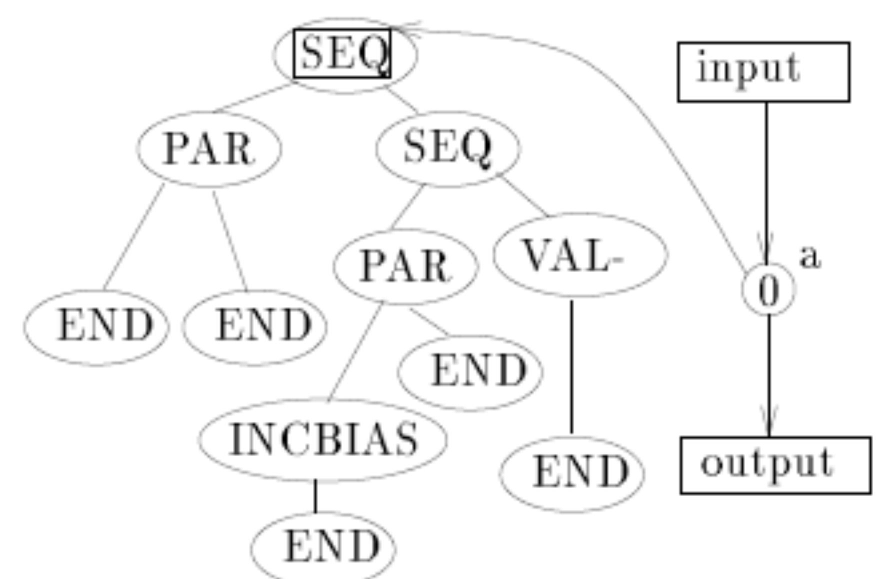


Peter Eggenberger-Hotz (2003):
Evolving the Morphology of a Neural Network for Controlling a Foveating Retina and its Test on a Real Robot

Cellular Encoding (CE)

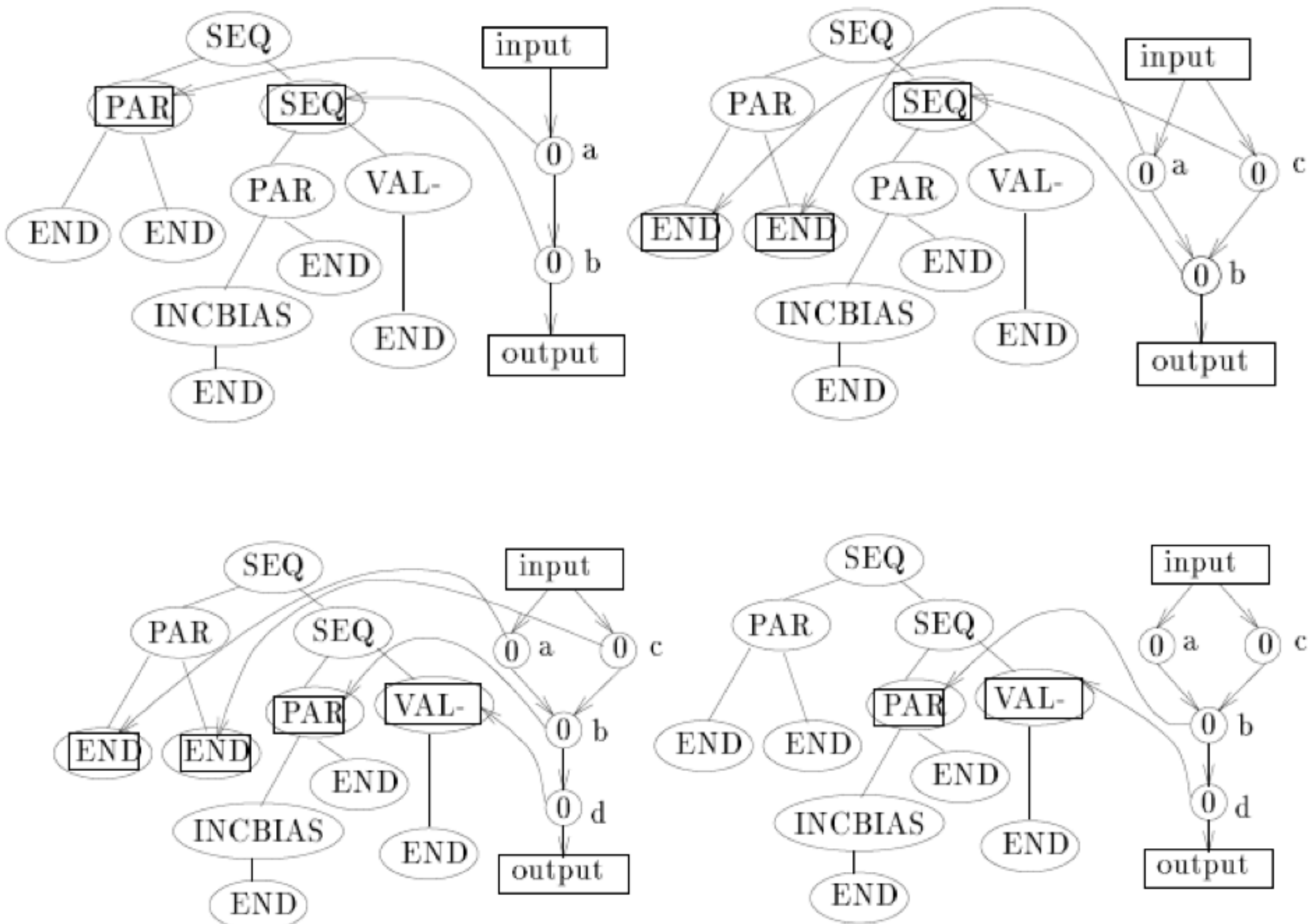
- 1993, Fréderic Gruau: indirect encoding example.
- Inspiration in embryo-genesis (cell division and differentiation). Cells → neurons.
- Program to "grow" ANN is represented by a tree (Genetic Programming).
- Operations: parallel/sequential divisions, connections change, change of weights/bias...





COMPUTATIONAL
INTELLIGENCE
GROUP

Cellular Encoding II

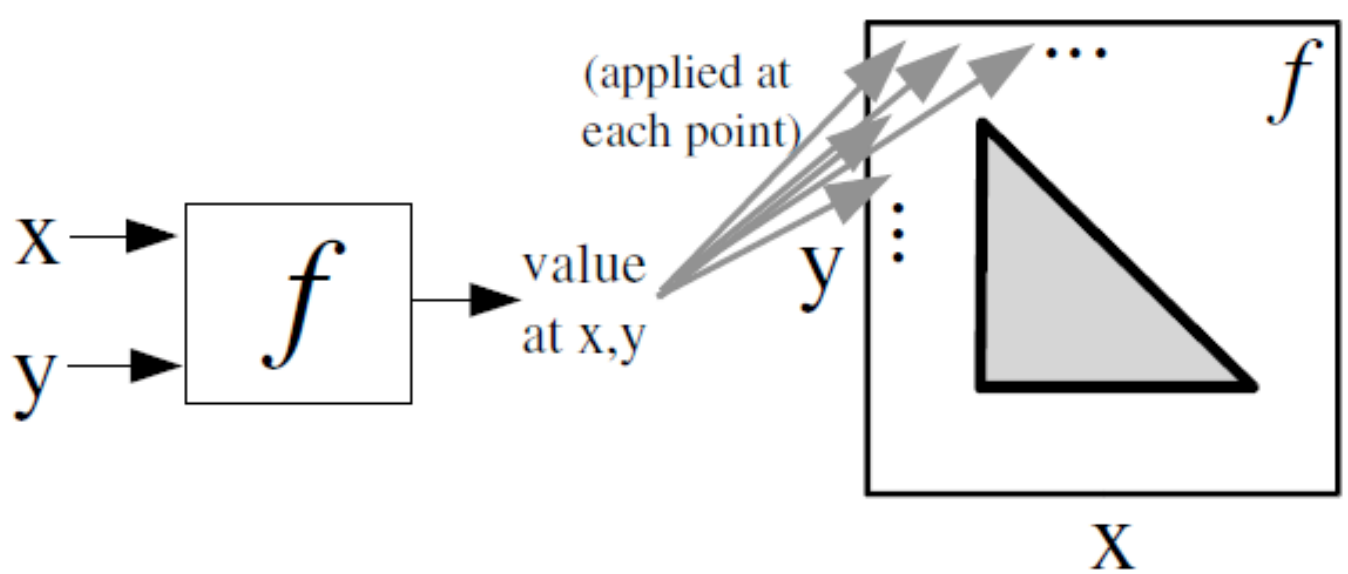


Cellular Encoding III

- May use operation which reads a sub-tree repeatedly → evolved a network representing parity of arbitrary number of inputs.
- Allows ANNs of arbitrary size: *neural module reuse*.

Compositional Pattern Producing Networks (CPPNs)

- Stanley 2006.
- **Can we create such regular patterns without development in time?**
- We can ask a special function called CPPN, where the cells are, using absolute coordinates.



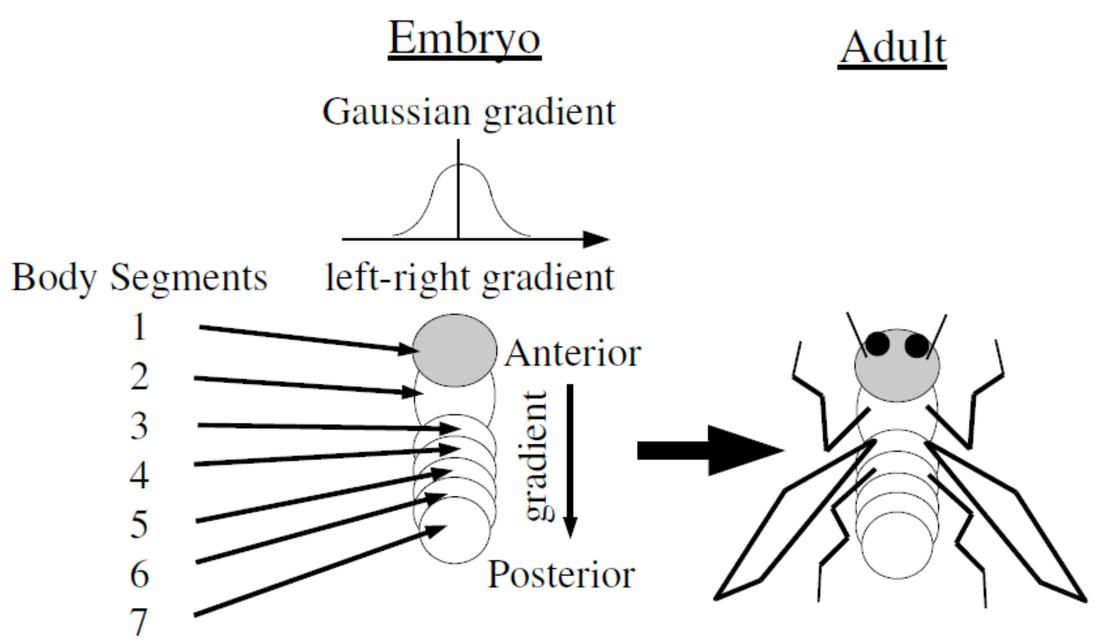
COMPUTATIONAL
INTELLIGENCE
GROUP

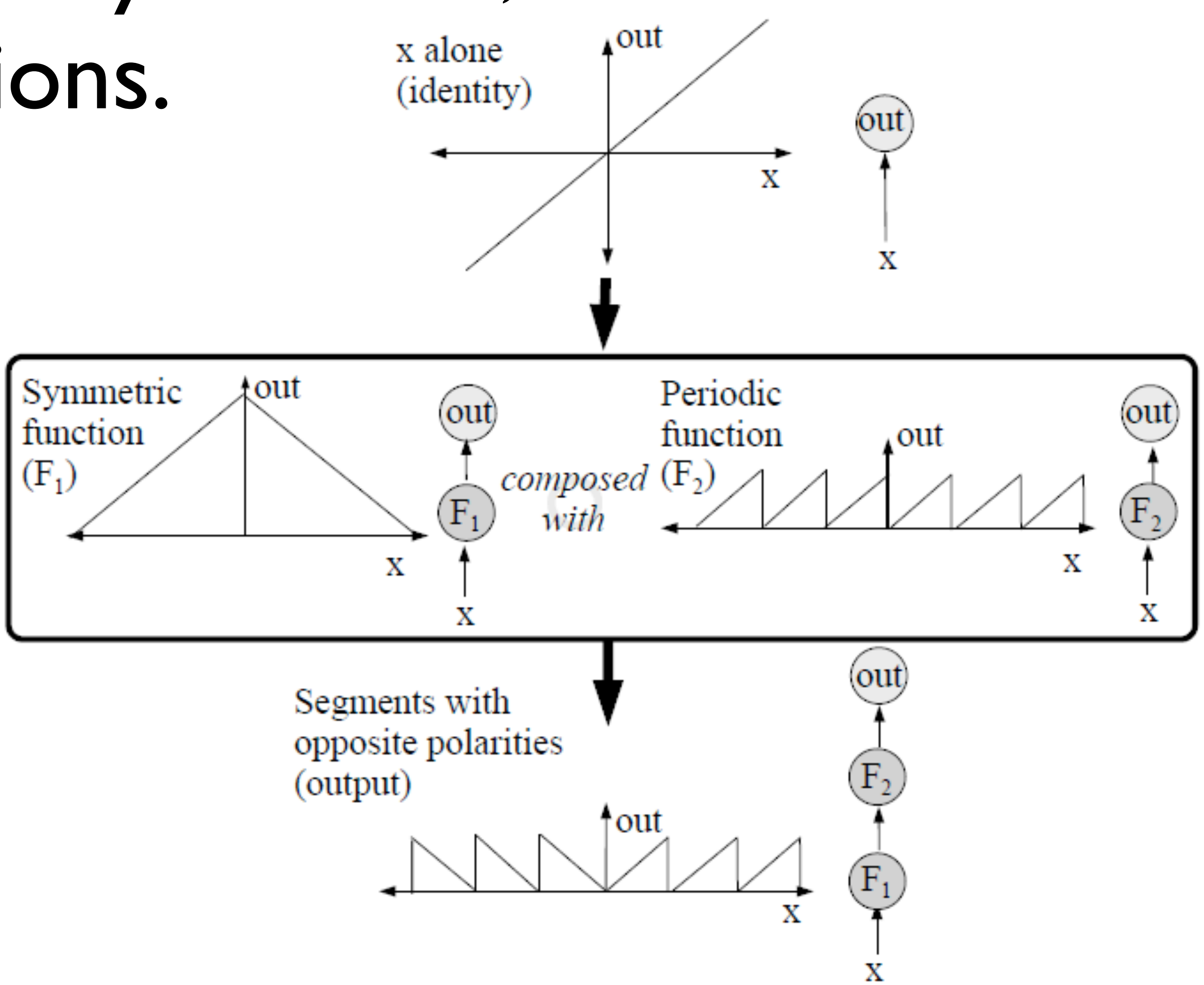
Kenneth O. Stanley (2006):
Compositional Pattern Producing Networks: A Novel Abstraction of Development

2012

Regularities by CPPN

- Nature uses concentration gradients of regulatory proteins to determine position.
- CPPN is a composition of symmetric, periodic and other functions.

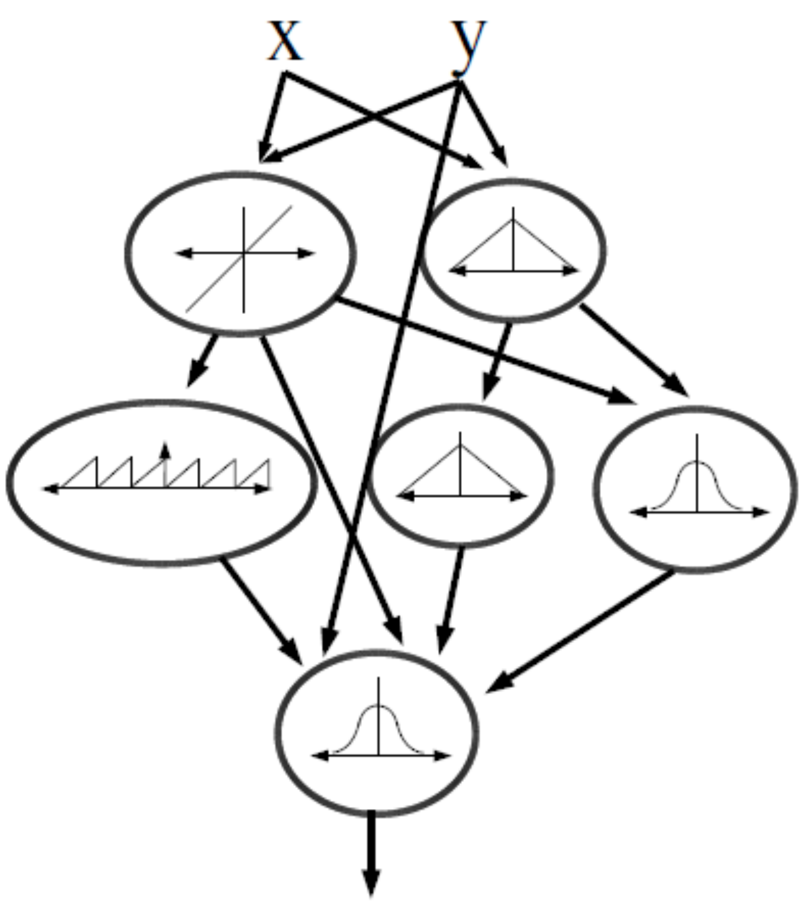




COMPUTATIONAL
INTELLIGENCE
GROUP

Regularities by CPPN II

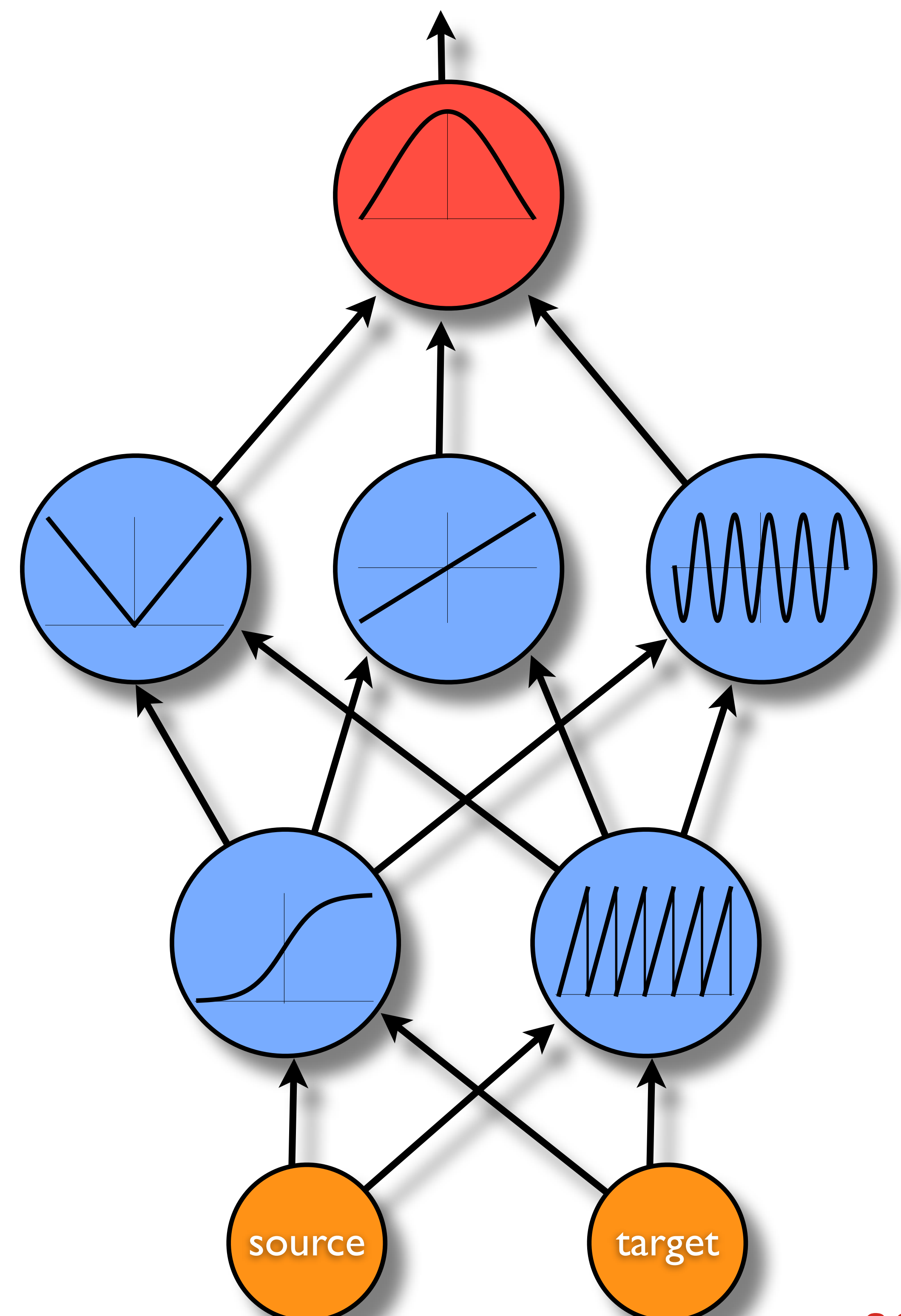
- **CPPN is a composition** of symmetric, periodic and other functions.



Name	Equation
Bipolar Sigmoid	$\frac{2}{1+e^{-4.9\,x}} - 1$
Linear	x
Gaussian	$e^{-2.5\,x^2}$
Absolute value	$\lvert x \rvert$
Sine	$sin(x)$
Cosine	$cos(x)$

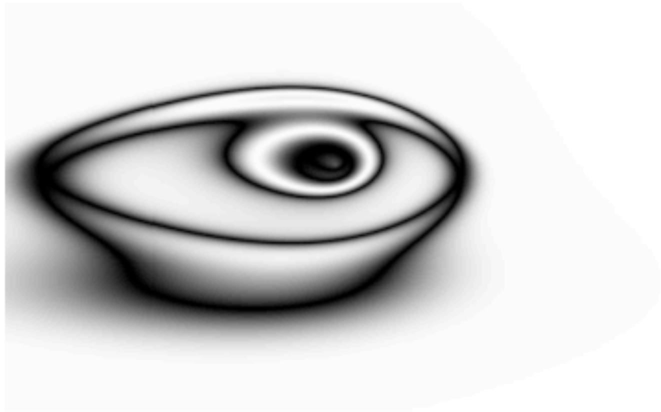
CPPNs in HyperNEAT

- **Compositional and Pattern Producing Network (CPPN).**
- CPPN is a **composition** of symmetric, periodic and other functions.
- In HyperNEAT it has a form of artificial neural network with heterogenous neuron types.

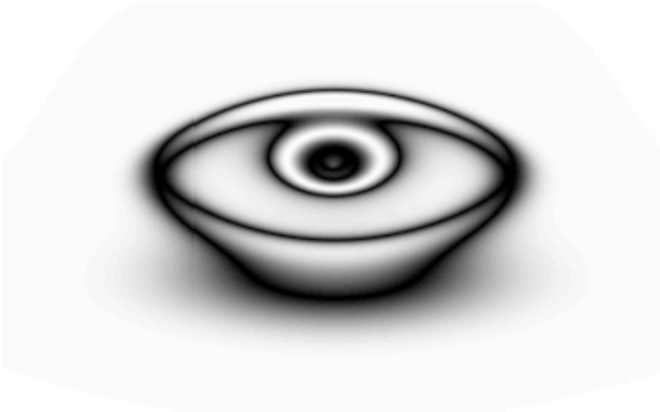


Picbreeder

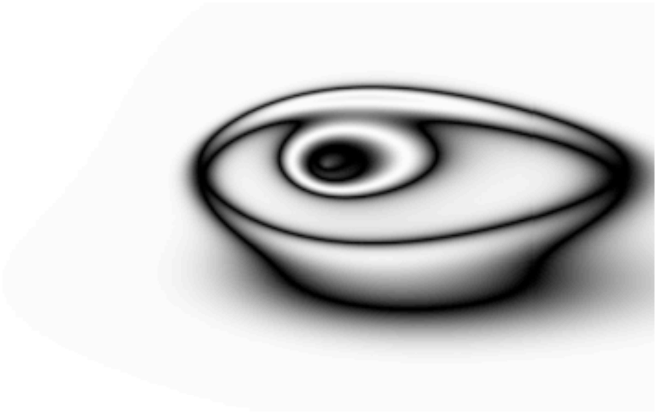
- **Interactive evolution** of images.
- CPPN output: level of grey.
- CPPNs evolved using NEAT.
- http://picbreeder.org/



(a) Eye warped left



(b) Symmetric eye

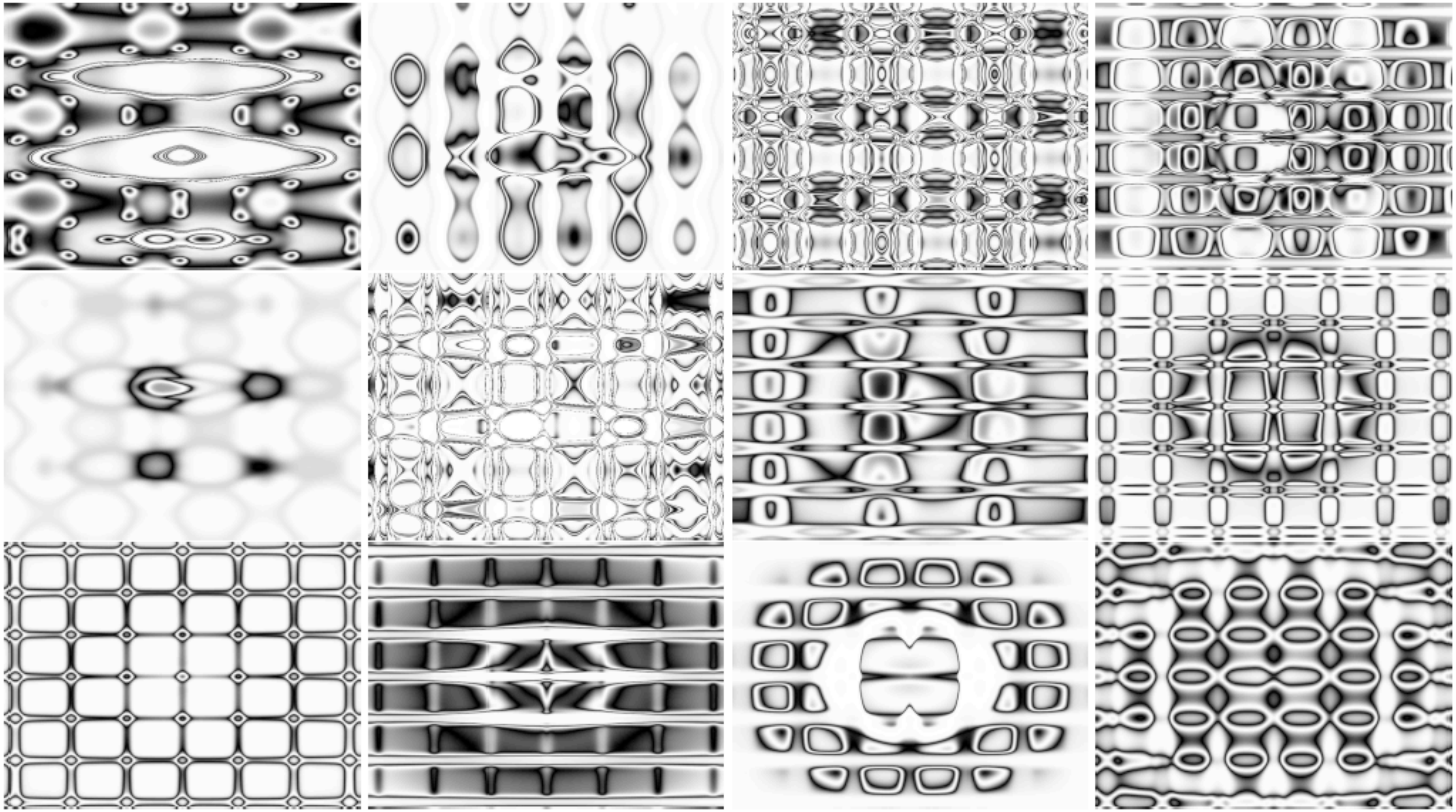


(c) Eye warped right

COMPUTATIONAL INTELLIGENCE GROUP

K. O. Stanley. Compositional pattern producing networks: A novel abstraction of development. *Genetic Programming and Evolvable Machines Special Issue on Developmental Systems*, 2007. To appear.

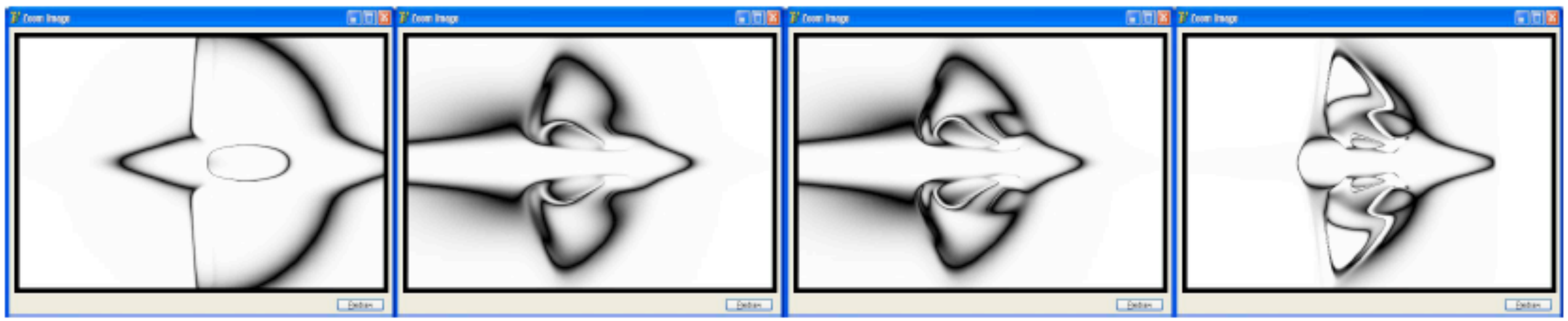
Picbreeder II



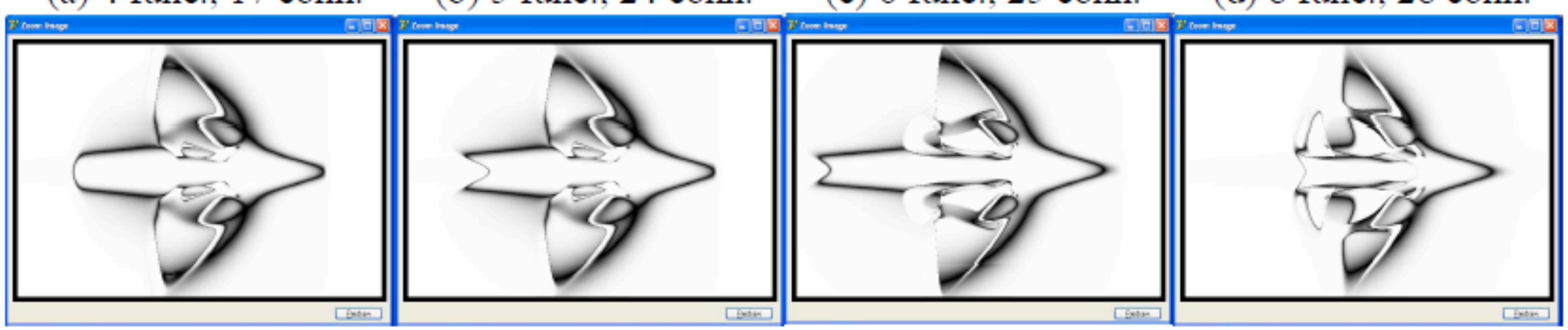
COMPUTATIONAL
INTELLIGENCE
GROUP

2012

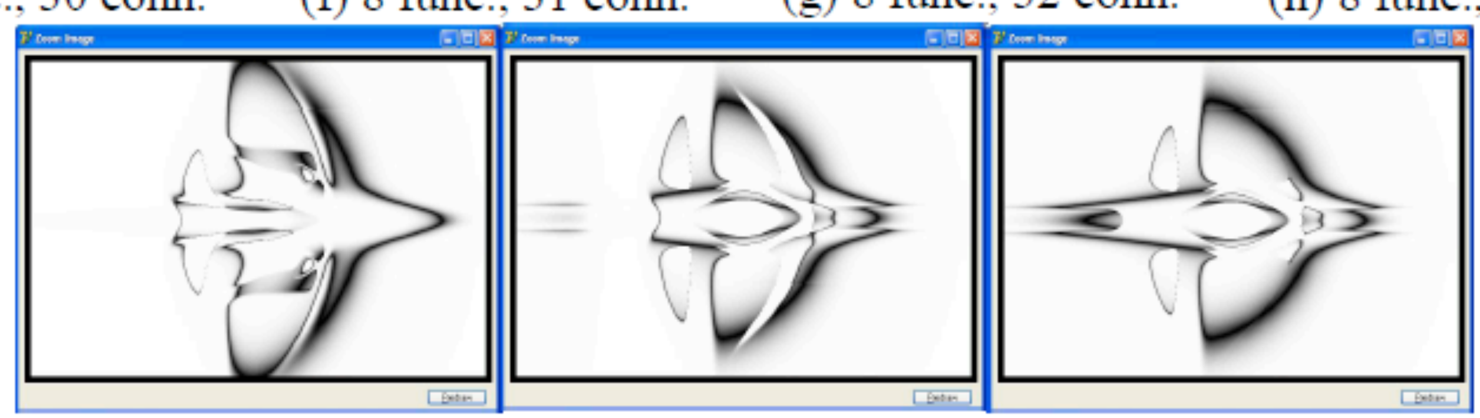
Picbreeder: Space Ship



(a) 4 func., 17 conn. (b) 5 func., 24 conn. (c) 6 func., 25 conn. (d) 8 func., 28 conn.



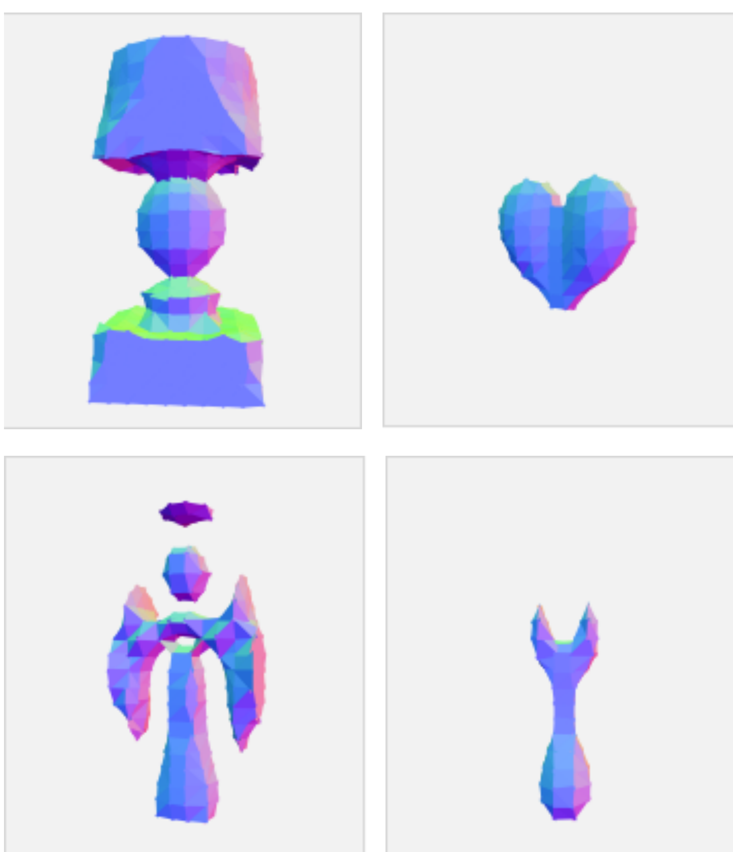
(e) 8 func., 30 conn. (f) 8 func., 31 conn. (g) 8 func., 32 conn. (h) 8 func., 34 conn.

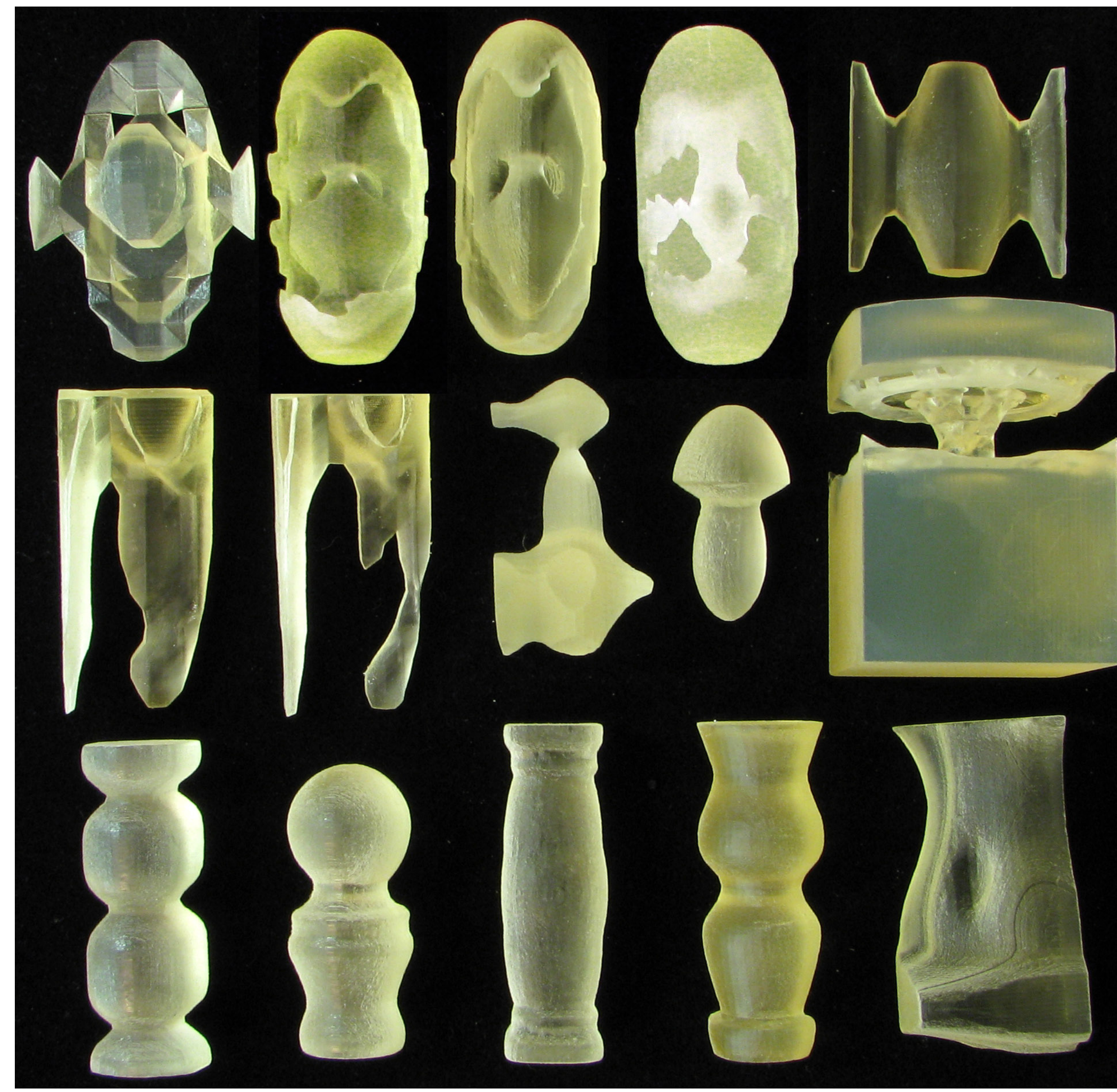


(i) 8 func., 36 conn. (j) 9 func., 36 conn. (k) 9 func., 38 conn.

Endless Forms

- Similar approach in 3D.
- http://endlessforms.com





COMPUTATIONAL INTELLIGENCE GROUP

Jeff Clune, Hod Lipson (2011):
Evolving Three-Dimensional Objects with a Generative Encoding Inspired by Developmental Biology

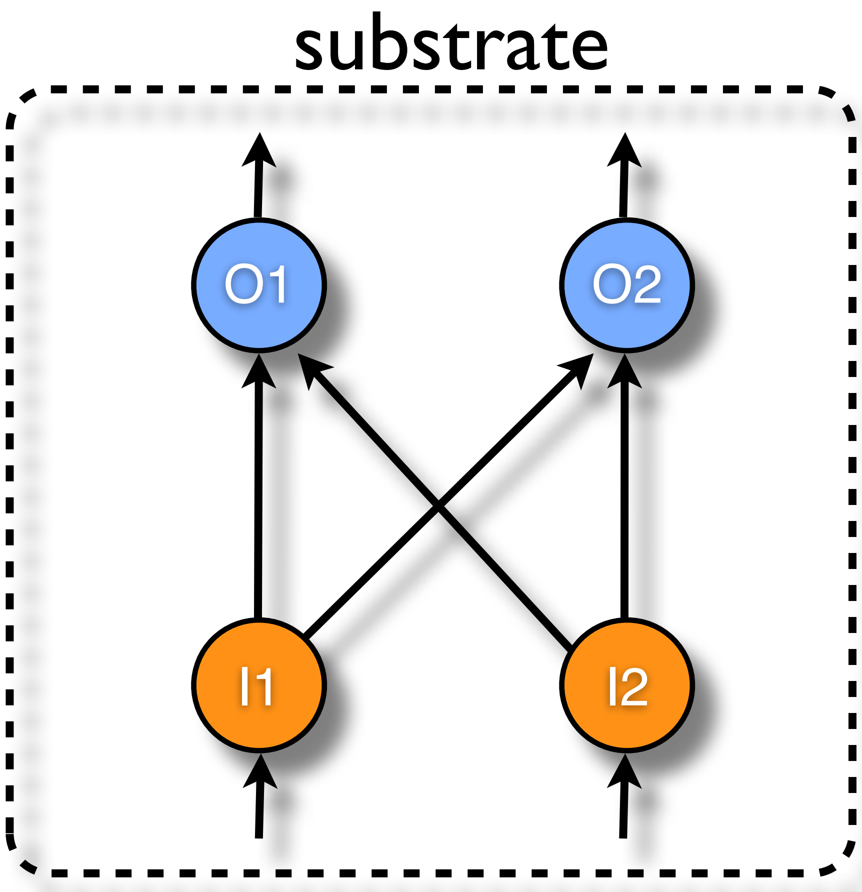
Hypercube-based Encoding

- Stanley 2007.
- Uses CPPNs in a similar way to Picbreeder: evolves **connectivity patterns**.
- Best known for **HyperNEAT** algorithm which evolves ANNs.

HyperNEAT

- Stanley et al. 2007: Hypercube-based encoding.

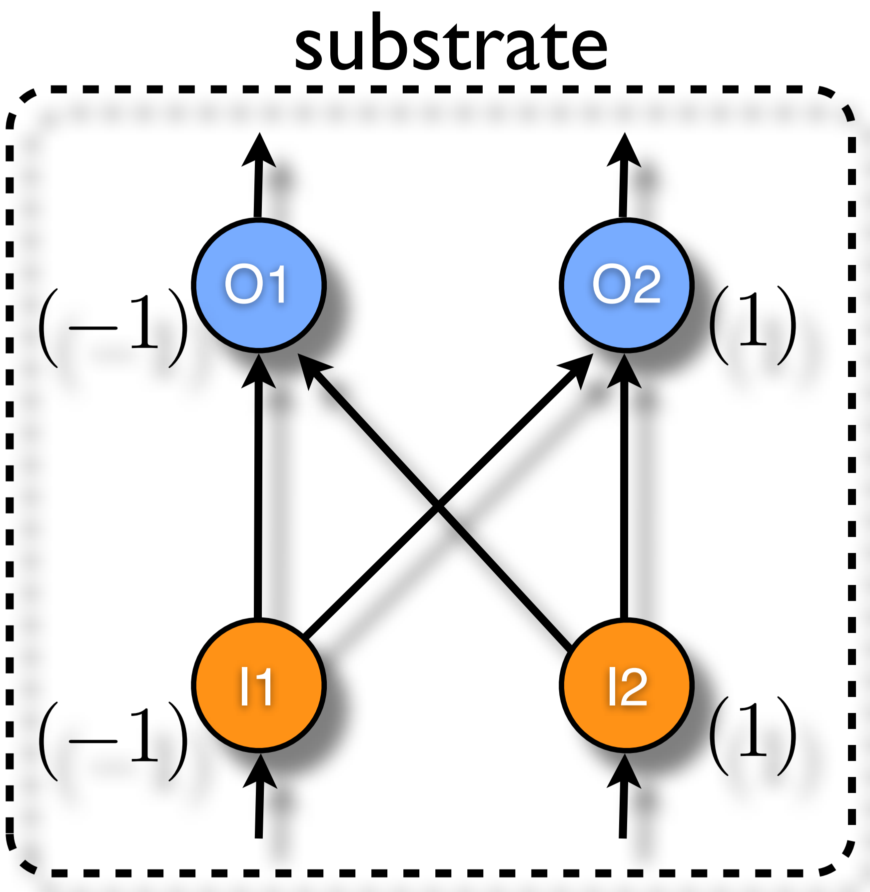
Substrate is a template for a possibly large-scale neural network.



HyperNEAT

- Stanley et al. 2007: Hypercube-based encoding.

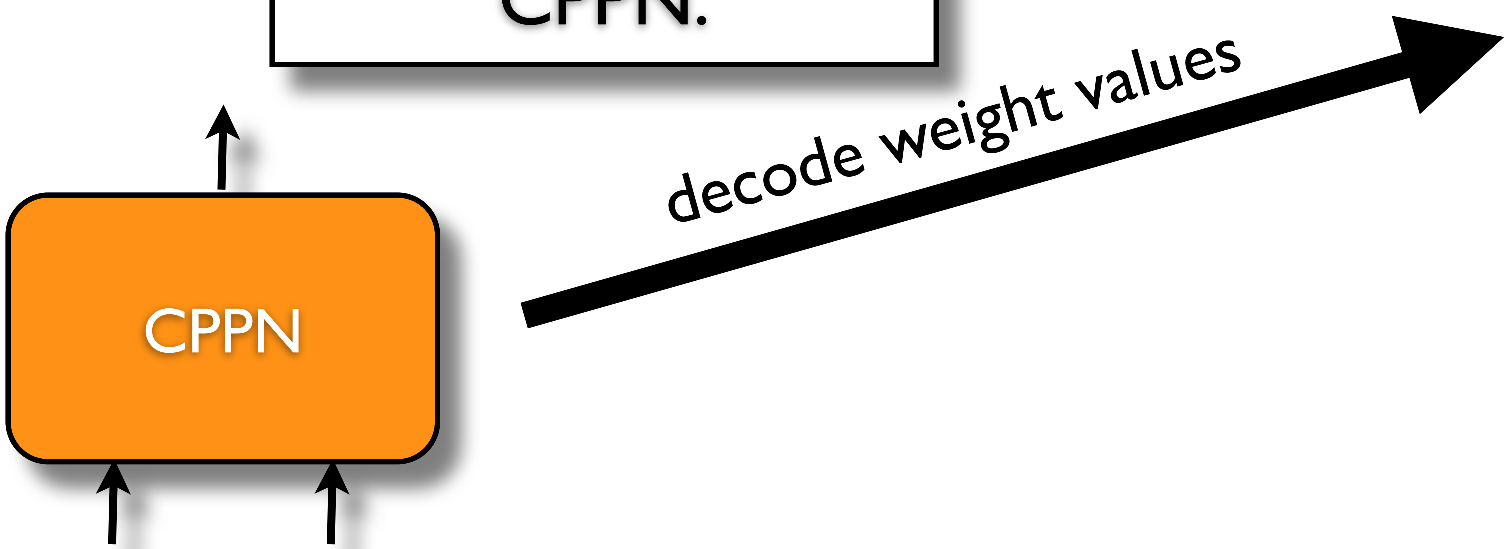
Each neuron is assigned coordinates. The weights of connections are unknown.

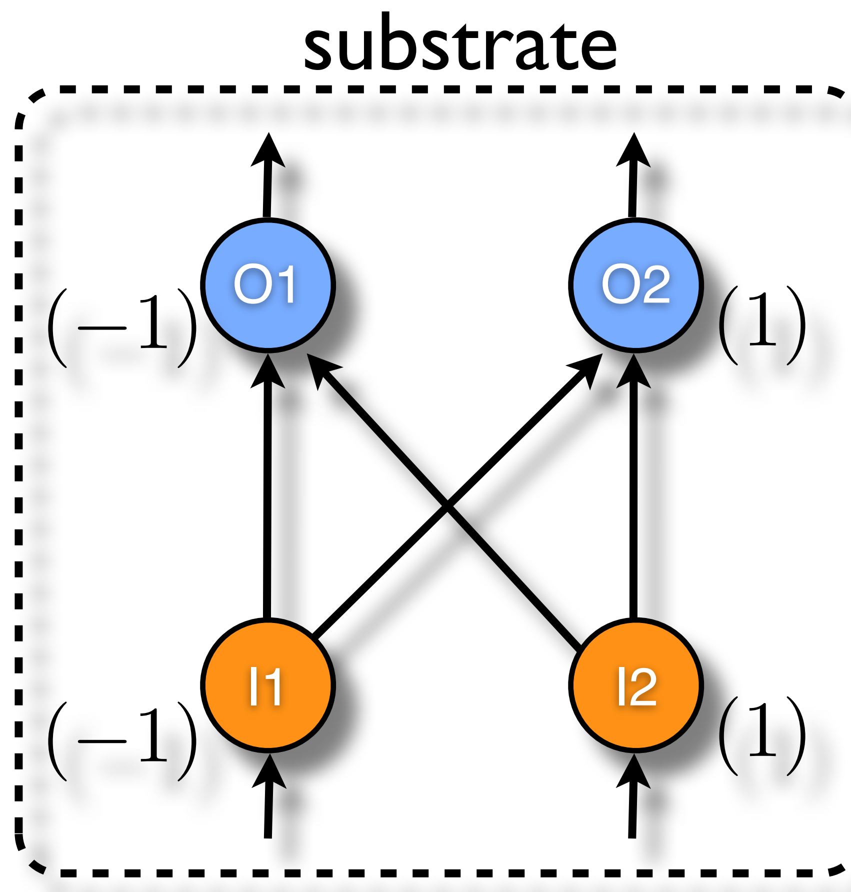


HyperNEAT

- Stanley et al. 2007: Hypercube-based encoding.

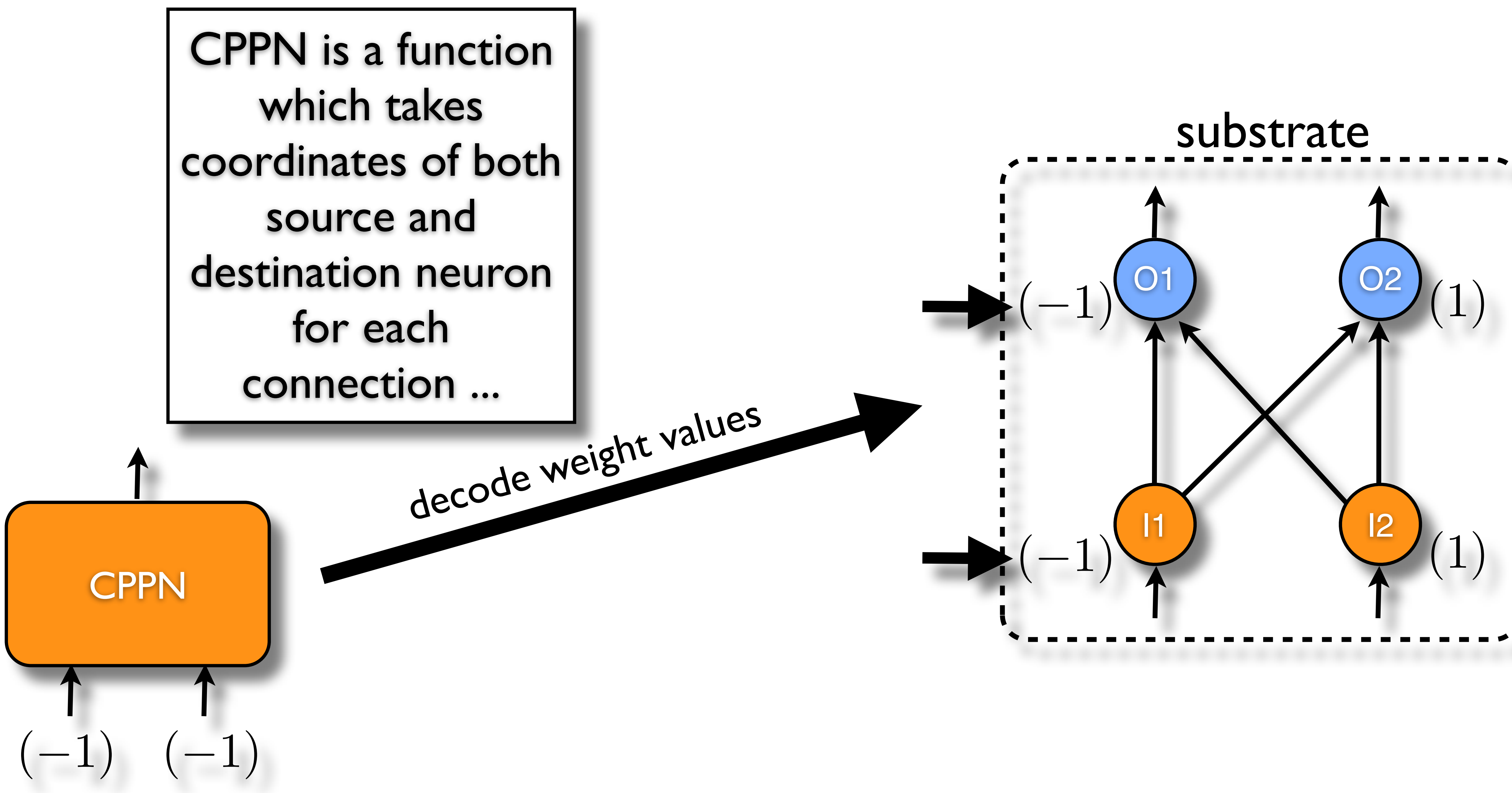
The *final network* is constructed out of *substrate* by computing all needed weights. This is done using CPPN.





HyperNEAT

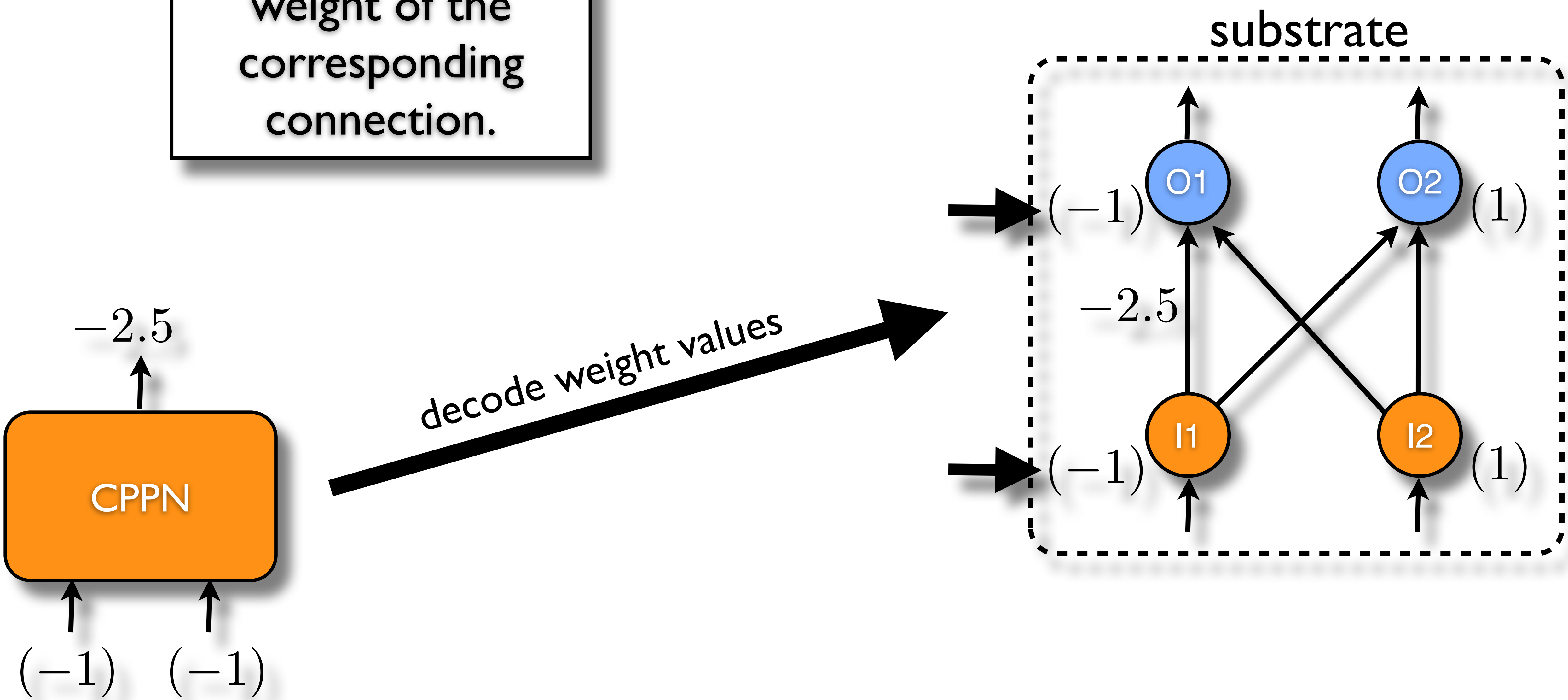
- Stanley et al. 2007: Hypercube-based encoding.



HyperNEAT

- Stanley et al. 2007: Hypercube-based encoding.

... and computes the weight of the corresponding connection.



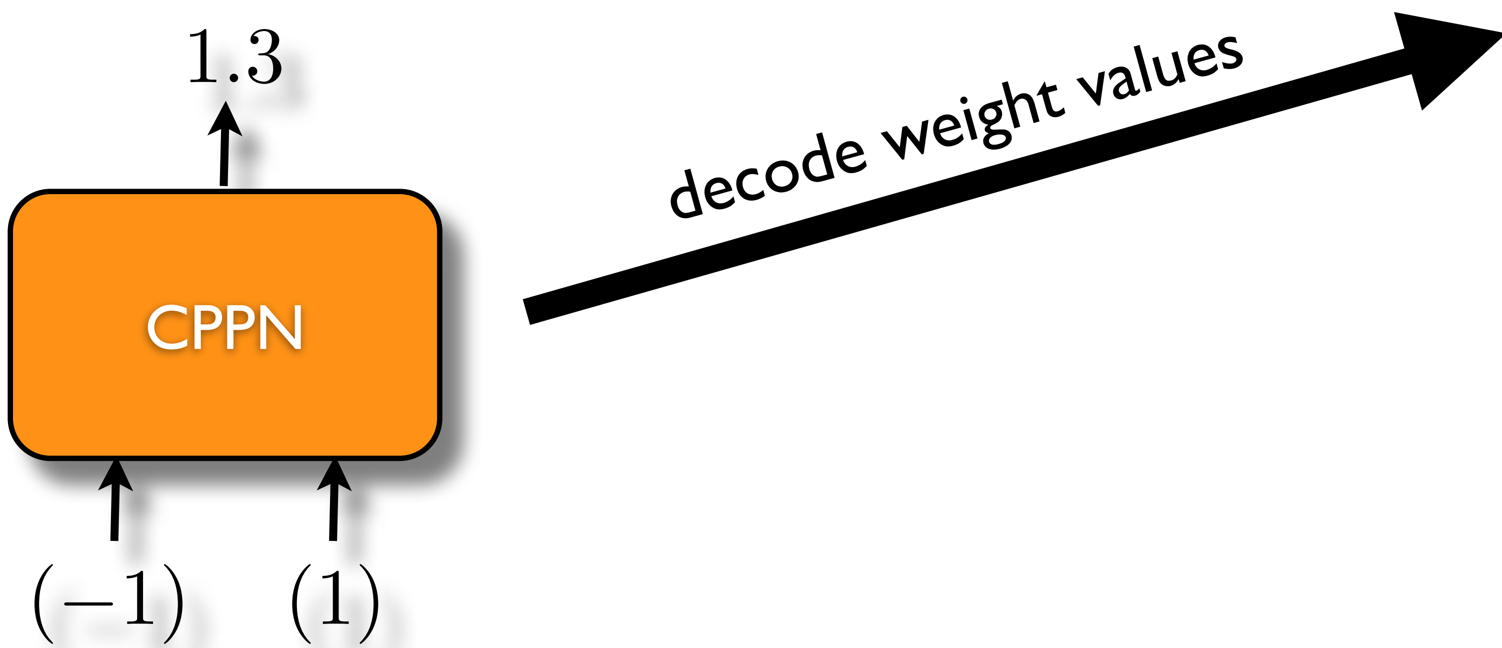
COMPUTATIONAL INTELLIGENCE GROUP

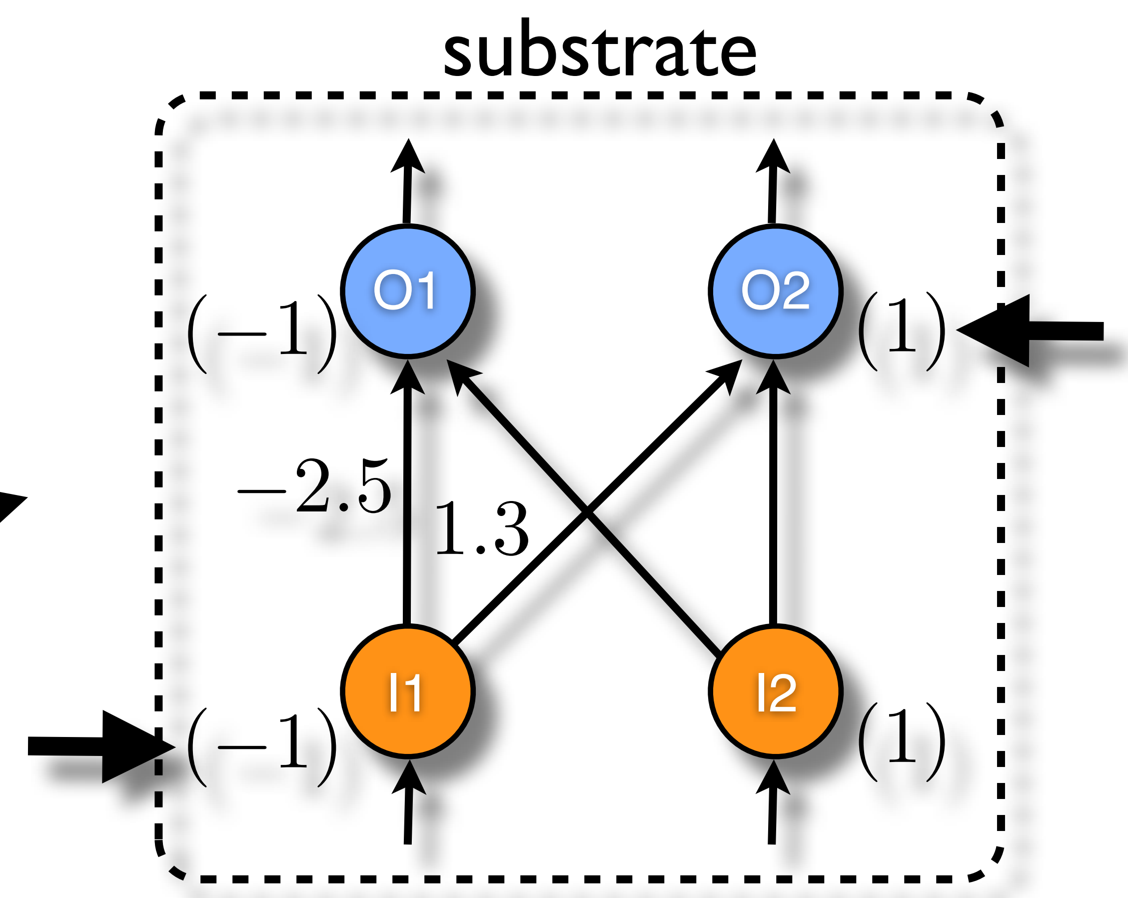
2012

HyperNEAT

- Stanley et al. 2007: Hypercube-based encoding.

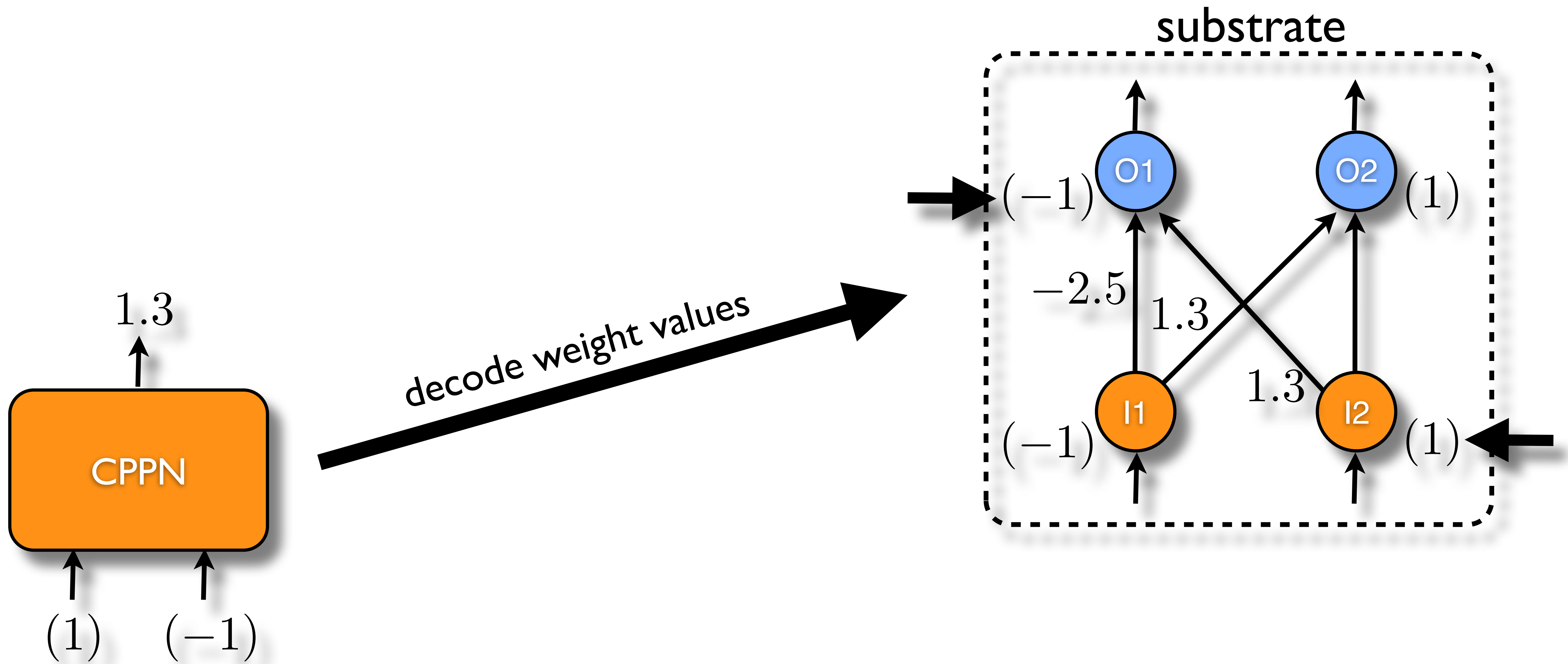
All weights are computed in a same way...





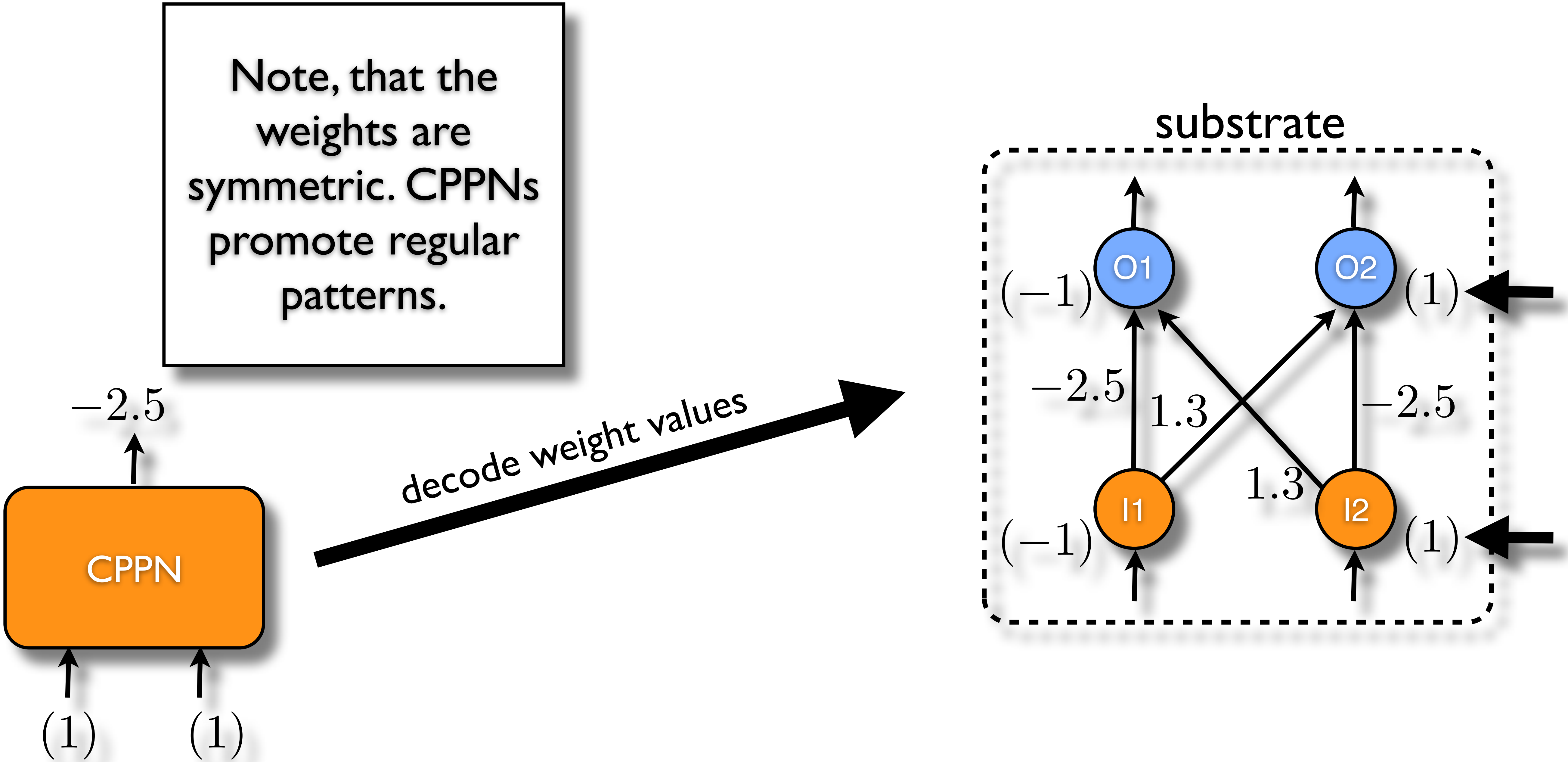
HyperNEAT

- Stanley et al. 2007: Hypercube-based encoding.



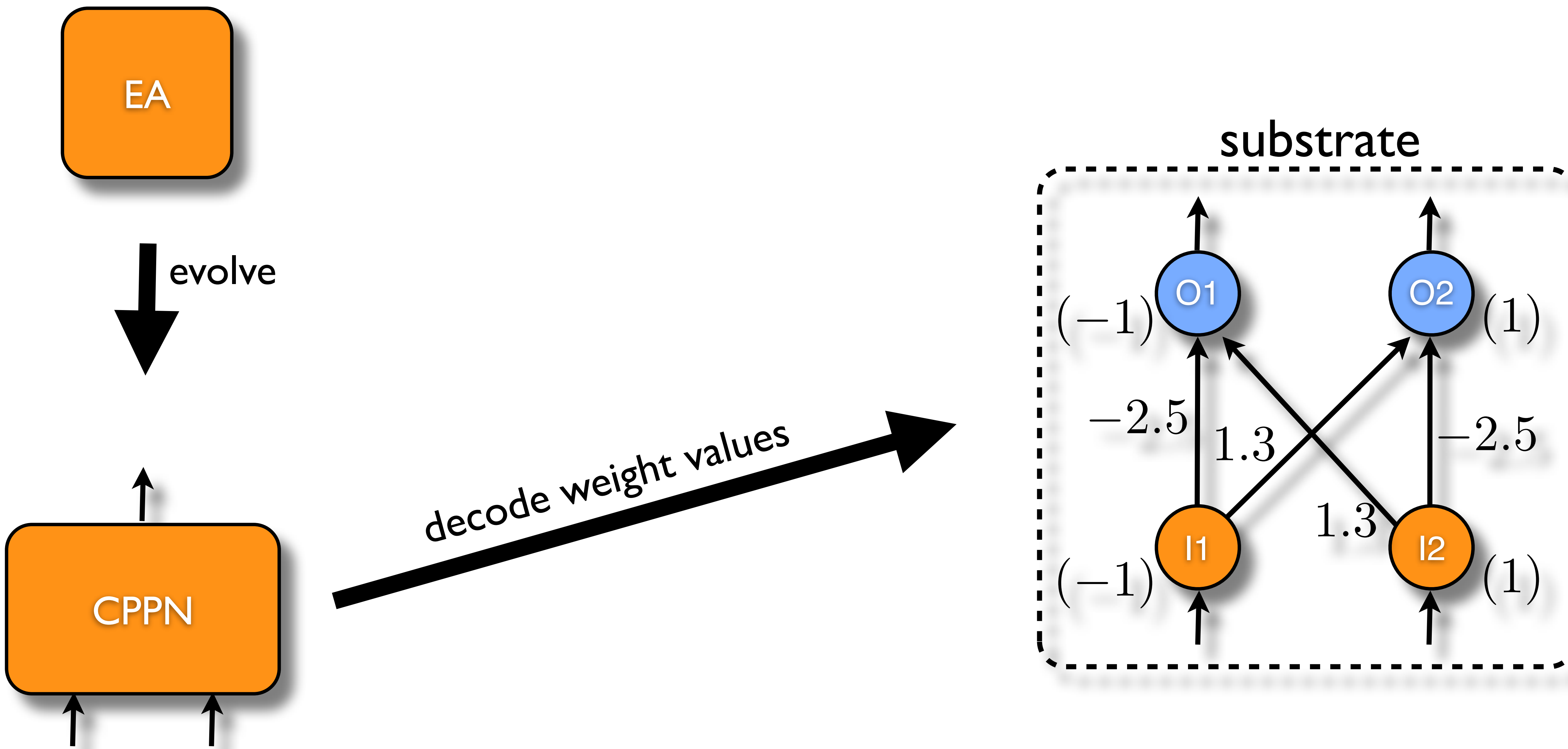
HyperNEAT

- Stanley et al. 2007: Hypercube-based encoding.



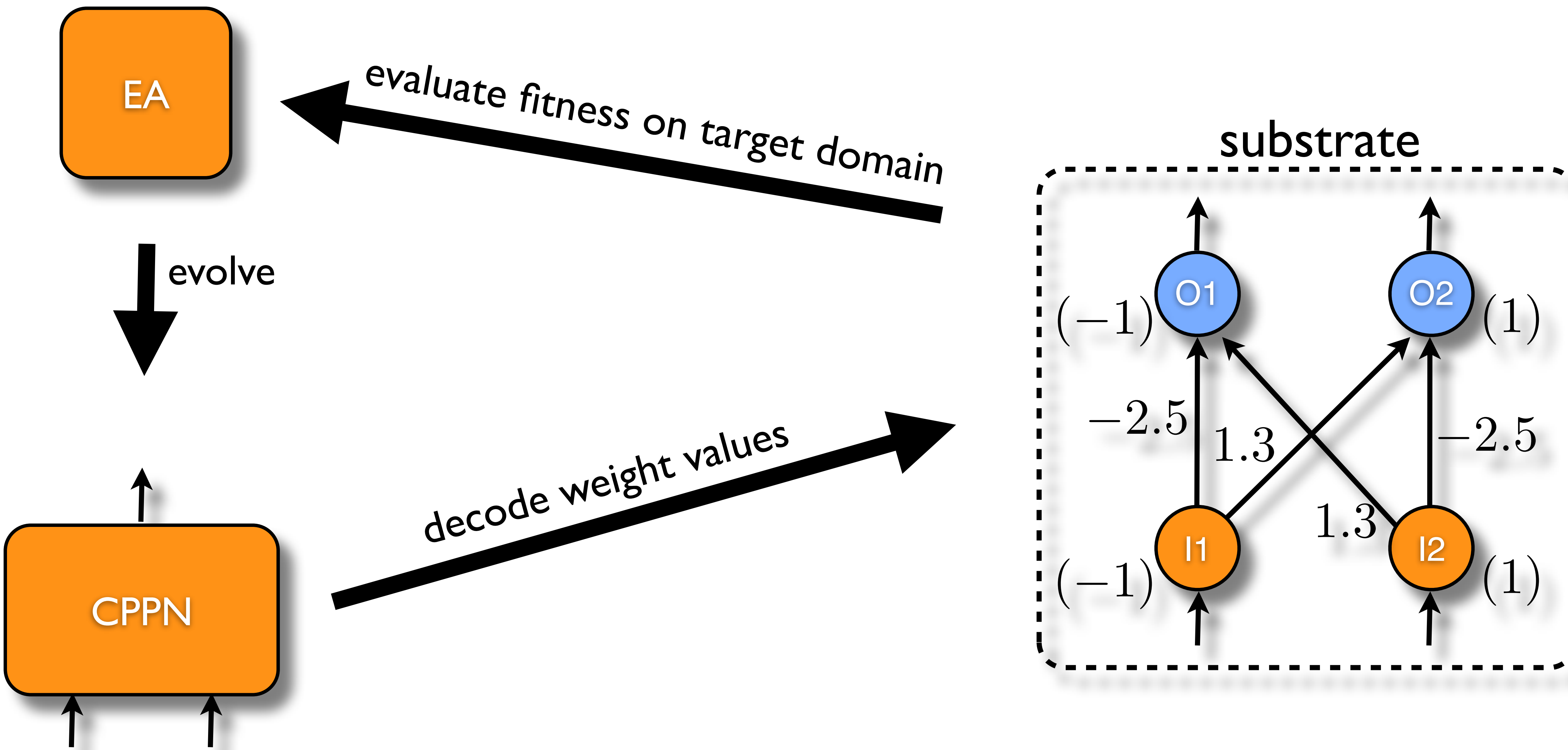
HyperNEAT

- Stanley et al. 2007: Hypercube-based encoding.

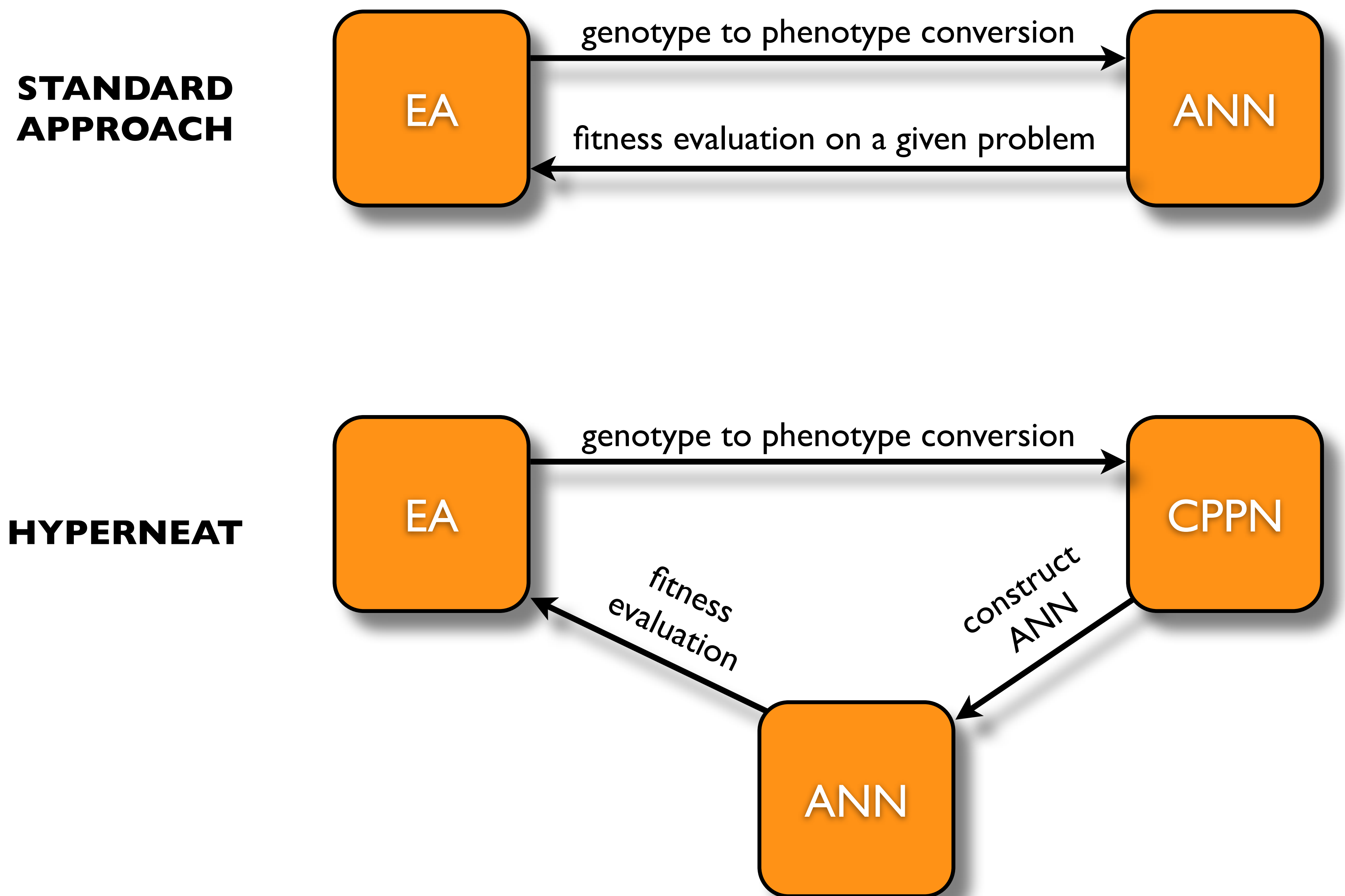


HyperNEAT

- Stanley et al. 2007: Hypercube-based encoding.

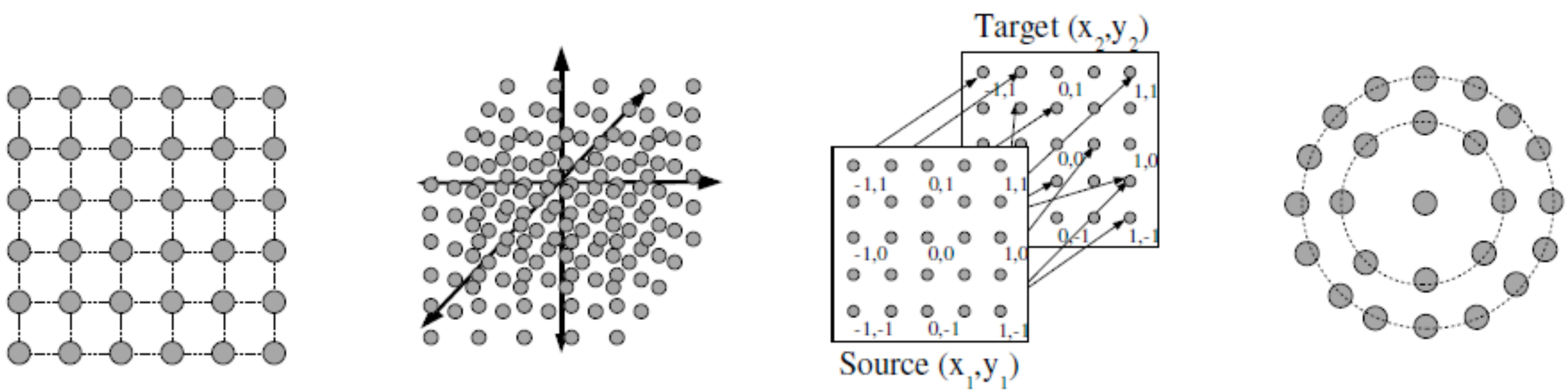


HyperNEAT vs. Standard Approaches



Types of Substrate?

- The list of neurons’ coordinates along with possible connections between them.



(a) Grid (b) Three-dimensional (c) Sandwich (d) Circular

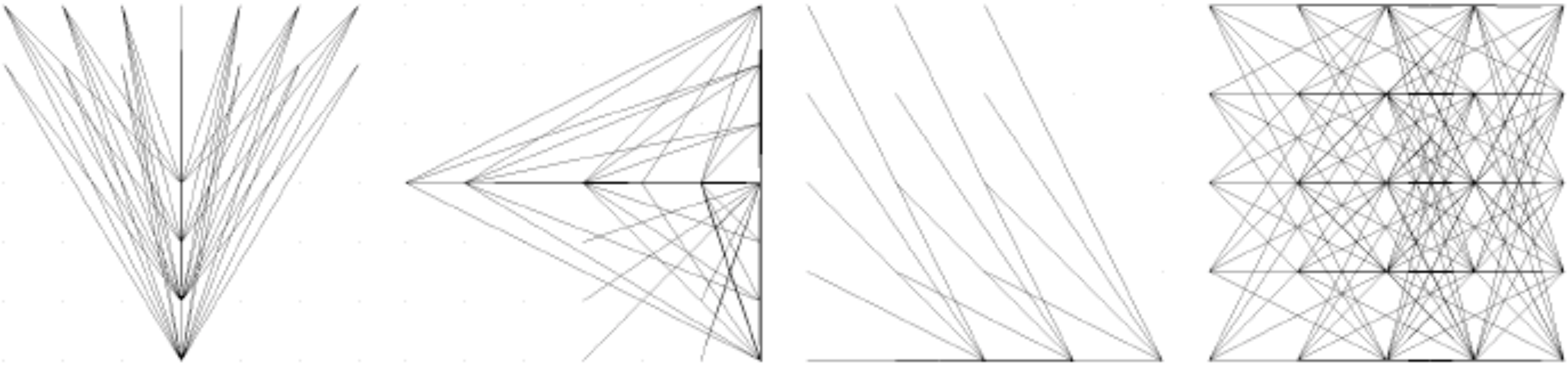
Create or not Create a Link?

- Substrates are often fully connected → lots of links → computationally infeasible → pruning is used.
- If CPPN outputs weights in range $[-3; 3]$ then
- links with weights < 0.2 are not expressed,
- $>= 0.2$ are scaled to magnitude between 0 and 3.

→ when using this approach the final ANN is a sub-graph of a substrate.

Connectivity Patterns

- Patterns evolved using interactive evolution:



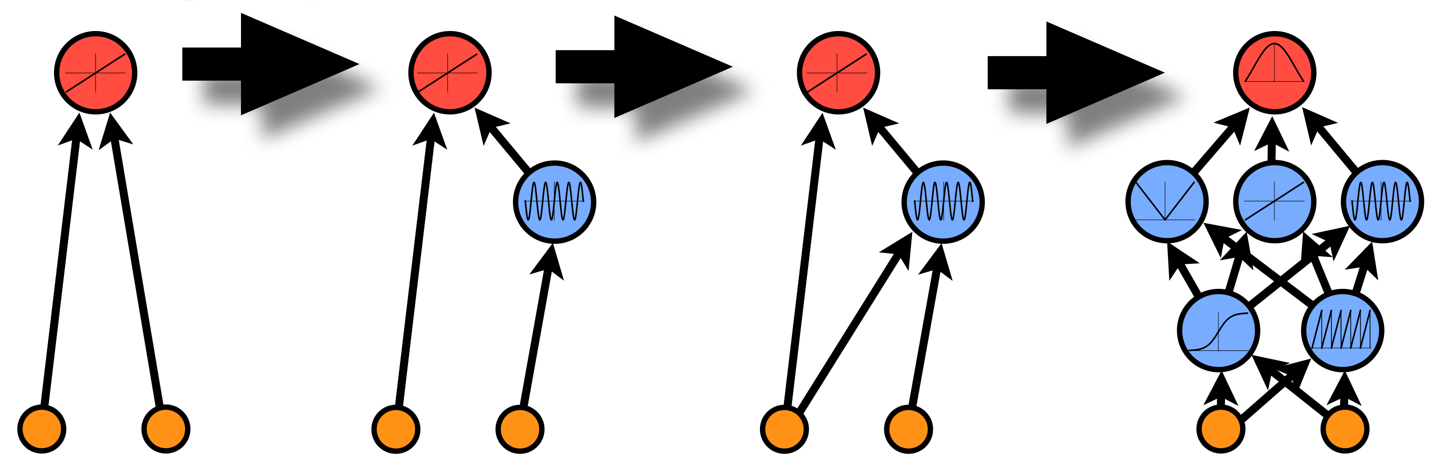
(a) Sym. (b) Imperf. (c) Repet. (d) Var.

Spatial Representation

- HyperNEAT **exploits spatial representation of a problem**. The same happens in Nature:
  - connection of eyes to brain hemispheres,
  - similar things processed nearby.
- We have to assign coordinates.
- **Does every problem have a reasonable spatial representation?**
  - It seams that most problems have. The others would not probably benefit from regularities in ANNs.

NEAT in HyperNEAT

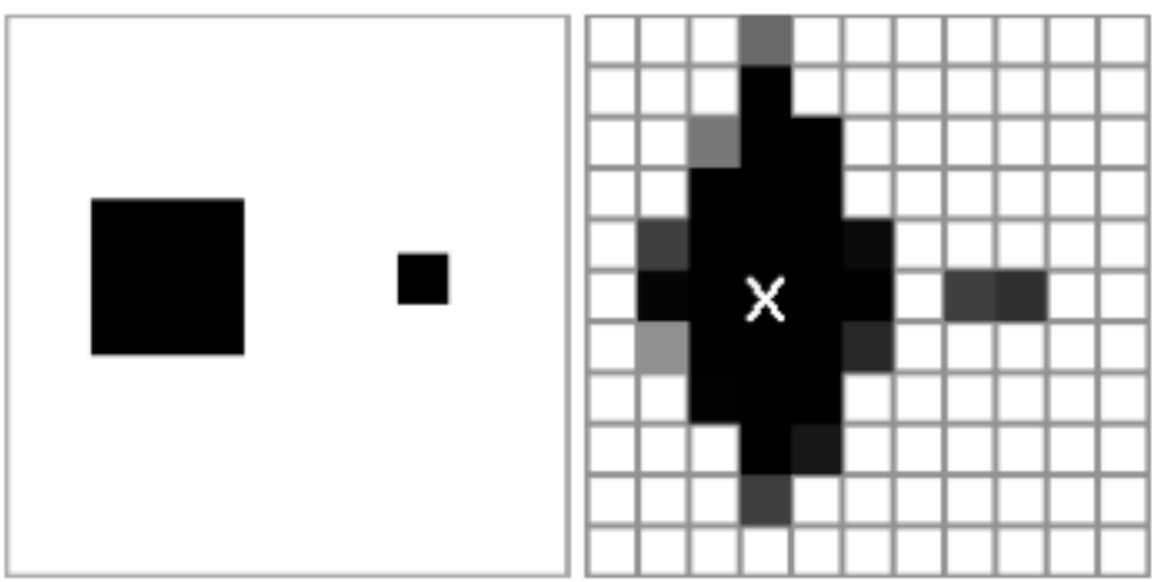
- HyperNEAT uses a slightly modified NEAT (Stanley 2001) as a **base algorithm** to evolve CPPNs.
- NEAT is neuro-evolutionary algorithm able to evolve ANNs of arbitrary topologies.



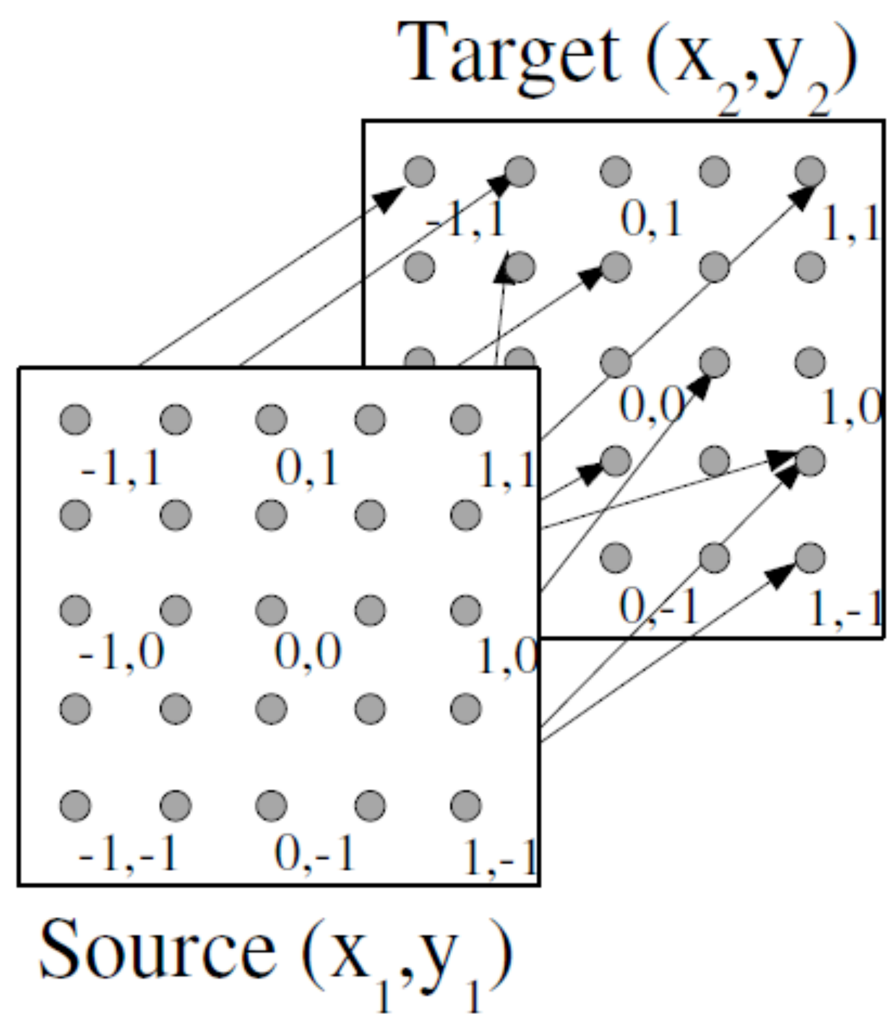
- It is based on:
  - **complexification** → evolving gradually more complex ANNs,
  - **innovation numbers** → track structural innovations,
  - **niching** → allows simultaneous evolution of small and large ANNs in one population. **Requires to define a distance measure for ANNs.**

Visual Discrimination

- Visual targeting: distinguish the larger object.
- "Sandwich substrate".



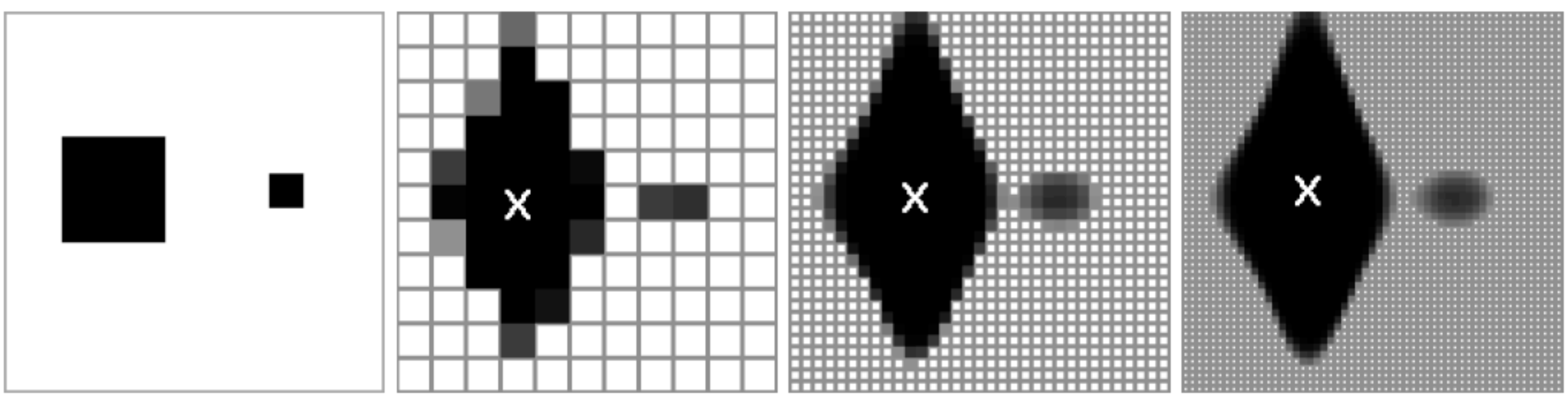
(a) Sensor Field (object placement) (b) 11x11 Target Field (12,827 connections)



Jason J. Gauci and Kenneth O. Stanley (2007):
Generating Large-Scale Neural Networks Through Discovering Geometric Regularities

Visual Discrimination II: Scaling the Substrate

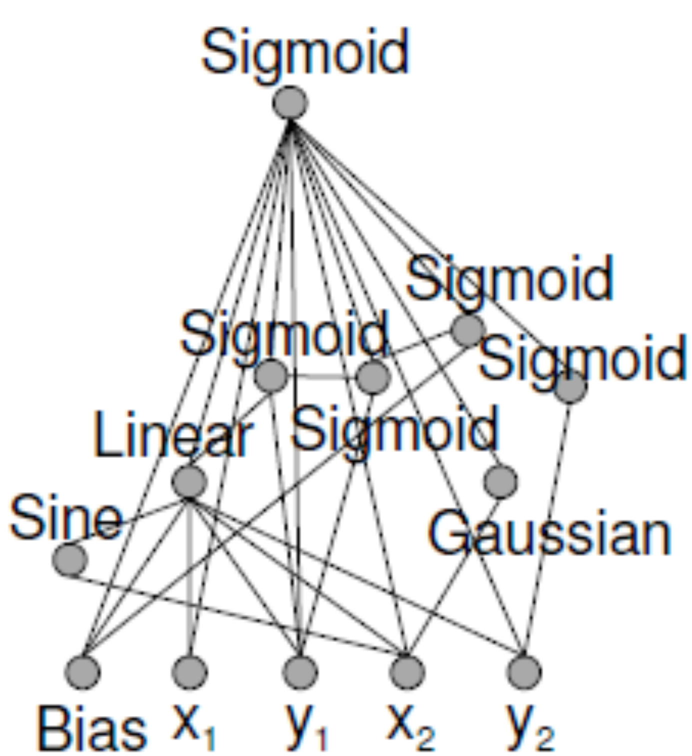
- The substrate density can be scaled using the same CPPN.
- The function of the final ANN is approximately preserved.
- We can train on small → get large.

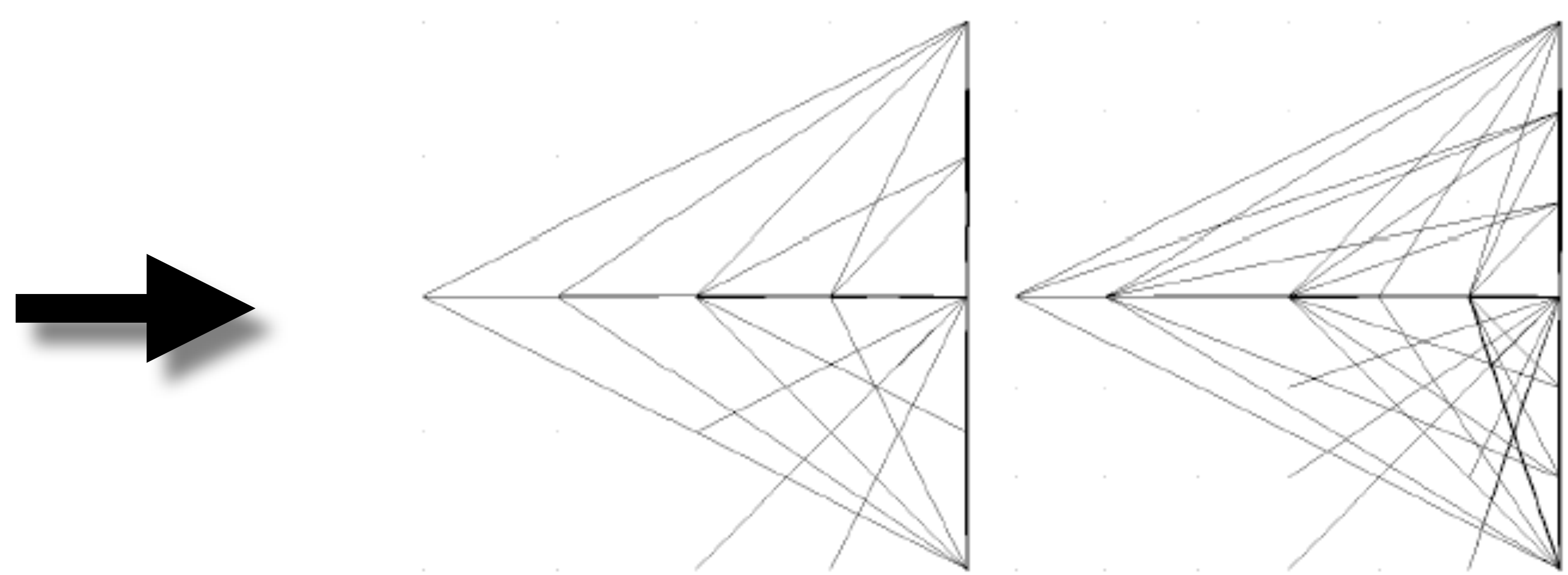


(a) Sensor Field (object placement)
(b) 11x11 Target Field (12,827 connections)
(c) 33x33 Target Field (1,033,603 connections)
(d) 55x55 Target Field (7,970,395 connections)

Visual Discrimination III: Scaling the Substrate

- An equivalent connectivity concept at different
- substrate resolutions.





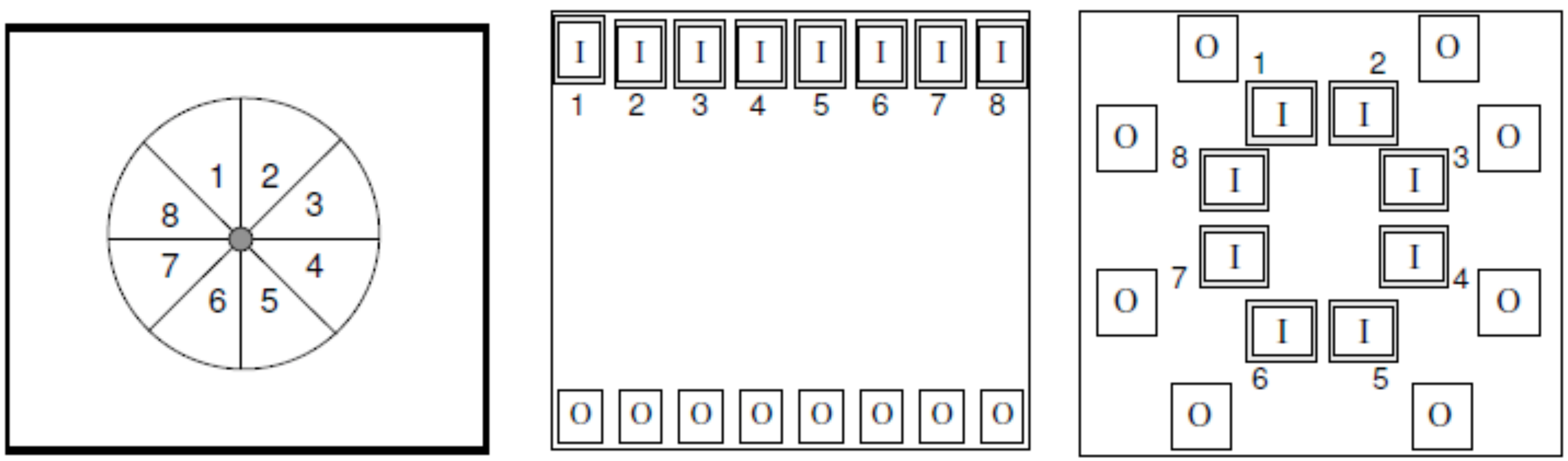
(a) 5 × 5 Concept (b) 7 × 7 Concept

COMPUTATIONAL INTELLIGENCE GROUP

2012

Food Gathering Problem

- Range-finder sensors detect food.
- More food eaten → higher fitness.
- Experiments with different sensor/effector placement – exploiting geometric relationships with "outer world".



(a) Robot (b) Parallel (c) Concentric

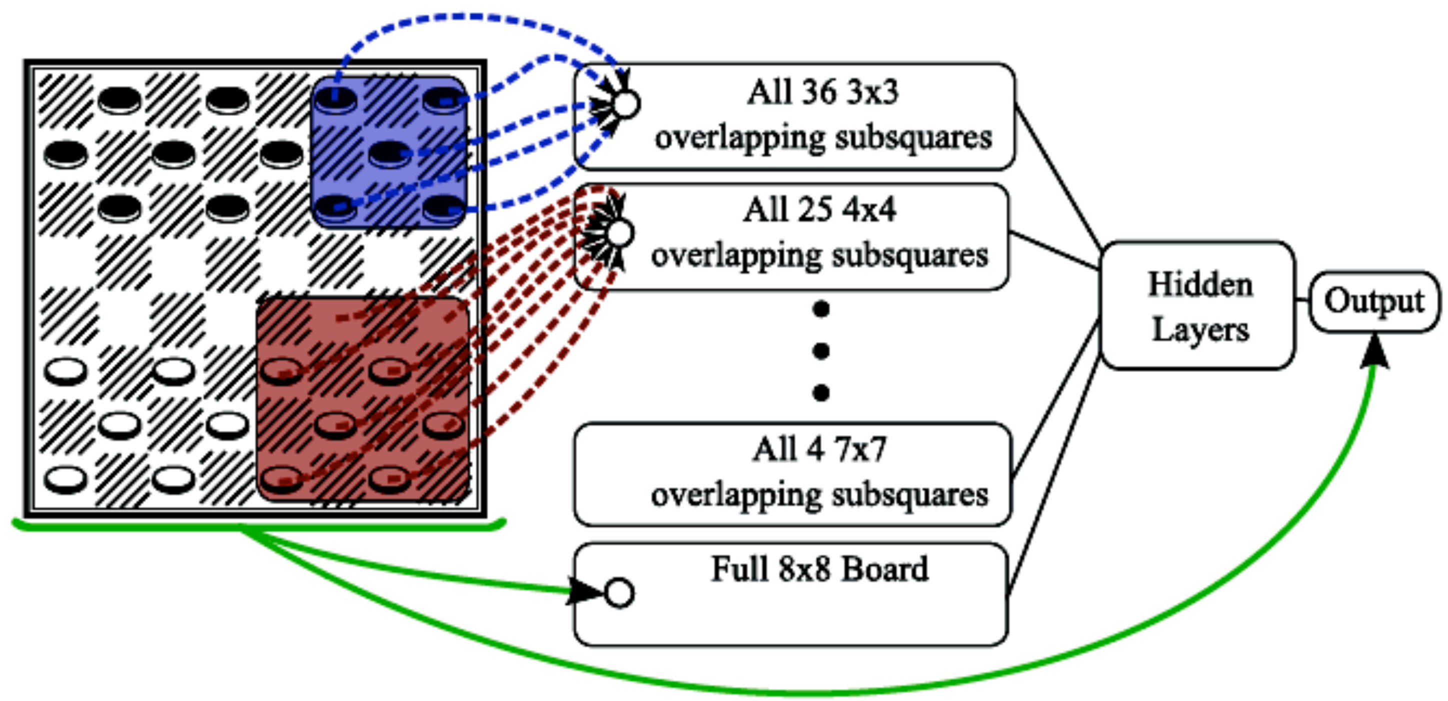
David B. D'Ambrosio and Kenneth O. Stanley (2007)
A Novel Generative Encoding for Exploiting Neural Network Sensor and Output Geometry

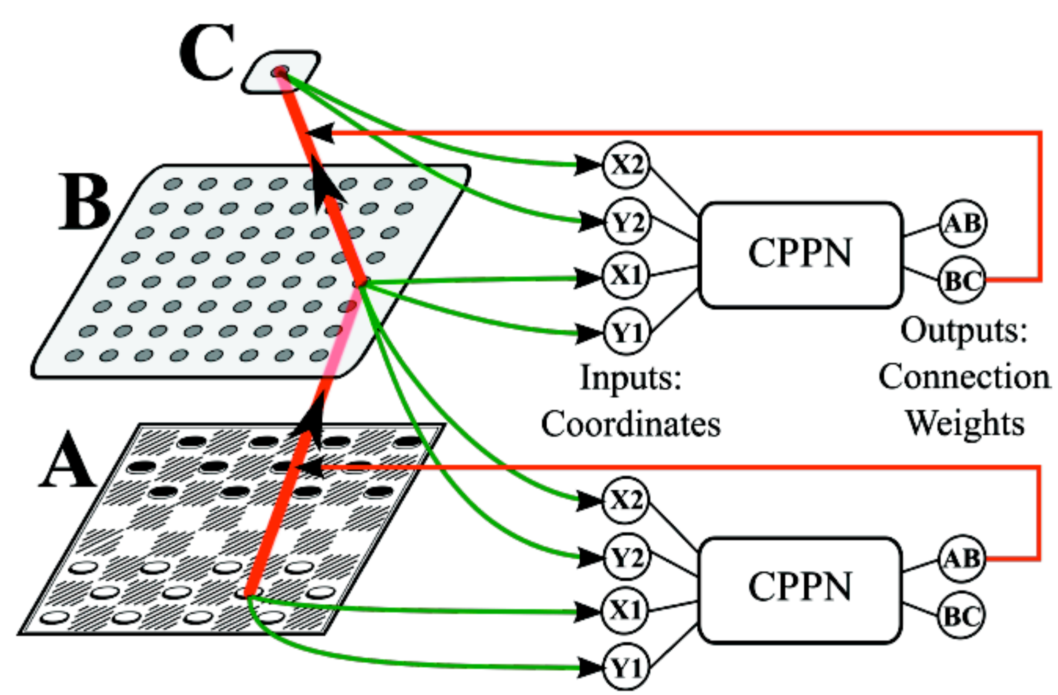
Food Gathering Problem II

- Parallel worked better than Concentric because less computation is needed for CPPN.
- New CPPN inputs added: the *distances*
- $(x1-x2)$ and $(y1-y2)$
- When CPPN is provided the distances, both work the same.

Checkers

- Comparison with classic NEAT.
- HyperNEAT is faster + generalizes.
- Single CPPN with multiple outputs.
- The output of the final net is a heuristic score for the minimax algorithm.



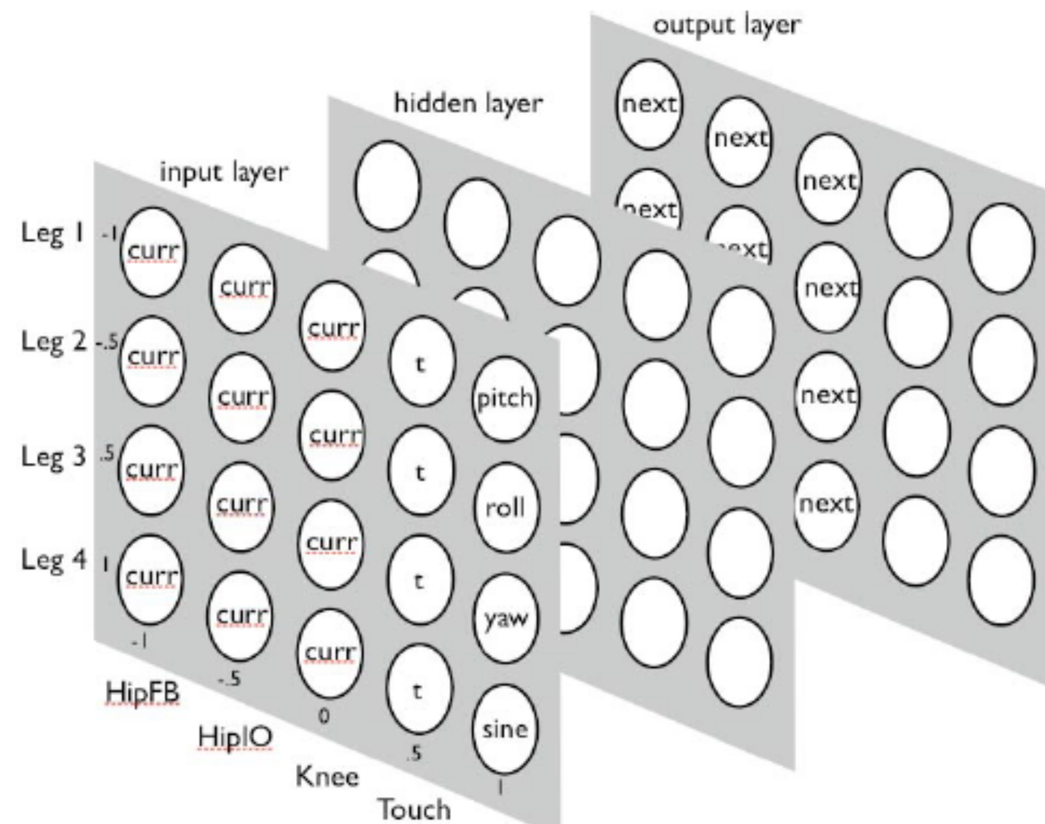


Jason Gauci and Kenneth O. Stanley (2008):
A Case Study on the Critical Role of Geometric Regularity in Machine Learning

COMPUTATIONAL INTELLIGENCE GROUP

HyperNEAT Coordinated Quadruped Gaits

- Simulation of four legged walker robot.
- Comparison with classic NEAT.
- Other experiments show that HyperNEAT can deal with random substrates.



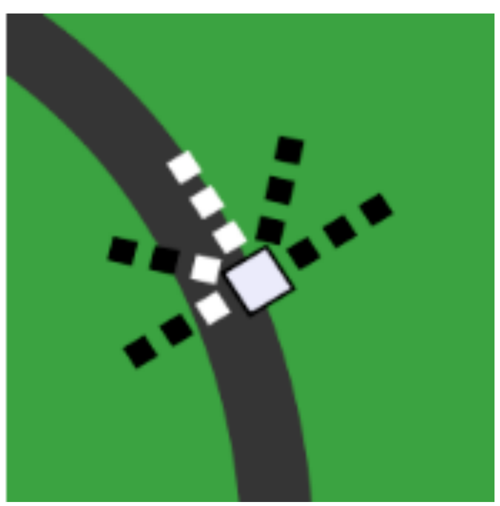
COMPUTATIONAL INTELLIGENCE GROUP

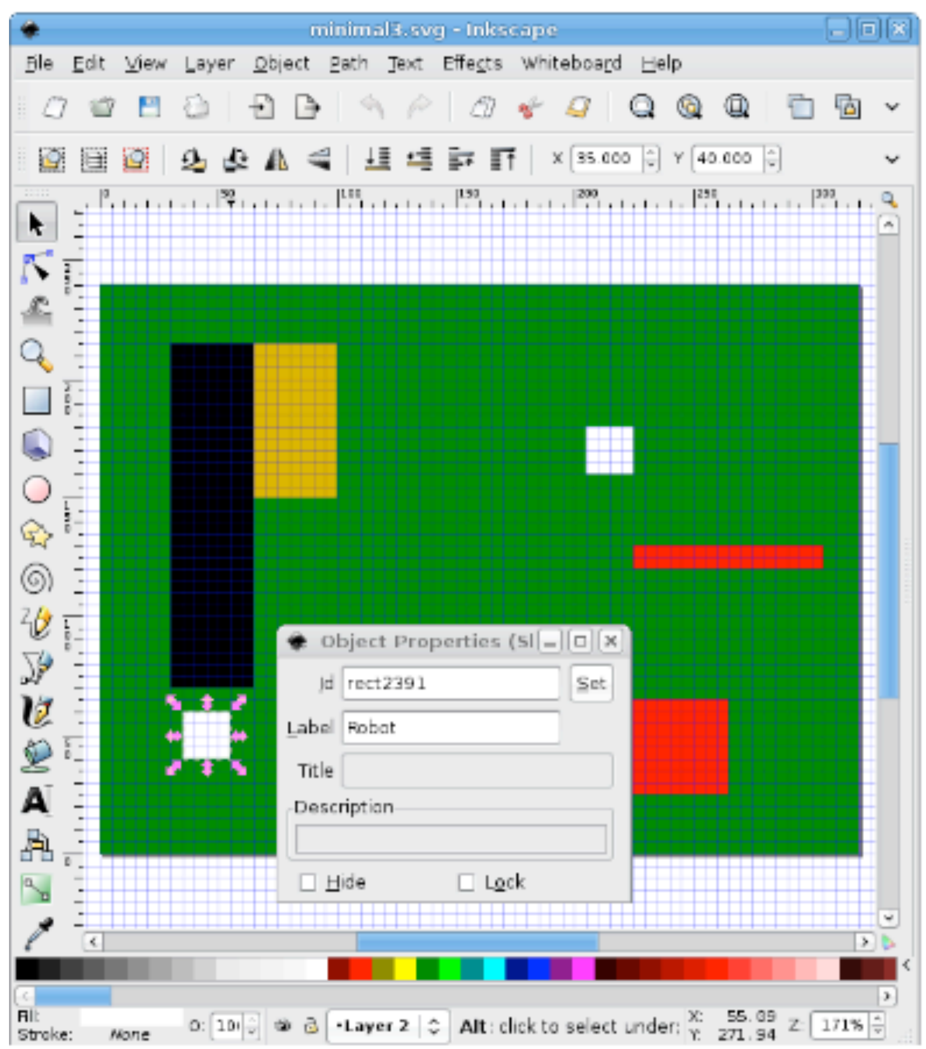
Jeff Clune:
Evolving Coordinated Quadruped Gaits with the HyperNEAT Generative Encoding

2012

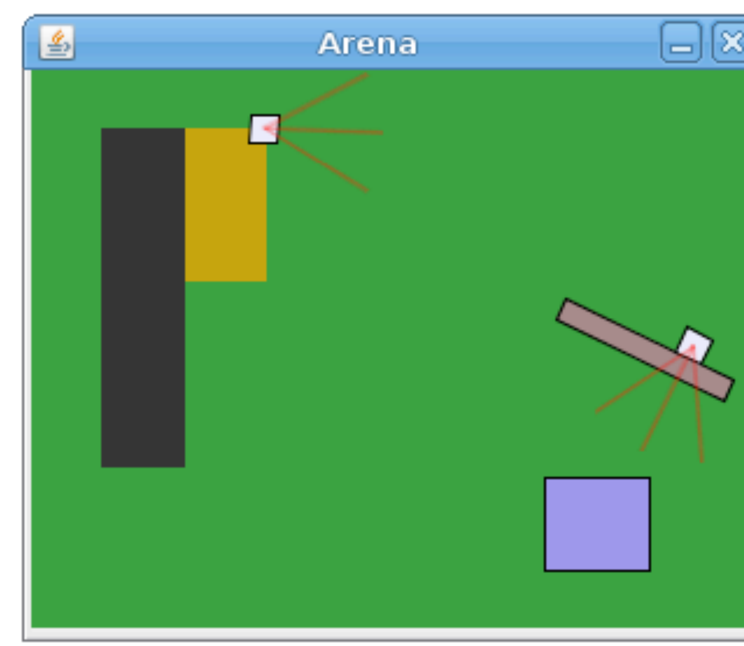
Mobile Robot Navigation

- HyperNEAT/HyperGP for robot control.
- ViVAE Simulated 2D environment with rigid body physics.





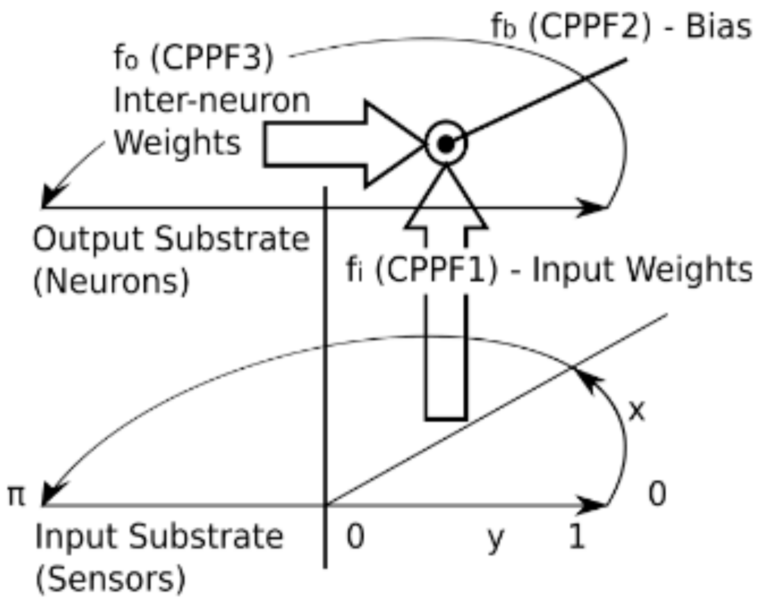
→

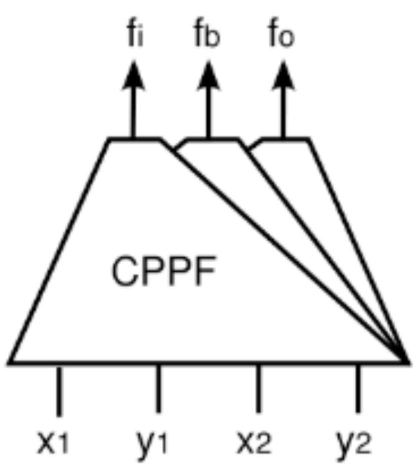


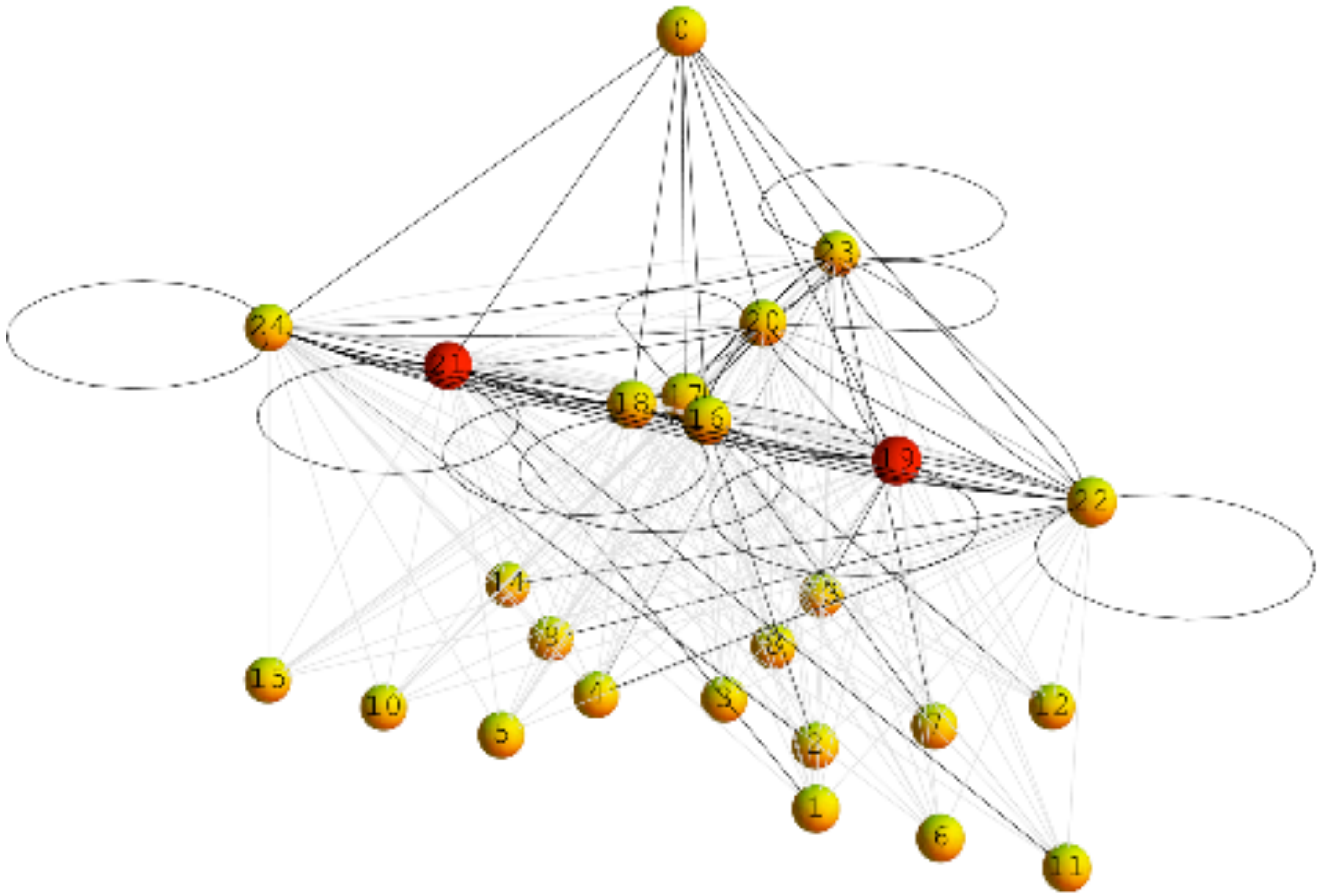
COMPUTATIONAL
INTELLIGENCE
GROUP

Mobile Robot Navigation II

- Substrate uses polar coordinates.
- Input + 1 fully recurrent layer
- See VIDEO...

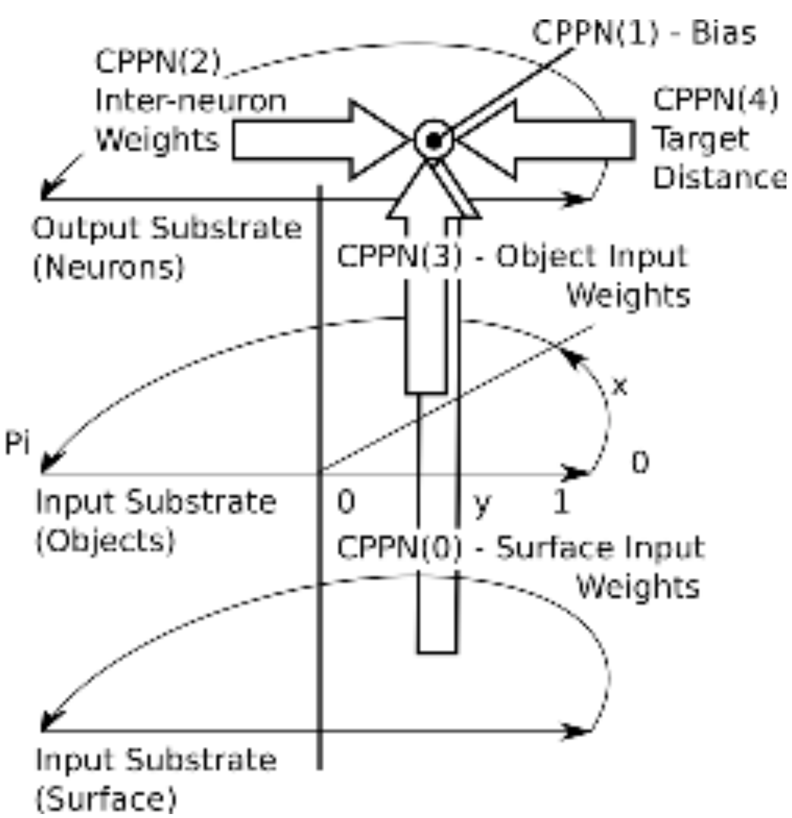


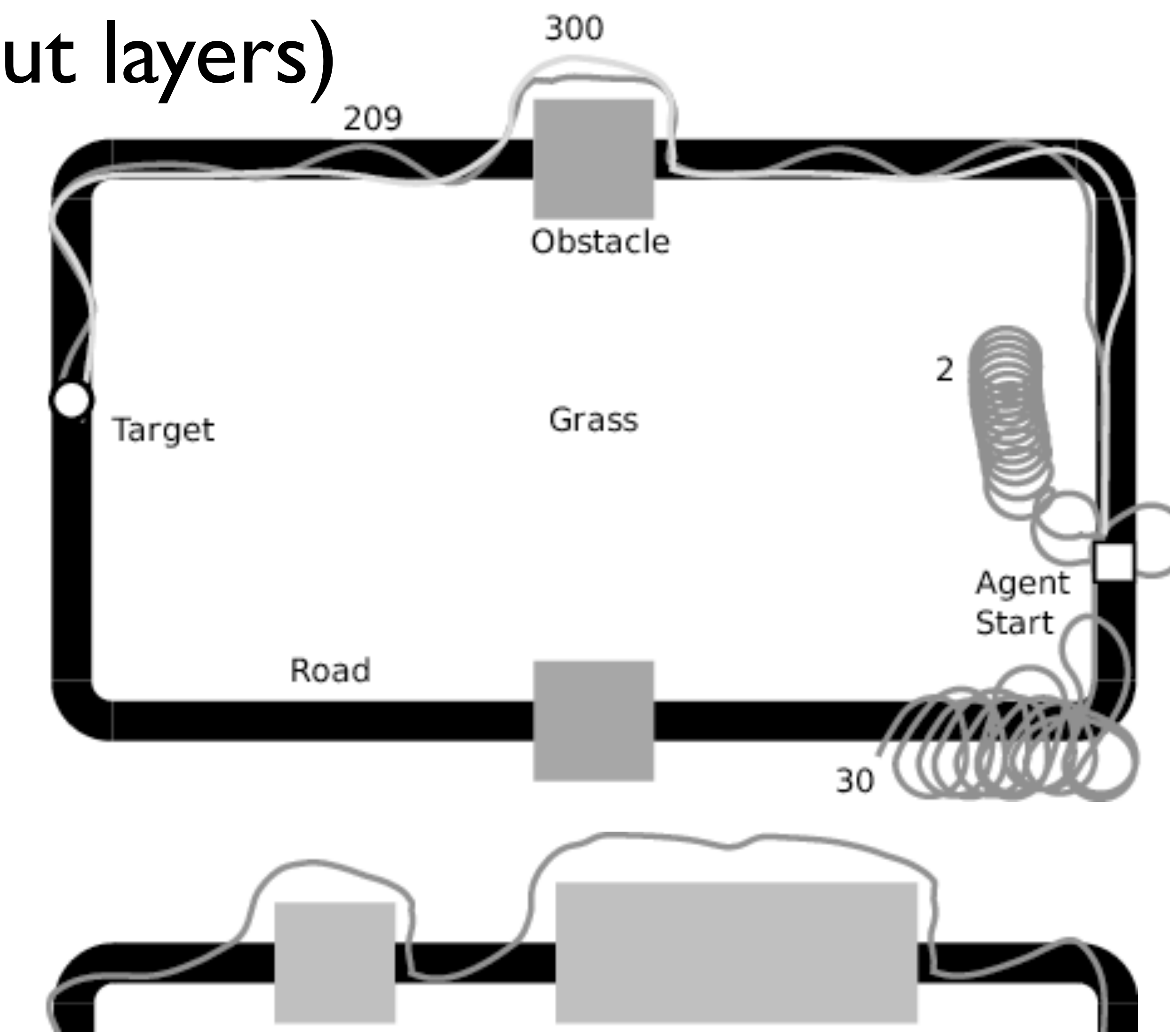




Mobile Robot Navigation III

- Obstacle avoidance.
- Object sensors added (two input layers)





COMPUTATIONAL
INTELLIGENCE
GROUP

$$f = \frac{distanceTravelled}{simulationSteps+1}\left(1 - \frac{targetDistance}{initialDistance}\right)$$

2012

Mobile Robot Navigation IV

Drchal, Koutník and Šnorek (2009):
HyperNEAT Controlled Robots Learn How to Drive on Roads in Simulated Environment

Buk, Koutník and Šnorek (2009):
NEAT in HyperNEAT Substituted with Genetic Programming

Drchal, Kapraľ, Koutník and Šnorek (2009):
Combining Multiple Inputs in HyperNEAT Mobile Agent Controller

Base Algorithms for Hypercube-based Encoding

- The large-scale networks produced by HyperNEAT can be very slow to simulate...
- We need to reduce the number of fitness function evaluations as much as possible.
- Can we do better when NEAT is replaced by a different *base algorithm*?

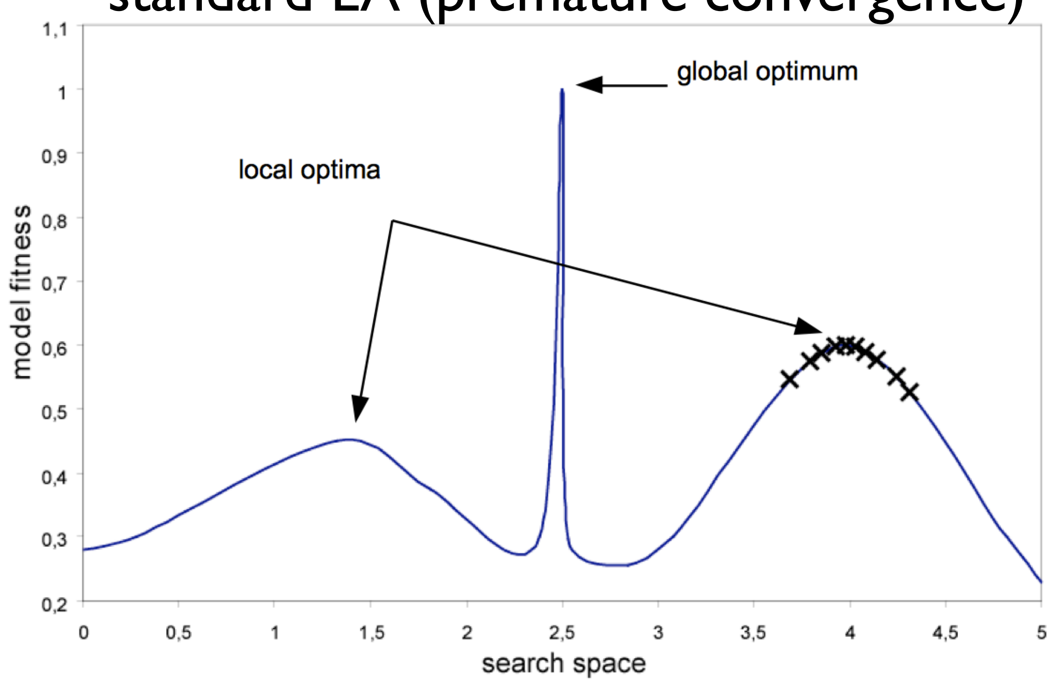
What about Genetic Programming (GP)?

- Zdenek Buk and Jan Koutnik replaced NEAT in HyperNEAT by GP (2009).
- Experiments on a single domain shown that HyperGP outperforms HyperNEAT.
- Can we do even better?

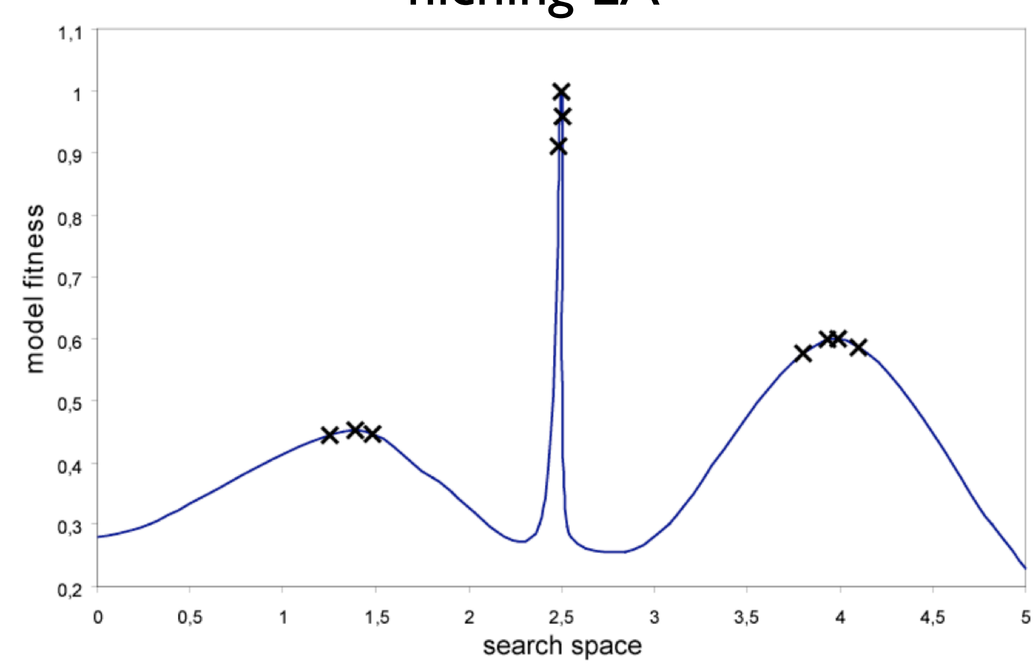
Niching EA

- Originally methods to search all optima in multimodal domain.
- Used to propose diversity in population in order to avoid premature convergence.
- Population split into separate subpopulations of similar individuals.
- **Distance measure is required.**

standard EA (premature convergence)



niching EA



GPEFS Overview

- Genetic Programming with Explicit Fitness Sharing (GPEFS).
- GPEFS is basically NEAT which evolves forests of trees using standard GP genetic operators.
- There is no complexification in GPEFS but niching is essential part which preserves diversity and prevents premature convergence.
- We do not employ crossover.
- The idea to combine GP and niching is not new but:
  - the version of fitness sharing used in NEAT was not employed,
  - we experiment with six distance measures (both our and already published),
  - we focus on Hypercube-based indirectly encoded problems.

GPAT Overview

- We propose Genetic Programming of Augmenting Topologies (GPAT).
- GPAT is basically NEAT which evolves forests of trees.
- Uses complexification and niching (Explicit Fitness Sharing).
- We do not employ crossover.
- It is much simpler to design an efficient distance measure for trees than for neural networks: *there is no need for innovation numbers*.
- GPAT is general algorithm, here we focus on Hypercube-based indirectly encoded problems.

GPAT Genotypes

- GPAT evolves trees (forests) but:
  - nodes have a variable arity,
  - constants are stored in links (similar to synaptic weights of ANN).

GPEFS & GPAT Results

- GPEFS and GPAT have similar performance.
- Significant improvement to GP and NEAT.

Q&A

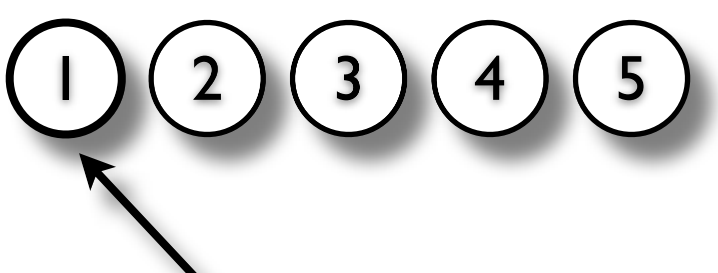
Additional Slides

GPEFS

Assign Species

- Explicit Fitness Sharing as in NEAT.
- Species assigned according to distance d and threshold δ.

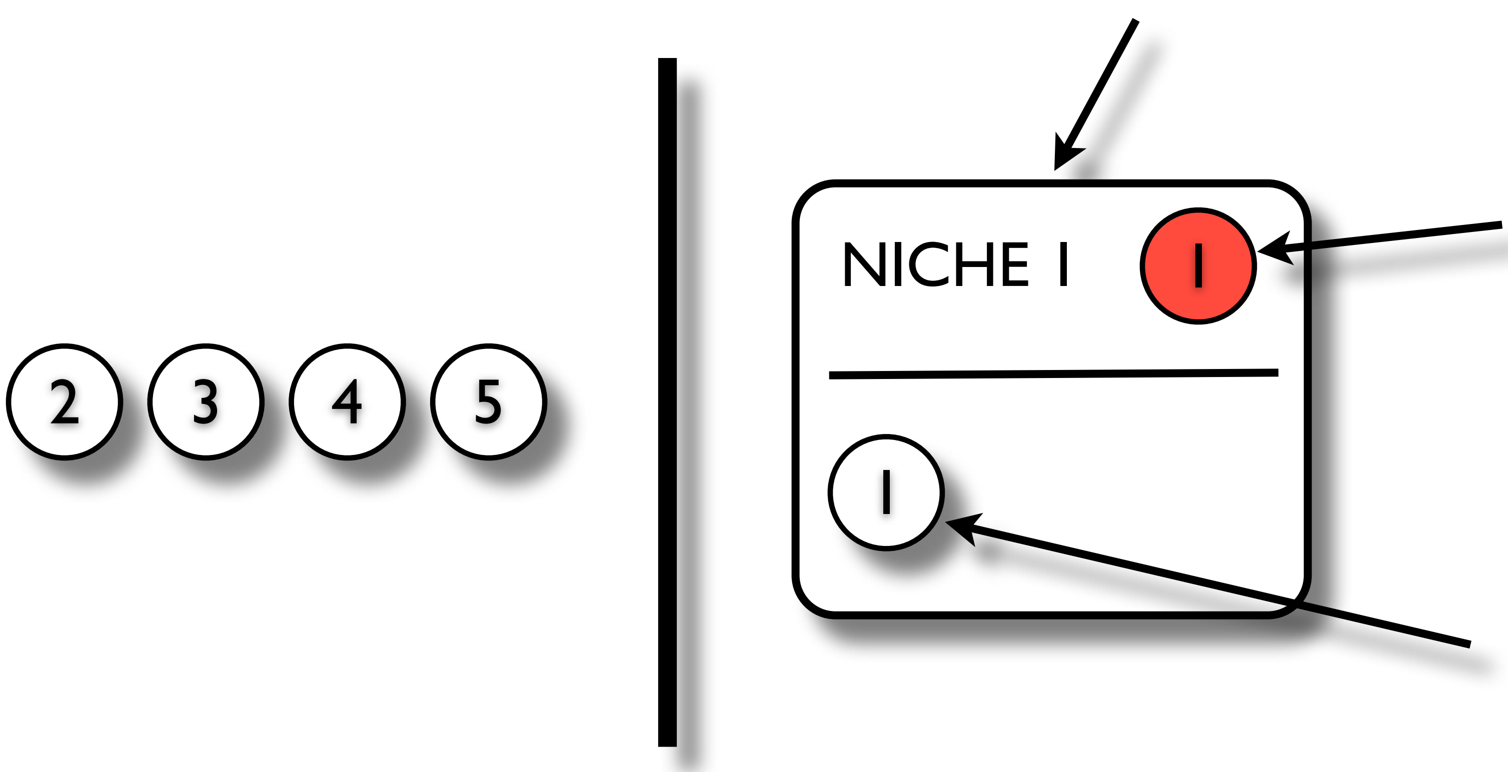
population to assign



start with the first individual

Assign Species

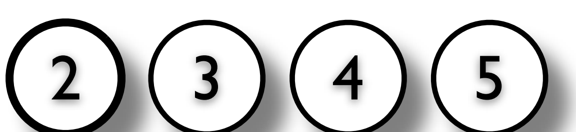
- Explicit Fitness Sharing as in NEAT.
- Species assigned according to distance *d* and threshold δ.

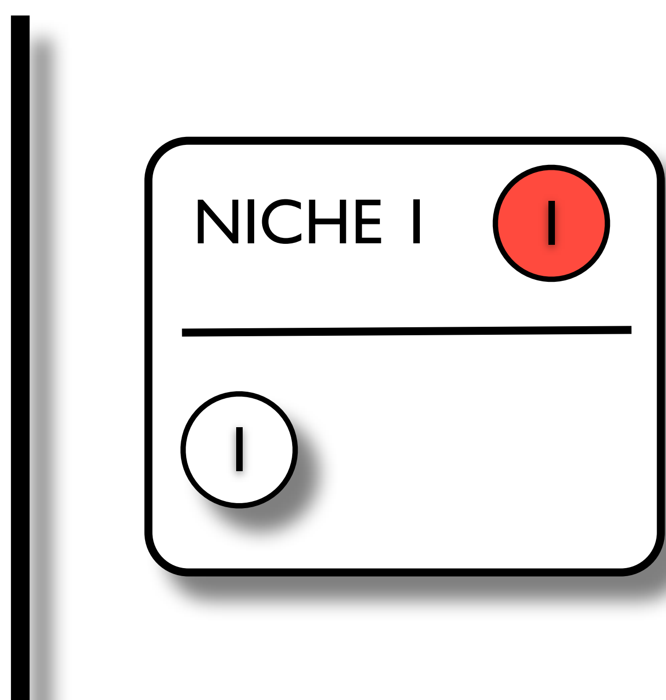


Assign Species

- Explicit Fitness Sharing as in NEAT.
- Species assigned according to distance d and threshold δ.

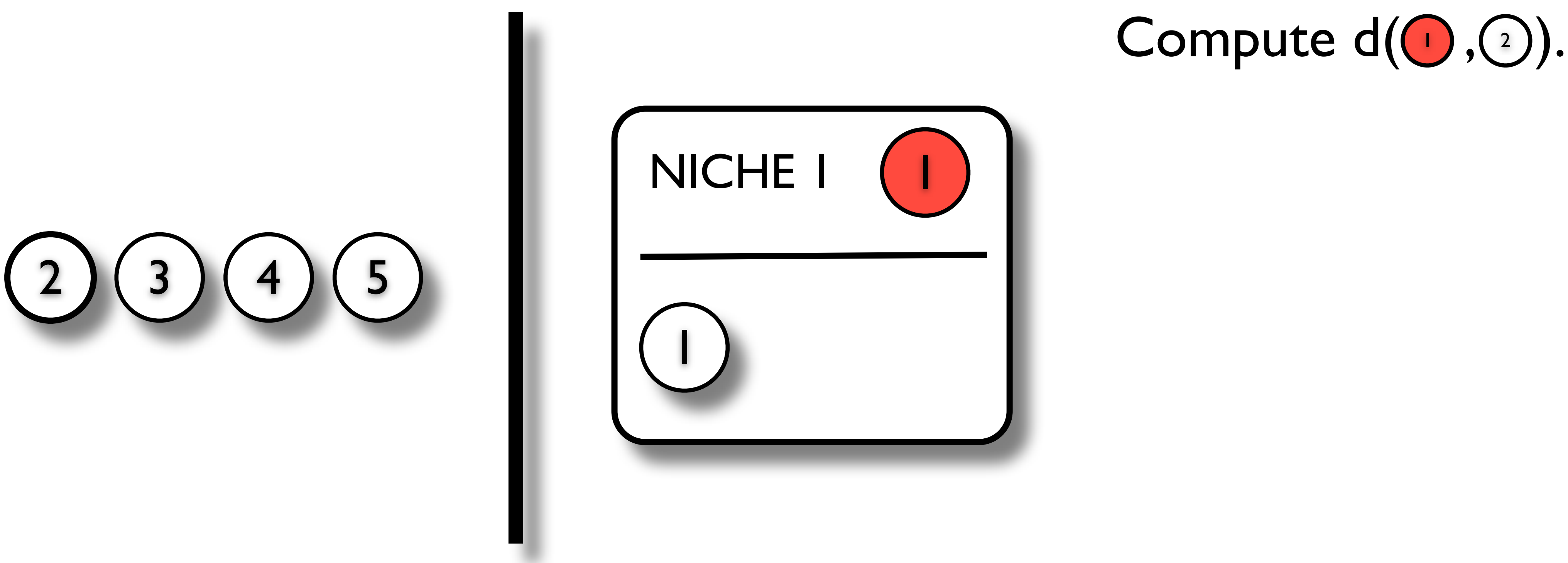
Now continue
with the second.





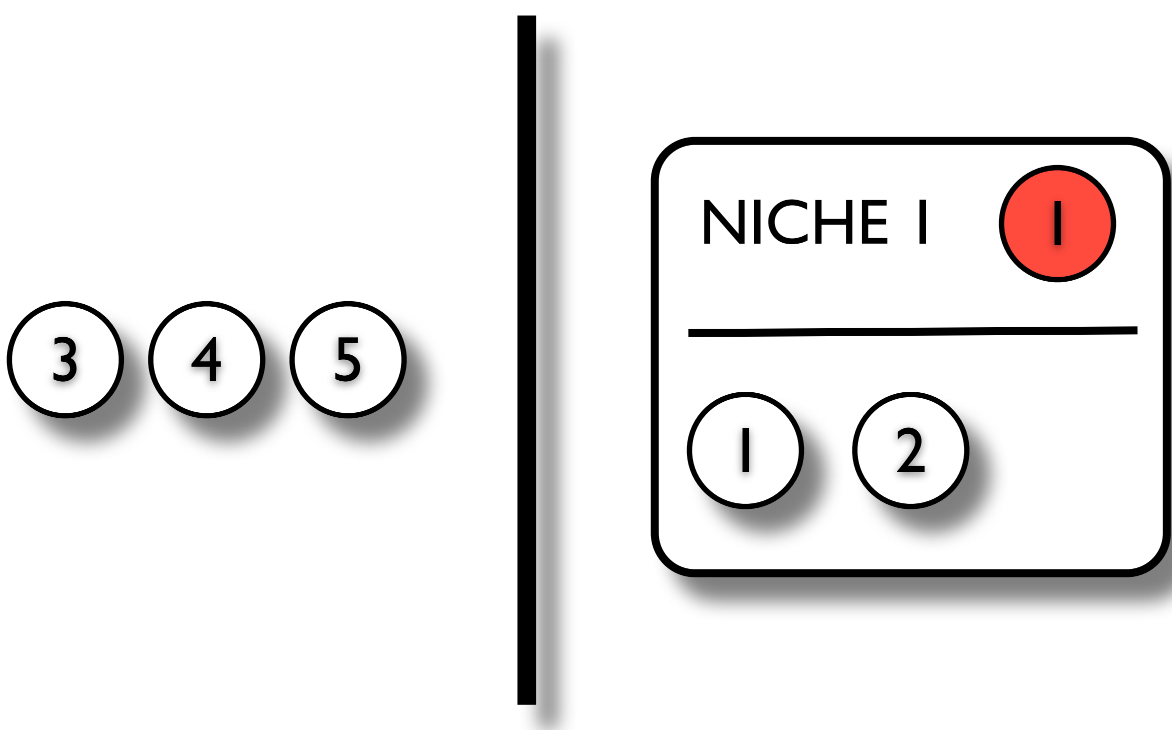
Assign Species

- Explicit Fitness Sharing as in NEAT.
- Species assigned according to distance *d* and threshold δ.



Assign Species

- Explicit Fitness Sharing as in NEAT.
- Species assigned according to distance *d* and threshold δ.

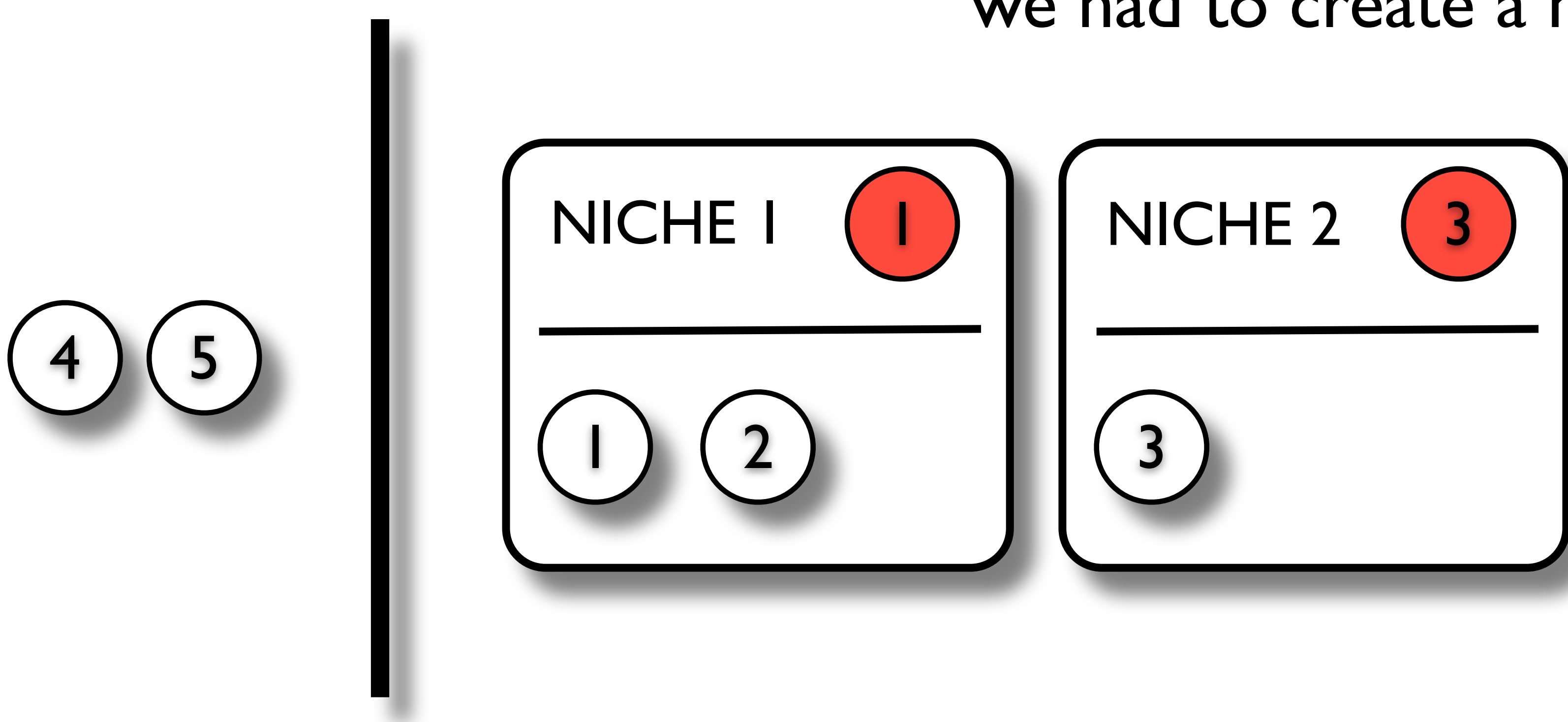


Compute d(1,2).
If $d < \delta$ like here,
make him a member.

Assign Species

- Explicit Fitness Sharing as in NEAT.
- Species assigned according to distance *d* and threshold δ.

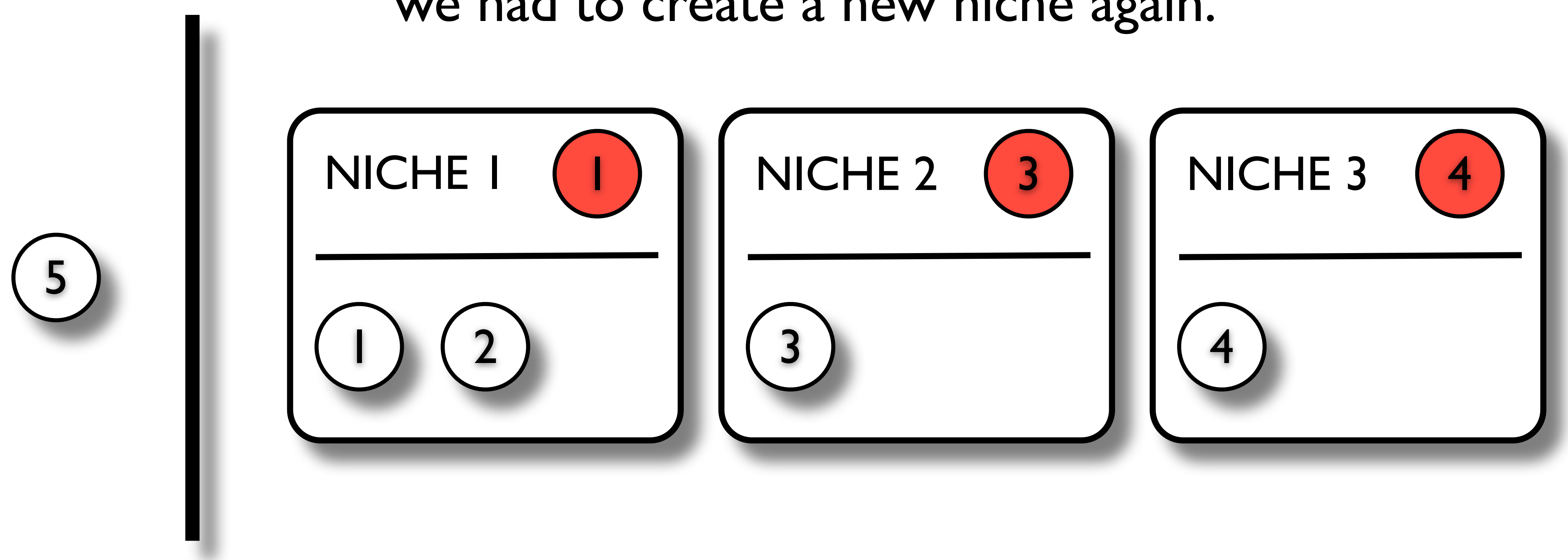
Because d(1,3) $\geq \delta$ here,
we had to create a new niche.



Assign Species

- Explicit Fitness Sharing as in NEAT.
- Species assigned according to distance *d* and threshold δ.

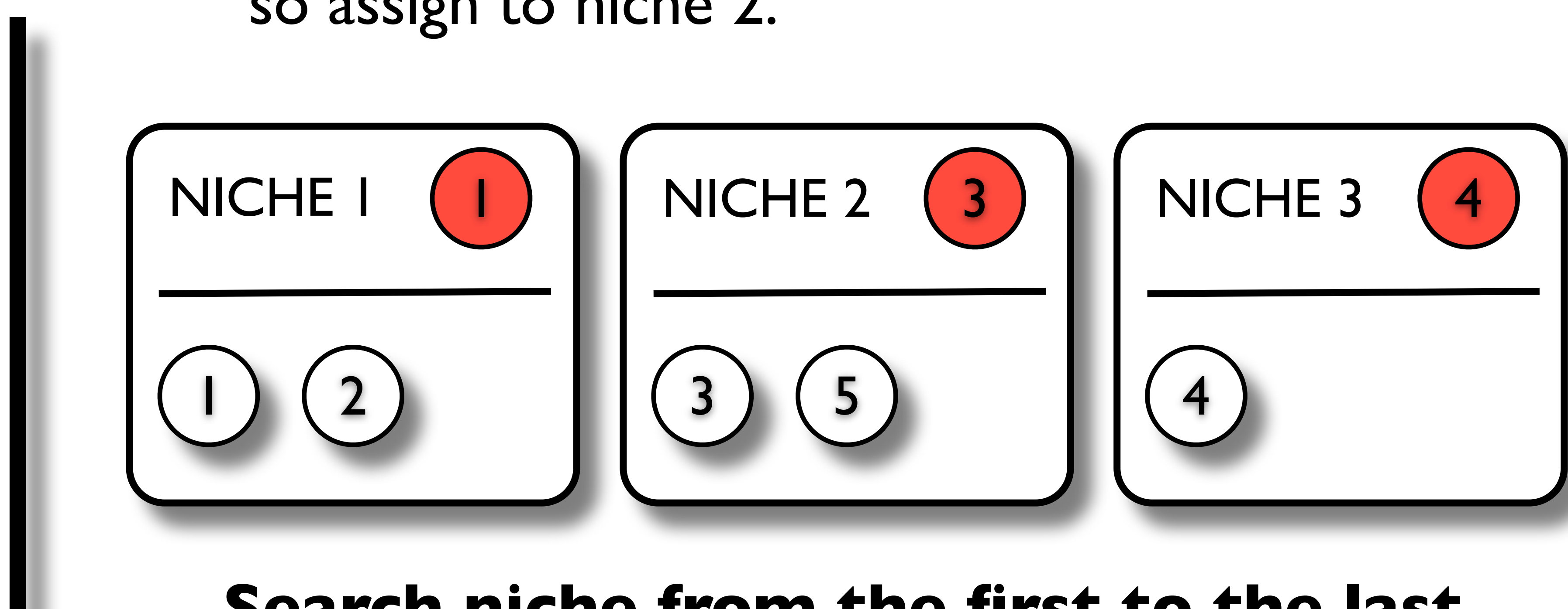
Both d(1,4) $\geq \delta$ and d(2,4) $\geq \delta$,
we had to create a new niche again.



Assign Species

- Explicit Fitness Sharing as in NEAT.
- Species assigned according to distance *d* and threshold δ.

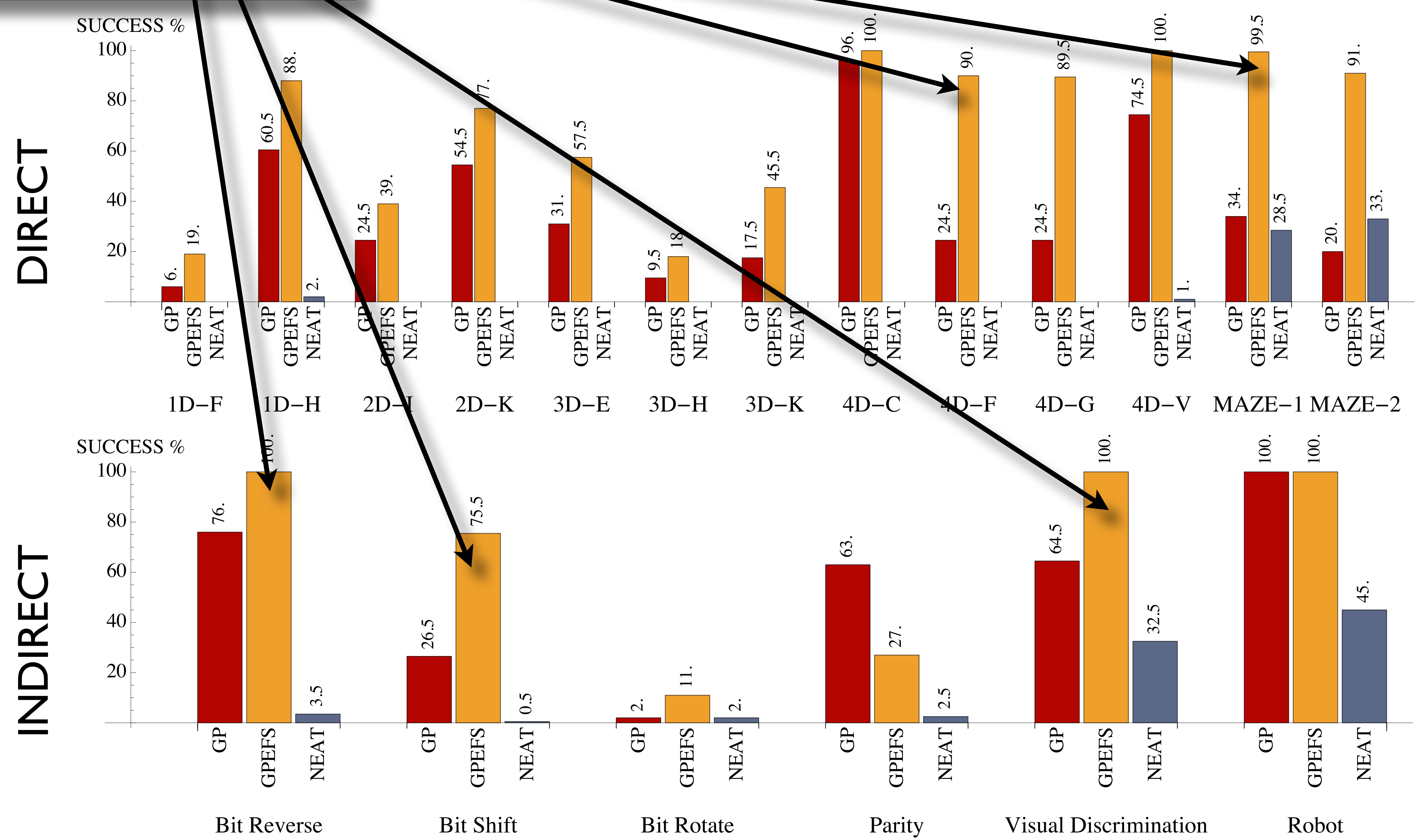
Here, d(1, 5) $\geq \delta$ but d(2, 5) $< \delta$,
so assign to niche 2.



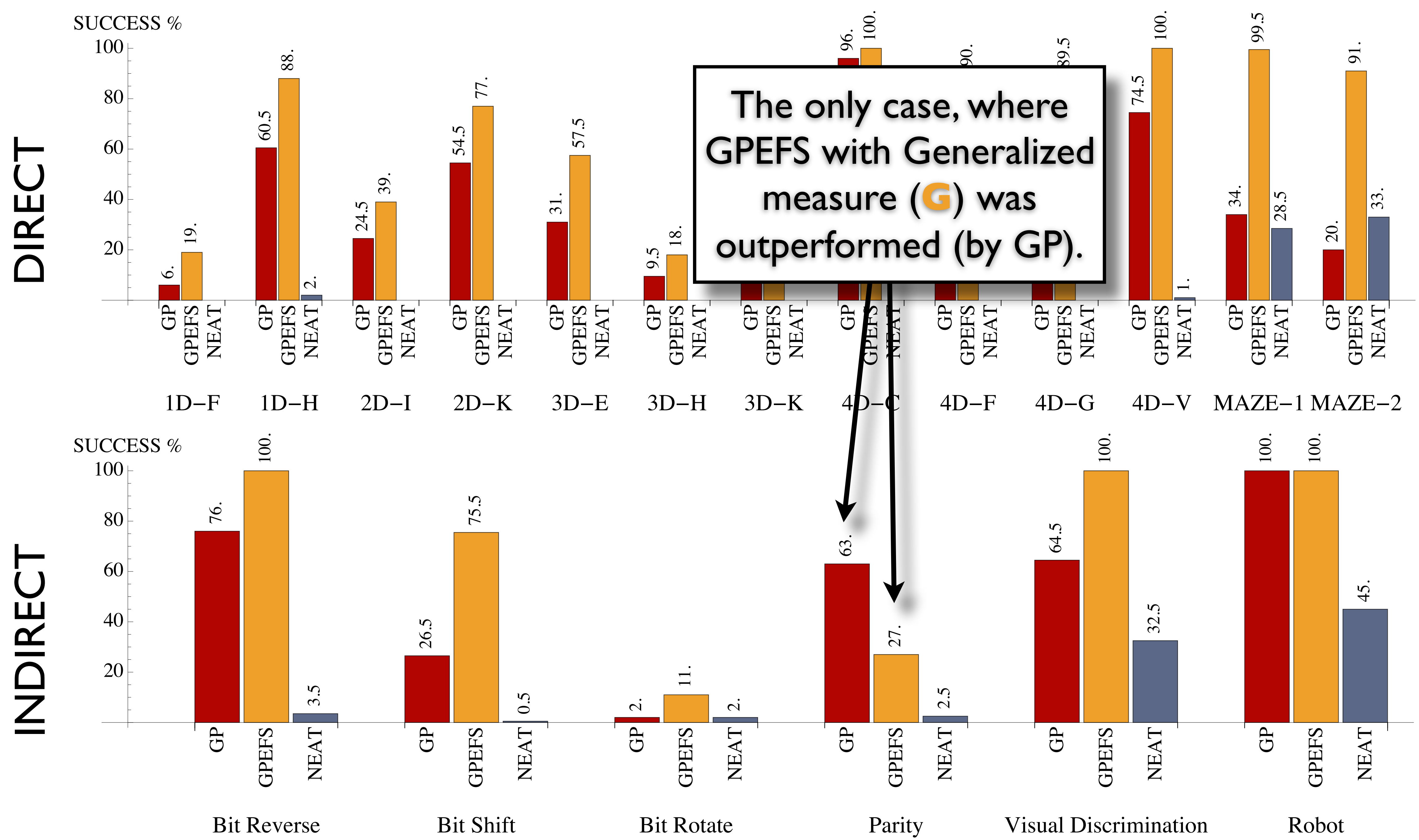
Search niche from the first to the last until sufficiently similar is found. If none such exists, create a new.

GP, GPEFS & NEAT Compared

GPEFS with Generalized measure (G) is the winner (mostly significant).



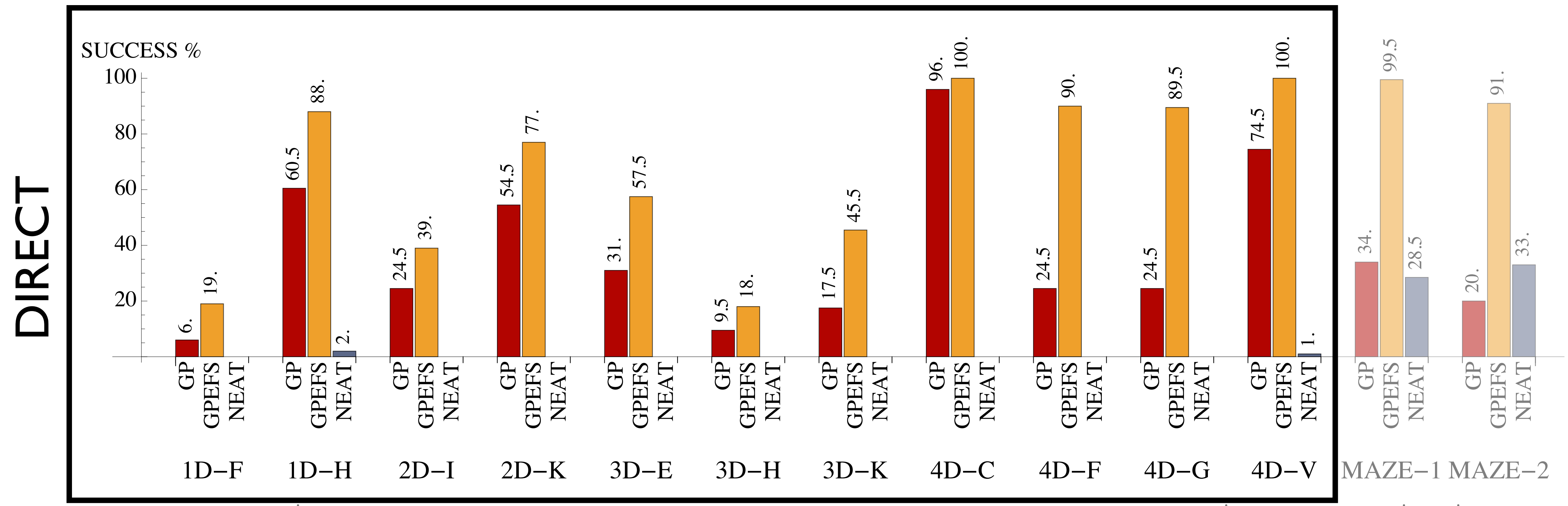
GP, GPEFS & NEAT Compared

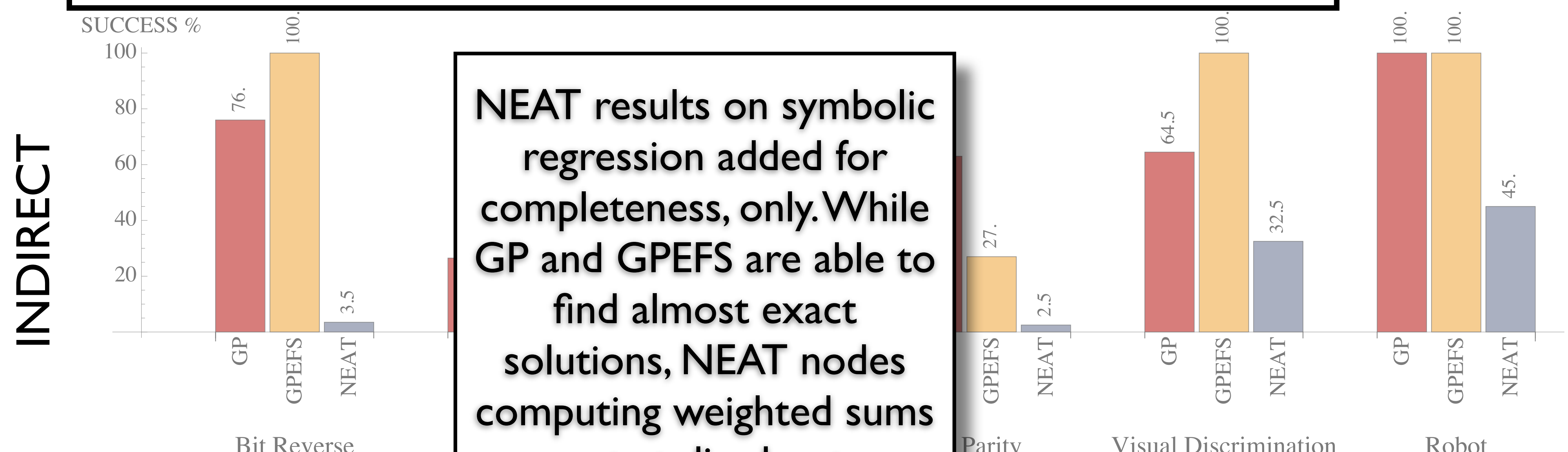


COMPUTATIONAL
INTELLIGENCE
GROUP

2012

GP, GPEFS & NEAT Compared





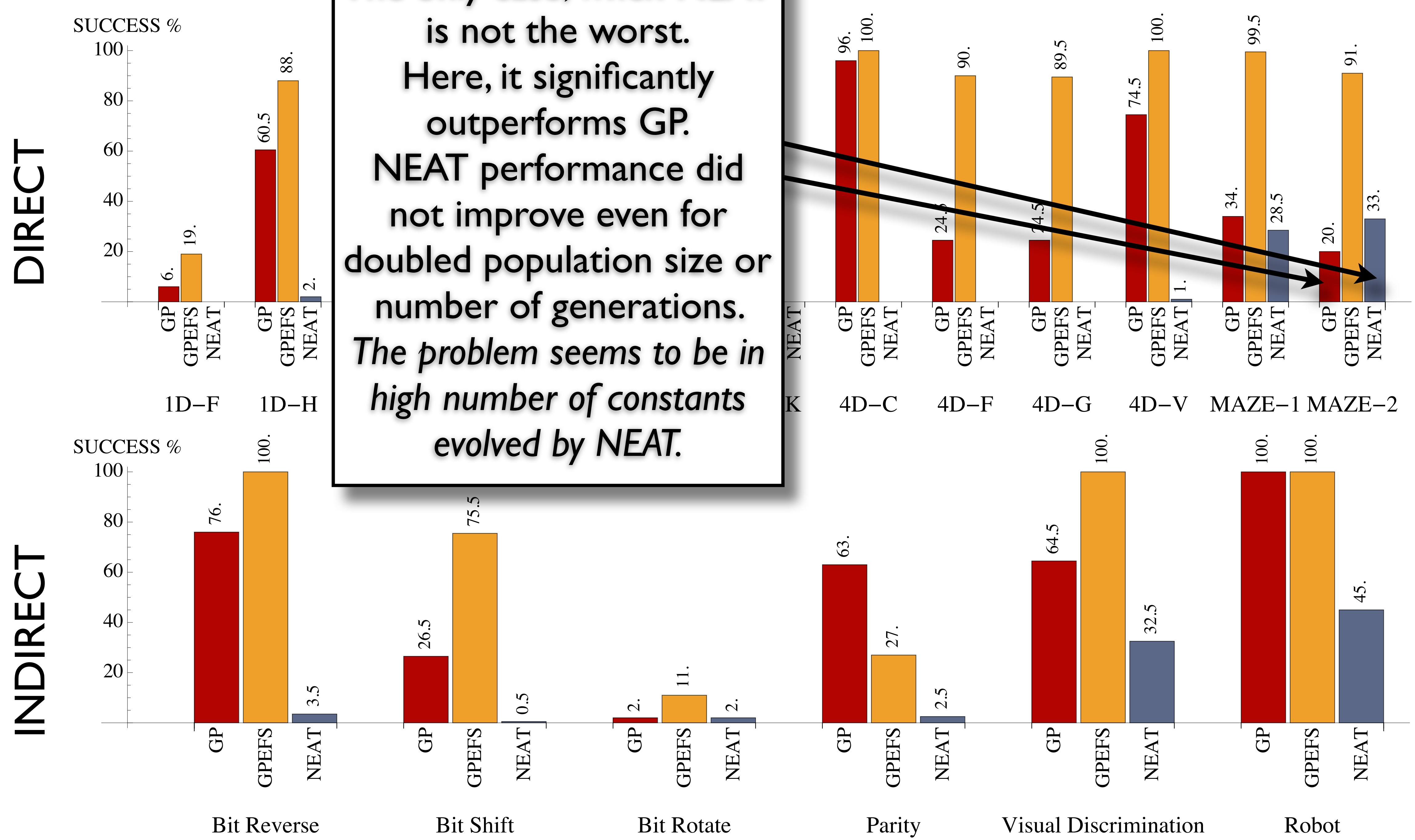
NEAT results on symbolic regression added for completeness, only. While GP and GPEFS are able to find almost exact solutions, NEAT nodes computing weighted sums are put at disadvantage.

COMPUTATIONAL
INTELLIGENCE
GROUP

2012

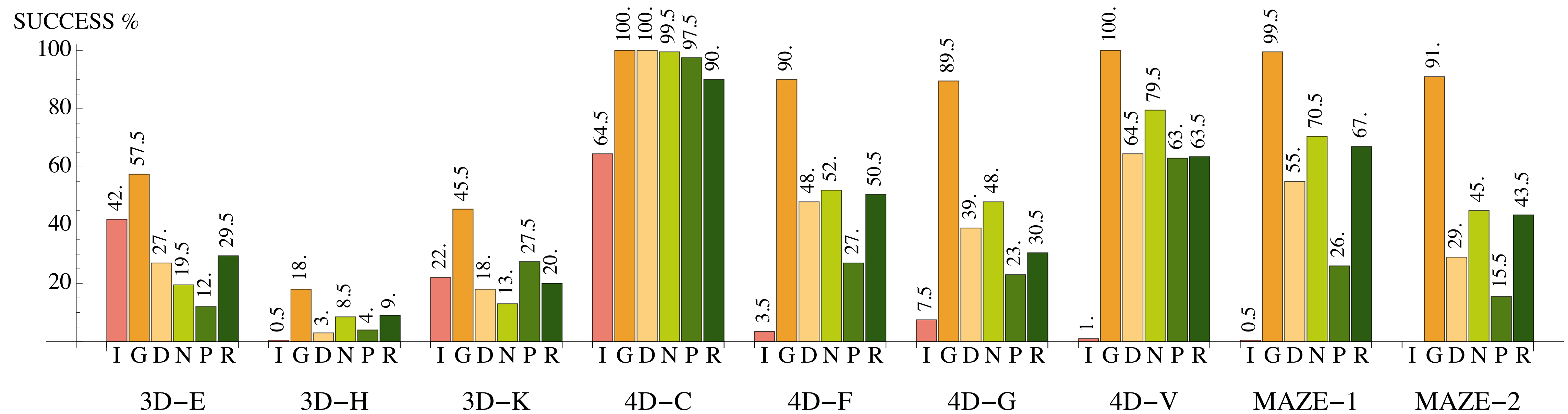
GP, GPEFS & NEAT Compared

The only case, when NEAT is not the worst.
Here, it significantly outperforms GP.
NEAT performance did not improve even for doubled population size or number of generations.
The problem seems to be in high number of constants evolved by NEAT.



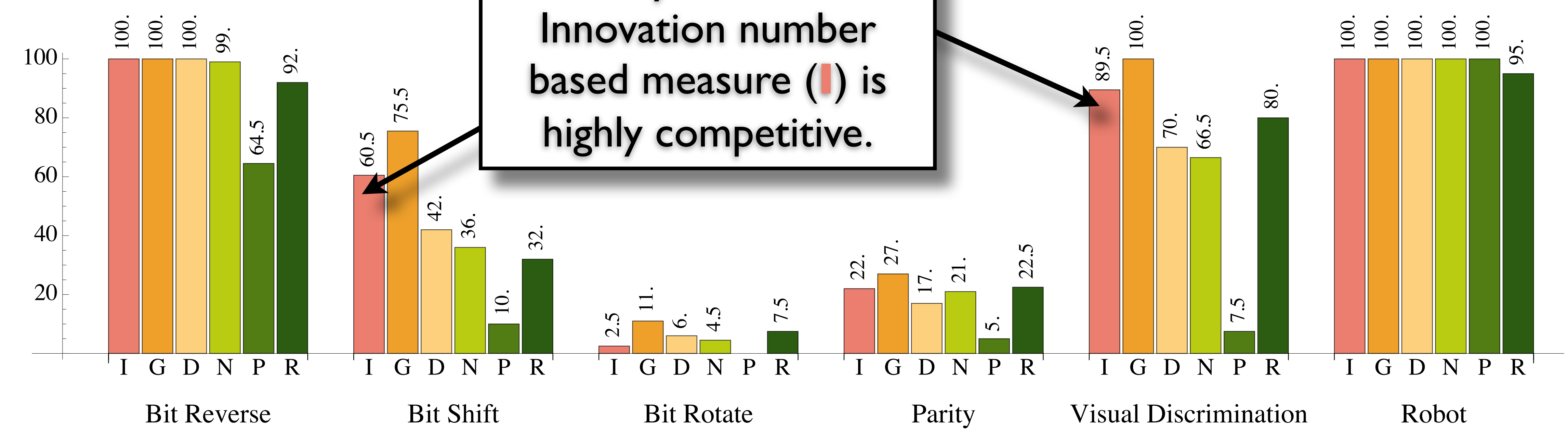
GPEFS: Distance Measures

DIRECT



Unlike for most directly encoded problems, for indirectly encoded the Innovation number based measure (I) is highly competitive.

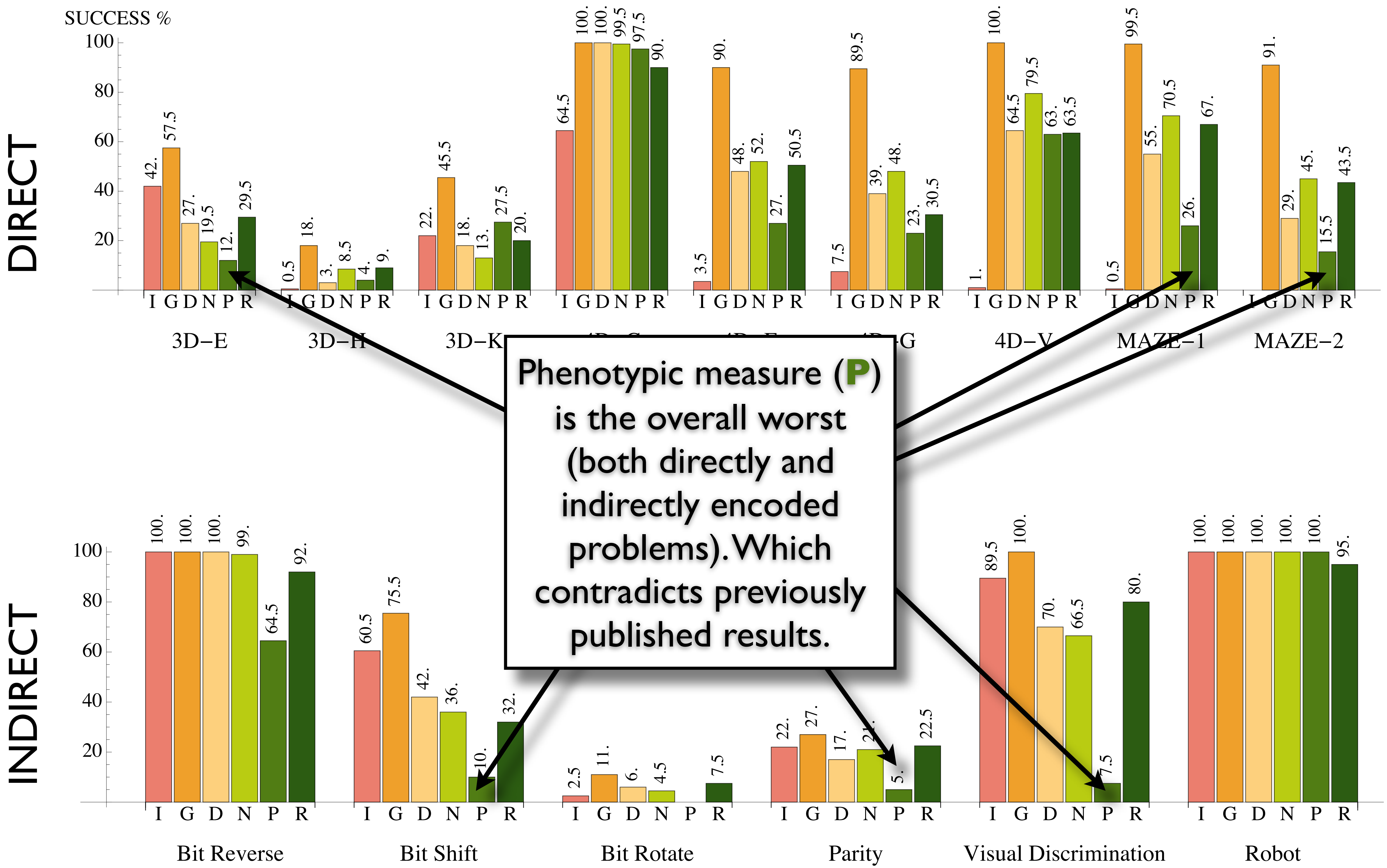
INDIRECT



COMPUTATIONAL
INTELLIGENCE
GROUP

2012

GPEFS: Distance Measures

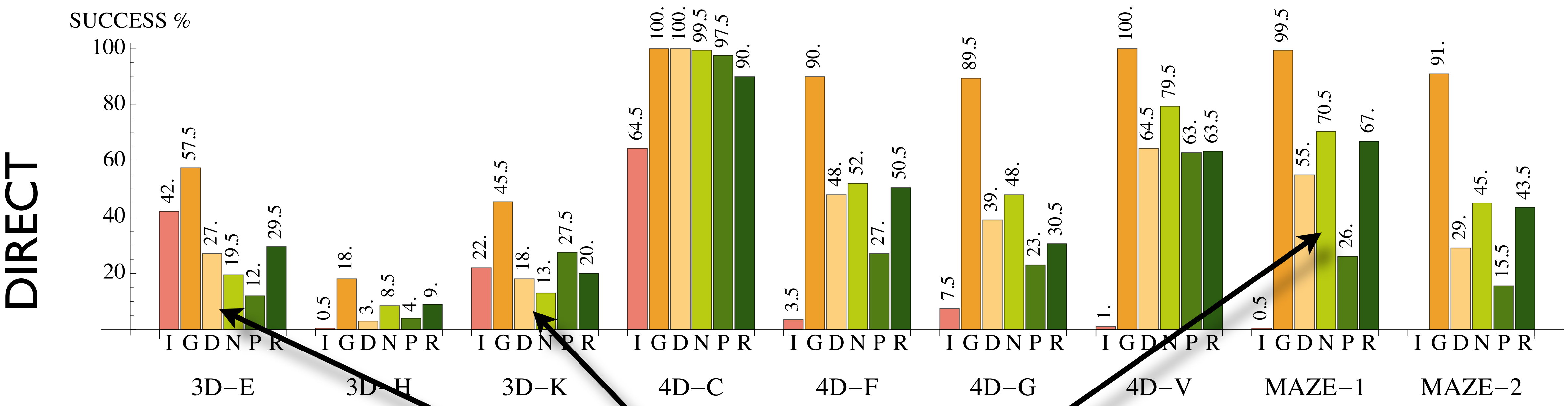


COMPUTATIONAL INTELLIGENCE GROUP

2012

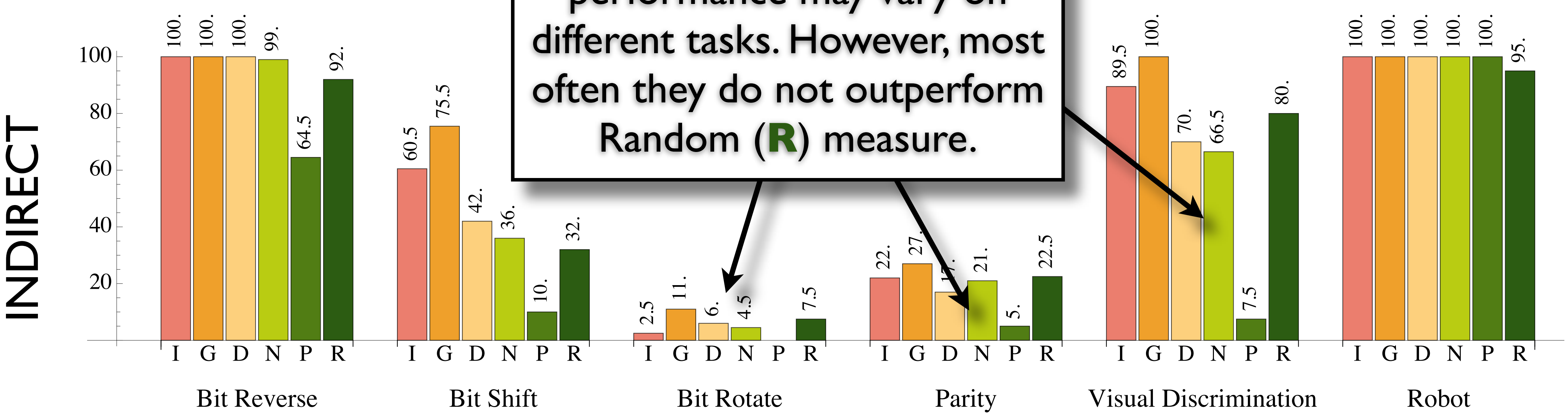
GPEFS: Distance Measures

DIRECT

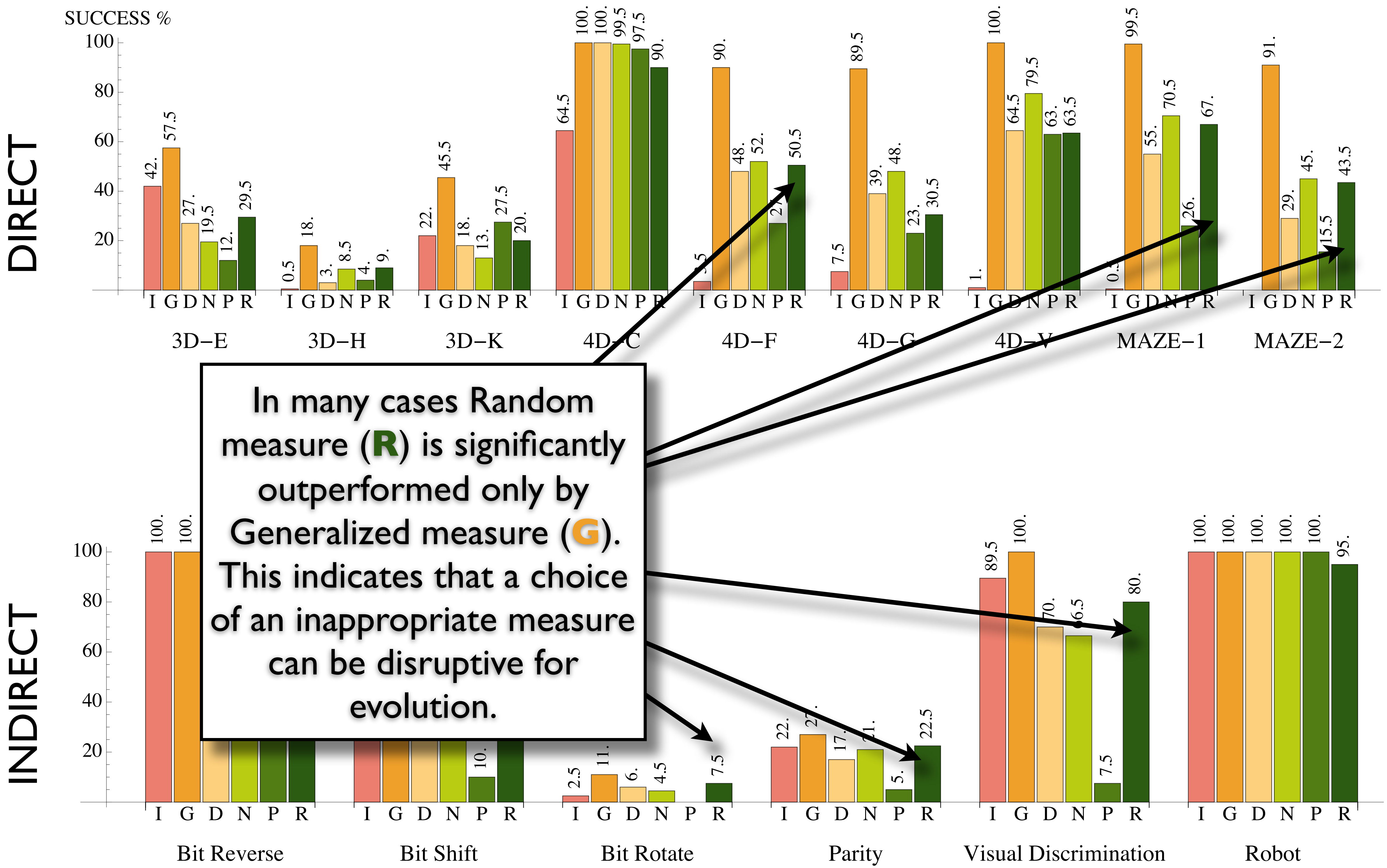


Tree Description (D) and Node Count (N) measures performance may vary on different tasks. However, most often they do not outperform Random (R) measure.

INDIRECT

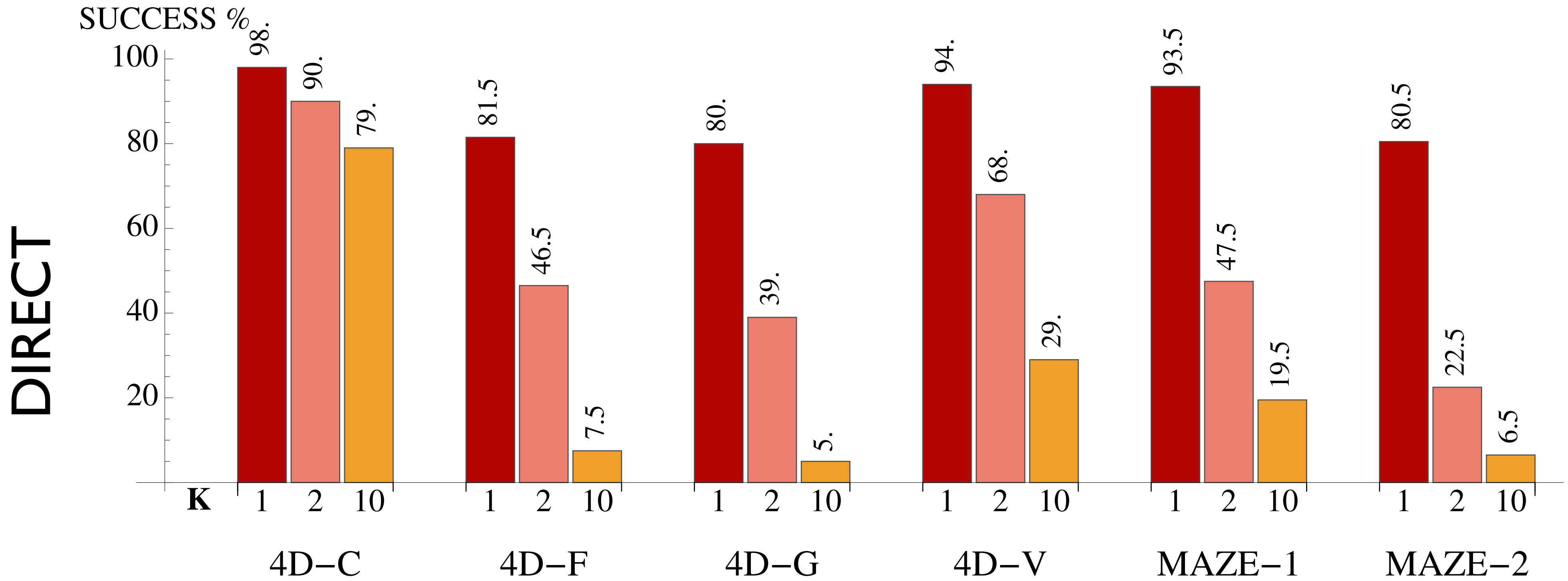


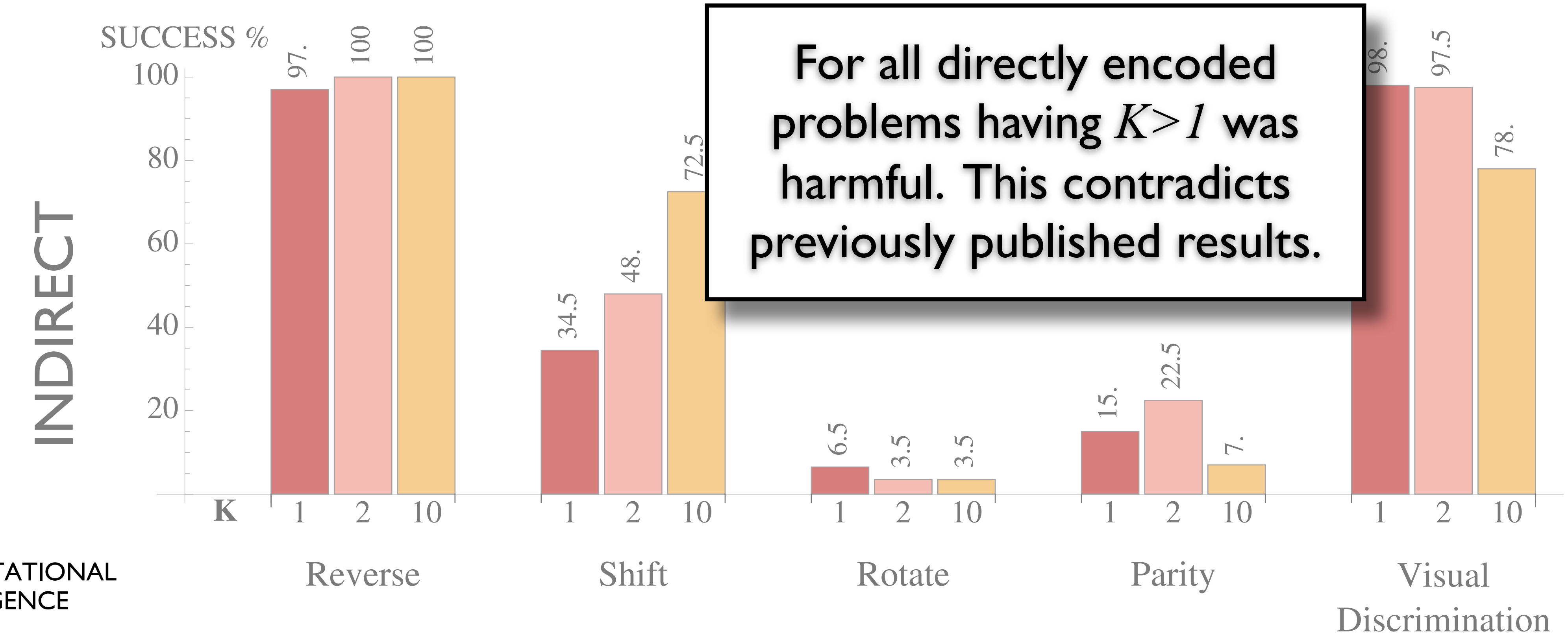
GPEFS: Distance Measures



In many cases Random measure (**R**) is significantly outperformed only by Generalized measure (**G**). This indicates that a choice of an inappropriate measure can be disruptive for evolution.

Generalized Measure Parameters



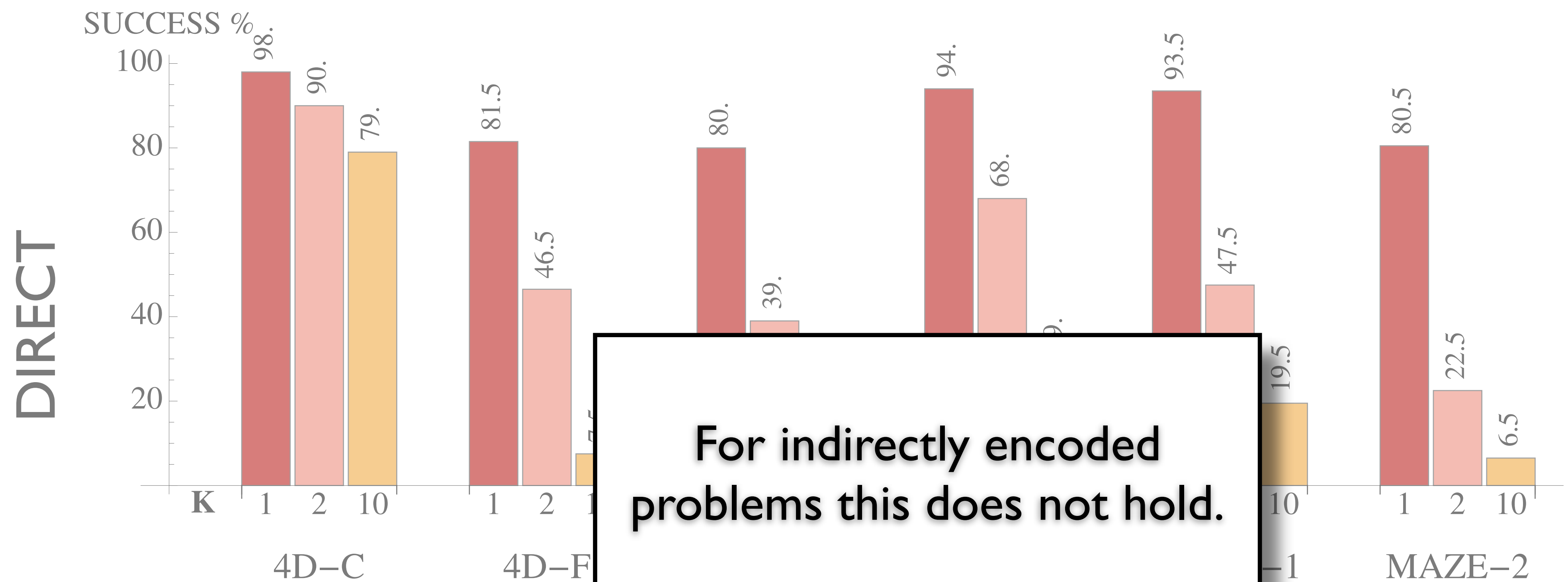


For all directly encoded problems having $K>1$ was harmful. This contradicts previously published results.

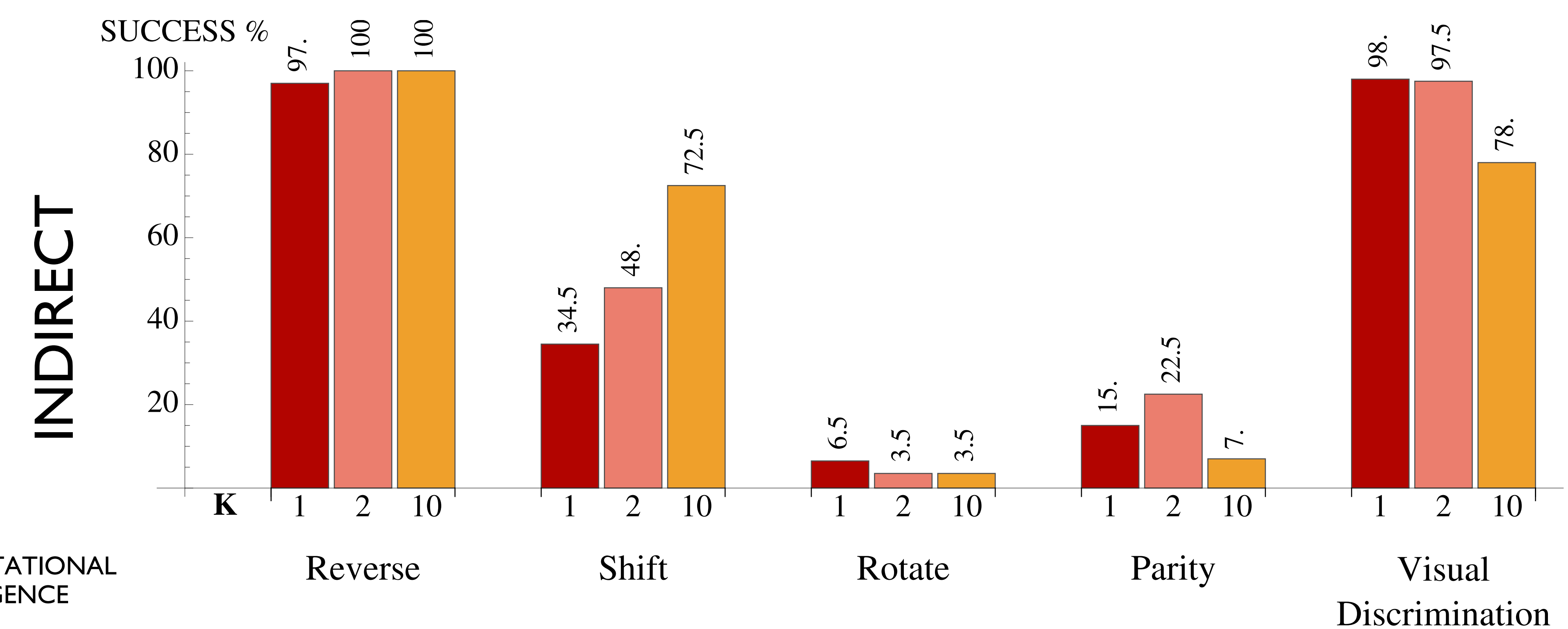
COMPUTATIONAL INTELLIGENCE GROUP

2012

Generalized Measure Parameters



For indirectly encoded problems this does not hold.

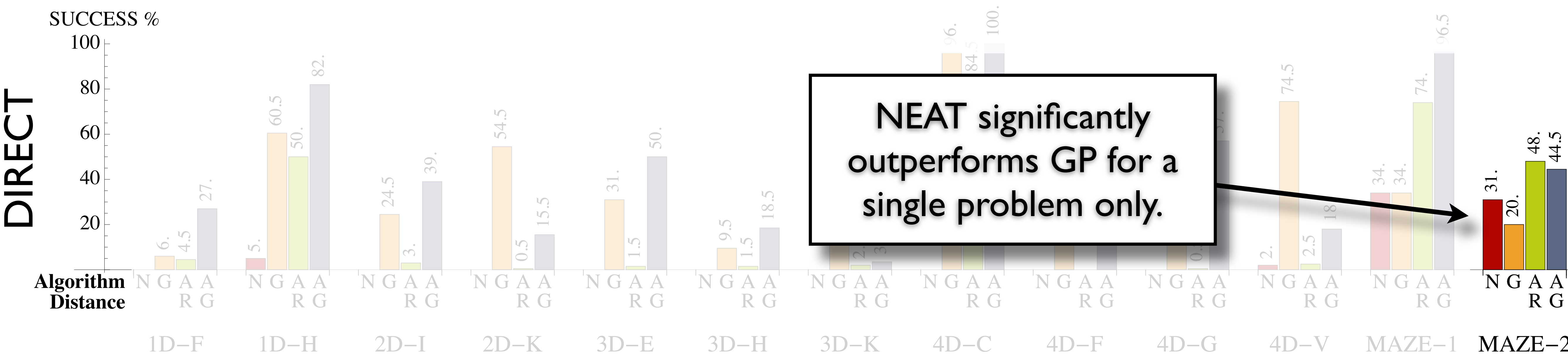


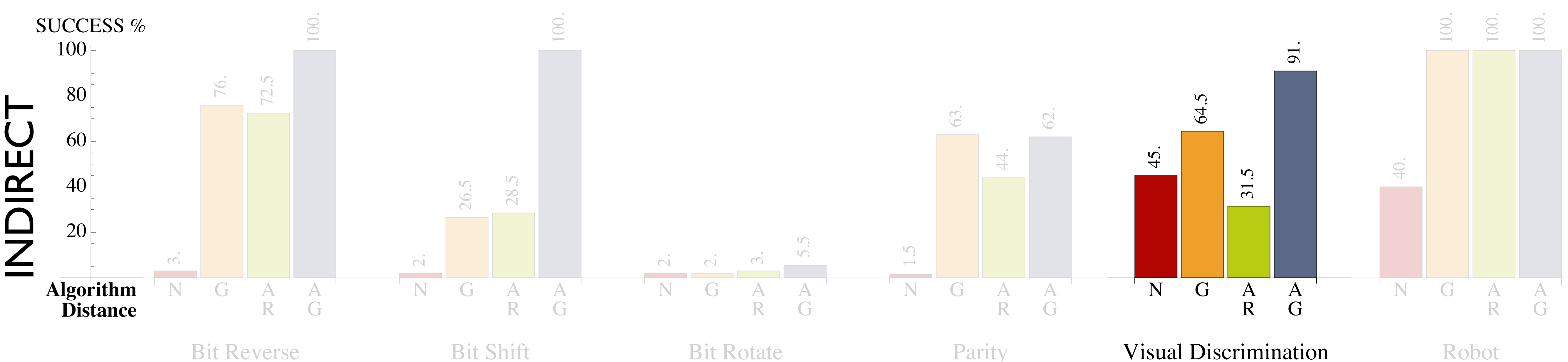
COMPUTATIONAL
INTELLIGENCE
GROUP

2012

GPAT

NEAT, GP & GPAT Compared

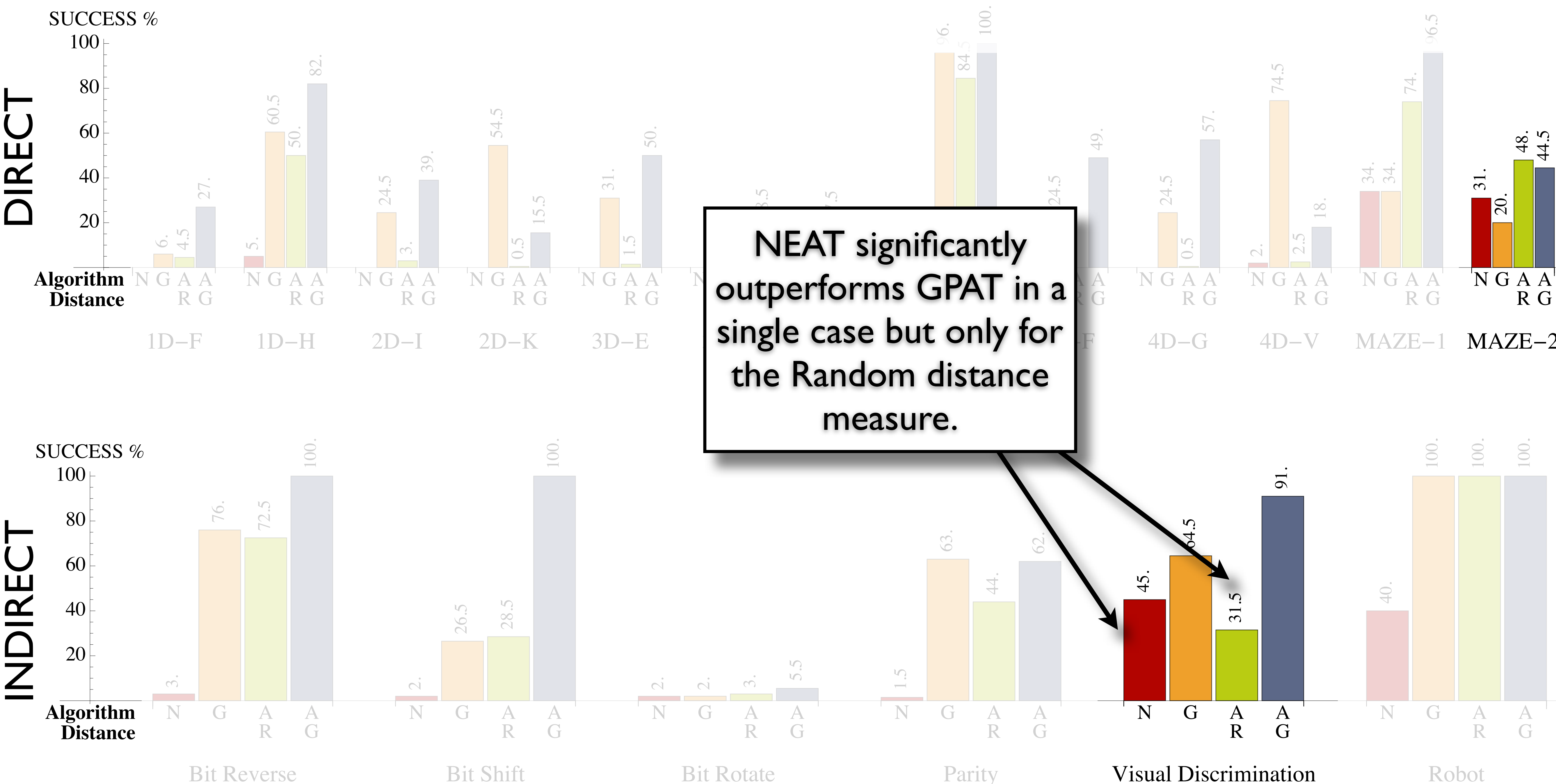




COMPUTATIONAL INTELLIGENCE GROUP

2012

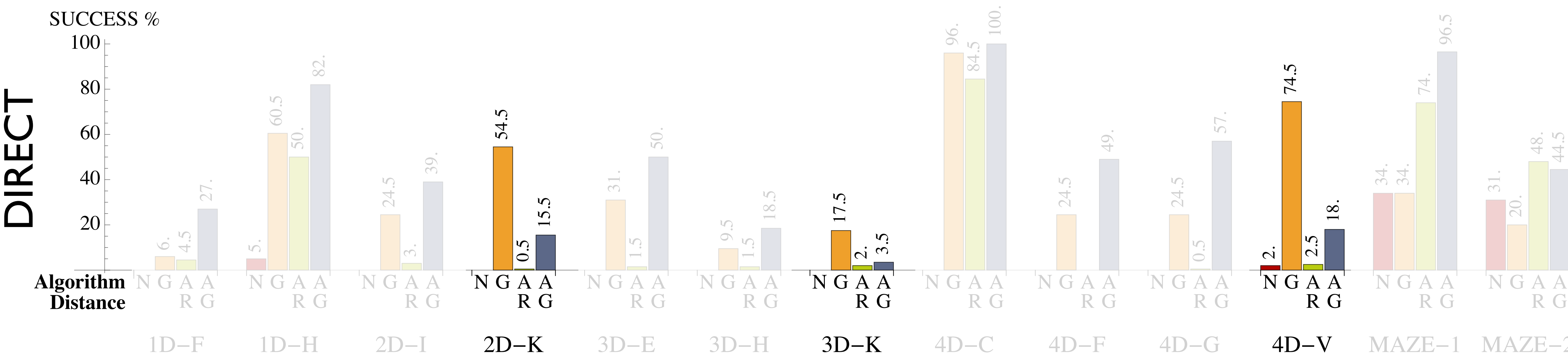
NEAT, GP & GPAT Compared

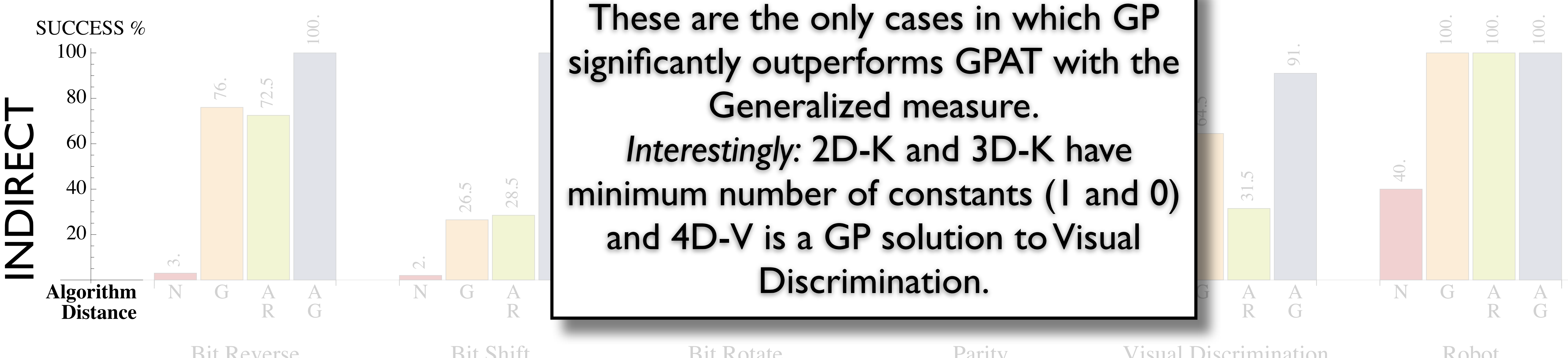


COMPUTATIONAL
INTELLIGENCE
GROUP

2012

NEAT, GP & GPAT Compared



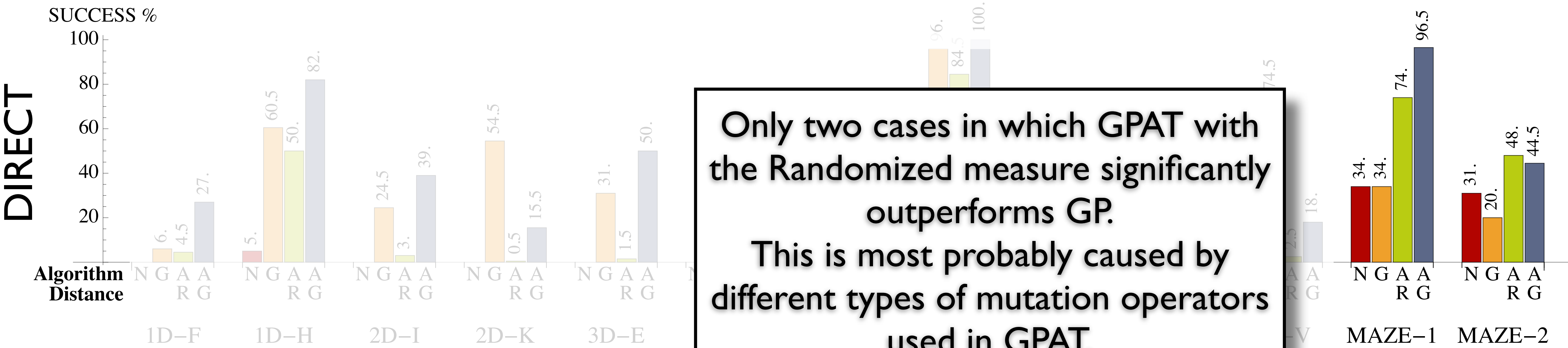


These are the only cases in which GP significantly outperforms GPAT with the Generalized measure.
Interestingly: 2D-K and 3D-K have minimum number of constants (1 and 0) and 4D-V is a GP solution to Visual Discrimination.

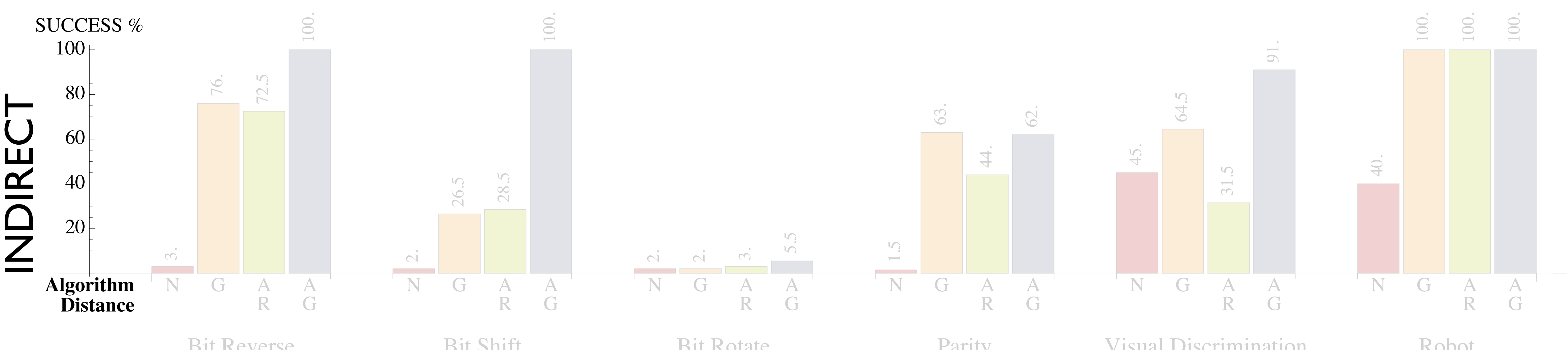
COMPUTATIONAL
INTELLIGENCE
GROUP

2012

NEAT,GP & GPAT Compared



Only two cases in which GPAT with the Randomized measure significantly outperforms GP.
This is most probably caused by different types of mutation operators used in GPAT.

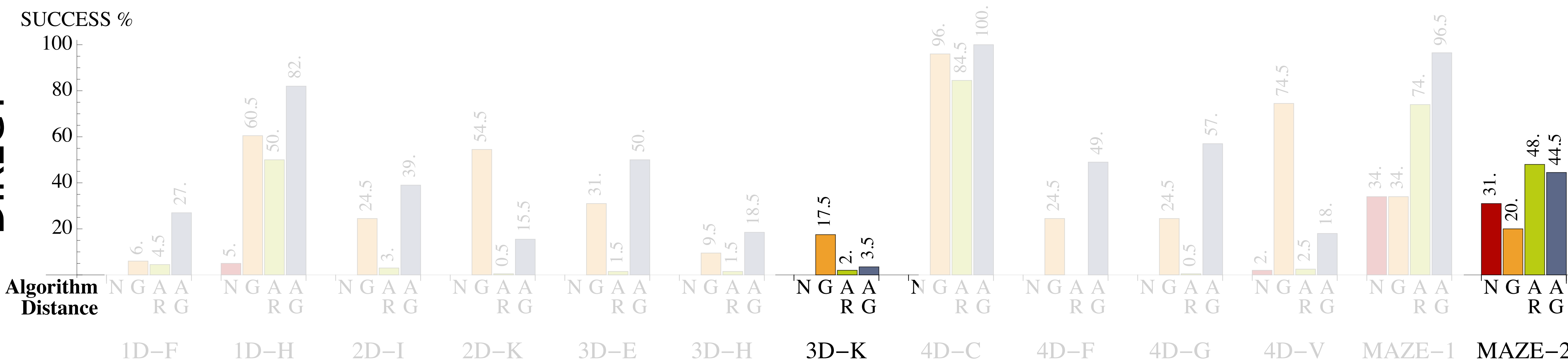


COMPUTATIONAL
INTELLIGENCE
GROUP

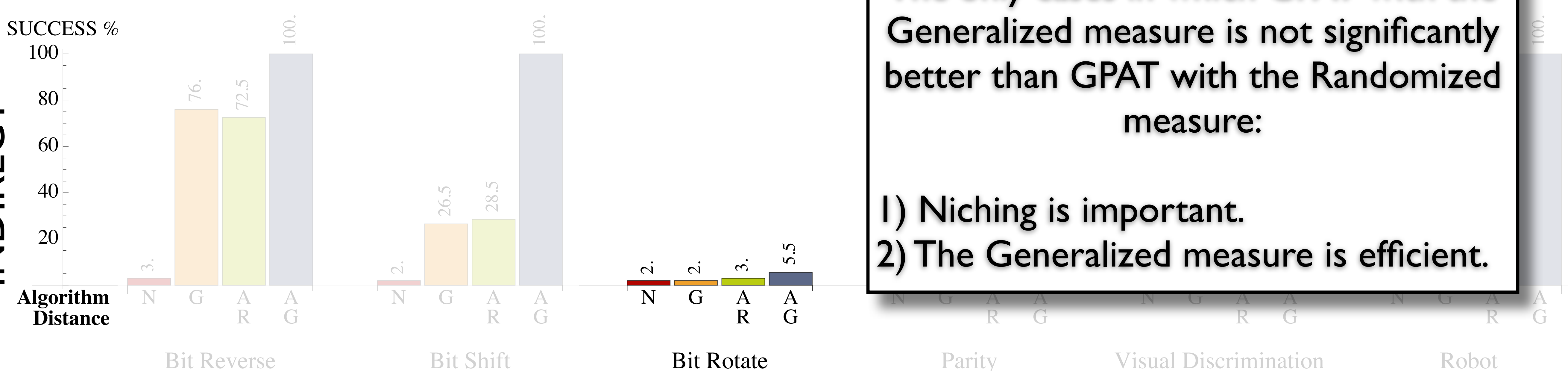
2012

NEAT,GP & GPAT Compared

DIRECT



INDIRECT



The only cases in which GPAT with the Generalized measure is not significantly better than GPAT with the Randomized measure:

1) Niching is important.
2) The Generalized measure is efficient.

COMPUTATIONAL
INTELLIGENCE
GROUP

2012