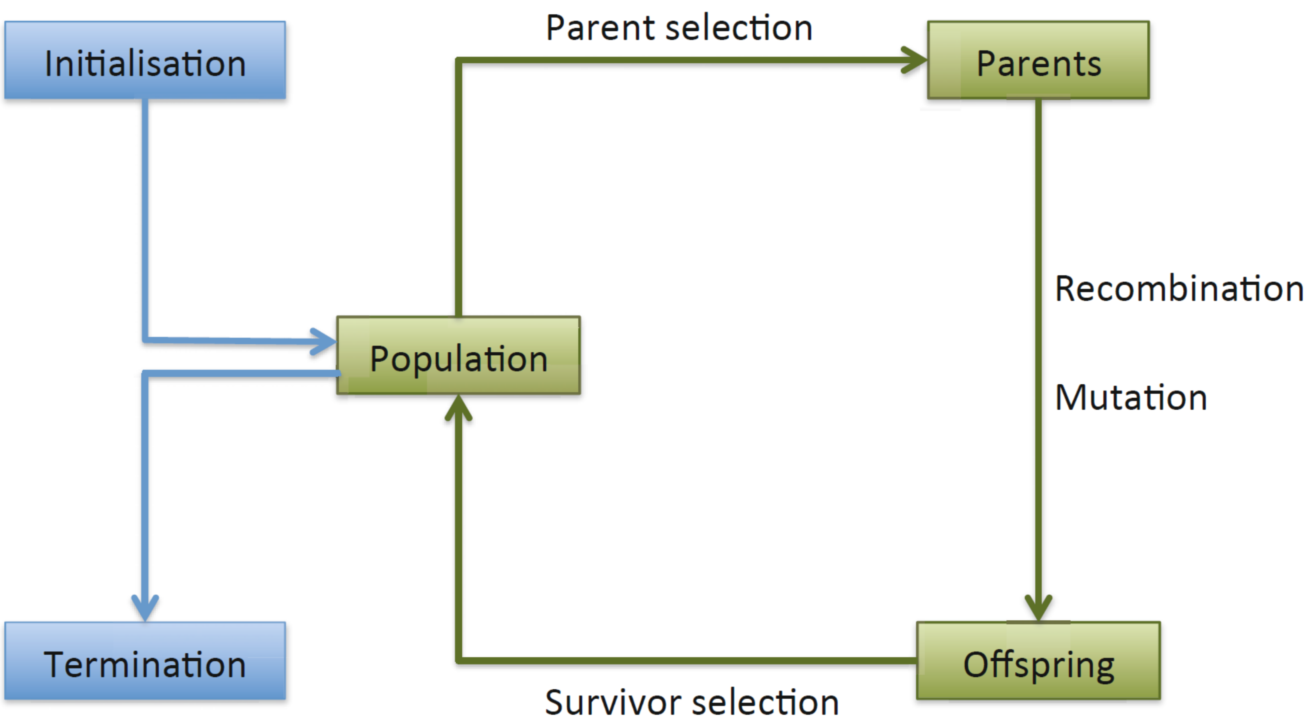
NEAT

Neuroevolution of augmenting topologies

Contens

- Evolutionary computation
  - basic idea, parts of EA
- Neural Network
  - Percepton, Connections, Activation functions
- NEAT
  - Introduction, Encoding, Species, Mutation, Crossover, Performance, Verification
- Examples of NEAT
  - Sharp-NEAT
    - Xor Black box
    - Pendulum

Evolutionary Computation



Evolutionary Computation

- Initialization
  - Random creation of candidates solution

- Candidate
  - Genotype vs. Fenotype
  - Encoding
  - Population
  - Generation
  - Offspring

Evolutionary Computation

- Evalution of the solution
  - Termination condition
  - Fitness value
  - Problem, Enviroment

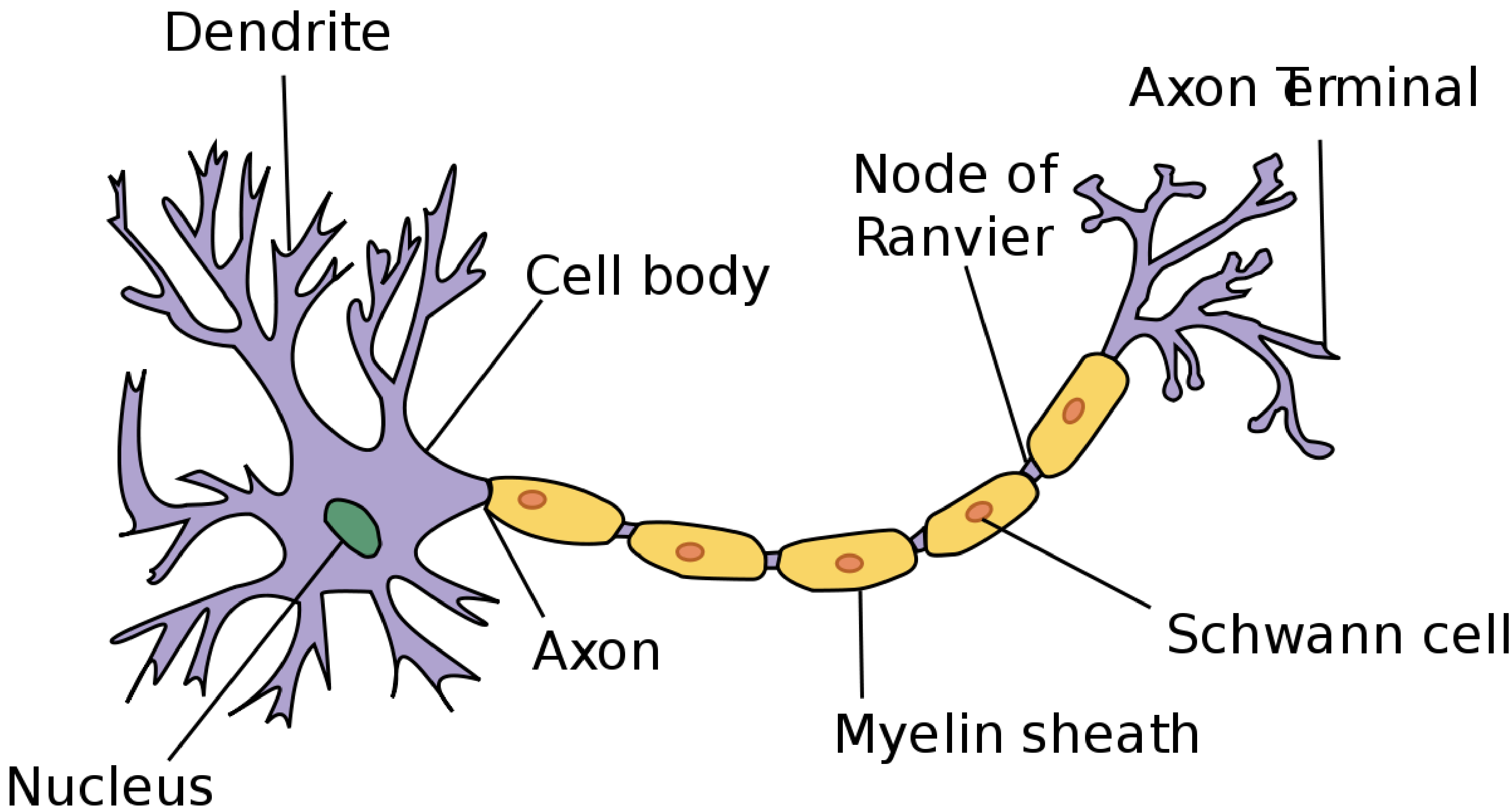
- Selections
  - Parental Selection
  - Enviroment(Survivor) Selection

Evolutionary Computation

- Variation operators
  - Rekombination(Crossover) operator
  - Mutation operator

- History
  - Genetic algorithm (c: bit string)
  - Evolution strategies (c: vector of real numbers)
  - Evolutionary programming (c: finite automata)

Neural Network



Neural Network

- Neuron
  - Percepton, Dedrites, Axon
- Neural Network
  - Deep networks, input, hidden, output layer
- Activation functions
  - Step func, Linear, tanh(x) etc.

NEAT – Introduction

- Presented in Evolving Neural Networks through Augmenting Topologies
  - Introduced by:
    - Kenneth O. Stanley (Austin)
    - Risto Miikulainen (Austin)
- NEAT = NeuroEvolution of augumenting topologies
- Evolving topologies along weights

Authors

NEAT – Why?

- NE of fully connected topologies
  - NEAT is faster
- NE of fixed topologies
  - Neat do not require decission before NE
  - Neat can not so easily stucked
- NEAT topologies attempt to stay small

NEAT – TWEANN

- Topology and Weight Evolving Artificial Neural Networks
- ENCODING:
  - Direct:
    - Binary(connection matrix, linear string of bits to represent graph)
    - Graph(dual representation) [for subgraph crossover]
  - Indirect:
    - Cellular encoding(graph encoding language)
- NONMATING in TWEANN

TWEANN – Encoding Problems

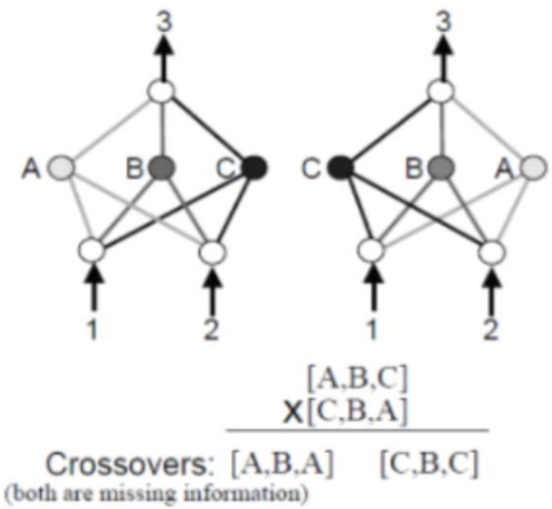


Figure 1: The competing conventions problem. The two networks compute the same exact function even though their hidden units appear in a different order and are represented by different chromosomes, making them incompatible for crossover. The figure shows that the two single-point recombinations are both missing one of the 3 main components of each solution. The depicted networks are only 2 of the 6 possible permutations of hidden unit orderings.

TWEANN – Encoding Problems

- Permutation of nodes & same topologies
  - damaged offspring
- Different topologies & similar solution
  - Memory requirements, unnecessary complex solutions
- Homology
  - E Coli motivation
  - NEAT : historical origin of node

TWEANN– Protecting Innovation

- New structure
  - By mutation
  - Not optimal from creation

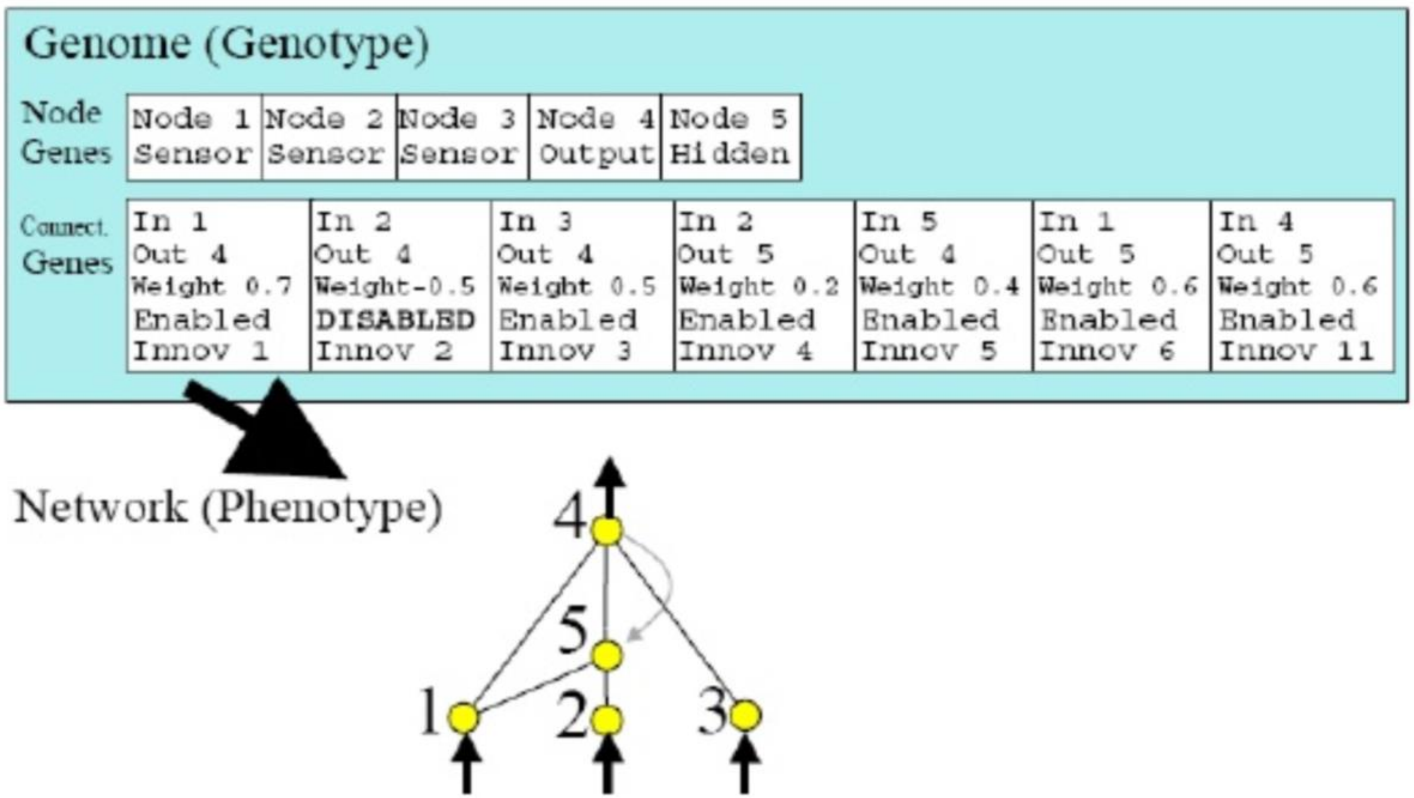
- Species
  - „Nitching“ (koutek, místečko)
  - Not compete with population at large
  - How to select species
    - NEAT – fitness sharing

NEAT – Init Population & Topological inovation

- TWEANN –
  - random topologies from beginning
    - incorrect topologies
    - Non minimal solutions

    - Fitness penalization
- NEAT
  - From minimaly solution
  - Specing

NEAT – encoding



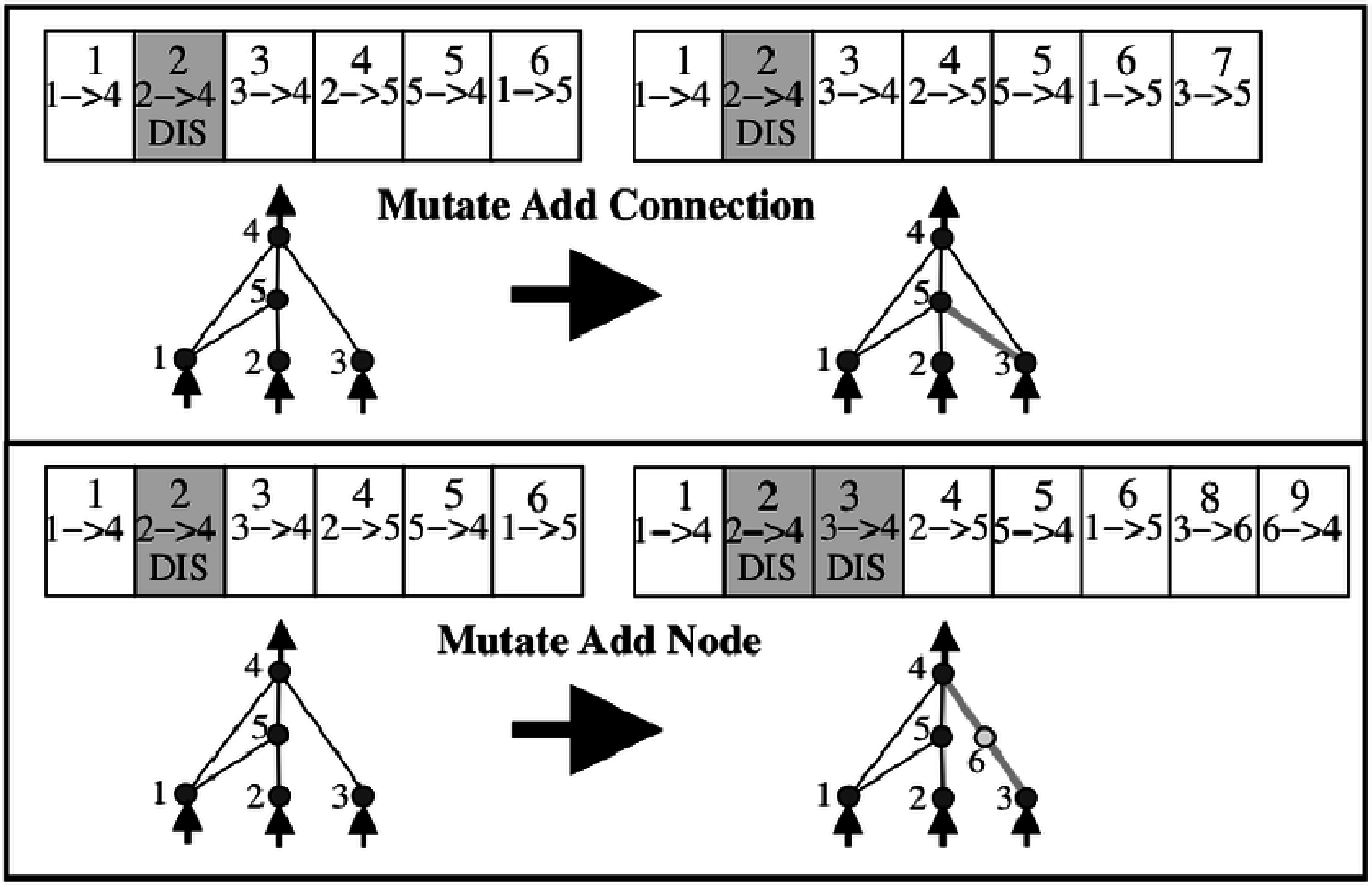
NEAT – Encoding

- Genome
  - List of network connectivity
  - List of network nodes (input, output, hidden)
- Node
  - Number of node
  - Input(Sensor), Hidden, Output
- Connection
  - In-node, out-node
  - weight
  - enable-bit
  - Innovation number

NEAT – Mutation

- Can change weights, network structures
- Same for weights (even if perturbed, not in all)
- Changes in structure
  - Add connection
  - Add node

NEAT – mutation structure

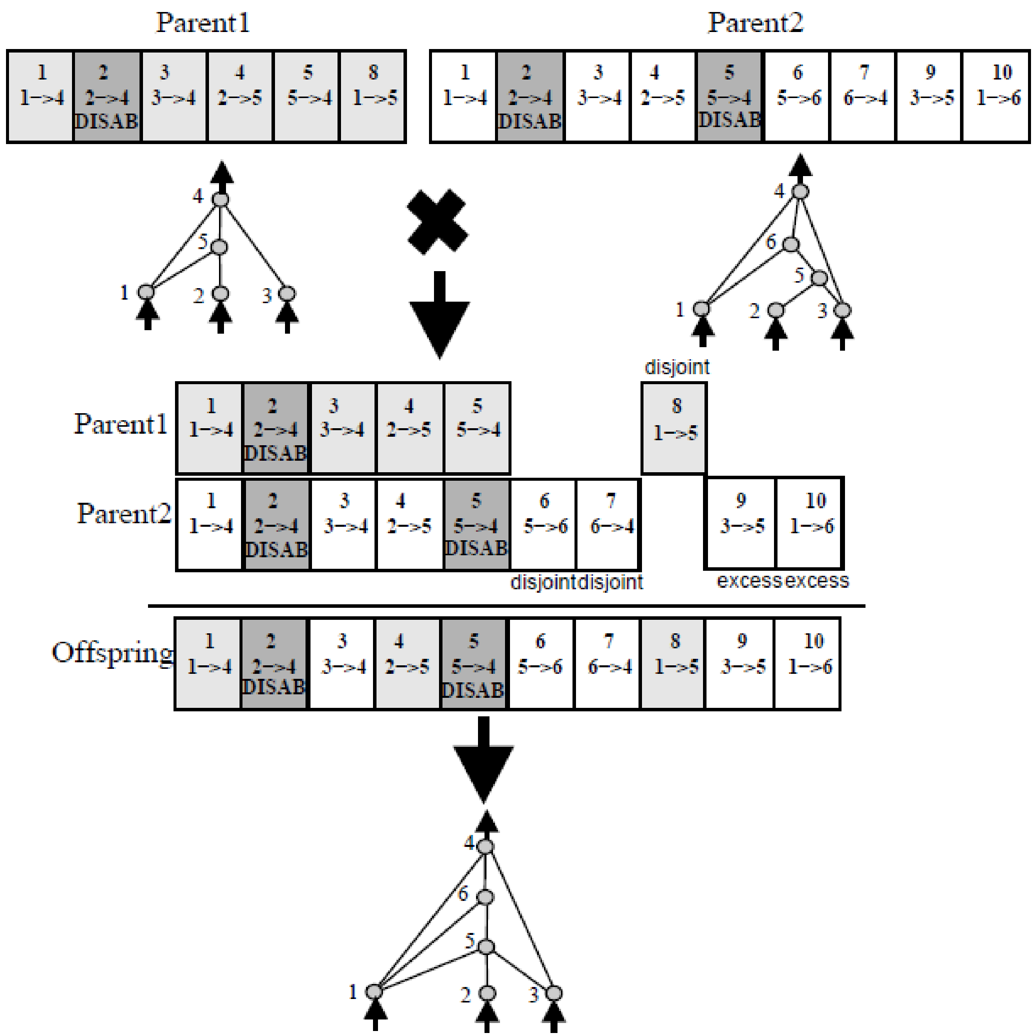


NEAT – tracking genes

- Historical markings
  - Global innovation number
  - No changes after crossover
  - Easy genes matching

- Not matching genes
  - Excess, dissjoint – from more fit parent

NEAT – Crossover



Protecting Innovation

- Speciating – niche competion
- Niche = topology matching
  - By historical markings
  - Compability distance
    - number of excess & disjoints
  - Representing by one random genome from previous generation

$$\delta = \frac{c_1 E}{N} \frac{c_2 D}{N} + c_3 \overline{W}.$$

NEAT – Fitness sharing

- Same species share fitness
- Explicit fitness sharing
- Threshold:
  - Distance between i,j
  - sh function:

$$f_i' = \frac{f_i}{\sum_{j=1}^{n} sh(\delta(i,j))}$$

$$\delta(i,j) = 0$$

$$\delta(i,j) > \delta_t \implies sh(\delta(i,j)) = 0$$

$$\delta(i,j) < \delta_t \implies sh(\delta(i,j)) = 1$$

NEAT – Incremental Grow

- From minimal candidate
- All inputs connected directly to the outputs
- Survival only of the useful through fitness evaluation

NEAT – Validation

- XOR Network
  - Does NEAT evolve something?
- POLE balancing test
  - Does NEAT work efficiently than other NE?
- Sharp-NEAT
  - http://sharpneat.sourceforge.net/
  - https://github.com/colgreen/sharpneat

SharpNEAT
Evolution of Neural Networks

Videos:

MARIO

- https://www.youtube.com/watch?v=qv6UVOQ0F44

Flappy Birds

- https://www.youtube.com/watch?v=L6bbFgjkqK0

Reference:

A. E. Eiben and J.E. Smith, Introduction to Evolutionary Computing, Springer, First edition, 2003, ISBN 3-540-40184-9

Kenneth O. Stanley; Bobby D. Bryant & Risto Miikkulainen (2003). "Evolving Adaptive Neural Networks with and without Adaptive Synapses" . *Proceedings of the 2003 IEEE Congress on Evolutionary Computation (CEC-2003*

Reference:

STANLEY KO a MIIKKULAINEN R.
Evolving neural networks through augmenting topologies.
Evolutionary Computation[online].
2002, **10**(2), 99-127 [cit. 2017-10-30]. ISSN 10636560.
Available:
http://nn.cs.utexas.edu/downloads/papers/stanley.ec02.pdf