# NeuroEvolution of Augmenting Topologies (NEAT)

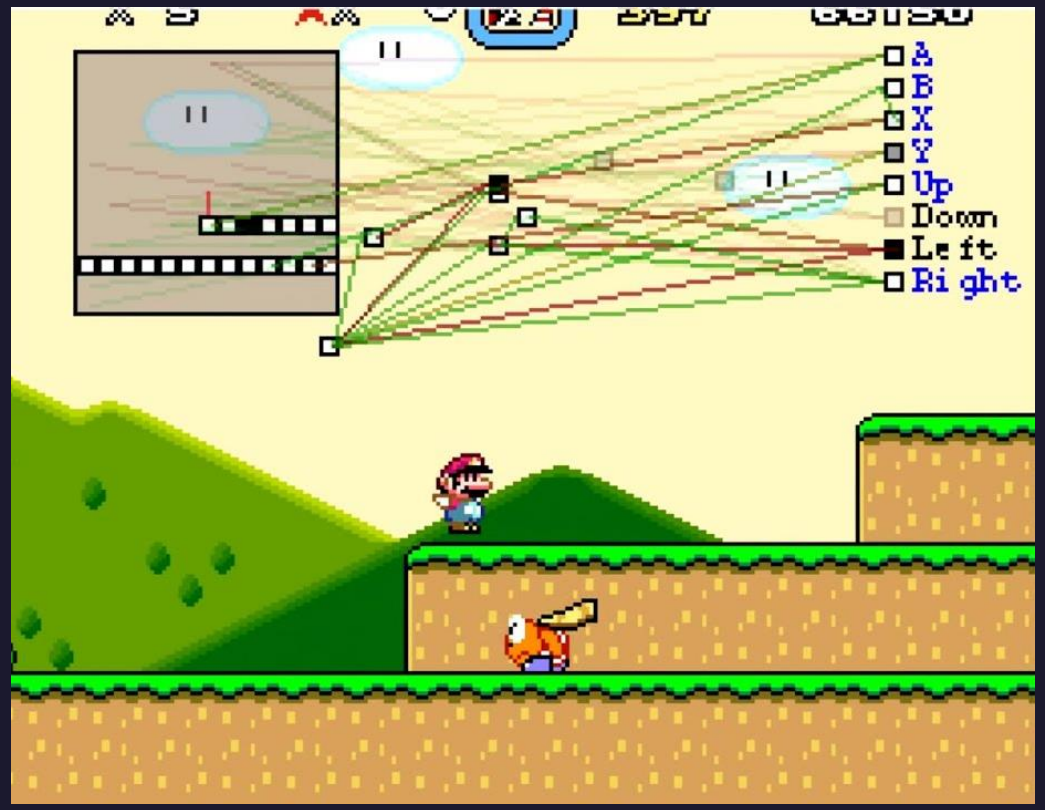
Silviu Stăncioiu

# Motivație

- Probleme generale de machine learning (ex: XOR)
- Pole Balancing
- Agenți virtuali

# Exemplu

- It's a me, Mario!

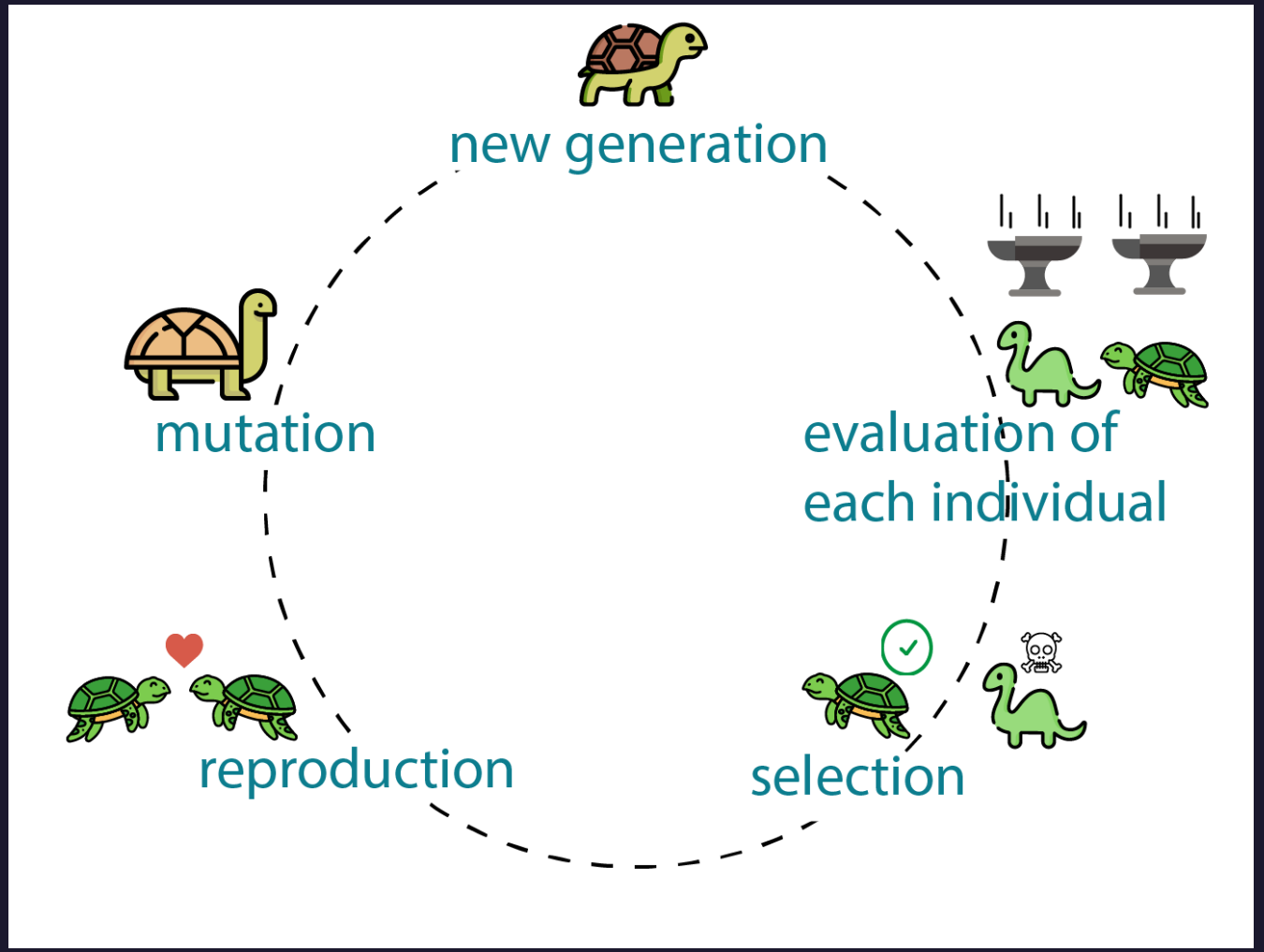



https://youtu.be/qv6UVOQ0F44

# Algoritmi genetici

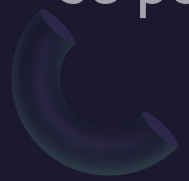
- Inspirați din evoluția naturală
- Se pornește de la o populație de indivizi codificați într-un anumit fel.
- Se măsoară performanța acestora și li se atribuie un scor (numit **fitness**).
- Cei mai buni indivizi sunt selectați, iar genele lor sunt combinate pentru a forma noi indivizi (**crossover**).
- Codificările noilor indivizi sunt modificate puțin pentru a avea diversitate (**mutații**).
- Când populația nouă este pregătită, aceasta înlocuiește populația veche.

# Algoritmi genetici



# Algoritmi genetici - optimizări

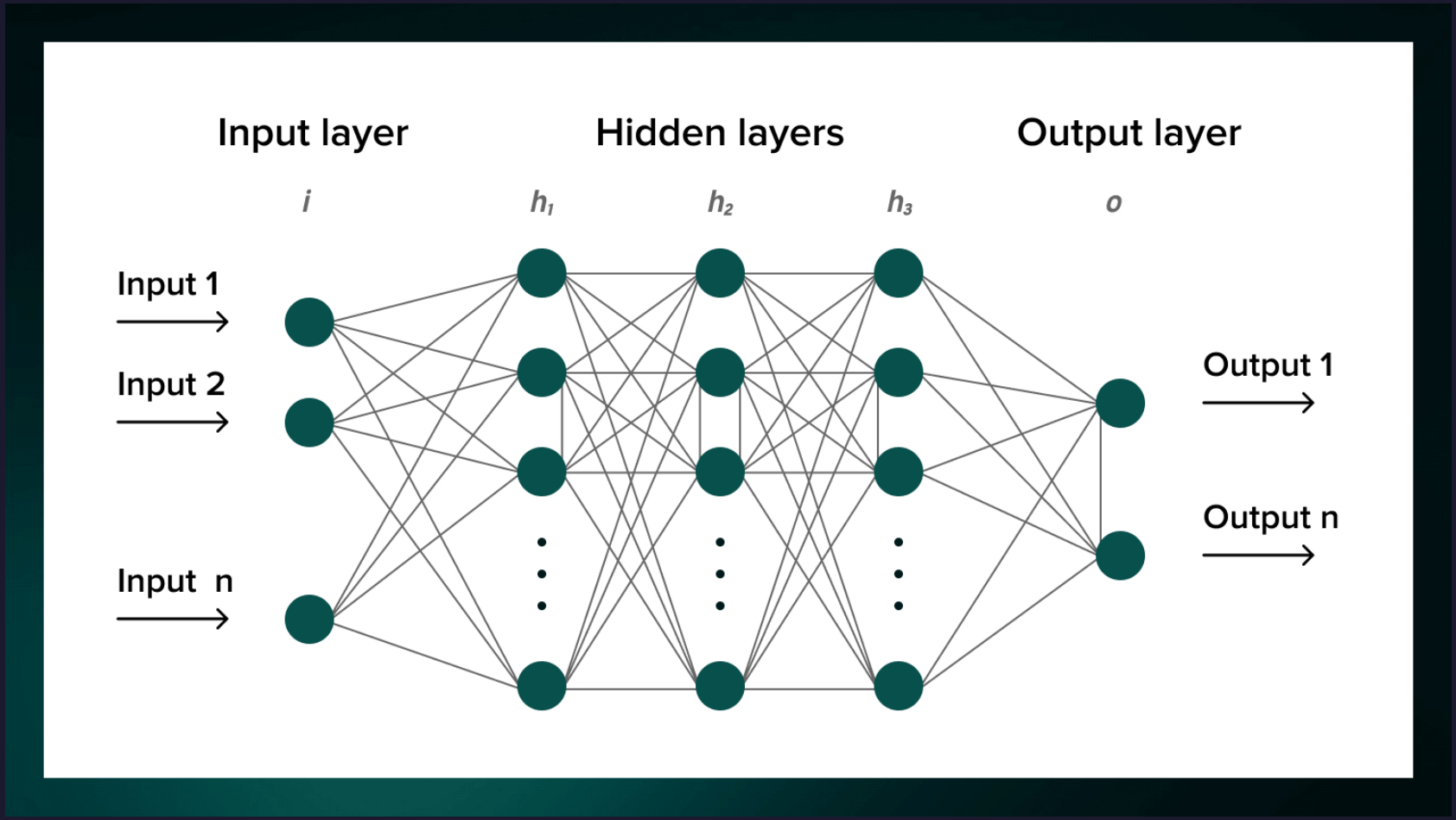
- Se pot păstra copii ale celor mai buni indivizi de la o generație la alta pentru a asigura că **fitness**-ul nu scade.
- Se pot introduce noi indivizi aleatori în populație pentru a păstra diversitatea.
- Se pot ajusta fitness-urile indivizilor după ce evaluarea lor a luat sfârșit.
- Se pot schimba parametrii de evoluție în timpul rulării.
- Se poate schimba mediul în care agenții sunt evoluați.



# Rețele neuronale

- Grafuri formate din noduri (neuroni) de 3 tipuri: **input**, **output** și **hidden**.
- Există muchii care unesc aceste noduri și care au atașate un număr numit **weight**.
- Valorile nodurilor sunt înmulțite cu valorile muchiilor, iar rezultatele sunt acumulate în nodurile următoare.
- Nodurile **hidden** și cele de **output** pot apela o funcție asupra valorilor acumulate în acestea.

# Rețele neuronale



# Rețele neuronale

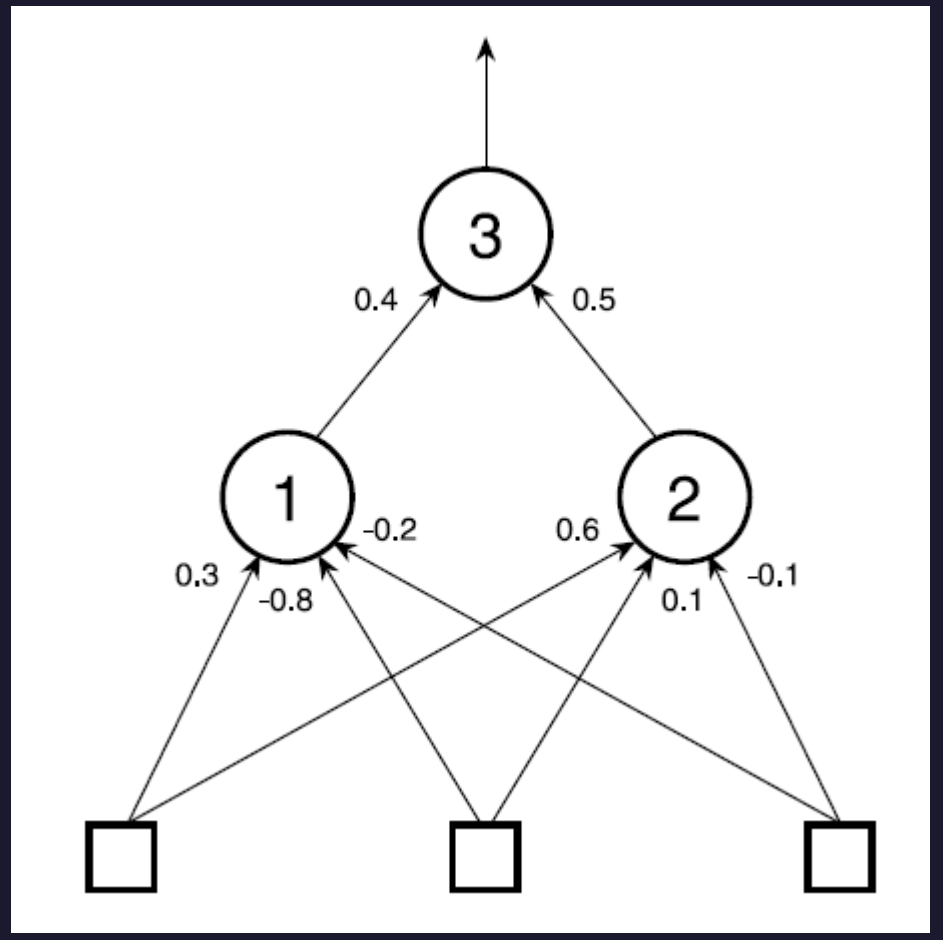
- Pot fi antrenate în mai multe moduri
- **Backpropagation**
- Algoritmi Genetici

# Rețele neuronale antrenate folosind algoritmi genetici – v1

- Populația este reprezentată de rețele neuronale cu aceeași structură
- Codificarea unui individ este reprezentată de **weight**-urile acestuia (un **weight** reprezintă o genă).
- **Crossover**: Pentru fiecare **weight** al unei rețele copil, se aleg **weight**-uri fie de la un părinte, fie de la celălalt.
- **Mutații**: Există șansă ca weight-urile copiilor să fie perturbate puțin.

# Rețele neuronale antrenate folosind algoritmi genetici – v1

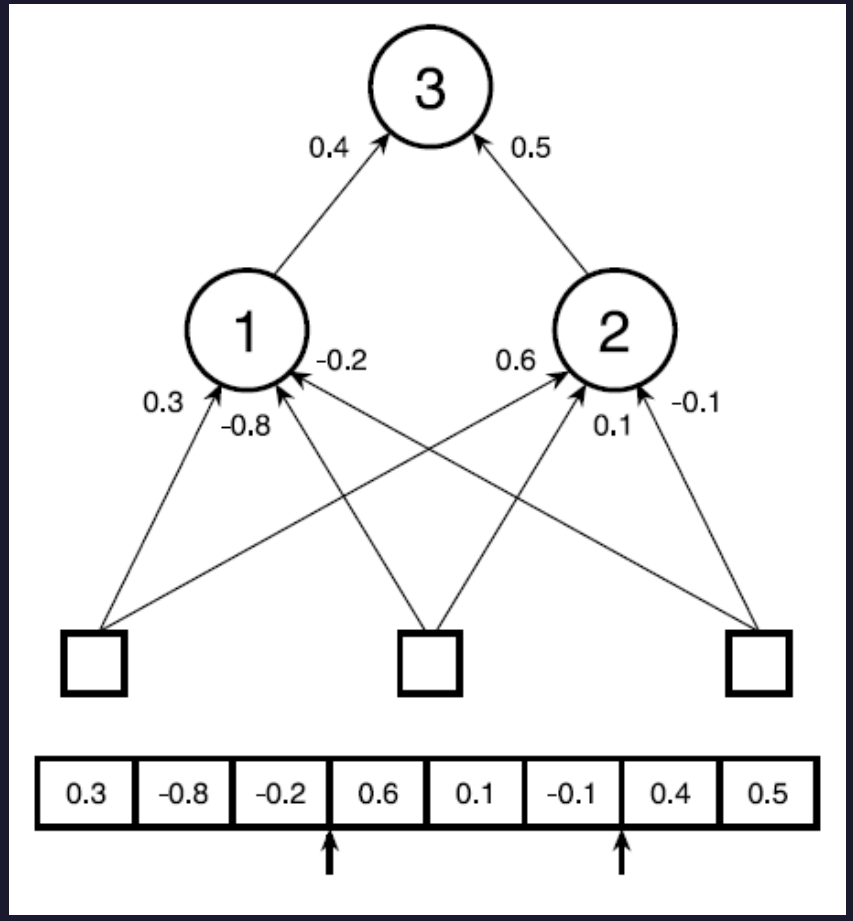
- Exemplu codificare individ:
- 0.3, -0.8, -0.2, 0.6, 0.1, -0.1, 0.4, 0.5



# Rețele neuronale antrenate folosind algoritmi genetici – v2

- Varianta anterioară făcea **crossover**-ul să se comporte ca o **mutație** (neuroni combinați doar parțial)
- Putem considera o genă ca fiind toate **weight**-urile care duc către un neuron.

# Rețele neuronale antrenate folosind algoritmi genetici – v2

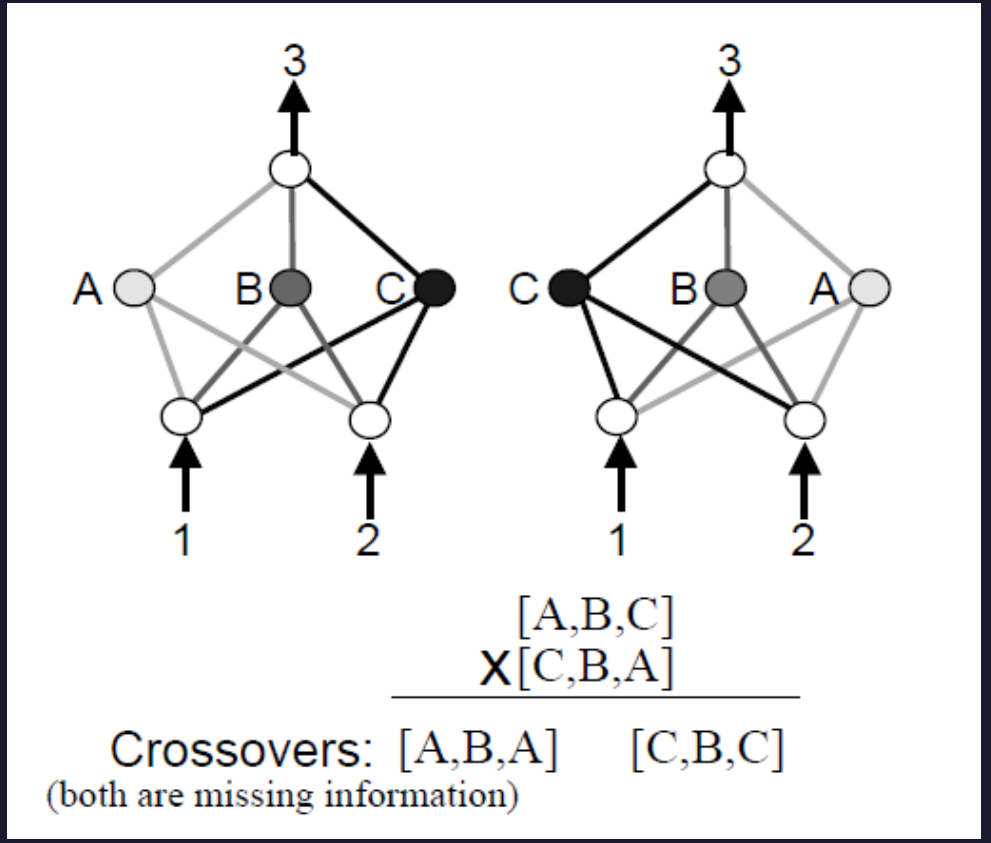


# NeuroEvolution of Augmenting Topologies (NEAT)

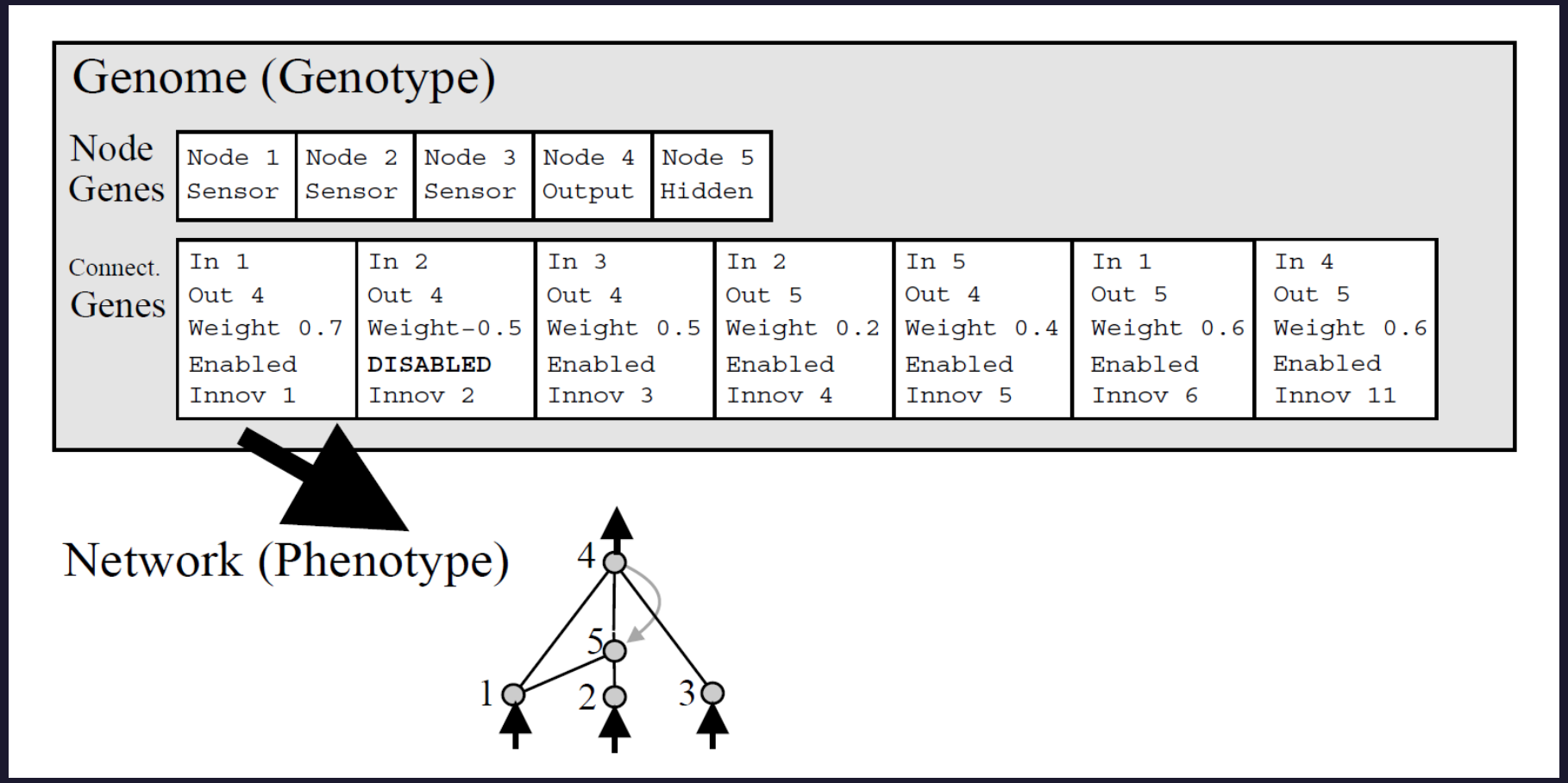
- Structura rețelelor nu mai este fixă, și este și ea la rândul ei evoluată folosind algoritmi genetici.
- Evoluția este mai naturală, genele sunt adăugate incremental.
- Învățarea este mai eficientă deoarece se pornește de la rețele neuronale mici.
- Nu mai este nevoie de o configurare manuală a structurii rețelei (de regulă prin trial & error).

# NeuroEvolution of Augmenting Topologies (NEAT)

- Cum codificăm rețelele?
- Cum facem **Crossover**? Dar **mutații**?
- Cum păstrăm inovațiile structurii?



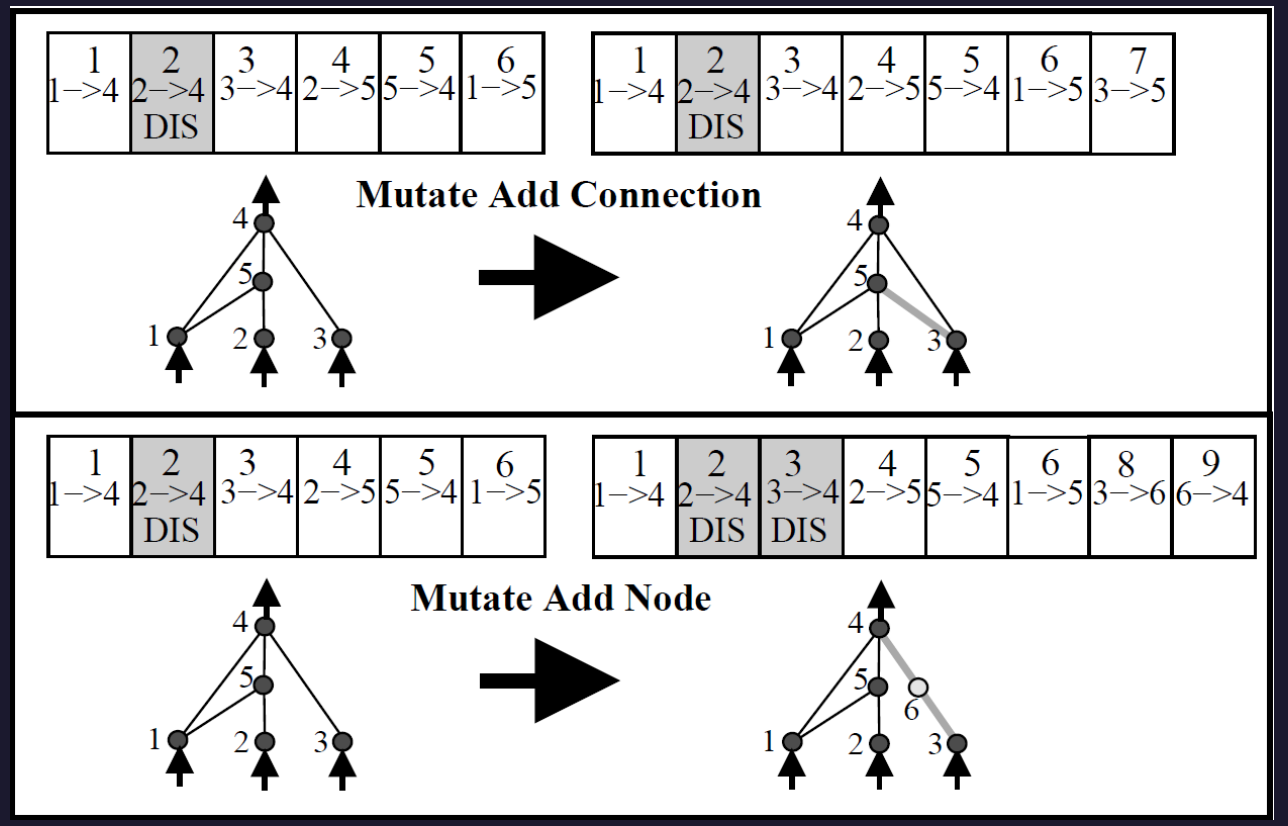
# NeuroEvolution of Augmenting Topologies (NEAT)



# NeuroEvolution of Augmenting Topologies (NEAT)

- Avem 3 tipuri de mutații
- Perturbăm **weight-**uri (ca în cazul rețelelor anterioare)
- Adăugăm muchii între noduri neconectate.
- Adăugăm noduri de-alungul unor muchii (muchia inițială este dezactivată, iar alte două sunt adăugate în loc. Una din ele are **weight** 1, iar cealaltă are **weight-**ul muchiei dezactivate pentru a nu perturba prea mult comportamentul rețelei.)

# NeuroEvolution of Augmenting Topologies (NEAT)



# NeuroEvolution of Augmenting Topologies (NEAT)

- Dimensiunea rețelelor tot crește
- Cum știm ce gene putem combina?

# NeuroEvolution of Augmenting Topologies (NEAT)

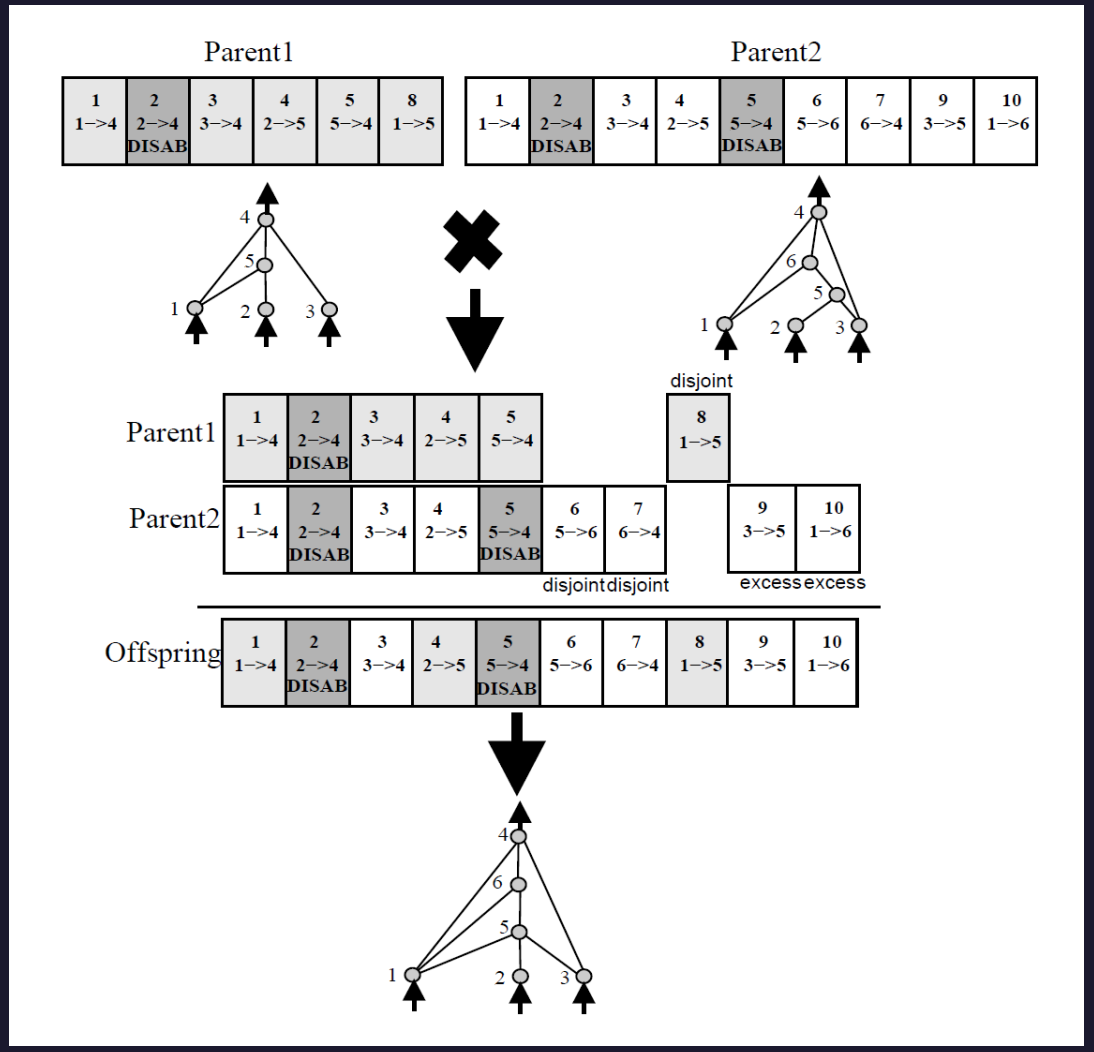
- Având două rețele diferite dorim să aflăm care sunt genele comune ale acestora pentru a putea face crossover.
- Atunci când neuroni sau muchii sunt adăugate la o anumită epocă, acesora li se atribuie un ID numit **innovation number**.
- Se poate implementa folosind o variabilă globală **global innovation number**.
- Dacă în cadrul unei epoci se adaugă același caracteristică, atunci va avea același innovation number.
- Se poate implementa prin folosirea unei liste în care se află inovațiile epocii respective.

# NeuroEvolution of Augmenting Topologies (NEAT)

- Crossover-ul se face între 2 părinți. Aceștia pot avea gene comune sau specifice (**disjoint**, **excess**).
- În cazul genelor comune, copiii primesc aleator gena fie de la un părinte, fie de la celălalt.
- În cazul celor specifice (**disjoint**, **excess**), copiii primesc genele părintelui cu **fitness**-ul mai mare.
- Dacă o genă este activă în cazul unui părinte și inactivă în cazul celuilalt, atunci se alege aleator dacă aceasta va fi activă sau nu în cazul copiilor.

# NeuroEvolution of Augmenting Topologies (NEAT)

- În acest exemplu, părinții au **fitness**-ul egal.



# NeuroEvolution of Augmenting Topologies (NEAT)

- Nu are sens să antrenăm rețelele neuronale mici în același mod ca pe cele mari.
- Rețelele mici evoluează mai repede.
- Rețelele mici sunt mai sensibile la adăugarea de noi muchii și noduri, fapt ce le poate scădea fitness-ul mai dramatic.
- Este nevoie de un mecanism de specii.

# NeuroEvolution of Augmenting Topologies (NEAT)

- Fiecare individ aparține unei specii
- O specie este reprezentată de către un individ aleator g al acesteia.

# NeuroEvolution of Augmenting Topologies (NEAT)

- Pentru a determina dacă un individ se află într-o specie se calculează $\delta$ astfel
- $\delta = \frac{c_1 E}{N} + \frac{c_2 D}{N} + c_3 \cdot \overline{w}$
- Unde E este numărul de gene **excess**, D este numărul de gene **disjoint** iar $\overline{w}$ este suma diferențelor **weight**-urilor dintre individ și individul reprezentant al speciei. $c_1, c_2, c_3$ și N sunt constante.
- $\delta$ se compară cu o valoare $\delta_t$. Dacă $\delta < \delta_t$, atunci individul se află în specie.
- Dacă individul nu se poate încadra în nicio specie atunci când este adăugat în populație, se va crea o specie nouă, în care acest individ este reprezentant.

# NeuroEvolution of Augmenting Topologies (NEAT)

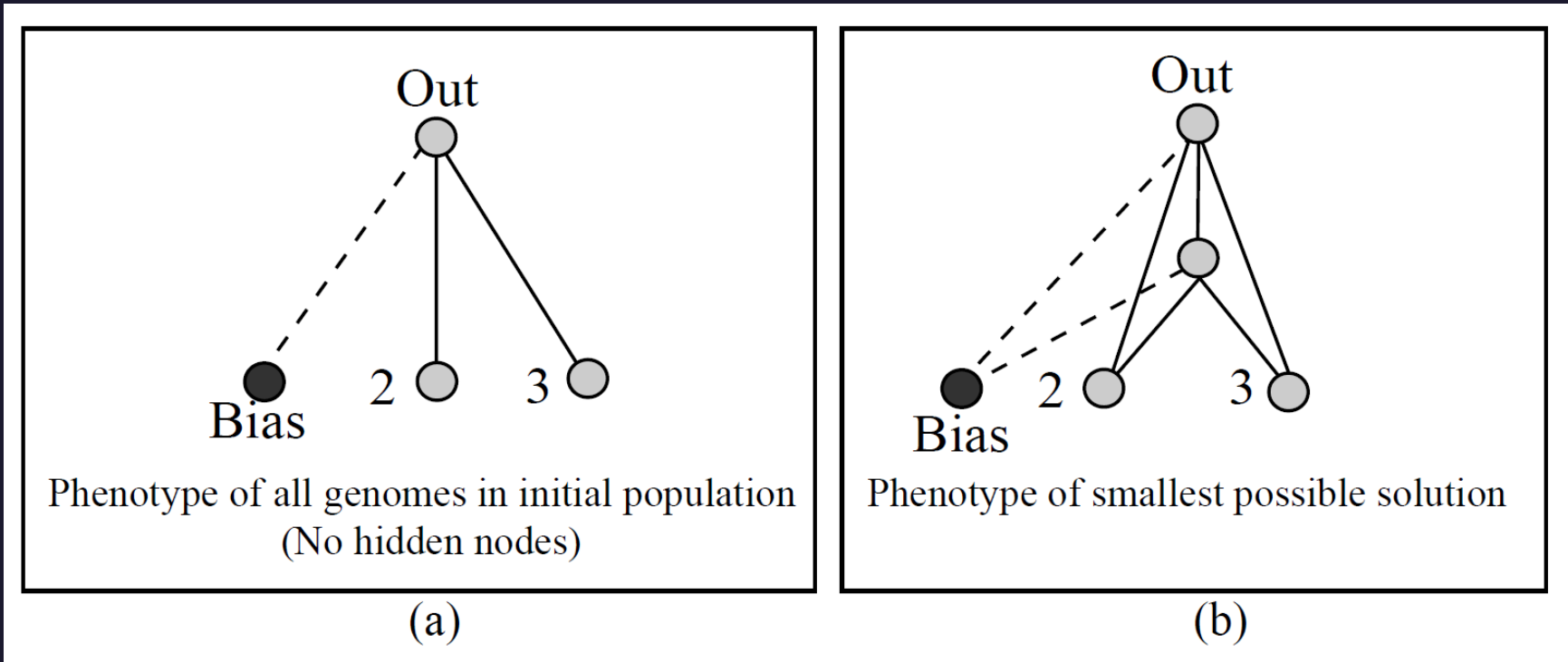
- Pentru **fitness** se folosește **explicit fitness sharing**.
- Speciile nu pot deveni prea mari/ dominante deoarece **fitness**-ul acestora va scădea din cauză mărimii speciei.
- $f_i = \frac{f_i}{\mathrm{E}_{j=1}^{n} sh(\delta(i,j))}$
- Funcția sh returnează 0 atunci când $\delta(i,j)$ este mai mare decât threshold-ul $\delta_t$ și 1 în caz contrar.
- Astfel, cu cât specia individului i este mai mare, cu atât fitness-ul acestuia va deveni mai mic.

# NeuroEvolution of Augmenting Topologies (NEAT)

- Posibili parametrii:
- Populație: 150 indivizi
- Coeficienți: $c_1 = 1.0, c_2 = 1.0, c_3 = 4.0$
- $\delta_t$=3.0
- Mutație: 80% șansă de perturbare a **weight**-urilor, 10% șansă **weight**-uri noi.
- Crossover: 75% șansă ca o genă dezactivată a unui părinte să fie dezactivată și pentru copil.
- 25% din populație rezultă din mutații fără crossover.
- Șansă de crossover între specii: 0.001
- Șansă de a adăuga un nod nou: 0.03
- Șansă de a adăuga o muchie nouă: 0.05
- Funcție de activare a nodurilor:
- $\varphi(x) = \frac{1}{1+e^{-4.9x}}$

# NeuroEvolution of Augmenting Topologies (NEAT)

- Problema XOR



# Referințe


- *Kenneth O. Stanley & Risto Miikkulainen (2002). "Evolving Neural Networks Through Augmenting Topologies" (PDF). Evolutionary Computation. **10** (2): 99–127. CiteSeerX 10.1.1.638.3910. doi 10.1162/106365602320169811. PMID 12180173. S2CID 498161.*
- *Mat Buckland and Mark Collins. 2002. AI Techniques for Game Programming. Premier Press.*