Biologically-inspired algorithms and models

7. Cooperative and competitive coevolution

Maciej Komosinski

Many species (groups of organisms) – at least two – influence each other's evolutionary processes.

Question: How does this affect the fitness function landscape?

A famous competitive example: predators and prey – the Lotka–Volterra model

Fundamental issues in cooperative optimization

The example of the cooperative architecture discussed in this section is based on [PD00].

The implementation of this architecture is available in the DEAP library: https://deap.readthedocs.io/en/master/examples/coev_coop.html.

- The optimization of complex problems will be more effective if we can decompose them into components.
- As we know from previous lectures (cf. epistasis, hierarchical GA), unfortunately, the effect of different parts of a solution on the value of the objective function is usually strongly coupled.
- In a traditional EA, individuals are evaluated completely independently – so there is no room for them to cooperate with each other.
- If we managed to decompose the problem (even manually, for example by splitting the individual – the vector of variables – into individual variables), we would have to solve the following problems: 1) the dependence of the fitness landscape from the point of view of each component on the values of other cooperating components, 2) attributing credit to components (*credit assignment*), 3) maintaining diversity within the population of components.
- Suppose we divide the solution into components that go into separate (genetically independent) populations – so the genetic representation and operators in each population may be completely different. How do we evaluate the quality of each part?
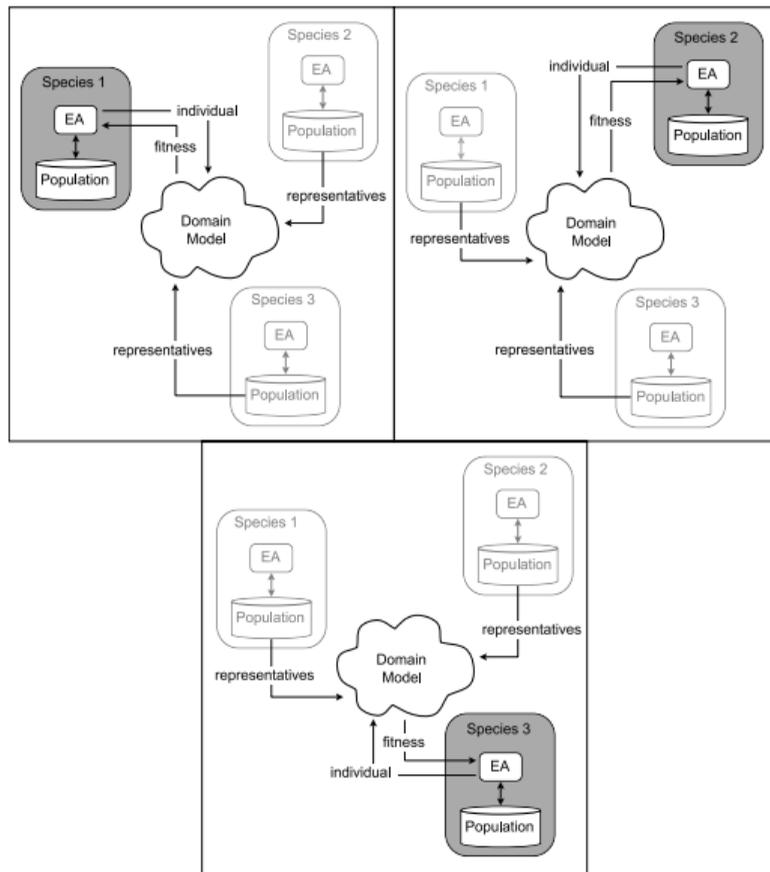
Adaptation of the number of species

- For a function with n variables, we can hand-decompose the task into n species.
- For an agent (e.g. a robot) with a rule-based control system, we can hand-decompose the set of rules into two species, each for a class of behaviors: finding the target and waiting for the target to appear.
- Considering a task in which the number of components can change dynamically, it would be great if the algorithm itself could adjust the number of species and their function ($\rightarrow$ "niches"...) in cooperation with other species. What could trigger the addition of a new species and the removal of an existing one?
  - Adding a new species, for example initialized randomly: when the system is in stagnation – no increase in the quality of the best individual.
  - Removing an existing species: for example when its contribution to cooperation (the difference between the quality of individuals with and without it) is below a specified threshold.
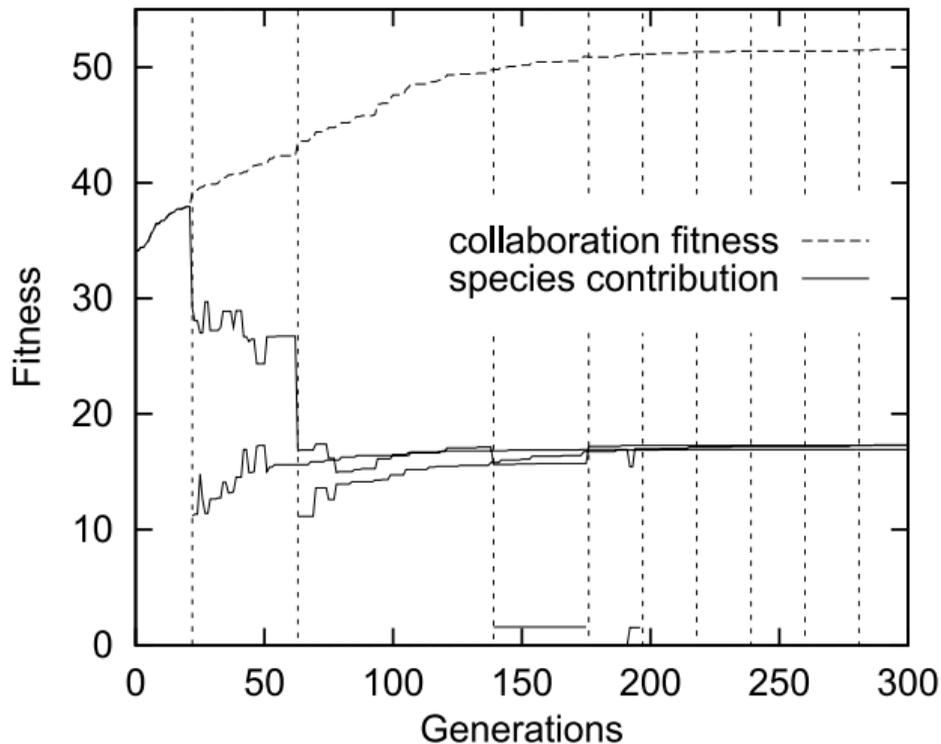
Experiment 1: binary string covering

- Find a (*match*) set M of m binary vectors that match another (*target*) set T of t binary vectors, $m \ll t$.
- Match strength of two vectors is the number of identical bits.
- Match strength of M is the average of (for each vector in T find its best covering match from M).
- Thus M must contain frequent patterns in T – the match set must generalize.
- How did the species that constitute M discover patterns deliberately hidden in T?

T_1:
```
111111111111111111111111111111#############################
############################111111111111111111111111111111
```

T_2:
```
111111111111111##############################################
##############111111111111111##############################
##############################111111111111111##############
##############################################111111111111111
```

T_3:
```
11111111####################################################
########11111111############################################
##############11111111######################################
######################11111111##############################
##############################11111111######################
######################################11111111##############
##############################################11111111######
######################################################11111111
```

Automatically determined number of species; vertical lines are the generations when a species was added as a result of detected stagnation (22, 63, 138) or removed due to its small contribution (oscillations starting from 176).
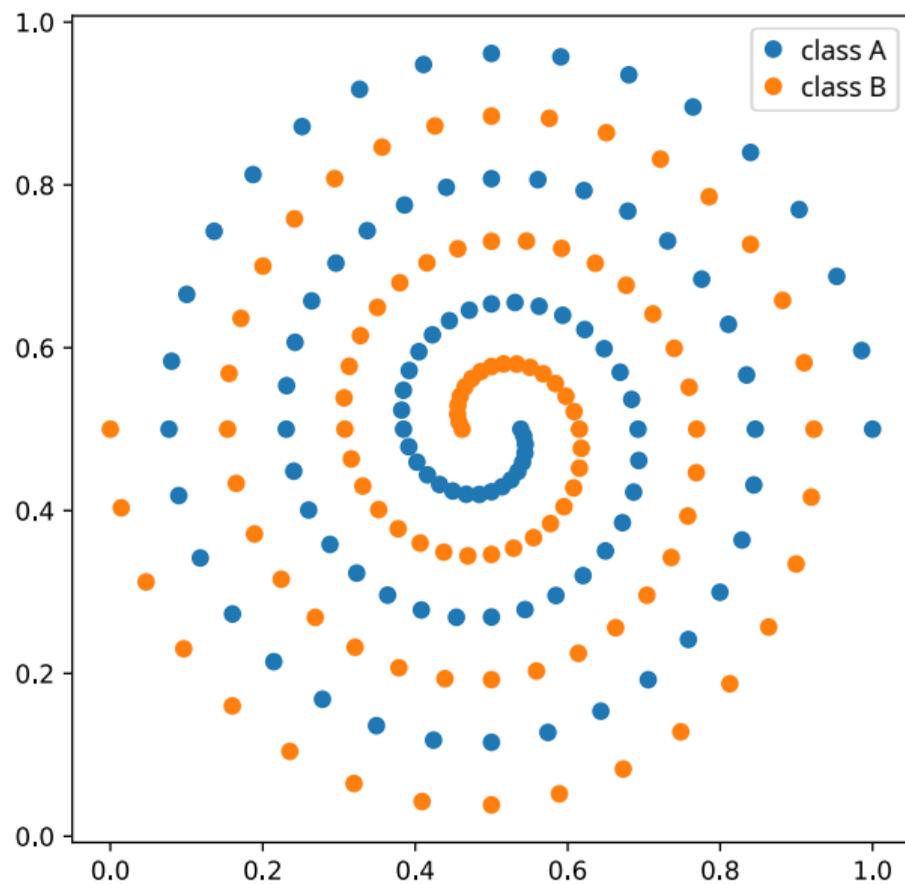
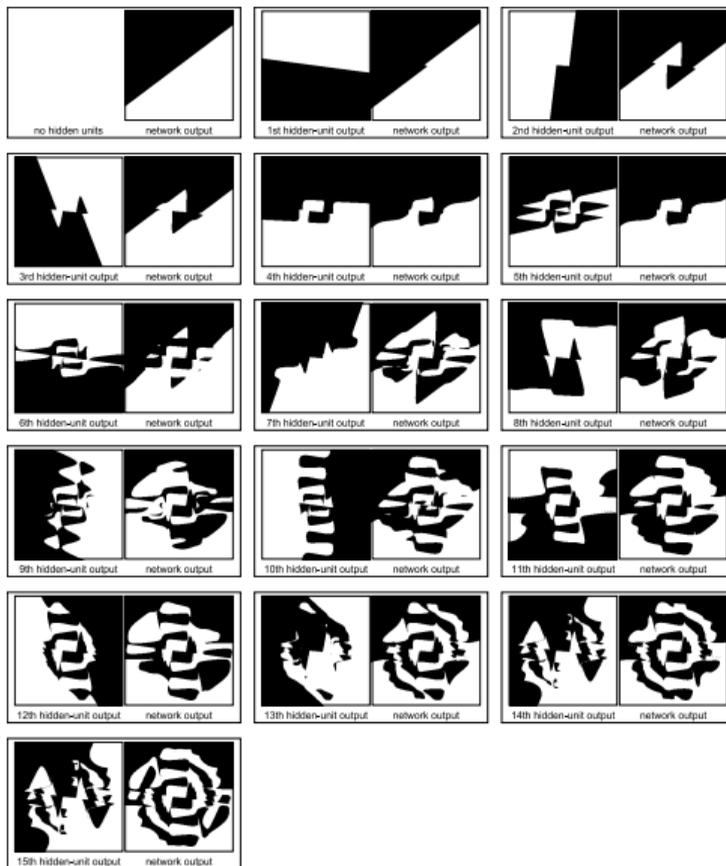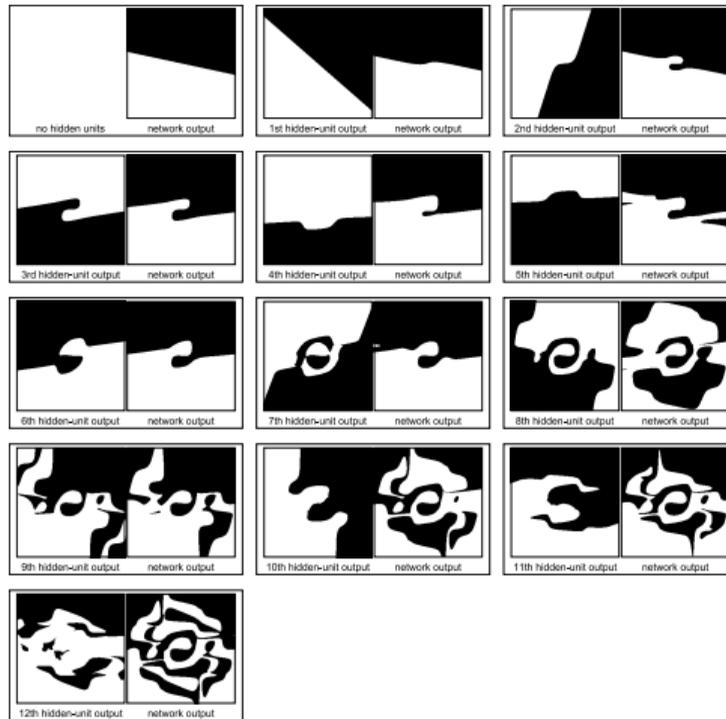Experiment 2: classification of a tricky dataset

Experiment 2: classification of a tricky dataset

Cascade neural network results: cooperative coevolution

Competitive coevolution – the desired behavior

A typical example: coevolution (optimization) of a strategy [Elf+21]. An individual represents the knowledge reflected by a strategy (e.g., it may be the weights of the criteria used to evaluate the situation on the board in a game). The evaluation of an individual is obtained, for example, by playing many games against the other individuals (each playing according to their own strategy).

Discussion: if we start such a process and wait long enough, do we get the master strategy? If not, why not? (provide reasons – the list of possible problems).

Difficulties: we want an *arms race* (perpetual competition), but we may end up in an *MSS – Mediocre Stable State* (stagnation – poor, lasting condition). Too strong an opponent will not allow distinguishing between average and bad solution; too weak – between average and good. The evaluation of each solution depends on the others (an external, objective "teacher" solves this problem while eliminating the advantages of coevolution). The evaluation of a strategy may not be transitive.

Competitive coevolution – problems and remedies

- Discussion of sample scenarios: GP (a population of expressions and a population of tests [ML analogy]), chess playing strategies, soccer, tennis and the intransitivity of "betterness", rock–paper–scissors, a local fencing school and diversity (cf. *exploiter agents* in AlphaStar), nature.
- Concepts: arms race*, Red Queen**, MSS.
- Problems: the lack of or the loss of gradient, looping (cycles, non-transitive relation of comparison that arises from evaluation), the lack of monotonicity/progress [Mic09].
- Remedies: *competitive fitness sharing* (increasing the value of those solutions that win against tests (opponents) difficult for other solutions [RB95]), a specific selection of the test set, maintaining *hall of fame* or sets of Pareto-nondominated solutions and tests.

---

*https://en.wikipedia.org/wiki/Evolutionary_arms_race
**https://en.wikipedia.org/wiki/Red_Queen_hypothesis

[Elf+21]   Ehab Z. Elfeky et al. "A systematic review of coevolution in real-time strategy games". In: *IEEE Access* 9 (2021), pp. 136647–136665. URL: https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=9548932.

[Mic09]   Thomas Miconi. "Why coevolution doesn't "work": superiority and progress in coevolution". In: *12th European Conference on Genetic Programming – EuroGP*. Springer, 2009, pp. 49–60. URL: https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=c51f7914019c020d2386b4880cdc59b22dbbd7c4.

[PD00]   Mitchell A. Potter and Kenneth A. De Jong. "Cooperative coevolution: an architecture for evolving coadapted subcomponents". In: *Evolutionary computation* 8.1 (Mar. 2000), pp. 1–29. DOI: 10.1162/106365600568086. URL: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.35.5861&rep=rep1&type=pdf.

[RB95]   Christopher Rosin and Richard Belew. "Methods for competitive co-evolution: finding opponents worth beating". In: *Proceedings of the Sixth International Conference on Genetic Algorithms*. Morgan Kaufmann, 1995, pp. 373–380. URL: https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.52.9359&rep=rep1&type=pdf.