Bredariol Francesco

EvES

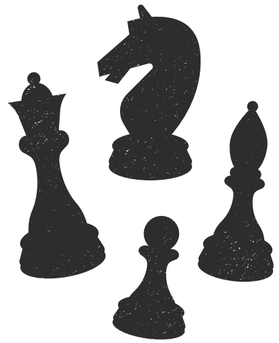
Evolutionary Elo System

An optimization techniques for zero sum games

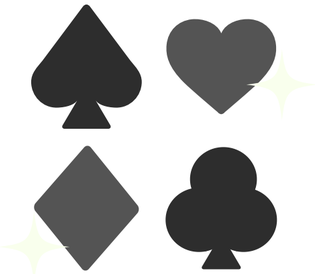
The problem

Zero Sum Games

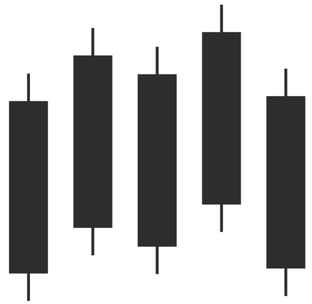
The game theory describes zero-sum game as situations where gains and losses add up to zero.
There could be two or more players involved.



chess



poker



futures trading

The problem

Zero Sum Games

While optimizing **zero sum games** there are some non-trivial tasks to face (other than the **optimization task itself**):

- compute how an individual **perform**
- select **fair matches** for the individual
- obtain a **variety of solutions** among the entire population
- map solution into an “**human-like**” level

The idea

Elo Rating System

The Elo rating system is a method for **calculating the relative skill** levels of players in zero-sum games.

It is a **statistical estimation** of the probability of winning based on the **historical perfomances** of an individual among the others

It can be used to **match individuals** with similar skills in order to obtain the **most fair possible matches**.

Arpad Elo
inventor of the Elo rating system

Elo Rating System

Winning probability

Given an initial population of n players, each of whom has an **Elo level** γ,
the probability for the i-th individual of winning against the j-th individual is computed using
a **sigmoid function***:

$$p_i(win|x_j) = \frac{1}{1 + e^{\frac{\gamma_i - \gamma_j}{\lambda}}}$$

*please note that there are many possible ways to compute this probability. this is just the one I choose

Elo Ranking System

Update rule

In a zero-sum games an individual can either win or lose (we assume no tie).
Encoding the result r of a match as 1 for win and 0 for lose, the elo update rule* is the following (from the perspective of the i-th individual against the j-th one):

$$\gamma_i = \gamma_i + k(r - p_i(r = 1|x_j))$$

*please note that there are many possible ways to update the elo. this is just the one I choose

On the way to the EvES

Online/Offline optimization

The Elo described until now is good for two main reasons:

- measurement of the fitness of an individual
- improvement of the matchmaking function

Fitness approaches

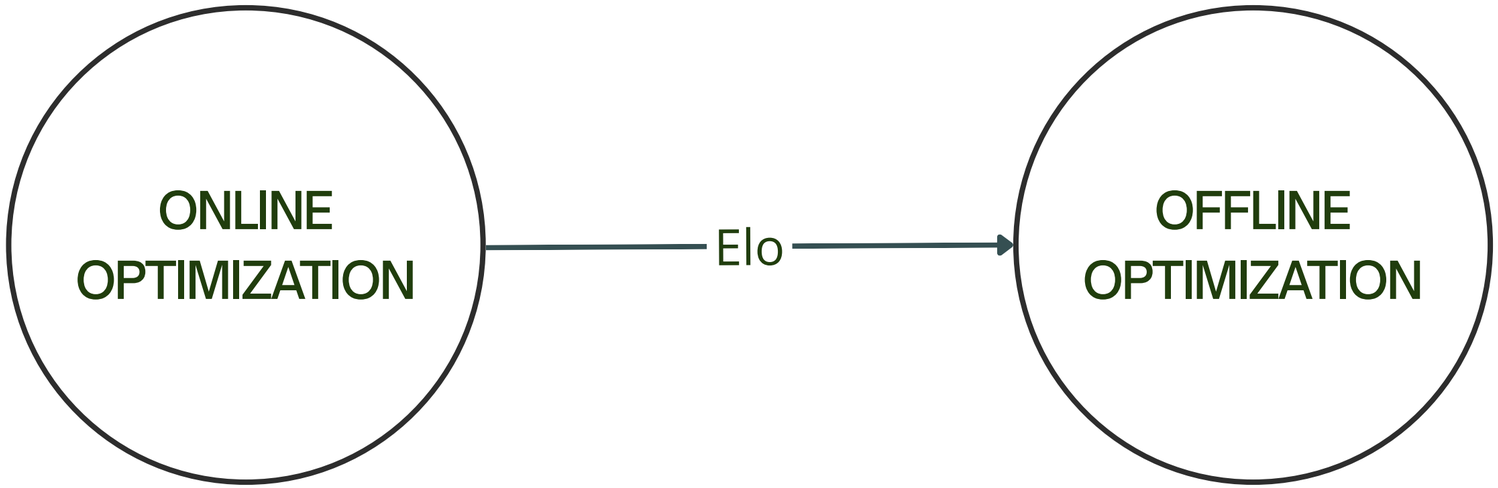
evolutionary strategies
genetic algorithms
...

Reward signal approaches

reinforcement learning
learning classifier system
...

On the way to the EvES

Elo as link



during the online optimization
the elo is computed and used internally

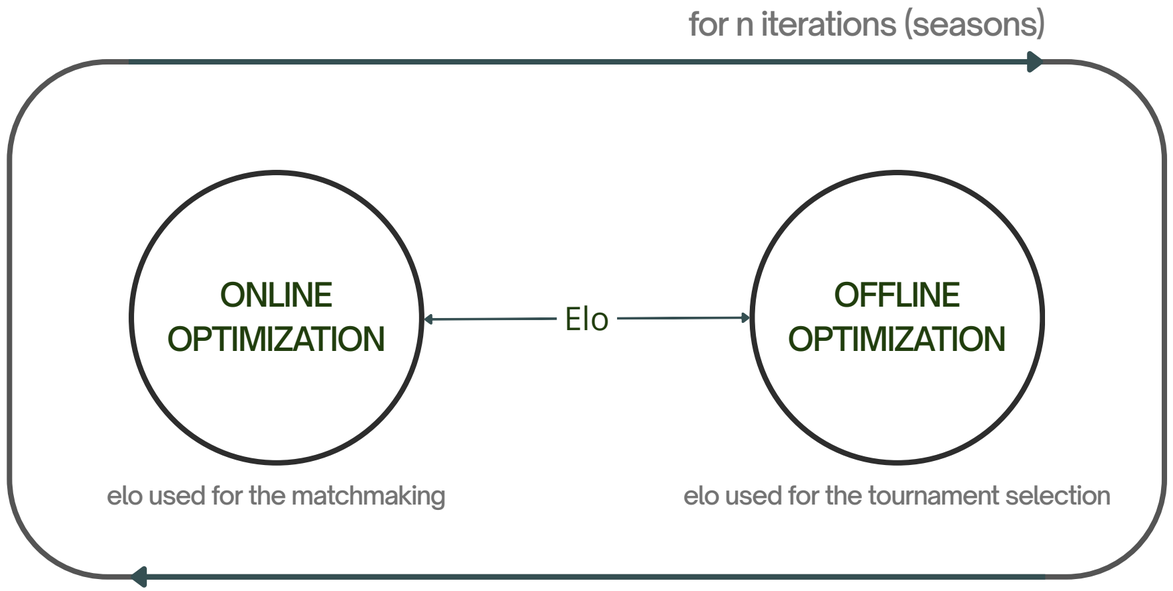
during the offline optimization the elo
is used to improve the solutions

Bringing all together

EvES framework

It makes sense to develop an **hybrid method**
to leverage the strenghts of
both optimization paradigms

In this framework individuals
must follow both protocols at the same time



Key components

Environment

The environment must send signals to individuals in order to make the online optimization possible

A good approach is to follow the Gym/ALE protocol

Individuals are indirectly part of the environment and this is indeed fundamental

```
env = Env()
obs, info = env.reset()
done, truncate = False, False
while not done or truncated:
    actions = individuals.move(obs)
    new_obs, rewards, done, truncated, info = env.step(actions)
    players.observe(obs, actions, new_obs, rewards, done)
    obs = new_obs
env.close()
```

Key components

Individuals

Individuals must implement a **variety of methods** in order to accomplish both online and offline optimization

In order to produce **good individuals** one must **ensure the capability of producing such indivuals** from the online optimization

Reactivness is a good feature

```python
class Individual:

    def observe(self, obs, action, rew, new_obs, done, **kwargs):
        pass

    def move(self, **kwargs):
        pass

    def reset(self, **kwargs):
        pass

    def update(self, **kwrags):
        pass

    def mutate(self, **kwargs):
        pass
```

Minimal requirements for an individual

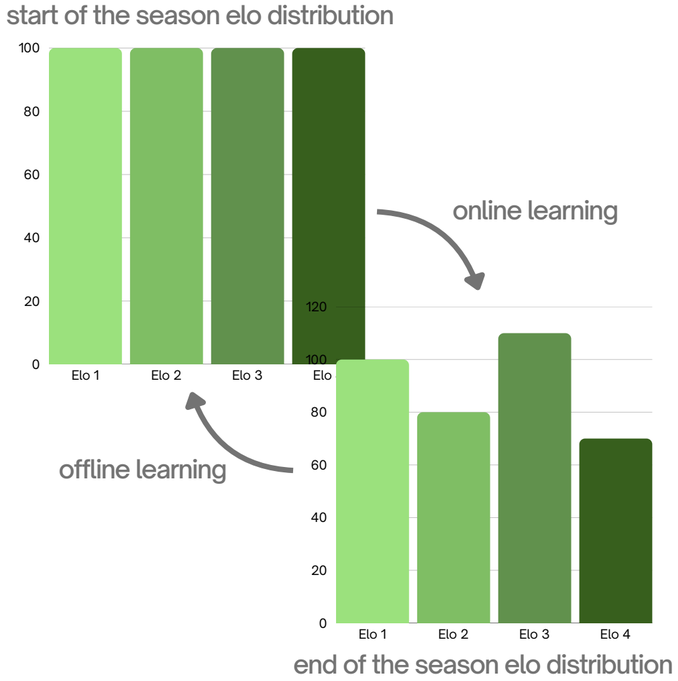
Key components

Elo Ranking System

An **Elo Ranking System** must be implemented inside the framework

It can just follow everything descripted up until now

The elo of individuals should be **reset at the beginning of a new season** to a fixed level since there are **no way to map the effect of a mutation to the Elo**

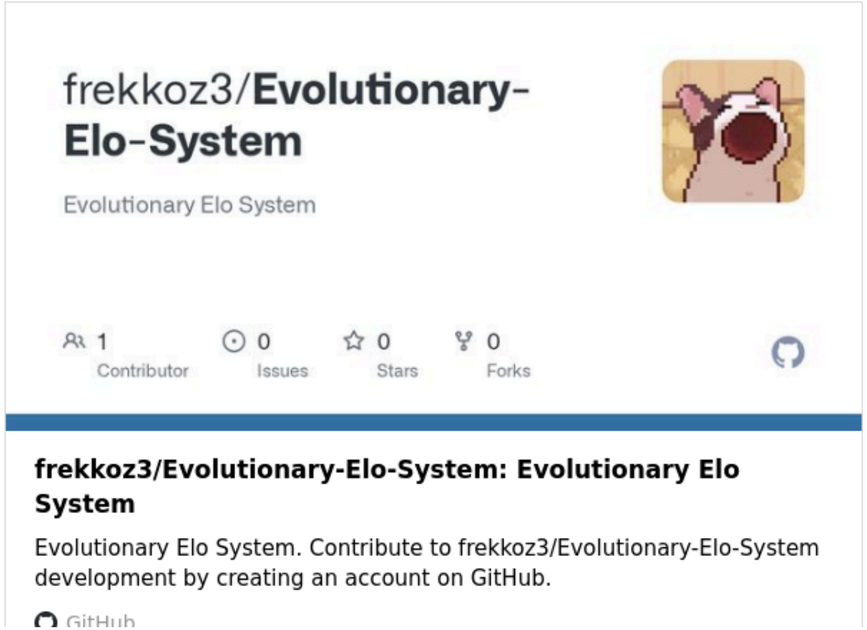


About the

Python implementation

The framework described until now is been implemented in **Python**

Several components are mappable **1:1 from the idea to the source code**, while others have a **variety of details better explained in their own README files**



Details

DQN

The main online optimization techniques that I decided to use is the **DQN**, a technique that comes from the **Reinforcement Learning** area

It is a powerful technique but there are **many caveaut** one should be aware of

A better choice could be the R2D2, capable of learning **more complex dependencies**

Details

Evolutionary Strategy

I decided to use a simple **ES with elitism**

There are **no crossovers** involved in the evolutionary cycle

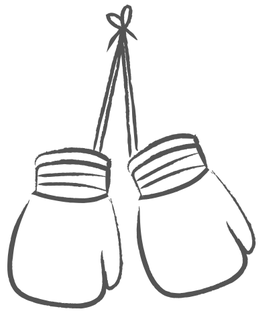
The **mutations** are uniformly random

The **tournament selection** is the default selection type

Details

Benchmarks

Boxing



Rectangular ring
8 types of action
Stamina resource
Complex internal dynamic
The first to hit 15 punches wins

Grab ‘n go

Rectangular ring
5 types of action
Maximum time 30 seconds

Problem in the Benchmarks

Boxing

The framework is too computational heavy

DQN does not capture time relationships

Individuals are way too few reagents

Elo to fine-tune

Grab ‘n go

There are problem with the DQN

A really naive approach with a Lamarckian GP didn’t produce any results (LCS would have been better)

The Elo is not really necessary

Conclusion

Take home lecture

This could have been a really interesting optimization technique but there are a lot of caveaut that make it really hard to work properly

Combining this technique with proper distributed methods could solve some of the main problems

Focusing on the online optimization is crucial in order to obtain good results

The Elo can be a bad measurement under certain conditions and other metrics should be used instead (Glicko, TrueSkill2, BayesElo)

Thank for your attention

Bredariol Francesco