INSTITUT PASTEUR

Paradiseo (in 5 minutes)

Randomized Optimization Framework tailored for Benchmarking & Auto-Design

Johann Dreo

2022-05-30

What is Paradiseo?

https://nojhan.github.io/paradiseo/

- **Open-source**
- **Evolutionary / Metaheuristic**
- **Framework**
- **C++**
- **20 years stability**

Name	Language	Update	License	Contributors	kloc
ParadisEO	C++	2021	LGPLv2	33	82
jMetal	Java	2021	MIT	29	60
ECF	C++	2017	MIT	19	15
ECJ	Java	2021	AFLv3	33	54
DEAP	Python	2020	LGPLv3	45	9
CIlib	Scala	2021	Apachev2	17	4
HeuristicLab	C#	2021	GPLv3	20	150
Clojush	Clojure	2020	EPLv1	17	19

(as of 2021)

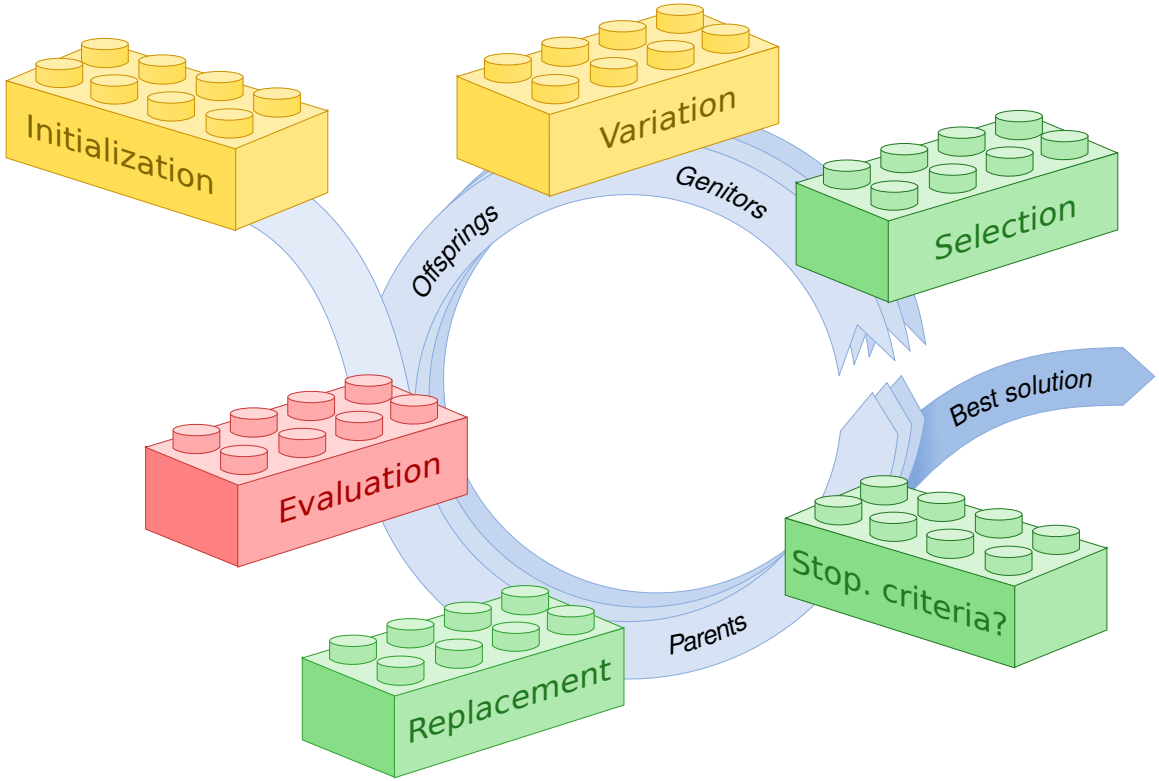
Why Paradiseo?

- a ***modular design*** for several types of paradigms,
- the ***largest codebase*** of existing components,
- tools for **benchmarking** and ***automated design/selection*** of algorithms,
- a focus on ***speed*** and several ***parallelization*** options.

Name	Language	Update	License	Contributors	kloc	Evol.	EDAs	PSO	Local Search	Cluster	Multicore	GPGPU	Multiobjective	Landscapes	States	Auto. Design
ParadisEO	C++	2021	LGPLv2	33	82	Y	Y	Y	Y	Y	Y	~	Y	Y	Y	Y
jMetal	Java	2021	MIT	29	60	Y	N	Y	N	Y	N	N	Y	N	?	N
ECF	C++	2017	MIT	19	15	Y	N	Y	N	Y	N	N	N	N	Y	N
ECJ	Java	2021	AFLv3	33	54	Y	Y	Y	N	Y	Y	Y	Y	N	Y	N
DEAP	Python	2020	LGPLv3	45	9	Y	N	N	N	Y	Y	N	Y	N	Y	N
Cllib	Scala	2021	Apachev2	17	4	Y	N	N	N	N	N	N	N	N	?	N
HeuristicLab	C#	2021	GPLv3	20	150	Y	N	Y	Y	Y	Y	N	Y	~	Y	N
Clojush	Clojure	2020	EPLv1	17	19	Y	N	N	N	N	N	N	N	N	N	N

How does `Paradiseo` feels like?

- ***Plug in*** Lego bricks
- Enjoy reality-proof helper features
  - Automatic CLI
  - Parallelization
  - Suspend on state
  - etc.




```
29
30      edoNormalAdaptive<R> gaussian(dim);
31
32      auto& obj_func = state.pack< eoEvalFuncPtr<R> >(sphere);
```

```
47      auto& selector  = state.pack< eoRankMuSelect<R> >(dim/2);
48      auto& estimator = state.pack< edoEstimatorNormalAdaptive<R> >(gaussian);
49      auto& bounder   = state.pack< edoBounderRng<R> >(R(dim, -5), R(dim, 5), gen);
50      auto& sampler   = state.pack< edoSamplerNormalAdaptive<R> >(bounder);
51      auto& replacor  = state.pack< eoCommaReplacement<R> >();
```

```
56      auto& algo = state.pack< edoAlgoAdaptive<CMA> >(
57           gaussian , pop_eval, selector,
58           estimator, sampler , replacor,
59           pop_continue, distrib_continue);
```

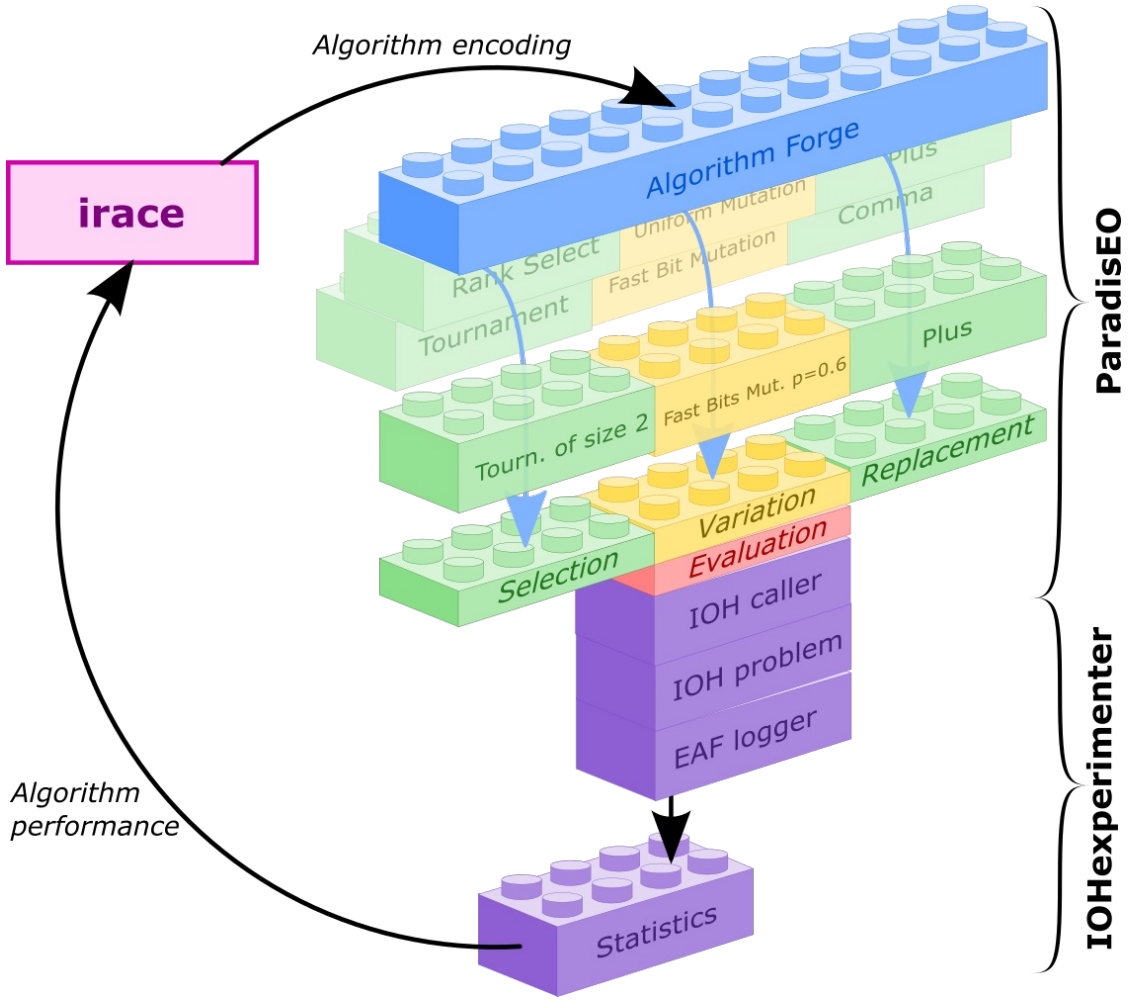
How is Paradiseo useful?

Benchmarking:

- ***Faster*** experiments
- Larger ***scale*** experiments
- Binding with **IOHexperimenter**
  - Problems
  - Loggers/Performance

Auto-Design:

- **Wider** search space
  - \+ Parameters
  - \+ Operators
  - \+ Hybridization
- Automated interface export (ex.: *irace*)
- Better benchmarking (?)



What's next in Paradiseo?

- Interfaces between the algorithms instances' exe and other tools:
  - with HPO:
    - Algorithm Configuration/Selection solvers
    - [...]
    - Design of Experiments / Pipelines managers
  - Types of interfaces:
    - Configuration files (e.g. irace)
    - Network C/S messages: https://github.com/IOHprofiler/IOHexperimenter/tree/master/server/ (common to all expensive optimization problems?)

https://nojhan.github.io/paradiseo

INSTITUT PASTEUR