INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

INRIA

CNRS CENTRE NATIONAL DE LA RECHERCHE SCIENTIFIQUE

LIFL

Reusable Design of Metaheuristics for Multi-objective Optimization

May 2007

E.-G. Talbi and the ParadisEO team

Paradiseo

http://paradiseo.gforge.inria.fr
paradiseo-help@lists.gforge.inria.fr

Framework and Tutorial Application

- A framework for the design of metaheuristics for multi-objective optimization (mainly evolutionary algorithms)

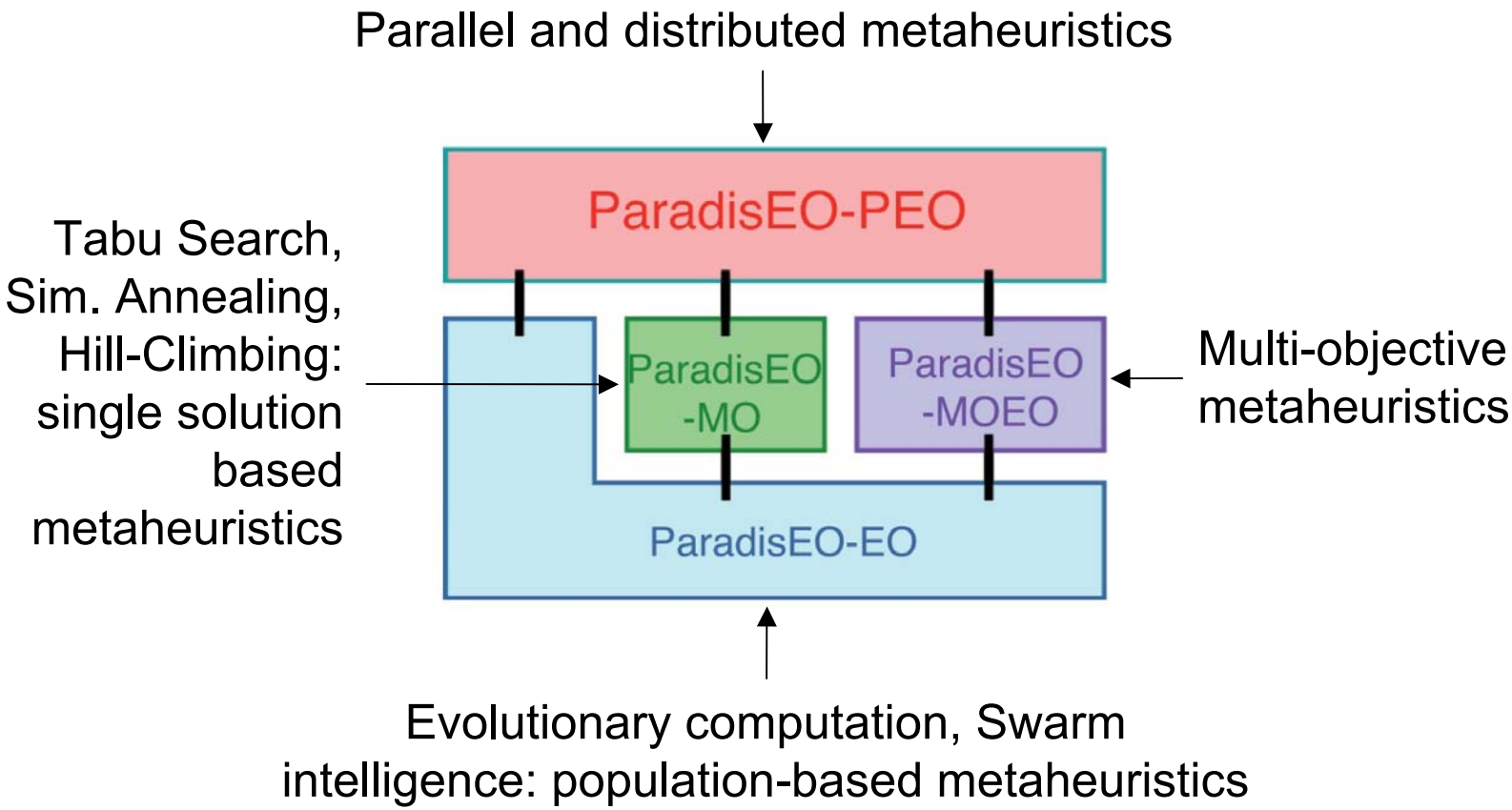
➔ ParadisEO-MOEO (Multi-Objective Evolving Objects)

A. Liefooghe, M. Basseur, L. Jourdan, E.-G. Talbi: "**ParadisEO-MOEO: A Framework for Evolutionary Multi-Objective Optimization**", EMO 2007, LNCS vol. 4408, pp. 386-400, Matsushima, Japan, 2007

- Tutorial application

➔ A bi-objective flow-shop scheduling problem

ParadisEO: Module-based Architecture



Multi-Objective Evolving Objects (MOEO)

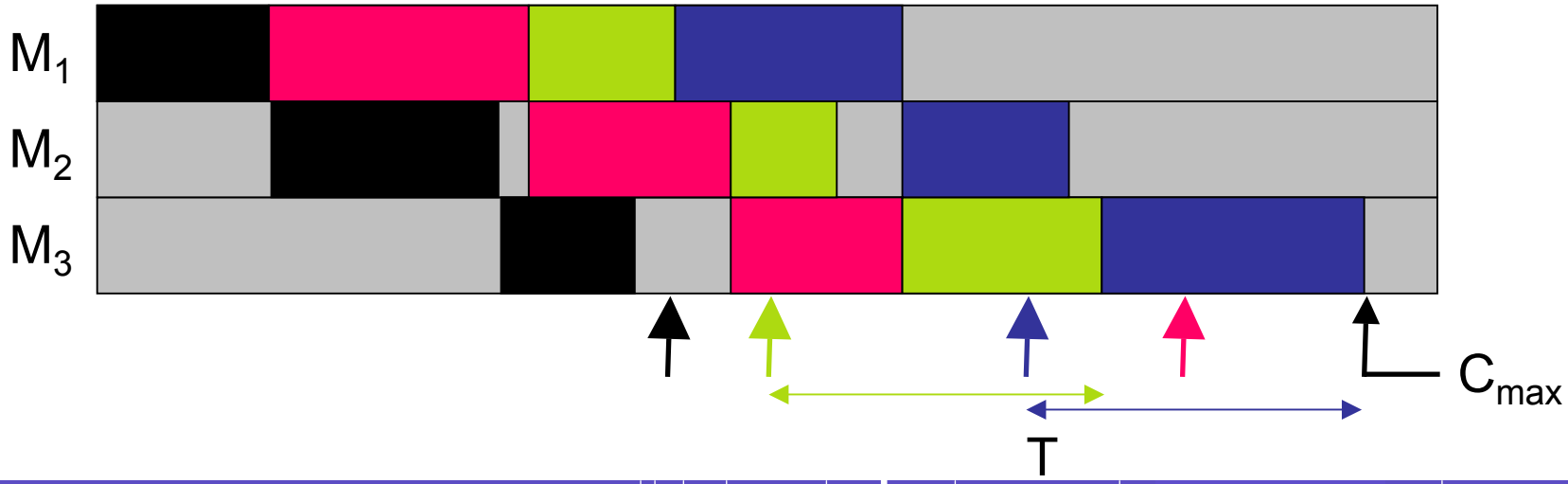
- A framework dedicated to the flexible design of metaheuristics for multi-objective optimization
  - Extends the EO library
  - ANSI-C++
  - Open-source
  - White-box
  - Object-oriented
  - Template-based
  - Paradigm-free
- Design/development team:
  - A. Liefooghe, T. Legrand, S. Cahon, L. Jourdan and E-G Talbi
    OPAC team (LIFL) / Dolphin project (INRIA)

Design Issues

- Maximum design and code reuse
  - Distinction between resolution methods and tackled problems
- Flexibility and Adaptability
  - Adding or updating other optimization methods, search mechanisms, operators, representation...
- Utility
  - Large panel of methods, strategies…
- Portability
  - Deployment on different platforms (thanks to ParadisEO-PEO)
- Transparent access to performance and robustness
  - Parallel and hybrid implementation is transparent to the target hardware platform

Flow-shop Scheduling Problem

- N jobs to schedule on M machines
- Machines are critical resources
- 2 objectives to optimize (minimize)
  - Makespan (C_{max})
  - Total tardiness (T)



Multi-Objective EAs Design Issues

The goal of approximating the Pareto set is itself bi-objective

- Minimizing the distance of the solutions to the real Pareto set
  - ➔ well-converged Pareto set approximation
- Maximizing the diversity of the solutions
  - ➔ well-diversified Pareto set approximation

Three specific aspects have to be taken into account

- Fitness assignment
- Diversity preservation
- Elitism

A Multi-Objective Evolutionary Algorithm

1. Representation
2. Initialization
3. Evaluation
4. Variation operators
5. Fitness assignment
6. Diversity assignment
7. Selection
8. Replacement
9. Stopping criteria

A Multi-Objective Evolutionary Algorithm

1. Representation
2. Initialization
3. Evaluation
4. Variation operators
5. Fitness assignment
6. Diversity assignment
7. Selection
8. Replacement
9. Stopping criteria

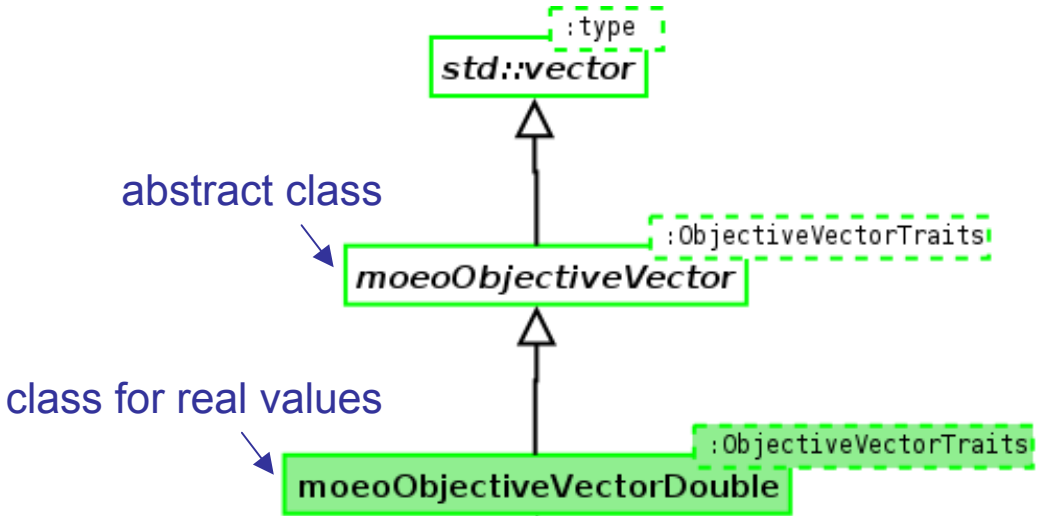
Representation in the Objective Space

- In mono-objective optimization (EO)
  - fitness value = objective value
- In multi-objective optimization (MOEO)
  - fitness value ≠ objective values

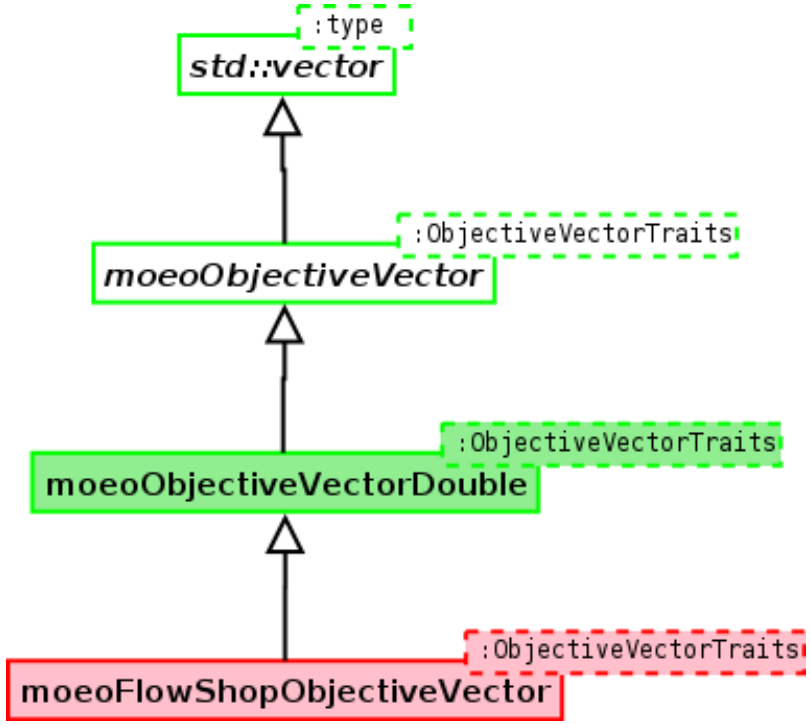
- Individuals are represented in the objective space as a tuple of n values

$$F(x) = \begin{bmatrix} f_1(x) \\ f_2(x) \\ \mathrm{M} \\ f_n(x) \end{bmatrix} \qquad F : X \rightarrow Z$$

Objective Vector: Core Classes

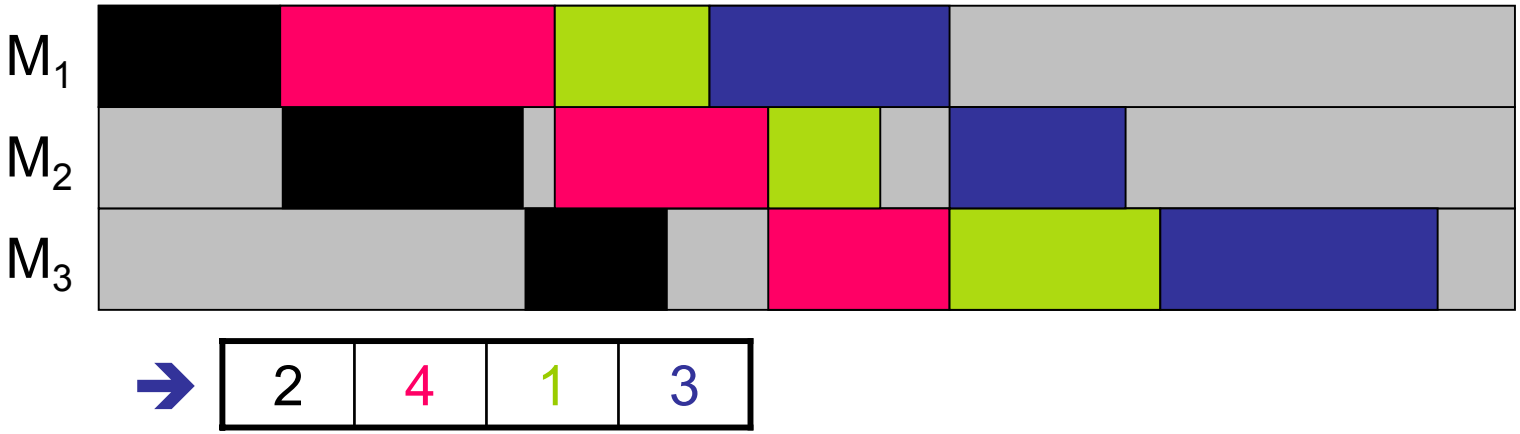


Representation: Application to the Flow-shop

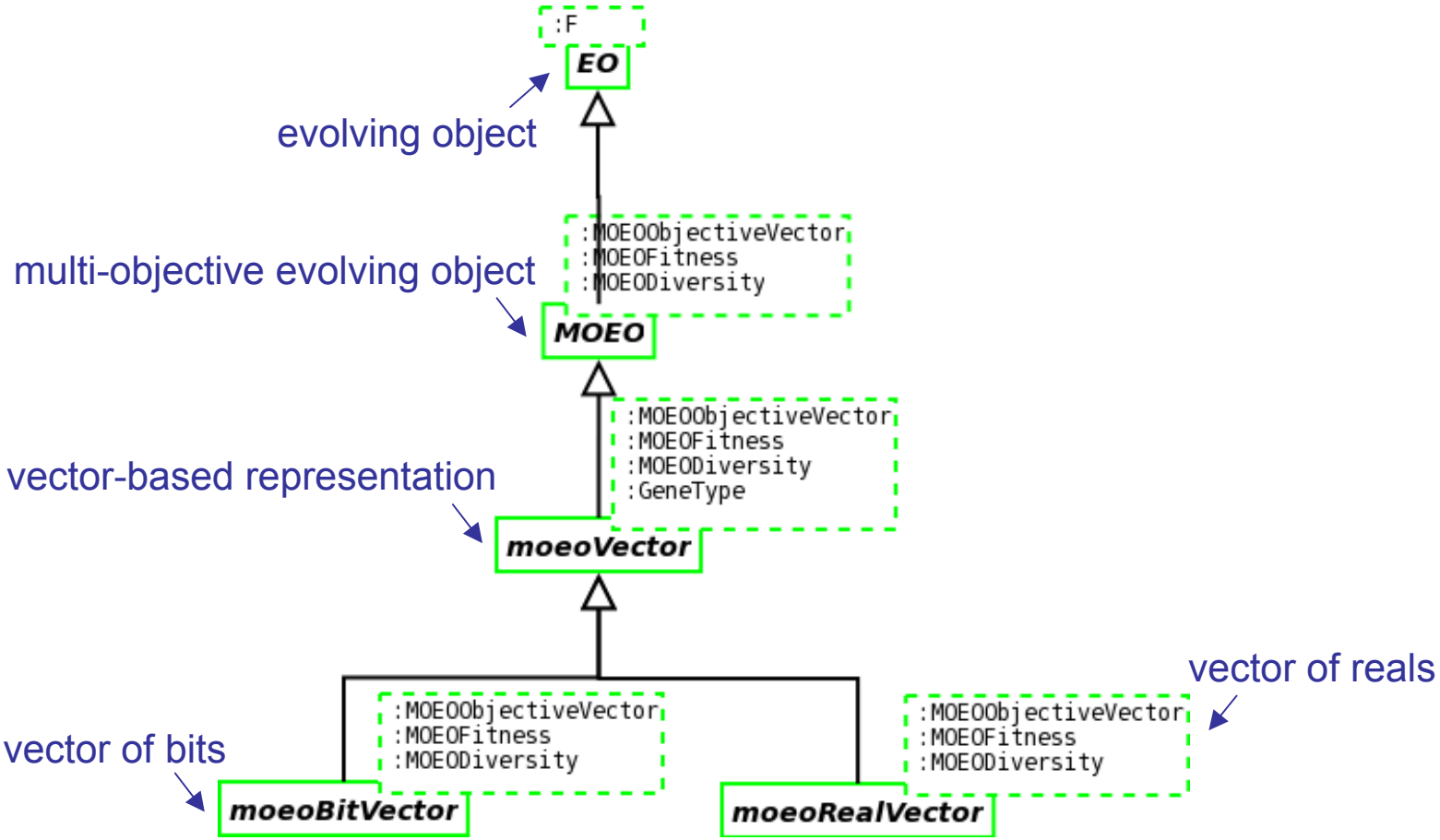


Representation in the Decision Space

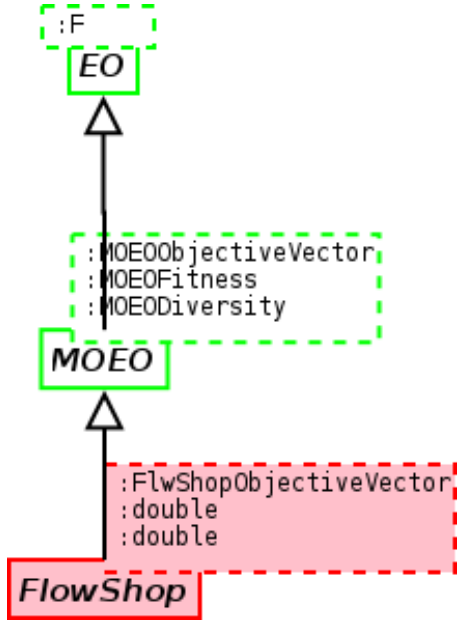
- As in the mono-objective case, it depends of the tackled problem
  - Discrete representation
  - Real-valued representation
  - …
- Flow-shop: order-based representation
  - Scheduling of the jobs on one machine



Representation: Core Classes



Representation: Application to the Flow-shop



A Multi-Objective Evolutionary Algorithm

1. Representation
2. Initialization ➔ same as within ParadisEO-EO
3. Evaluation
4. Variation operators
5. Fitness assignment
6. Diversity assignment
7. Selection
8. Replacement
9. Stopping criteria

A Multi-Objective Evolutionary Algorithm

1. Representation
2. Initialization
3. Evaluation
4. Variation operators
5. Fitness assignment
6. Diversity assignment
7. Selection
8. Replacement
9. Stopping criteria

Evaluation

- Most computationally expensive step for real-world problems
- Evaluation of a solution in the objective space for every objective

- Application to the flow-shop
  - Makespan
    - $C_{max} = \max [s_{iM} + p_{iM}]$
  - Total tardiness
    - $T = \sum \max(0 , s_{iM} + p_{iM} - d_i)$

A Multi-Objective Evolutionary Algorithm

1. Representation
2. Initialization
3. Evaluation
4. Variation operators ➔ same as within ParadisEO-EO
5. Fitness assignment
6. Diversity assignment
7. Selection
8. Replacement
9. Stopping criteria

A Multi-Objective Evolutionary Algorithm

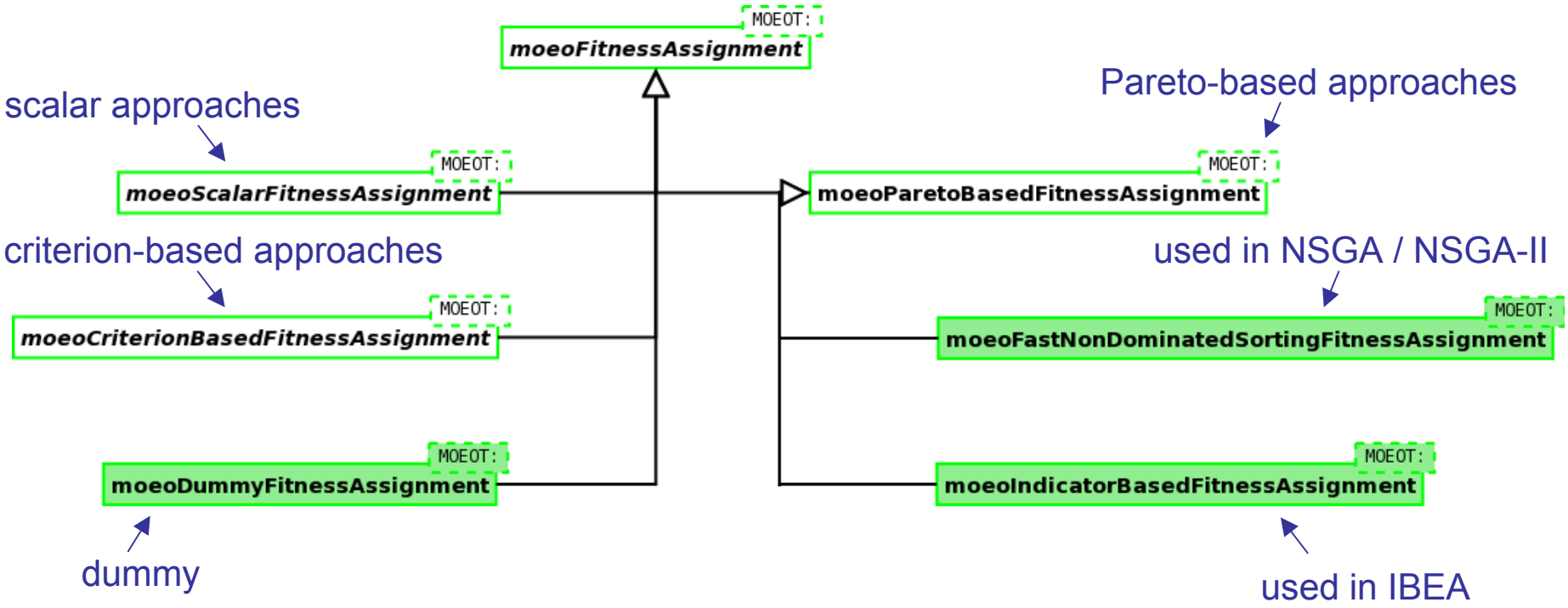
1. Representation
2. Initialization
3. Evaluation
4. Variation operators
5. Fitness assignment
6. Diversity assignment
7. Selection
8. Replacement
9. Stopping criteria

Fitness Assignment

- Fitness (x) reflects the quality of x in term of convergence

- Three kinds of fitness assignment strategies
  - Scalar approaches
  - Criterion-based approaches
  - Pareto-based approaches

Fitness Assignment: Core Classes



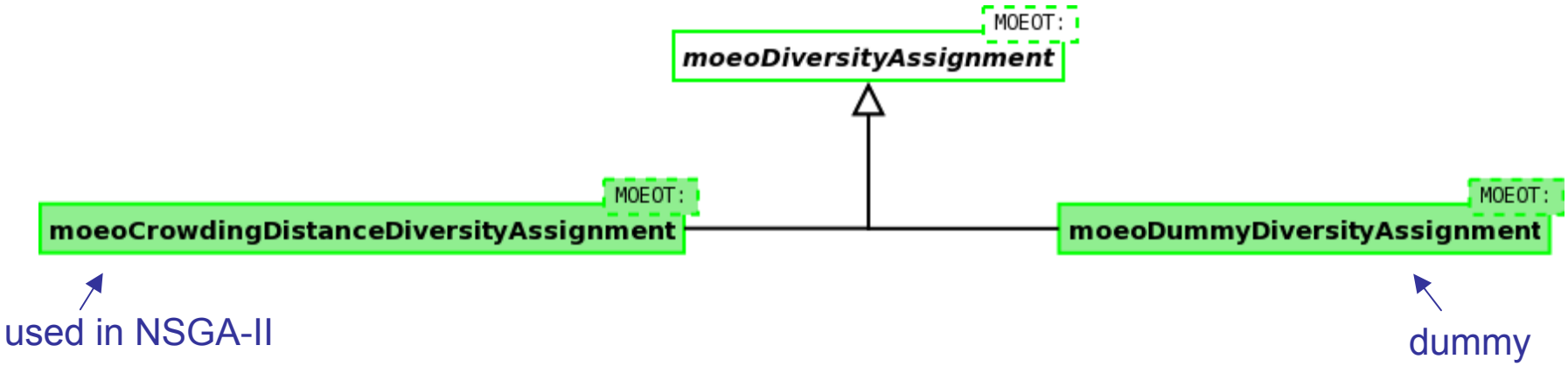
A Multi-Objective Evolutionary Algorithm

1. Representation
2. Initialization
3. Evaluation
4. Variation operators
5. Fitness assignment
6. Diversity assignment
7. Selection
8. Replacement
9. Stopping criteria

Diversity preservation

- Incorporating density information into the selection processes
- The probability to select an individual decreases the greater is the density of individuals in its neighbourhood
- Various kinds of methods for diversity preservation
  - Kernel methods, nearest neighbour techniques, histograms (hypergrid), …
- Some methods need a distance measure that can be defined on
  - The objective space
  - The decision space

Diversity Assignment: Core Classes



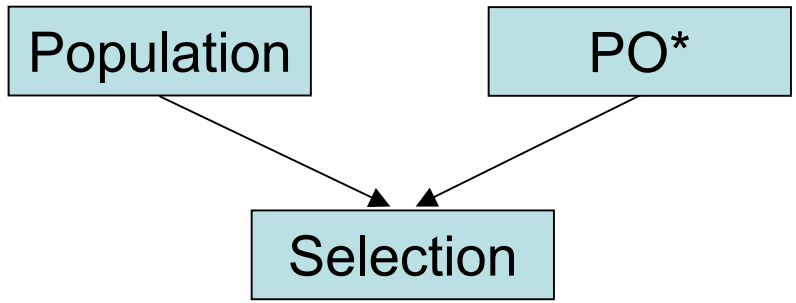
A Multi-Objective Evolutionary Algorithm

1. Representation
2. Initialization
3. Evaluation
4. Variation operators
5. Fitness assignment
6. Diversity assignment
7. Selection
8. Replacement
9. Stopping criteria

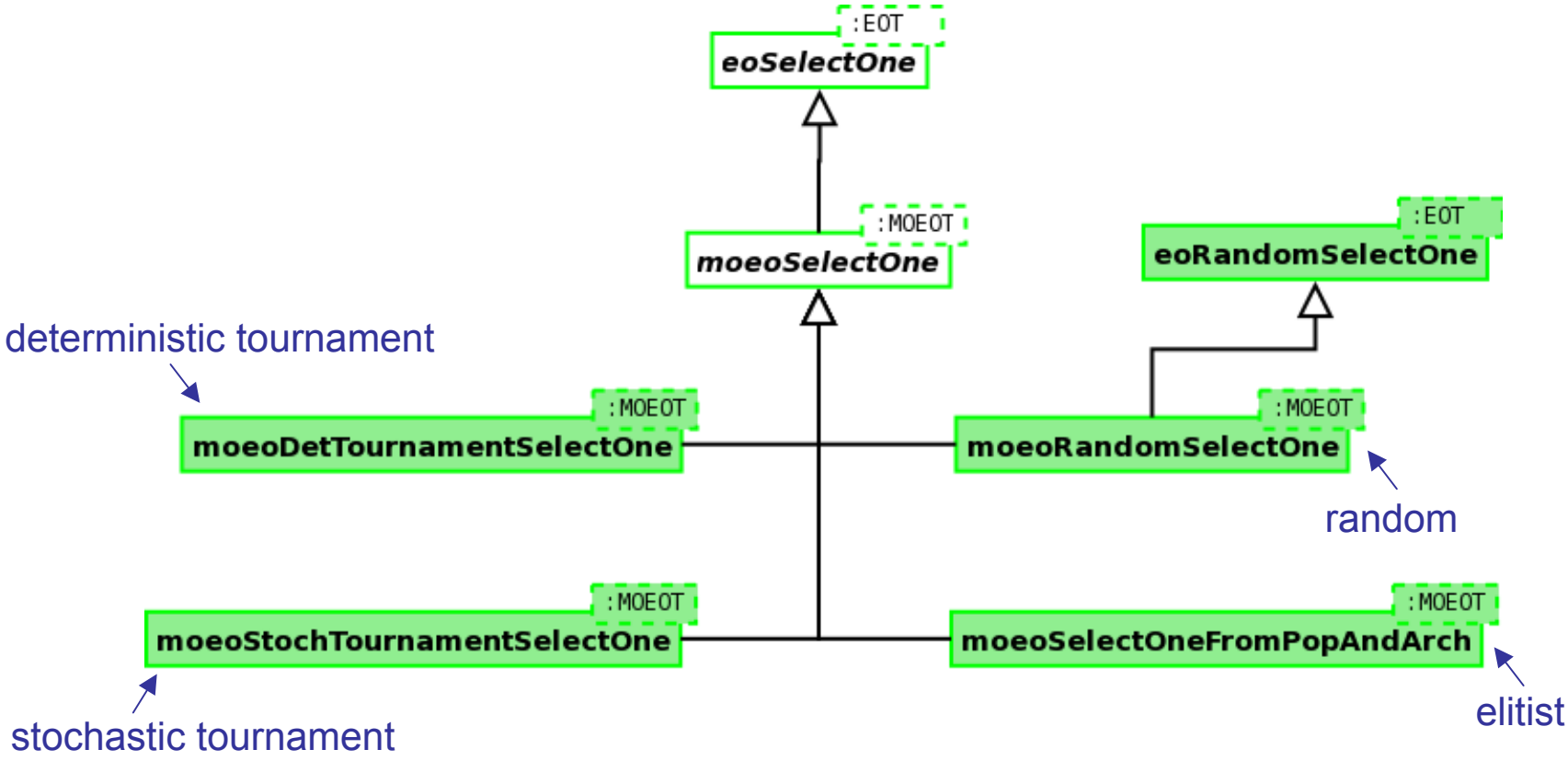
Selection

• The selection process differs of the mono-objective case as it have to use fitness and diversity values, and not objective value(s)

• Most common selection strategies
  - Random
  - Deterministic tournament
  - Stochastic tournament
  - Elitist selection



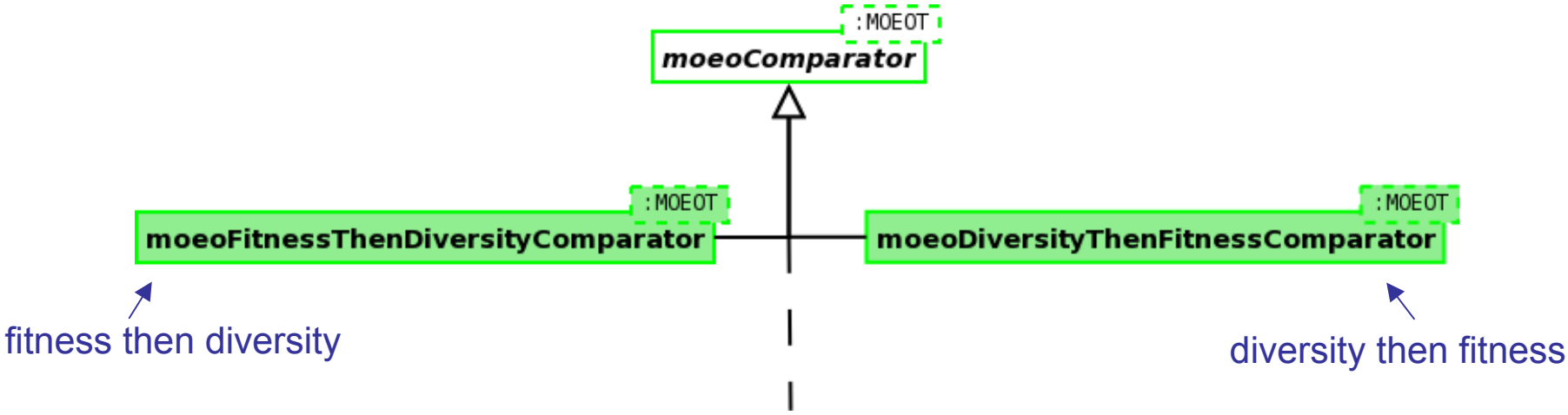
Selection: Core Classes



How to compare solution?

- According to fitness values?
- According to diversity values?
- Both?

Comparator: Core Classes



A Multi-Objective Evolutionary Algorithm

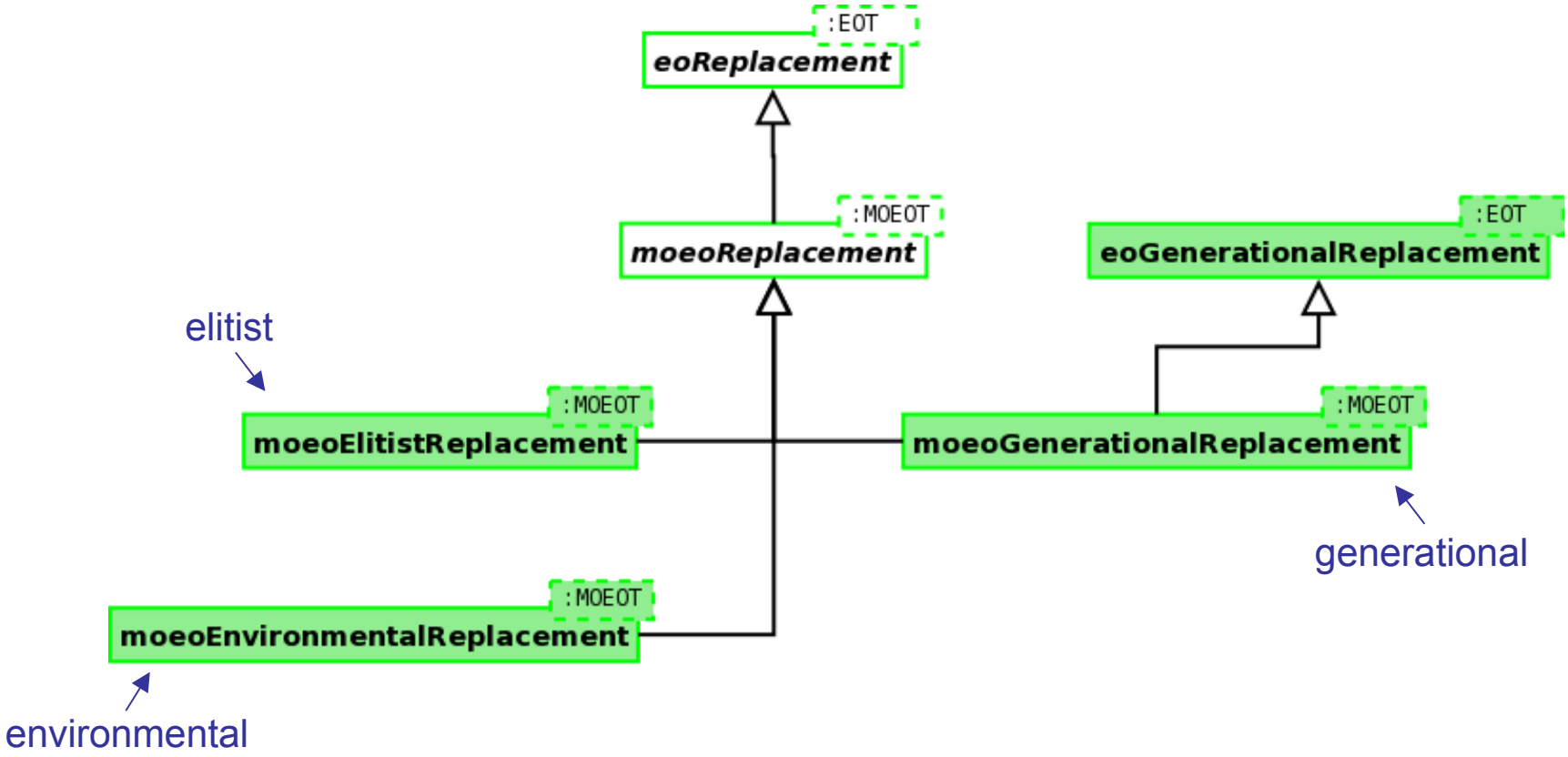
1. Representation
2. Initialization
3. Evaluation
4. Variation operators
5. Fitness assignment
6. Diversity assignment
7. Selection
8. Replacement
9. Stopping criteria

Replacement

- Generational
  - Only the offspring survives
- Elitist
  - The worst N solutions don’t survive in the population of generation G+1
- Environmental
  - Repeat until the population size = N
    - The worst solution is deleted
    - The fitness and diversity values are updated

- Solutions are compared using a *moeoComparator*

Replacement: Core Classes

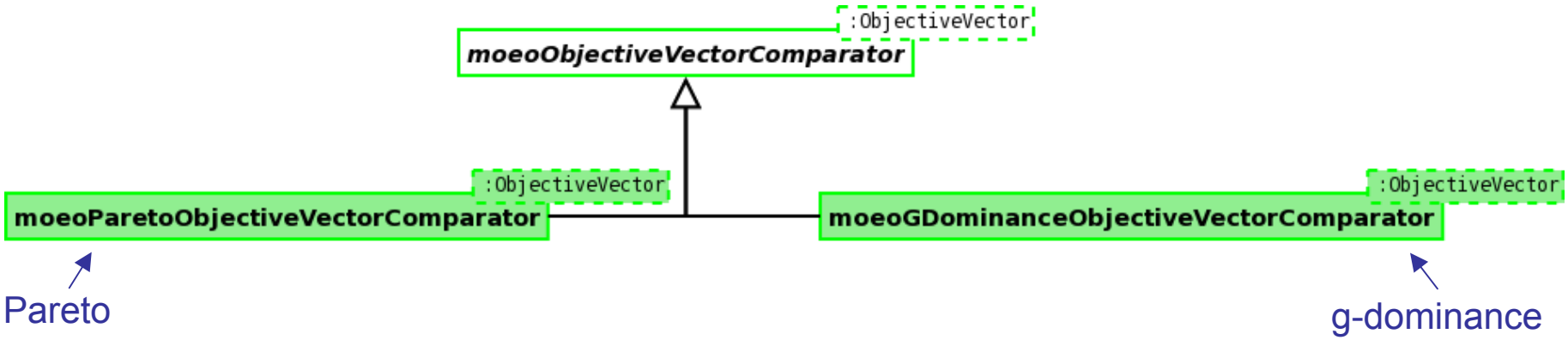


A Multi-Objective Evolutionary Algorithm

1. Representation
2. Initialization
3. Evaluation
4. Variation operators
5. Fitness assignment
6. Diversity Assignment
7. Selection
8. Replacement
9. Stopping criteria ➔ same as within ParadisEO-EO

Archiving

- Secondary population that stores non-dominated solutions found during the optimization process
  - Preserving non-dominated solutions
- Objective vector comparison strategies
  - Pareto dominance
  - G-dominance [Molina et al. 2007]



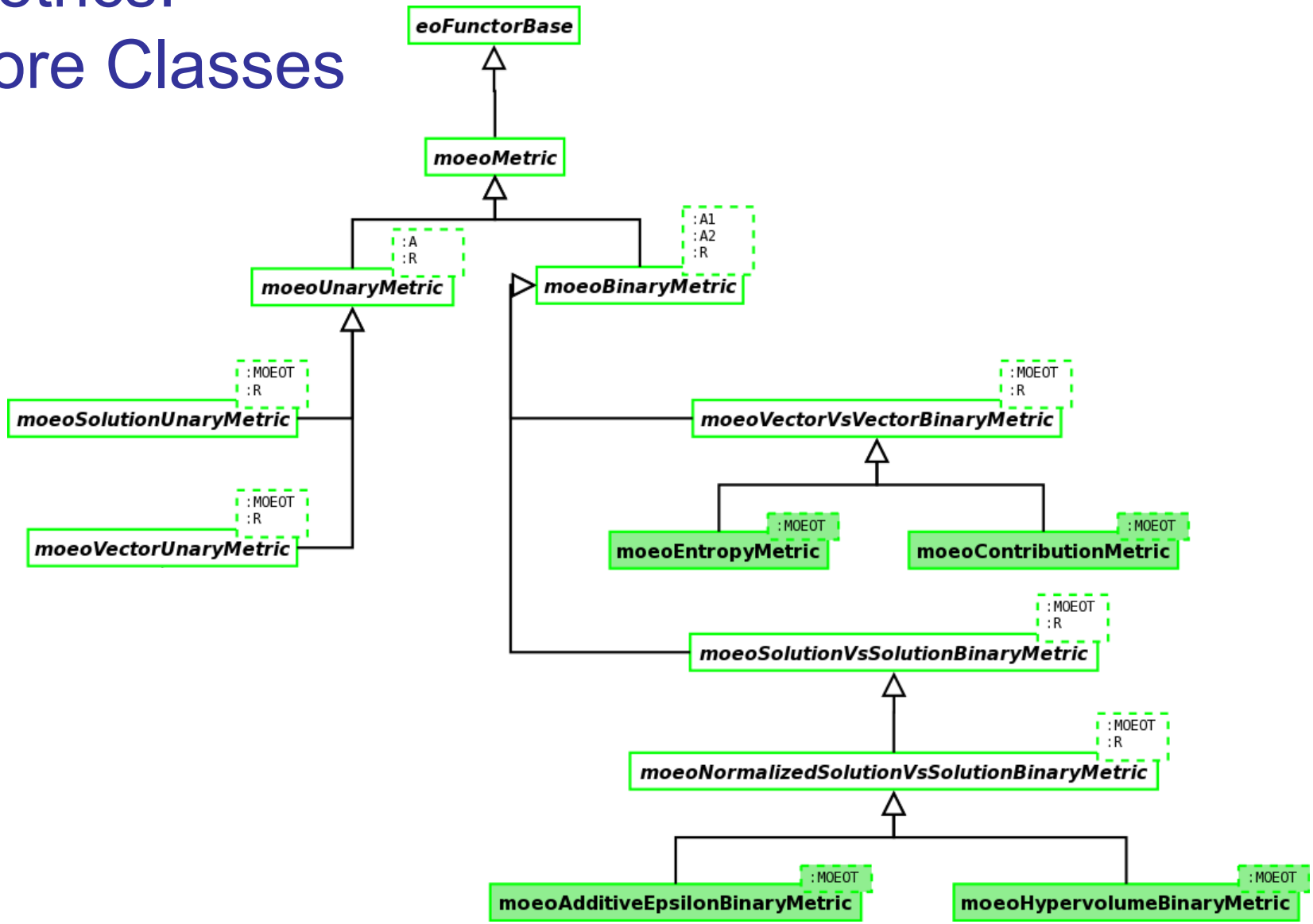
Statistical Tools

- Saving the Pareto front at each generation

- On-line metrics computation
  - Entropy
  - Contribution

- Metrics for pairwise comparisons (can be used within IBEA)
  - Additive epsilon metric
  - Hypervolume metric

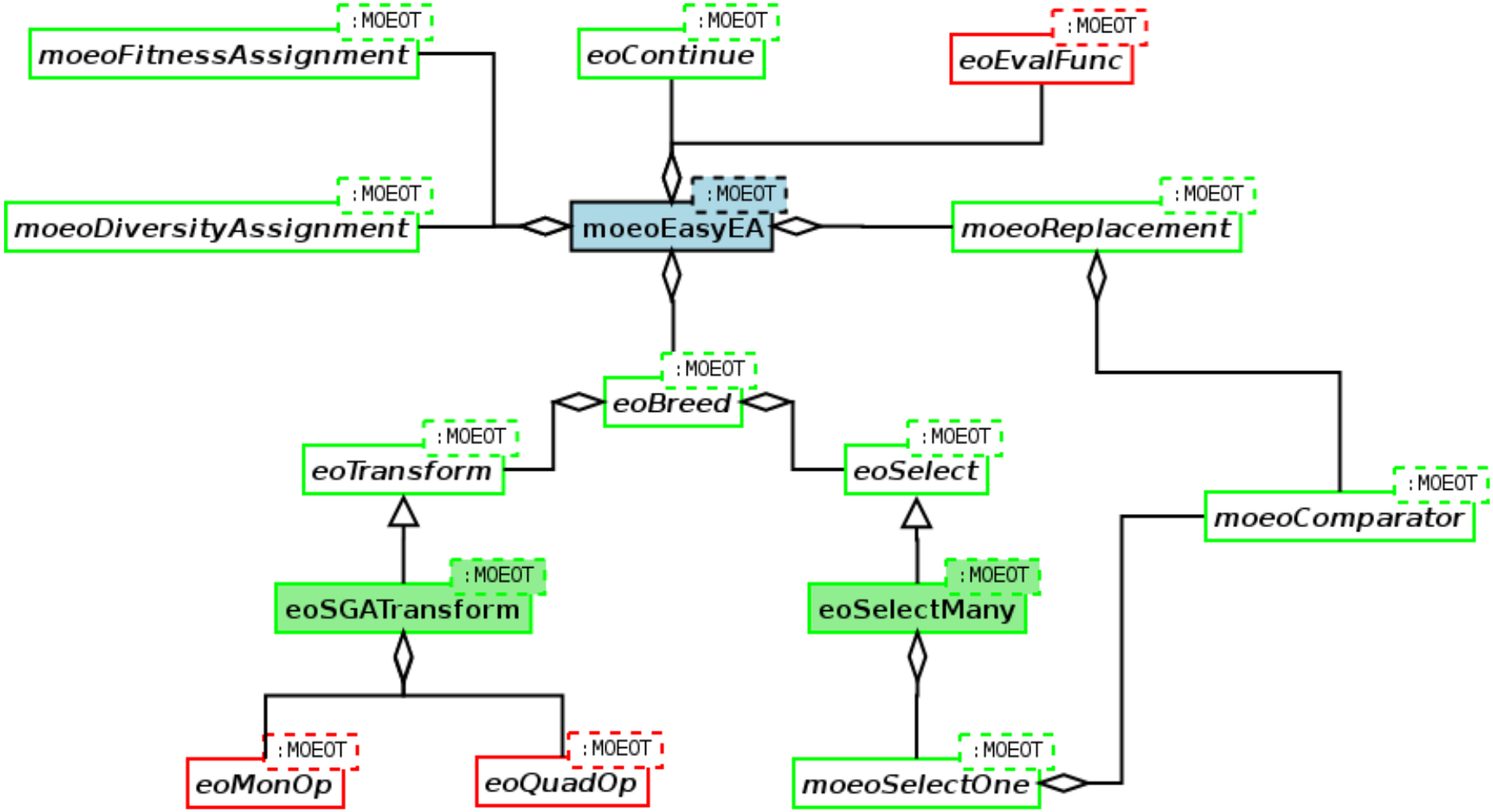
Metrics: Core Classes



Implementation of a Multi-Objective EA

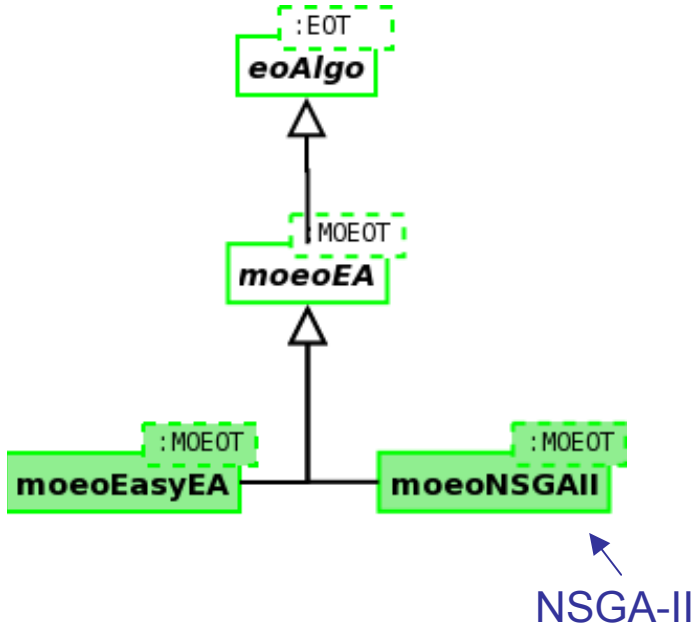
```
eoPop <MOEOT> pop;                          /* population */
moeoArchive <MOEOT> archive;                /* PO* */
eoInit <MOEOT> init;                        /* initialization */
eoEvalFunc <MOEOT> eval;                    /* evaluation */
eoTransform <MOEOT> transform;              /* variation operators */
moeoFitnessAssignment <MOEOT> fitness;
moeoDiversityAssignment <MOEOT> diversity;
moeoSelect <MOEOT> select;                  /* selection scheme */
moeoReplacement <MOEOT> replace;            /* replacement scheme */
eoCheckPoint <MOEOT> checkpoint;            /* checkpointing */
eoContinue <MOEOT> stop;                    /* stopping criteria */
moeoEasyEA <MOEOT> algo (stop, eval, select, transform,
replace, fitness, diversity);
algo(pop);                                  /* run the algorithm */
```

Core Classes of the EA



Easy-to-use EAs

Standard evolutionary algorithms even more easy-to-use than *moeoEasyEA*



Conclusion

- Download, test and enjoy
  - http://paradiseo.gforge.inria.fr
  - Free download of paradiseo-ix86-1.0-alpha
- Future works
  - More fitness and diversity assignment strategies
  - More easy-to-use EAs (IBEA, SPEA2…)
  - More metrics (Hypervolume)
  - More archiving strategies
  - Multi-objective local search (IBMOLS)
  - A-priori algorithms
  - Interactive resolution (coupled with GUIMOO)