INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

INRIA

CENTRE NATIONAL DE LA RECHERCHE SCIENTIFIQUE

ParadisEO-PEO
Hybrid and Parallel Metaheuristics

Nouredine MELAB
El-Ghazali TALBI

Paradiseo

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

Outline

- Contributions
- Hybrid Metaheuristics Development
- Case Study: Traveling Salesman Problem
- Low Level Relay GA ↔ HC
- High Level Relay GA + HC
- High Level Coevolutionary Island Model
  - HL Coevolutionary – Asynchronous Island Model
  - HL Coevolutionary – Synchronous Island Model
- ParadisEO-PEO – Parallel Models
  - The Parallel Evaluation of the Population
  - Parallel Evaluation of the Objective Function
- Conclusions

Contributions

ParadisEO-EO - a framework of Evolutionary Algorithms

ParadisEO-MO

solution-based metaheuristics

- Hill Climbing
- Simulated Annealing
- Tabu search

ParadisEO-PEO

parallel and distributed metaheuristics

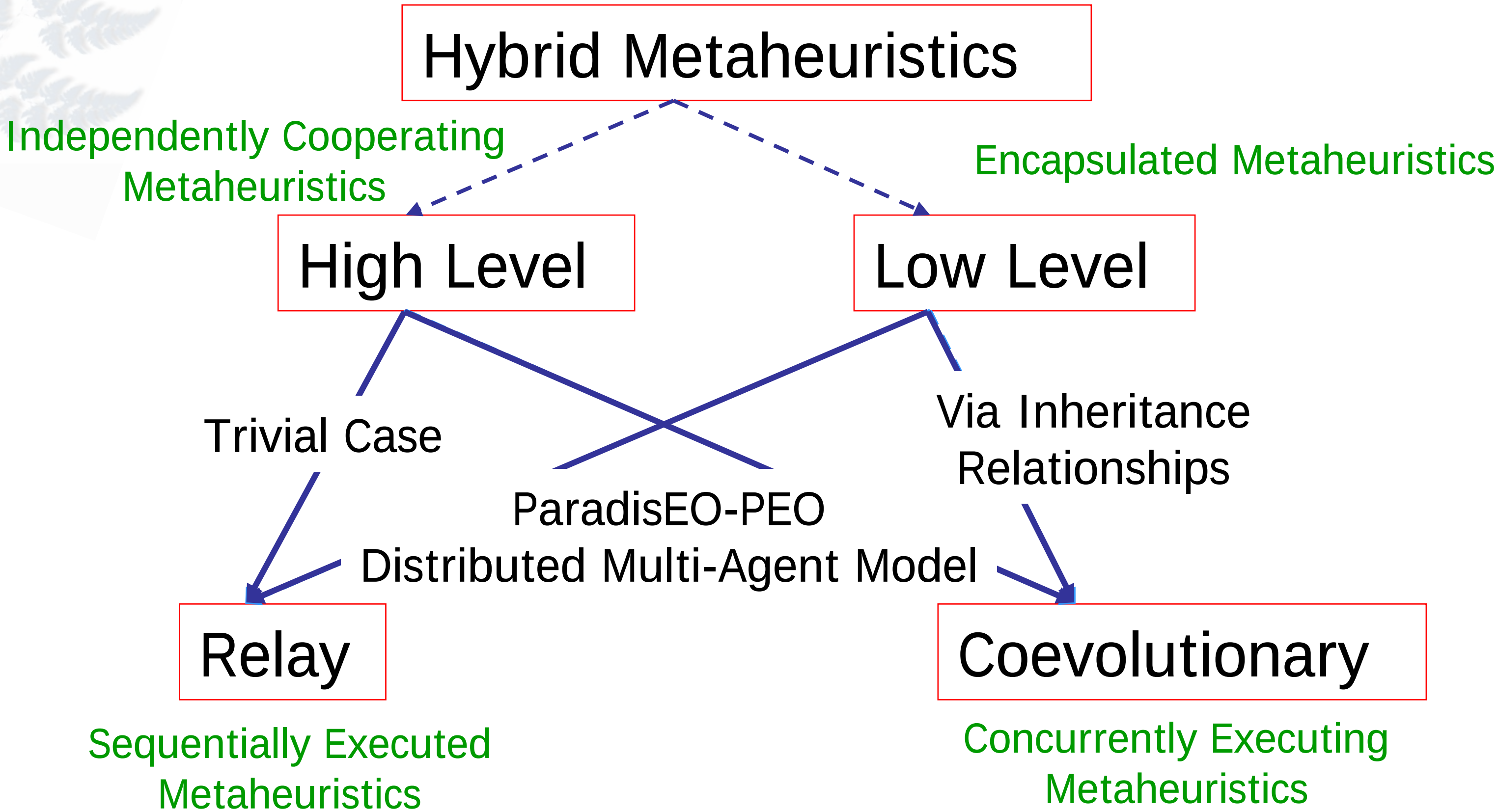
- **Parallelism**
  partitioning solutions at several steps
- **Cooperation / Hybridization**
  algorithms exchange solutions
  Ex. island cooperation

ParadisEO-MOEO

techniques related to multi-objective optimization

http://paradiseo.gforge.inria.fr

Hybrid Metaheuristics Development



E-G. Talbi, **A taxonomy of hybrid metaheuristics**, Journal of Heuristics, 2002.

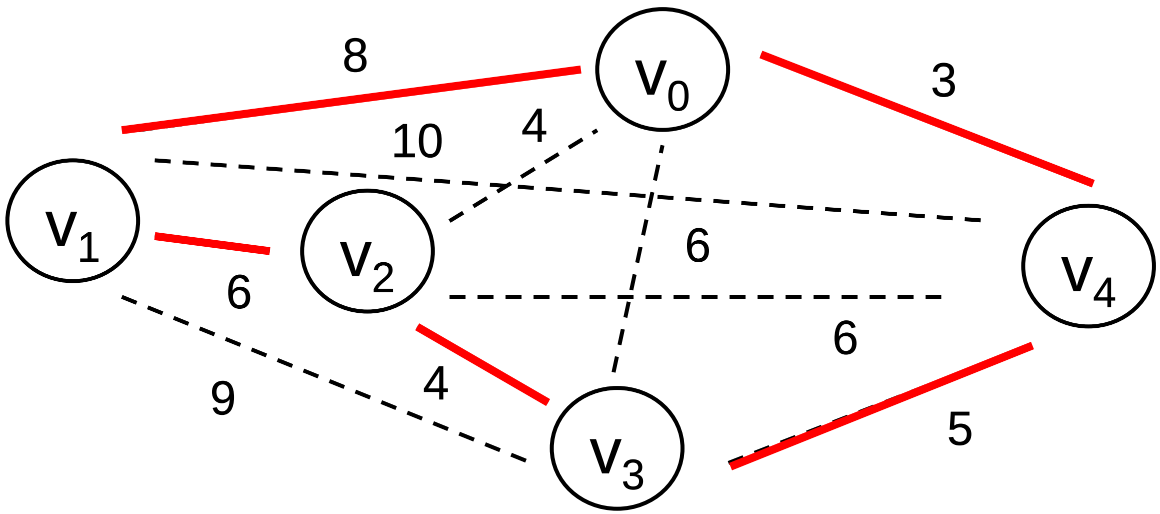
Outline

- Contributions
- Hybrid Metaheuristics Development
- **Case Study: Traveling Salesman Problem**
- Low Level Relay GA ↔ HC
- High Level Relay GA + HC
- High Level Coevolutionary Island Model
  - HL Coevolutionary – Asynchronous Island Model
  - HL Coevolutionary – Synchronous Island Model
- ParadisEO-PEO – Parallel Models
  - The Parallel Evaluation of the Population
  - Parallel Evaluation of the Objective Function
- Conclusions

ParadisEO-PEO

Case Study - Traveling Salesman Problem (TSP)

"Given a collection of N cities and the distance between each pair of them, the TSP aims at finding the shortest route visiting all of the cities".

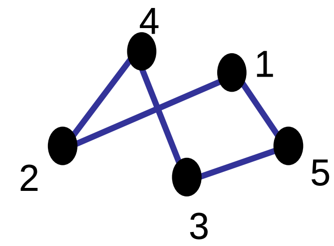


Shortest Route Length: 26

ParadisEO-PEO
Case Study - Traveling Salesman Problem (TSP)

Encoding: ordered sequence of visited vertices

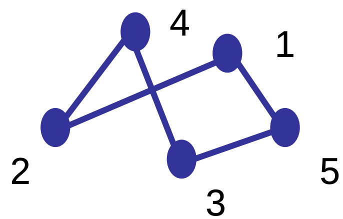
2	1	5	3	4	⟺	1	5	3	4	2	⟺	4	3	5	1	2

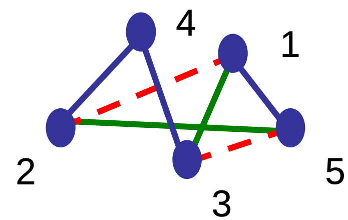


Operators:

Two-Opt: two points within the string are selected and the segment between them is inverted

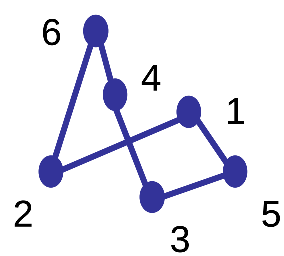
2	1	5	3	4
2	5	1	3	4

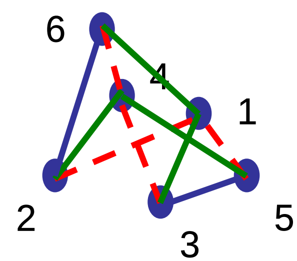




City-Swap: the values contained in two positions are exchanged, thus introducing four new edges in the string

2	1	5	3	4	6
2	4	5	3	1	6

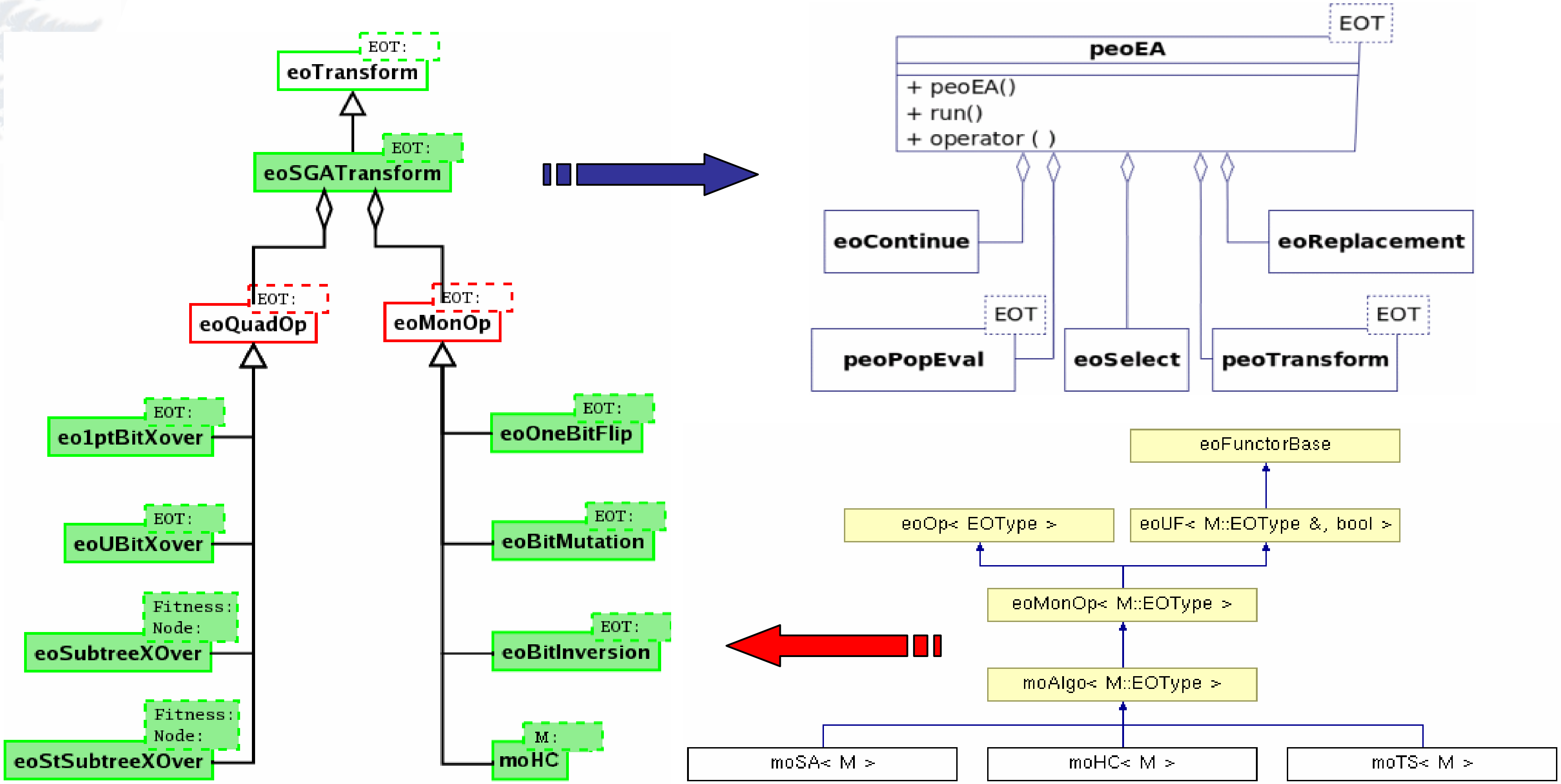




Outline

- Contributions
- Hybrid Metaheuristics Development
- Case Study: Traveling Salesman Problem
- **Low Level Relay GA ↔ HC**
- High Level Relay GA + HC
- High Level Coevolutionary Island Model
  - HL Coevolutionary – Asynchronous Island Model
  - HL Coevolutionary – Synchronous Island Model
- ParadisEO-PEO – Parallel Models
  - The Parallel Evaluation of the Population
  - Parallel Evaluation of the Objective Function
- Conclusions

Low Level Relay - GA ↔ HC (1)



A single solution method is a unary operator !!

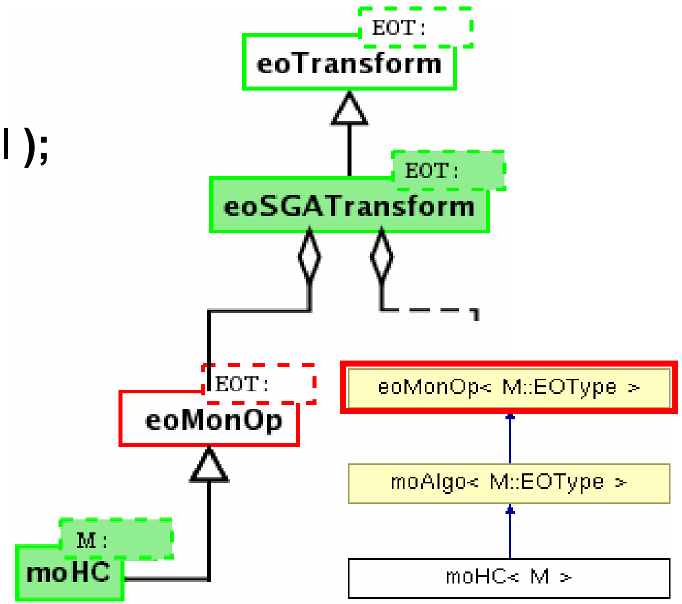
Low Level Relay - GA ↔ HC (2)

#1 Define a Local Search Method, *e.g.* Hill Climbing

```
…
moHC<TwoOpt> hc( pmx_two_opt_init, ..., full_eval );
…
```

#2 Define the Evolutionary Algorithm

```
…
eoSGATransform<Route> ox_transform(
          order_cross, CROSS_RATE,
          city_swap_mut, MUT_RATE );
…

peoEA<Route> ox_ea( ox_checkpoint, ..., ox_transform, ox_replace);
…
```



The mutation operator can be replaced by a local search method as all the MO algorithms have as base class in the hierarcy the eoMonOp class.

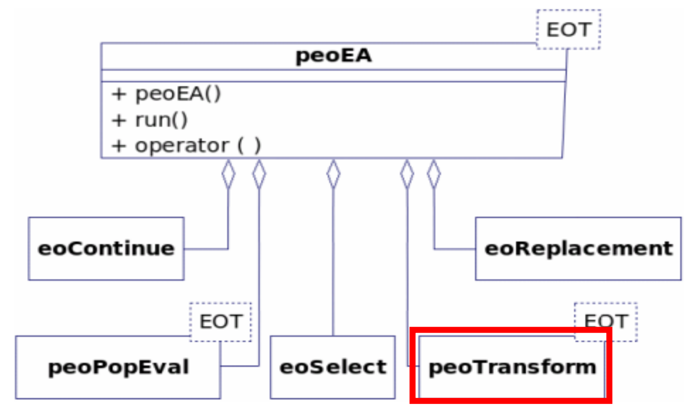
Low Level Relay - GA ↔ HC (3)

#1 Define a Local Search Method, *e.g.* Hill Climbing

```
…
moHC<TwoOpt> hc( pmx_two_opt_init, ..., full_eval );
…
```

#2 Replace the mutation operator by a Local Search

```
…
eoSGATransform<Route> ox_transform(
            order_cross, CROSS_RATE,
            hc, MUT_RATE );
…

peoEA<Route> ox_ea( ox_checkpoint, ..., ox_transform, ox_replace);
…
```



The Hill Climbing local search operator is applied with a MUT_RATE probability.

Low Level Relay - GA ↔ HC (4)

#1 All the required source files are located under $PEO_HOME/tutorial/Walkthrough/
(where the PEO_HOME variable indicates the location of your ParadisEO-PEO install).

#2 EDIT: switch to your ParadisEO-PEO install directory. Assumming you are already in the install directory you may type the followings (in a console):

$> cd tutorial/Walkthrough/	switch to the walkthrough dir.
$> ls –al	list the files inside the dir.
… exampleA.cpp ... ←	Low Level Relay Example
… exampleB.cpp ...	
… exampleC.cpp ...	
… exampleD.cpp ...	
… exampleE.cpp ...	
$> emacs exampleA.cpp & ←	open the file inside an editor

#3 The exampleA.cpp file already contains the required code for building the hill climbing method and the evolutionary algorithm.

Edit the file in order to replace the mutation operator by the hill climbing method!

Low Level Relay - GA ↔ HC (5)

#1 If the ParadisEO package is corectly installed, you should already have a Makefile in the $PEO_HOME/tutorial/Walkthrough/ directory.

#2 COMPILE: switch to your ParadisEO-PEO install directory. Assumming you are already in the install directory you may type the following (in a console):

```
$> cd tutorial/Walkthrough/        step into the source directory
$> make                            COMPILE the modified files!
```

#3 In case you have errors, switch back to the $PEO_HOME/tutorial/Walkthrough/ directory if outside and modify the exampleA.cpp file.

#4 EXECUTE: in case there are no errors, the executable is placed in the $PEO_HOME/tutorial/Walkthrough/ directory. In the build directory type:

```
$> ./exampleA @config/lessonA.param        run the low level relay example
```

Experiment by changing the different parameters of the algorithms!

Outline

- Contributions
- Hybrid Metaheuristics Development

- Case Study: Traveling Salesman Problem

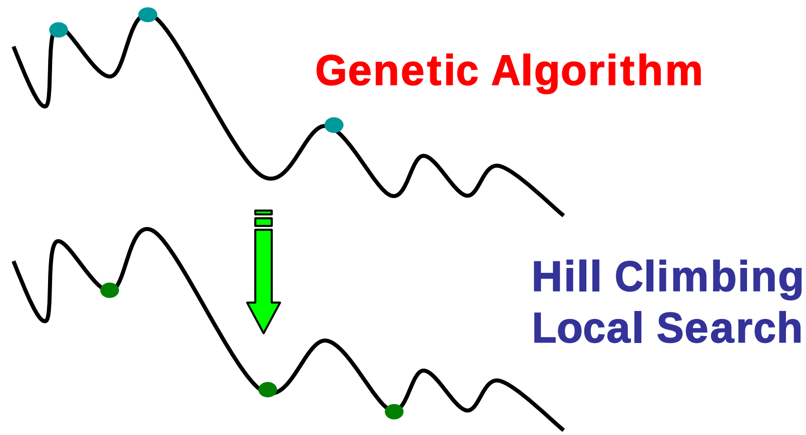
- Low Level Relay GA ↔ HC

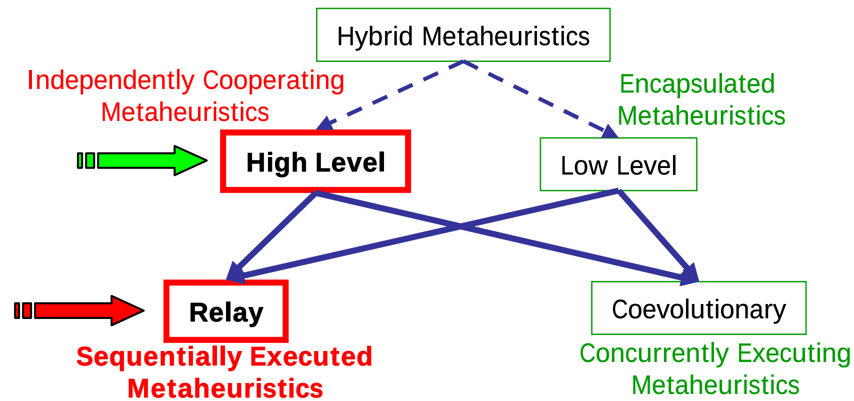
- **High Level Relay GA + HC**

- High Level Coevolutionary Island Model
  - HL Coevolutionary – Asynchronous Island Model
  - HL Coevolutionary – Synchronous Island Model

- ParadisEO-PEO – Parallel Models
  - The Parallel Evaluation of the Population
  - Parallel Evaluation of the Objective Function

- Conclusions





High Level Relay - GA + HC (1)

#1 Define a Local Search Method, *e.g.* Hill Climbing

```
…
moHC<TwoOpt> hc( pmx_two_opt_init, ..., full_eval );
…
```

#2 Define and Execute an Evolutionary Algorithm

```
…
peoEA<Route> ox_ea( ox_checkpoint, ..., ox_transform, ox_replace);
ox_ea( ox_pop );
…
```

#3 Apply the Local Search Method on the resulting population

```
…
std :: cout << "Initial value: " << ox_pop[ 0 ].fitness() << std :: endl;
hc( ox_pop[ 0 ] );
std :: cout << "Final value: " << ox_pop[ 0 ].fitness() << std :: endl;
…
```

The Hill Climbing local search operator is applied on the resulting individuals.

High Level Relay - GA + HC (2)

Add a loop for applying the Local Search algorithm on all the obtained solutions!

```
…
moHC<TwoOpt> hc( pmx_two_opt_init, ..., full_eval );
…
peoEA<Route> ox_ea( ox_checkpoint, ..., ox_transform, ox_replace);
ox_ea( ox_pop );
…

…
for ( int index = 0; index < ox_pop.size(); index++ ) {

        std :: cout << "Initial value: " << ox_pop[ index ].fitness() << std :: endl;
        hc( ox_pop[ index ] );
        std :: cout << "Final value: " << ox_pop[ index ].fitness() << std :: endl;
}
…
```

Low Level Relay - GA + HC (3)

#1 All the required source files are located under $PEO_HOME/tutorial/Walkthrough/
(where the PEO_HOME variable indicates the location of your ParadisEO-PEO install).

#2 EDIT: switch to your ParadisEO-PEO install directory. Assumming you are already in the install directory you may type the followings (in a console):

$> cd tutorial/Walkthrough/ switch to the walkthrough dir.

$> ls –al list the files inside the dir.

… exampleA.cpp ...

… exampleB.cpp ... High Level Relay Example

… exampleC.cpp ...

… exampleD.cpp ...

… exampleE.cpp ...

$> emacs exampleB.cpp & open the file inside an editor

#3 The exampleB.cpp file already contains the required code for building the hill climbing method and the evolutionary algorithm.

Edit the file in order to add a loop surrounding the call of the local search method!

Low Level Relay - GA + HC (4)

#1 If the ParadisEO package is corectly installed, you should already have a Makefile in the $PEO_HOME/tutorial/Walkthrough/ directory.

#2 COMPILE: switch to your ParadisEO-PEO install directory. Assumming you are already in the install directory you may type the following (in a console):

```
$> cd tutorial/Walkthrough/        step into the source directory
$> make                            COMPILE the modified files!
```

#3 In case you have errors, switch back to the $PEO_HOME/tutorial/Walkthrough/ directory if outside and modify the exampleB.cpp file.

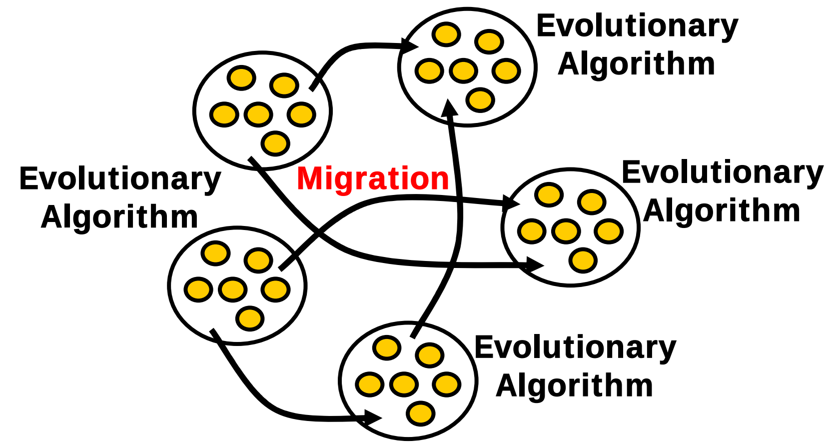
#4 EXECUTE: in case there are no errors, the executable is placed in the $PEO_HOME/tutorial/Walkthrough/ directory. In the build directory type:

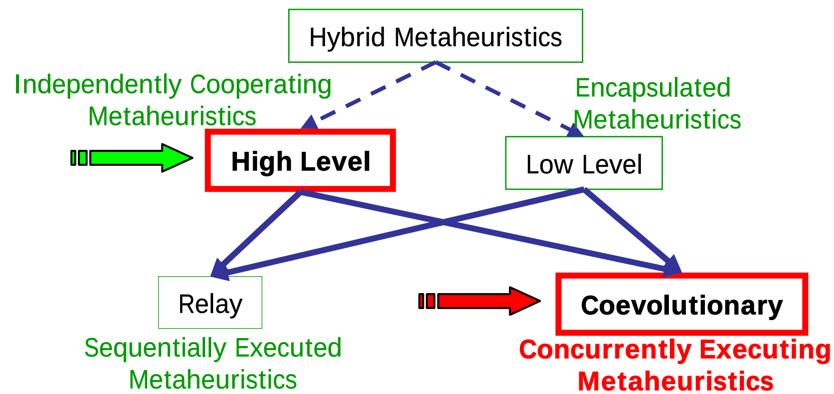
```
$> ./exampleB @config/lessonB.param        run the high level relay example
```

Experiment by changing the different parameters of the algorithms!

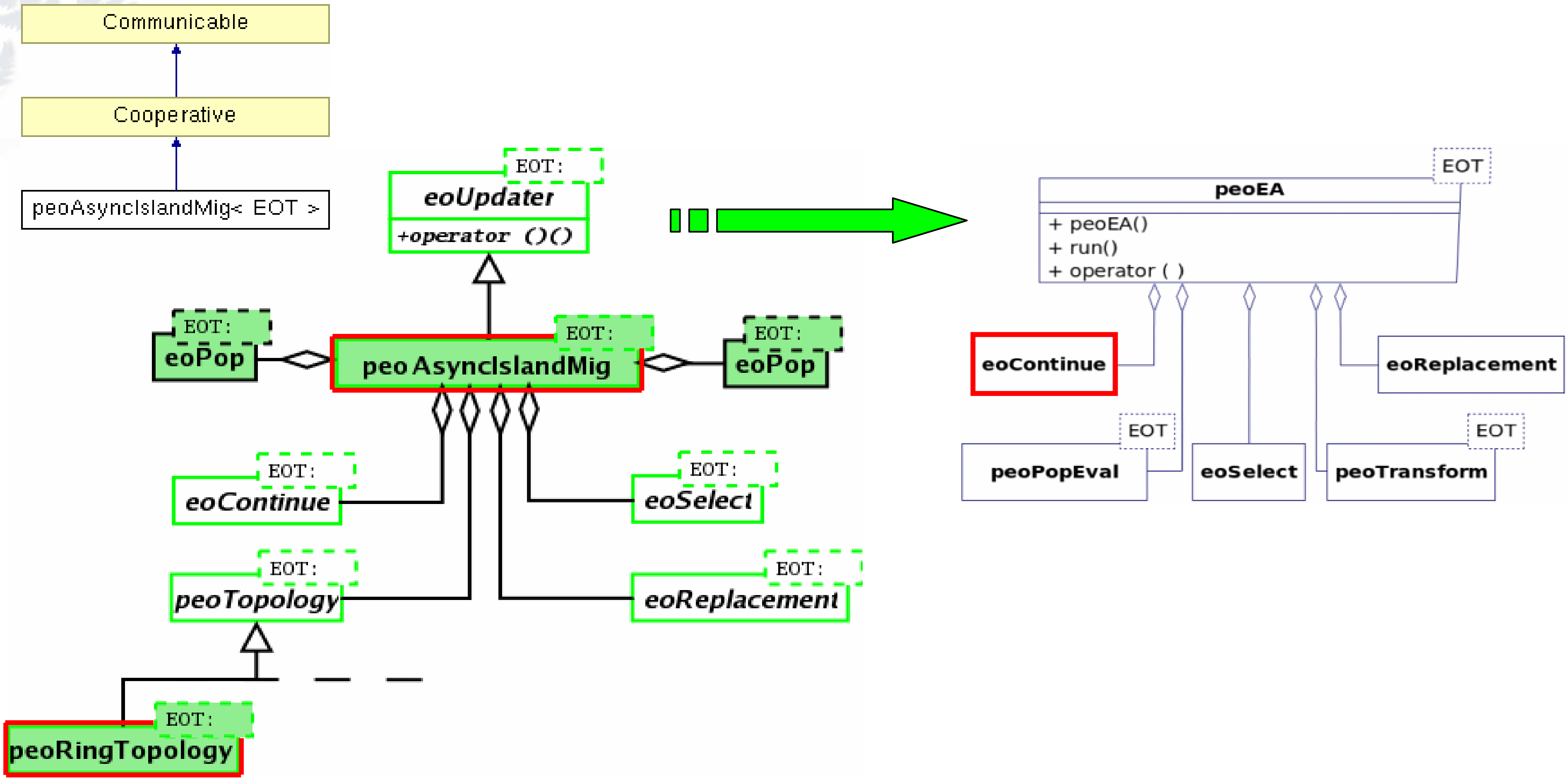
High Level Coevolutionary Island Model

- Contributions
- Hybrid Metaheuristics Development
- Case Study: Traveling Salesman Problem
- Low Level Relay GA ↔ HC
- High Level Relay GA + HC
- **High Level Coevolutionary Island Model**
  - HL Coevolutionary – Asynchronous Island Model
  - HL Coevolutionary – Synchronous Island Model
- ParadisEO-PEO – Parallel Models
  - The Parallel Evaluation of the Population
  - Parallel Evaluation of the Objective Function
- Conclusions





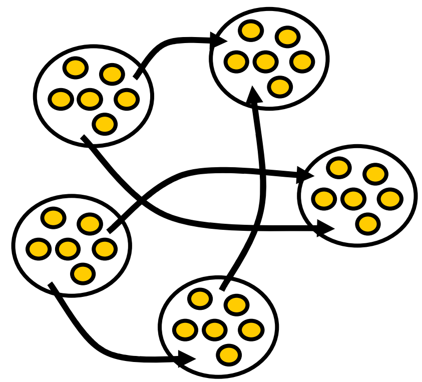
HL Coevolutionary – Async. Island Model (1)



HL Coevolutionary – Async. Island Model (2)

#1 Define an object specifying the Migration Topological Model

```
RingTopology topo;
```



#2 Define an Asynchronous Migration Object

```
...
eoPeriodicContinue<Route>        ox_mig_cont ( MIG_FREQ );

eoRandomSelect<Route>            ox_mig_select_one;
eoSelectNumber<Route>            ox_mig_select( ox_mig_select_one, MIG_SIZE );
eoPlusReplacement<Route>         ox_mig_replace;


peoAsyncIslandMig<Route> ox_mig(
            ox_mig_cont,         // migrations occur at every MIG_FREQ generations
            ox_mig_select,       // strategy of selection for obtaining the emigrants
            ox_mig_replace,      // strategy of replacement for integrating the immigrants
            topo,                // topology – migrations follow a pre-defined topology
            ox_pop, ox_pop );    // the source & destination populations
```

HL Coevolutionary – Async. Island Model (3)

#3 Link the Asynchronous Migration Object to an Evolutionary Algorithm

- **the Migration Object has to be called at every generation – checkpointing mechanism !**

```
...
eoCheckPoint<Route> ox_checkpoint( ox_cont );
ox_checkpoint.add( ox_mig );
...
```

- **the Evolutionary Object has to be set as OWNER of the Migration Object !**

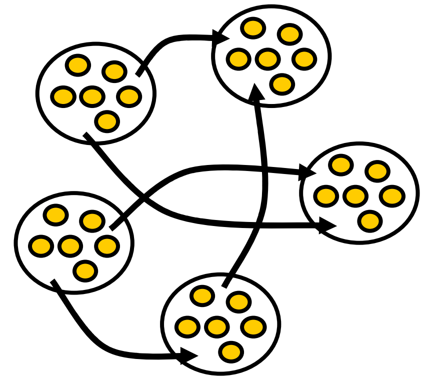
```
...
peoEA<Route> ox_ea( ox_checkpoint, ox_pop_eval, …, ox_replace );
ox_mig.setOwner( ox_ea );
ox_ea( ox_pop );
...
```

#4 Construct two or more Evolutionary Algorithms, each of them with an associated Migration Object! The EAs have to be part of the same topology (use the same var.)!

HL Coevolutionary – Sync. Island Model

#1 Define an object specifying the Migration Topological Model

```
RingTopology topo;
```



#2 Replace Async by Sync - Synchronous Migration Object !

```
...
eoPeriodicContinue<Route>        ox_mig_cont ( MIG_FREQ );

eoRandomSelect<Route>            ox_mig_select_one;
eoSelectNumber<Route>            ox_mig_select( ox_mig_select_one, MIG_SIZE );
eoPlusReplacement<Route>         ox_mig_replace;

peoSyncIslandMig<Route> ox_mig(
        MIG_FREQ,                // migrations occur at every MIG_FREQ generations
        ox_mig_select,           // strategy of selection for obtaining the emigrants
        ox_mig_replace,          // strategy of replacement for integrating the immigrants
        topo,                    // topology – migrations follow a pre-defined topology
        ox_pop, ox_pop );        // the source & destination populations
```

High Level Coevolutionary Island Model

#1 All the required source files are located under $PEO_HOME/tutorial/Walkthrough/ (where the PEO_HOME variable indicates the location of your ParadisEO-PEO install).

#2 EDIT: switch to your ParadisEO-PEO install directory. Assumming you are already in the install directory you may type the followings (in a console):

$> cd tutorial/Walkthrough/		switch to the walkthrough dir.
$> ls –al		list the files inside the dir.
… exampleA.cpp ...		
… exampleB.cpp ...		
… exampleC.cpp ...	←	HL Coevolutionary Example
… exampleD.cpp ...		
… exampleE.cpp ...		
$> emacs exampleC.cpp &	←	open the file inside an editor

#3 The exampleC.cpp file already contains the required code for building two coevolving evolutionary algorithms as part of an asynchronous insular model.

Edit the file in order to transform the asynchronous model into a synchronous one!

Message Passing Interface - MPI

#1 You should already have installed an MPI distribution (MPICH/LAM-MPI/OpenMPI, etc.)! If you installed the entire ParadisEO package, you should already have the MPICH-2 installed and configured on your system.

Depending on your MPI distribution, the followings might be different. For the following commands, MPICH-2 is assumed to be installed.

#2 INITIAL TEST - list the machines running MPDs (MPICH Daemons):

```
$> mpdtrace
        machineA
        machineB
        ...
```

#3 INITIAL TEST - send a message to be passed 100 times in the ring of MPI Daemons:

```
$> mpdringtest 100
        time for 100 loops = 0.0880811… seconds
```

#4 EXECUTION TEST – launch an executable across the cluster:

```
$> mpiexec -n 10 /bin/hostname -f
```

High Level Coevolutionary Island Model

#1 If the ParadisEO package is corectly installed, you should already have a Makefile in the $PEO_HOME/tutorial/Walkthrough/ directory.

#2 COMPILE: switch to your ParadisEO-PEO install directory. Assumming you are already in the install directory you may type the following (in a console):

```
$> cd tutorial/Walkthrough/
$> make
```

step into the source directory

COMPILE the modified files!

#3 In case you have errors, switch back to the $PEO_HOME/tutorial/Walkthrough/ directory if outside and modify the exampleC.cpp file.

#4 EXECUTE: in case there are no errors, the executable is placed in the $PEO_HOME/tutorial/Walkthrough/ directory. In the build directory type:

```
$> mpiexec –n 2 ./exampleC @config/lessonC.param
```

HL Coevol. example

Experiment by changing the different parameters of the algorithms!

ParadisEO-PEO – Parallel Models

- Contributions
- Hybrid Metaheuristics Development

- Case Study: Traveling Salesman Problem

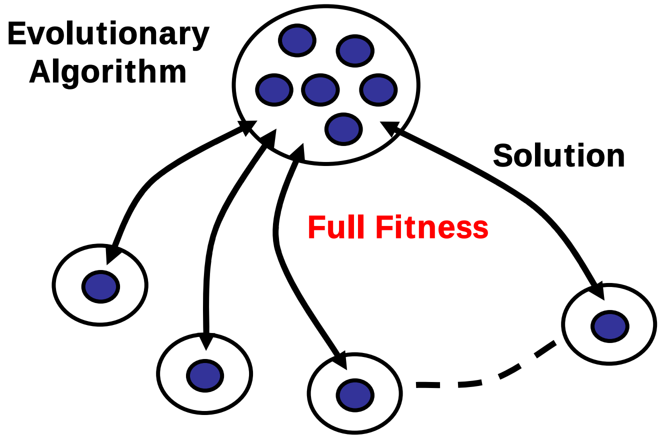
- Low Level Relay GA ↔ HC

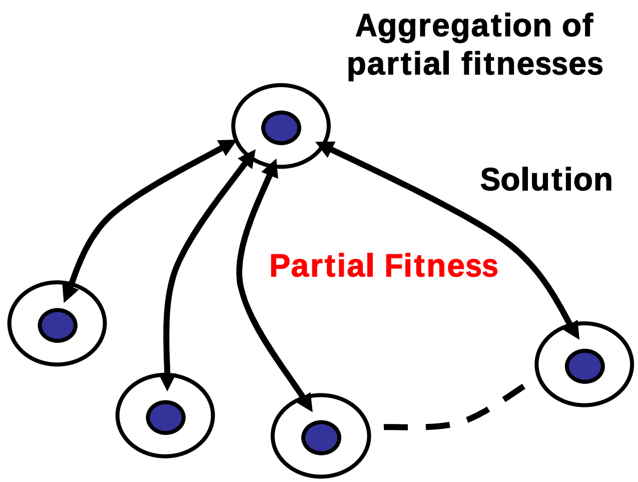
- High Level Relay GA + HC

- High Level Coevolutionary Island Model
    - HL Coevolutionary – Asynchronous Island Model
    - HL Coevolutionary – Synchronous Island Model

- **ParadisEO-PEO – Parallel Models**
    - The Parallel Evaluation of the Population
    - Parallel Evaluation of the Objective Function

- Conclusions





An unifying view of three parallel hierarchical levels

For both the population-based and solution-based metaheuristics

- the deployment of concurrent independent/cooperative metaheuristics
- the parallelization of a single step of the metaheuristic (based on distribution of the handled solutions)
- the parallelization of the processing of a single solution

The Parallel Evaluation of the Population (1)

#1 Define a Full Fitness Evaluator Object

…

RouteEval full_eval;

…

#2 Wrap the full evaluator into a parallel evaluator:

…

peoSeqPopEval<Route> ox_pop_eval(full_eval);

…

peoEA<Route> ox_ea(ox_checkpoint, ox_pop_eval, …);

ox_ea (ox_pop);

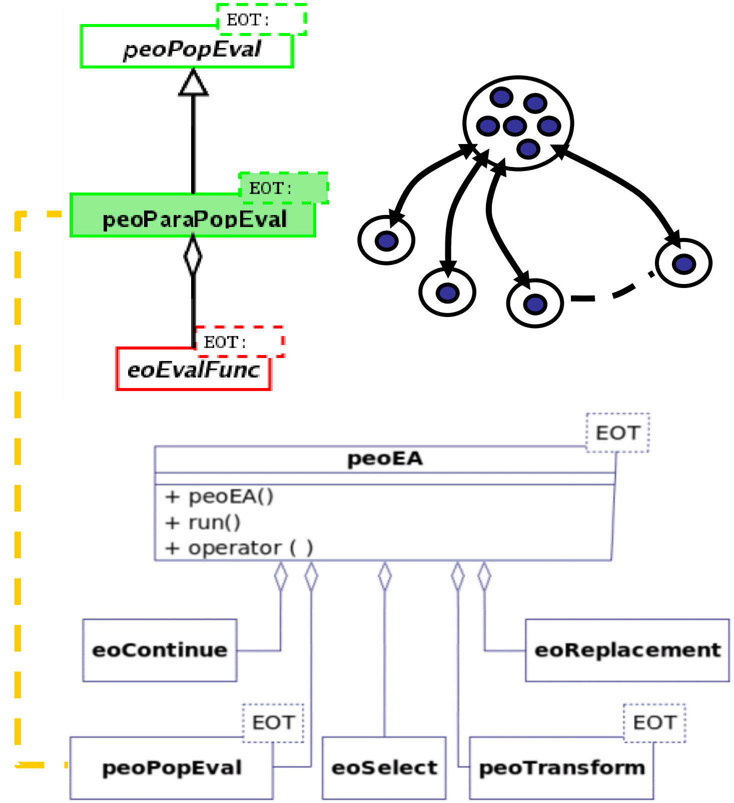
…

peoParaPopEval<Route> ox_pop_eval(full_eval);

…

peoEA<Route> ox_ea(ox_checkpoint, ox_pop_eval, …);

ox_ea (ox_pop);



The Parallel Evaluation of the Population (2)

#1 All the required source files are located under $PEO_HOME/tutorial/Walkthrough/ (where the PEO_HOME variable indicates the location of your ParadisEO-PEO install).

#2 EDIT: switch to your ParadisEO-PEO install directory. Assumming you are already in the install directory you may type the followings (in a console):

```
$> cd tutorial/Walkthrough/          switch to the walkthrough dir.
$> ls –al                            list the files inside the dir.
        … exampleA.cpp ...
        … exampleB.cpp ...
        … exampleC.cpp ...
        … exampleD.cpp ...   <----   Parallel Evaluation of the Pop.
        … exampleE.cpp ...
$> emacs exampleC.cpp &      <----   open the file inside an editor
```

#3 The exampleD.cpp file already contains the required code for building an evolutionary algorithm with sequential evaluation of the population.

Edit the file in order for the EA to evaluate the population in PARALLEL!

The Parallel Evaluation of the Population (3)

#1 If the ParadisEO package is corectly installed, you should already have a Makefile in the $PEO_HOME/tutorial/Walkthrough/ directory.

#2 COMPILE: switch to your ParadisEO-PEO install directory. Assumming you are already in the install directory you may type the following (in a console):

$> cd tutorial/Walkthrough/ — step into the source directory
$> make — COMPILE the modified files!

#3 In case you have errors, switch back to the $PEO_HOME/tutorial/Walkthrough/ directory if outside and modify the exampleD.cpp file.

#4 EXECUTE: in case there are no errors, the executable is placed in the $PEO_HOME/tutorial/Walkthrough/ directory. In the build directory type:

$> mpiexec –n 4 ./exampleD @config/lessonD.param — parallel evaluation

Experiment by changing the different parameters of the algorithms!

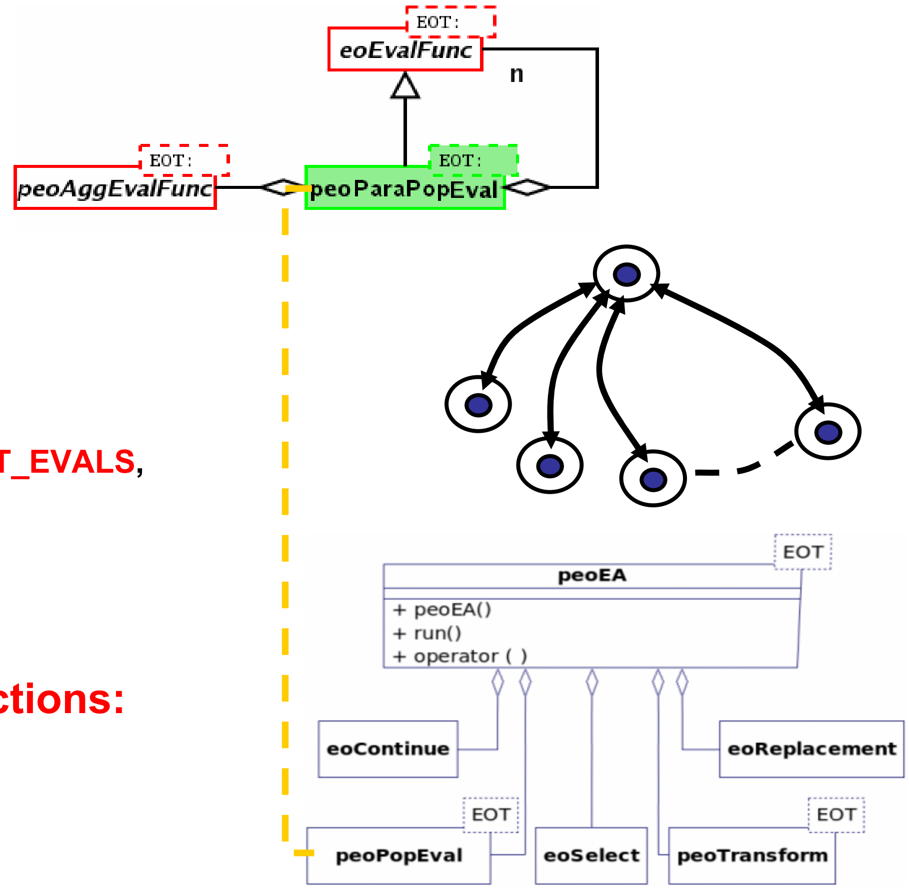
Parallel Evaluation of the Objective Function (1)

#1 Define an aggregation function and partial fitness function evaluators:

```
…
MergeRouteEval merge_eval;
std :: vector< eoEvalFunc<Route>* > part_eval;

for (int i = 1; i <= NUM_PART_EVALS; i ++) {

    part_eval.push_back (
            new PartRouteEval( (i-1) / (float) NUM_PART_EVALS,
                        i / (float) NUM_PART_EVALS)
    );
}
…
```

#2 Wrap the partial fitness eval. and merge functions:

```
…
peoParaPopEval<Route> ox_pop_eval( full_eval );

…
peoParaPopEval<Route> ox_pop_eval( part_eval, merge_eval );
```



Parallel Evaluation of the Objective Function (2)

#1 All the required source files are located under $PEO_HOME/tutorial/Walkthrough/ (where the PEO_HOME variable indicates the location of your ParadisEO-PEO install).

#2 EDIT: switch to your ParadisEO-PEO install directory. Assumming you are already in the install directory you may type the followings (in a console):

$> cd tutorial/Walkthrough/	switch to the walkthrough dir.
$> ls –al	list the files inside the dir.
… exampleA.cpp ...	
… exampleB.cpp ...	
… exampleC.cpp ...	
… exampleD.cpp ...	
… exampleE.cpp ...	Parallel Fitness Evaluation
$> emacs exampleC.cpp &	open the file inside an editor

#3 The exampleE.cpp file already contains the required code for building an evolutionary algorithm with parallel evaluation of the population.

Edit the file in order to have the EA to evaluate in parallel the objective function !

Parallel Evaluation of the Objective Function (3)

#1 If the ParadisEO package is corectly installed, you should already have a Makefile in the $PEO_HOME/tutorial/Walkthrough/ directory.

#2 COMPILE: switch to your ParadisEO-PEO install directory. Assumming you are already in the install directory you may type the following (in a console):

```
$> cd tutorial/Walkthrough/
$> make
```

step into the source directory
COMPILE the modified files!

#3 In case you have errors, switch back to the $PEO_HOME/tutorial/Walkthrough/ directory if outside and modify the exampleE.cpp file.

#4 EXECUTE: in case there are no errors, the executable is placed in the $PEO_HOME/tutorial/Walkthrough/ directory. In the build directory type:

```
$> mpiexec –n 4 ./exampleE @config/lessonE.param
```

parallel fitness evaluation

Experiment by changing the different parameters of the algorithms!

Outline

- Contributions
- Hybrid Metaheuristics Development
- Case Study: Traveling Salesman Problem
- Low Level Relay GA ↔ HC
- High Level Relay GA + HC
- High Level Coevolutionary Island Model
  - HL Coevolutionary – Asynchronous Island Model
  - HL Coevolutionary – Synchronous Island Model
- ParadisEO-PEO – Parallel Models
  - The Parallel Evaluation of the Population
  - Parallel Evaluation of the Objective Function
- **Conclusions**

Conclusions

ParadisEO-PEO is a white-box Object Oriented Framework

- clear conceptual separation of solution methods and problems
- maximum design and code reuse
- high flexibility - fine-grained EO objects

ParadisEO-PEO provides a broad range of features

- evolutionary algorithms, local searches and natural hybridization mechanisms (invariant parts provided).
- various transparent and portable parallel/distributed models

Experimental evaluation on academic problems and industrial applications

- high reuse capabilities, efficacy of the parallel/distributed models