Workgroup

Guido Perboli (guido.perboli@polito.it)
DAUIN - Politecnico di Torino

www.orgroup.polito.it

Rules

- Write a metaheuristics able to solve the symmetric TSP
- Given material
  - Data.zip
    - Instance files (.tsp)
    - Optimal solutions (.tour)
    - Parameters file: param.txt
  - Example.zip
    - Example of Tabu Search
    - Code compiled with Eclipse
    - Uses OpenTS by COINOR (http://www.coin-or.org/Ots/)
  - ExampleWorkgroup.zip
    - Example of presentation and results

- Given a list of towns and a distance between every pair of them, find the list of towns to be visited so that the travelling salesman will have the shortest route
- The distance matrix is simmetric

Instance file

NAME: berlin52
TYPE: TSP
COMMENT: 52 locations in Berlin (Groetschel)
DIMENSION: 52
EDGE_WEIGHT_TYPE: EUC_2D
NODE_COORD_SECTION
1 565.0 575.0
2 25.0 185.0
...
51 1340.0 725.0
52 1740.0 245.0
EOF

PARAMS

DataFileDir c:/MyDir

TSMaxIter 1000

TSRepetitions 1 (set TSRepetitions to 10 if a random process is used)

…..

ENDPARAMS

INSTANCES

berlin52.tsp

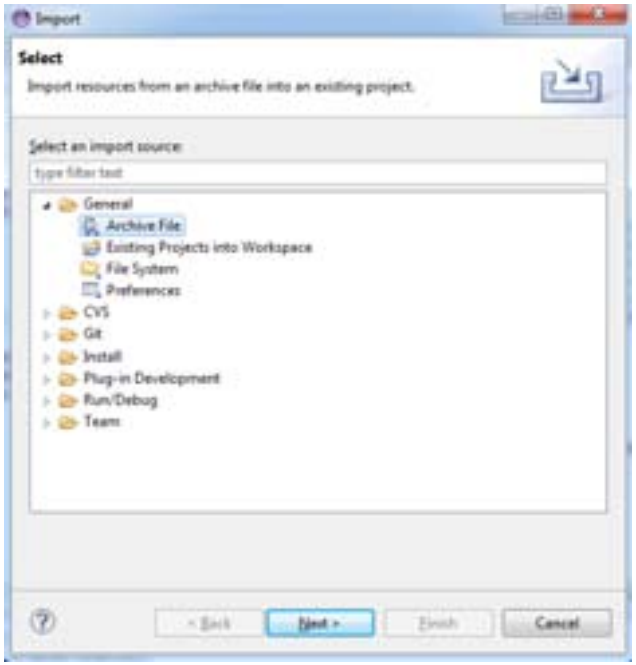
brazil58.tsp

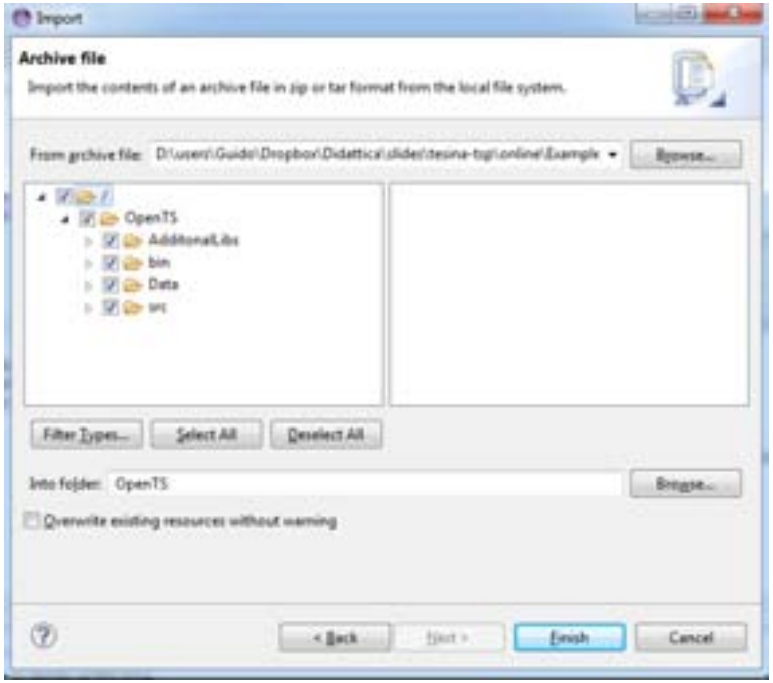
ENDINSTANCES

EOF

Compile the example

- Eclipse
  - File->Import





Compile the example (cont.)

- Select Project > Properties > Java Build Path > Libs
  - Click the button Add External JARs.
  - Select the file named OpenTS.jar in AdditionalLibs/OpenTS directory

Rules

- Upload your workgroup on the Web Portal (Section Elaborati) by January 3rd, 2014
  - Eclipse project, including source code
  - Compiled java file
    - It takes only one parameter form the command line, the parameter file
  - Excel file with the results
  - Presentation (ppt or pdf)
- For students resident outside Piedmont.
  - If any member of your group arrives from Christmas holidays after January 7th, send us an email by December 20th
  - At the presentation, at least half +1 members of the group must be compulsory present

Rules (cont.)

- Mark
  - Normal mark: Up to 10 points
  - Contest: the first three groups will take 2 additional point
    - We test your program on our instance set
- Workgroup
  - Program
    - Java
    - Metaheuristic of your choice
    - Tabu Search (http://www.coin-or.org/Ots/)
    - Genetic Algorithm (GALib http://sourceforge.net/projects/java-galib/)
    - Any other implemented by yourself

Rules (cont.)

- Presentation
  - Max 10 minutes
    - Title page
    - Metaheuristic (1 slide)
    - Metaheuristic blocks (2 slides)
    - Metaheuristic tuning (1 slides)
    - Computational results (2 slides)

Guido Perboli (guido.perboli@polito.it)

DAUIN - Politecnico di Torino

www.orgroup.polito.it

How to go on

- Choose you metaheuristic
  - Suggestion: TS, GA
- Search existing material
  - Google Scholar: e.g. keyword TSP tabu search
- Write the skeleton of your application
- Identify the blocks to implement
- Group management
  - Split the work among the different group members

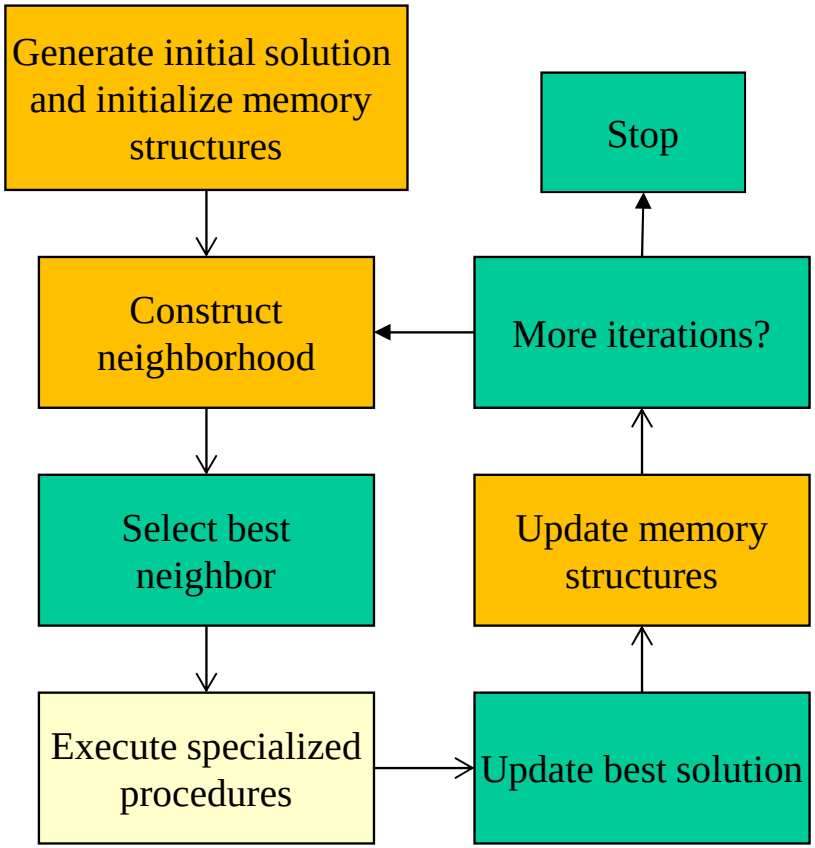
    E.g. Given the MH skeleton, assign to every member a different neighborhood

    Use the different skills. E.g. some one can be better in the statistical analisys of the results, someone else in the coding part

Example: Tabu Search

- I decide to implement a Tabu Search (TS) starting from OpenTS
- I implement a skeleton
- Solution representation
  - Ordered list of customers

Example: Tabu Search



Example: Tabu Search

- Define a solution
  - Extend SolutionAdapter and ObjectiveFunction
- Generate initial solution: extend SolutionAdapter
- Construct neighborhood: extend MoveManager
  - define the neighbors as a combination of current solution and move
  - Extend Move to define the specific move
- Update memory structures
  - In Move define the method hashCode()
    - It returns an integer, which is used to compare moves

Example: Tabu Search

- Additional dynamic behaviours: TabuSearchAdapter
  - All the methods take a ts object of type TabuSearchEvent
    - Used to have a hook to the actual algorithm

```
TabuSearch theTS = (TabuSearch)evt.getSource();
Solution   best  = theTS.getBestSolution();
SimpleTabuList mytl = (SimpleTabuList)theTS.getTabuList();
```

  - Methods
    - newBestSolutionFound: in the last iteration, the new Current becomes the Best solution
    - unimprovingMoveMade: in the last iteration, the new Current is not better than the Best solution
    - newCurrentSolutionFound: hook to the end of each iteration (before checking if better than the Best
    - tabuSearchStarted: start of the TS (useful in multithread)
    - tabuSearchStopped: end of the TS (useful in multithread)