

Projects - Metaheuristic Algorithms

2017-2018

I. Application oriented topics

1. **Cloud Resource Provisioning.** It refers to the provisioning of computational resources (e.g. virtual machines) for a specific task such that some quality of service requirements (storage, cost, energy consumption etc) are satisfied. The problem can be addressed from the point of view of the provider (efficiently allocate virtual machines on physical resources) and from the point of view of the user (select the best services). In both cases it can lead to constrained, multi-objective and/or dynamic optimization problems for which various metaheuristics have been proposed lately. To test various strategies one can use CloudSim (<https://github.com/Cloudslab/cloudsim>) which is a cloud simulation framework which allows to simulate data centers, virtual machines and various policies for resource allocation and energy consumption control.

Requirement: Select a problem and a method, implement the method and test it using CloudSim.

Variants: 4 (EA, ACO, PSO, other metaheuristics)

Biblio: applications/CloudProvisioning

CloudSim: <https://github.com/Cloudslab/cloudsim>

2. **VRP - Vehicle Routing Problem.** It is a combinatorial optimization problem which is a generalization of TSP. It appears in different real world situations and can be solved by using evolutionary algorithms, ant-based methods or other meta-heuristics. There are different variants of the problem (see for instance <http://neo.lcc.uma.es/vrp/vrp-flavors/>).

Requirement: Select a problem instance, describe an adequate method, implement and test it for one of the instances available at <http://neo.lcc.uma.es/vrp/>

Variants: 4 (EA, TS, PSO, VNS)

Biblio: applications/VehicleRouting

3. **Nurse rostering.** This is a practical problem asking to find a schedule of nurses in a hospital. It is in fact a constraint satisfaction problem which can be solved by using different meta-heuristics (evolutionary algorithms (EA), estimation of distribution algorithms (EDA), ant colony optimization (ACO), variable neighborhood search (VNS), tabu search (TS), hyperheuristics etc).

Requirement: Select a problem variant, identify and describe an adequate method (based on EA, EDA, ACO, VNS or TS), implement and test it.

Variants: 5 (EA, EDA, ACO, VNS, TS)

Biblio: applications/nurse_rostering

4. **Timetabling.** It is also a constraint satisfaction problem characterized by the existence of hard constraints (they should be satisfied) and soft constraints (they can be broken but the score of the timetabling becomes smaller). There are approaches based on simulated annealing (SA), evolutionary algorithms (EA), ant colony optimization (ACO), tabu search (TS), variable neighbourhood search (VNS), particle swarm optimization (PSO)

Requirement: Select a problem variant, identify and describe an adequate method (based on SA, EA, ACO, TS, VNS or PSO), implement a variant and test it.

Variants: 6 (SA, EA, ACO, TS, VNS or PSO)

Biblio: applications/timetabling

5. **Traffic Lights Optimization.** Controlling/optimization of traffic lights means establishing the time slot allocated to each possible state of a traffic light (e.g. red, green) in such a way that some criteria (e.g. the number of vehicles arriving at the destination) are optimized. When using metaheuristics to solve such problems, a candidate solution is encoded as a sequence of values corresponding to time duration in seconds of each state for all traffic lights included in the model. The evaluation of a candidate solution is based on the traffic simulation for a given amount of time.

Requirement: analysis of a metaheuristic approach for traffic lights optimization (see papers from the biblio collection), implementation of one variant and test on a simple scenario (e.g. 1-2 intersections in Timi'soara) using an existing traffic simulator (e.g. SUMO - http://sumo.dlr.de/wiki/Main_Page).

Biblio: applications/TrafficLightsOptimization

<http://neo.lcc.uma.es/problems/traffic-lights/index.html>

http://sumo.dlr.de/wiki/Main_Page

6. **Regular Expressions Inference.** The regular expressions are useful to extract information from text. Automatic synthesis of regular expressions starting from examples can be done by genetic programming.

Requirement: Analyze the implementation(s) available at <https://github.com/MaLeLabTs/RegexGenerator> and

<http://regex.inginf.units.it>, adapt and test them for generating regular expressions which allow to extract text fragments (a.g. numerical values in a text).

Biblio: applications/RegularExpressionsInference

7. **Portfolio optimization.** Portfolio optimization means to find financial or production portfolios which maximize the profit and minimize the risk. The problem can be formulated as a constrained combinatorial optimization problem which can be solved using various metaheuristics (e.g. GA, GP, PSO, Hopfield Neural Networks - HNN)

Requirement: Identify and describe an adequate method (based on GA, GP, PSO, HNN), implement a variant and test it.

Variants: 4 (GA, GP, PSO, HNN)

Biblio: applications/portfolio_optimization

8. **Community detection in networks.** The aim is to identify the groups of highly interconnected nodes in a network. The problem is similar to that of graph partitioning such that the

nodes in a partition are highly connected while the nodes in different partitions are sparsely connected. It can be solved by using clustering techniques but also evolutionary algorithms.

Requirement: Study of the community detection problem, implementation of an EA and its testing for a test problem (e.g. Zachary's club problem).

Biblio: applications/communities_detection

- 9. Intrusion detection systems.** Intrusion detection systems (IDS) are mainly based on classifying the behaviour of the user of a computing system in normal and abnormal, based on data related to the access of the user to the computing resource. IDSs usually monitor all events taking place in the computing system. Currently there are various approaches based on computational intelligence (CI) techniques: neural networks (NN), evolutionary algorithms (EA) and artificial immune systems (AIS). Current malware detection products (e.g. Bitdefender) incorporate various machine learning algorithms (e.g. perceptron-like classifiers) or metaheuristics of relevant attribute selection.

Requirement: Analyze the structure of a IDS and implement at least one component based on a CI technique: NN, EA or AIS.

Variants: 3 (NN, EA, AIS)

Biblio: applications/intrusion_detection_systems

- 10. Software testing.** The evolutionary algorithms and other nature inspire metaheuristics (e.g. ACO) can be used to generate test cases for software analysis.

Requirement: Study of the applicability of evolutionary algorithm in software testing. Choose an evolutionary algorithm to generate test cases for a software product and implement it.

Biblio. applications/software_testing

- 11. Automated Program Repairing.** Automatic generation of patches used to remove bugs from software or to make an application more efficient can be done using genetic programming. The patches are usually small pieces of code which are generated starting from some templates and are evaluated using the behavior of the program for some testcases.

Requirement: Comparative study of different approaches in automated patches generation. A Genetic Programming based example should be implemented.

Biblio. applications/AutomatedProgramRepair

- 12. Parameter tuning.** The behaviour of different software tools may depend on their control parameters and finding the appropriate parameter values can lead to difficult combinatorial optimization problems for which metaheuristics could be useful. Such cases may happen in search based software engineering, in tuning parameters of image processing tools (e.g. for registration, segmentation) or tuning the parameter of various physical models (e.g. weather models).

Requirement: Select a problem and a metaheuristic and implement a parameter tuning example.

Biblio. applications/ParameterTuning

- 13. Multilevel thresholding.** Image segmentation means identifying regions in the image in such a way that the pixels belonging to the same region are similar while those belonging

to different regions are dissimilar. For gray level images, one of the simplest segmentation technique relies on threshold(s). The threshold(s) estimation can be formulated as an optimization problem, e.g. find the threshold values which maximize the inter-regions variance.

Requirement: Implementing a metaheuristic algorithm (starting from one of the variants described in the papers provided in the bibliography) to estimate multiple thresholds + testing it on some test images (<http://sipi.usc.edu/database/>, http://www.imageprocessingplace.com/root_files_V3/image_databases.htm)

Biblio: applications/ImageThresholding

14. **Image registration.** The aim of image registration is to align images of the same scene/object taken from different viewpoints or using different devices. The task to be solved is to transform a source image such that it matches a target image. If some corresponding points in the two images are known then by using them a geometric transformation can be identified. The evolutionary algorithms can be used to estimate the parameters of transformation models.

Requirement: Choice and implementation of a metaheuristic (e.g. differential evolution, evolutionary algorithm or particle swarm optimization) for the estimation of the parameters of a transformation ensuring the registration of two medical or satellite images.

Biblio: applications/ImageRegistration

15. **Evolutionary Games** Evolutionary algorithms can be used to evolve strategies for various games (e.g. free cell, chess, Rubik cube etc). The main component of the approach is the choice of an appropriate encoding and variation operators.

Requirement: Implementation of an evolutionary approach for a game at your choice.

Biblio: applications/EvolutionaryGames

16. **Data analysis and prediction.** Lately, in the context of major conferences in evolutionary computing (e.g. GECCO, CEC) have been organized competitions on industrial challenges, most of them addressing prediction problems.

Requirement: Implementing a solution to one of the problems from:

- GECCO 2013 (<http://www.spotseven.de/gecco-challenge/gecco-challenge-2013/>)
- GECCO 2014 (<http://www.spotseven.de/gecco-challenge/gecco-challenge-2014/>)
- GECCO 2015 (<http://www.spotseven.de/gecco-challenge/gecco-challenge-2015/>)

II. Topics oriented towards techniques

1. **BBO - Biogeography Based Optimization.** This is a recent metaheuristic inspired by the properties of the geographical distribution of biological organisms. It uses a particular crossover operator inspired by the emigration/immigration processes.

Requirement. Study of BBO, implementation of a variant and its application for a simple optimization problem.

Biblio. techniques/BBO

2. **Cuckoo Search.** This technique is inspired by the cuckoo behaviour (the use the nests of other species to place their eggs). This idea is modelled by particular mutations involving Levy distributions.

Requirement. Study of the state-of-the-art of cuckoo search, software implementation and testing for some optimization problems. *Biblio.* techniques/CuckooSearch

3. **Firefly algorithms.** This technique is inspired by the fireflies "lightening" behavior. The biological inspiration is modelled through attraction/repelling mechanisms which control the search process.

Requirement. Study of the state-of-the-art of fireflies algorithm, software implementation and testing for some optimization problems.

Biblio. techniques/FireflyAlgorithm

4. **Bat algorithms.** This metaheuristic is inspired by the echo-location mechanism used by bats in the orientation process. The idea is modelled by combining a local search with a random search.

Requirement. Study of the state-of-the-art of bat algorithm, software implementation and testing for some optimization problems.

Biblio. techniques/BatAlgorithm

5. **VNS - Variable Neighborhood Search** It is a search technique involving selection of perturbed configurations inside a neighborhood of the current configuration.

Requirement: Study of VNS methods, implementation of a variant and application for an optimization problem.

Biblio. techniques/VNS

6. **Cooperative Coevolution.** Coevolution means the simultaneous evolution of several populations. Cooperative coevolution provides a general framework for solving complex problems by dividing them in simpler problems. This strategy can be combined with any type of evolutionary algorithms and has been successfully applied for high-dimensional optimization problems.

Requirement. Study of cooperative coevolution, implementation of a coevolutionary algorithms and its application for a optimization problem with a large number of variables (at least 100).

Biblio. techniques/coevolution

7. **Compact Evolutionary Algorithms.** They are EAs designed under the assumption that there is a limited amount of resources available for their execution (e.g. they should be executed on mobile devices). In order to deal with this problem they do not use explicit populations but only descriptions of the probability distributions which models the elements of the populations.

Requirement. Study of compact EAs, implementation of such an algorithm and its application for a simple optimization problem.

Biblio. techniques/compactEA

8. **EDA - Estimation of Distribution Algorithms.** They are stochastic optimization techniques that uses probabilistic models for exploring the search spaces (they are somewhat similar to compact EAs). *Requirement:* Study of EDA algorithms, implementation of a variant and testing for a simple optimization problem (e.g. Onemax).

Biblio. techniques/EDA

9. **Hyperheuristics.** They are heuristics which "chooses heuristics" in order to design a method for solving a problem. The selection of heuristics can be random, greedy, based on some learning processes or even based on genetic algorithms. They have successfully applied for combinatorial optimization problems.

Requirement: Study of hyperheuristics, implementation of a variant and testing it for a simple optimization problem.

Biblio. techniques/hyperheuristics

10. **Evolutionary Deep Neural Networks.** Neuroevolution refers to the evolutionary design of neural networks architectures. NEAT (Neuro Evolutionary Augmenting Topologies) is a strategy for evolving and training neural architectures of various complexities. There are implementations in C++ (<http://nn.cs.utexas.edu/?neat-c>), Python (<https://github.com/CodeReclaimers/nea-python>), Java (<http://nn.cs.utexas.edu/?jneat>). Recently the method has been applied for evolving deep architectures.

Requirement. Analyze the characteristics of various NEAT implementations and use them to solve a classification problem (e.g. CIFAR-10).

Biblio. techniques/EvolutionaryDeepNN

11. **GPU implementation of EAs.** Implementation of EAs on GPU became popular in the last years. The main issue is related to the choice of operation to be done on GPU such that the transfer between CPU and GPU is minimized.

Requirement: GPU implementation of a simple GPU.

Biblio. applications/EA+GPU

III. Comparative studies of tools based on metaheuristics.

Examples of tools which can be compared:

Metaheuristics:

1. **ECJ.** <http://cs.gmu.edu/~eclab/projects/ecj/>
2. **JGap.** <http://jgap.sourceforge.net/>
3. **JavaEVA.** <http://www.ra.cs.uni-tuebingen.de/software/EvA2/>
4. **JCLEC.** <http://jclec.sourceforge.net/>
5. **JaGA.** <http://www.jaga.org/>
6. **JMetal.** <http://jmetal.sourceforge.net/>
7. **EO.** <http://eodev.sourceforge.net/>
8. **ParadisEO.** <http://paradisEO.gforge.inria.fr/index.php?n=Main.HomePage>
9. **PISA.** <http://www.tik.ee.ethz.ch/sop/pisa/>
10. **OpenBeagle.** <https://github.com/chgagne/beagle>
11. **Robust Genetic Programming.** <http://robgp.sourceforge.net/about.php>
12. **DEAP - Distributed Evolutionary Algorithms in Python** <http://code.google.com/p/deap/>

13. **pySTEP - Python Strongly Typed gEnetic Programming** <http://pystep.sourceforge.net/>
14. **PerlGP** - Perl Genetic Programming System <http://perlgp.org/>
15. **MetsLib** - metaheuristic modeling framework and optimization toolkit in modern C++
<https://projects.coin-or.org/metslib>
16. **jmp75** - a simplified metaheuristics framework for .NET. <https://github.com/jmp75/metaheuristics>
17. **HEO** - cross-platform library that provides various parallel metaheuristic algorithms for hard optimization problems. <https://code.google.com/p/heo/>
18. **OPT4J** - open source Java-based framework for evolutionary computation.
<http://opt4j.sourceforge.net/>
19. **HeuristicLab** - a framework for heuristic and evolutionary algorithms.
<http://dev.heuristiclab.com/>

Requirement. At least two tools should be selected, described and tested for a simple problem. The project will contain a comparative study of the selected tools. Teams of 2-3 students can be involved in such a projects.

Structure of the project. Each project will consist of:

1. A report (6-12 pages) structured as follows:
 - abstract (5 lines): will describe the aim of the report and the main result
 - introduction: will contain a state of the art concerning the studied method/application; it is based on the study of the papers specified as references
 - detailed presentation of the models/algorithms/problems related to the subject
 - detailed presentation of the application + results
 - conclusions and open problems
 - references
2. A functional application illustrating the behaviour of the method(s) described in the report or using the software tools selected for comparison.

Remarks.

- The papers suggested for references can be downloaded from
<http://www.info.uvt.ro/~dzaharie/ma2017/projects>

These papers should be used to start the study - other references could be used as well. The report should be based on at least two references from the provided list.

- The report should not contain paragraphs directly copied from other papers. The existence of such paragraphs is considered to be plagiarism and such a report cannot be accepted. In order to avoid plagiarism you should synthesize the main ideas using you own words.