1. Study one of the following algorithms (theoretical and software implementation):
   - SEMO (Simple evolutionary multi-objective optimizer) [1]
   - FEMO (Fair evolutionary multi-objective optimizer) [1]
   - NSGA-II (Non-dominated Sorting Genetic Algorithm) [2]
   - SPEA2 (Improving the strength Pareto evolutionary algorithm) [3]
   - PAES (Pareto Archived Evolution Strategy)
   - PESA-II (Pareto Envelope-Based Selection Algorithm) [4]
   - OMOPSO (Optimal multi-objective optimization based on Particle Swarm Optimization - PSO) [5][6]
   - MOCell (A cellular genetic algorithm for multi-objective optimization) [7]
   - AbYSS (Adapting Scatter Search for Multi-objective Optimization) [8]
   - MOEA/D (Multi-objective Evolutionary Algorithm Based on Decomposition) [9]
   - CellDE (Dendritic Cell Algorithm) [10]
   - FastPGA (Fast Pareto Genetic Algorithm) [12]
   - IBEA (Indicator-based Evolutionary Algorithm) [13]
   - SMPSO (Speed-constrained Multi-objective PSO) [14].

2. Study two of the following Quality Indicators useful in multi-objective optimization techniques:
   - “7 point” average distance [15]
   - hypervolume[15]
   - coverage[15]
   - error ratio[15]
   - spread
   - generational distance[16][17]
   - inverted generational distance[5]
   - Additive epsilon.
   - Do they require the real Pareto front? If yes, there are alternatives like the Pareto front does not require actual knowledge?¹ Can you find other metrics that do not require actual knowledge of the Pareto front? [15]?

3. Please present the differences between:
   a. Genetic Algorithms and Differential Evolution [18]
   b. Genetic Algorithms and Cellular Genetic Algorithms

4. Analyzed comparatively by two operators in each class of operators. Present the advantages / disadvantages: Specify the types of representations of chromosomes (solution type in jMetal) that these operators may apply.
   Class operators:
   1. Crossover: HUX crossover, PMX crossover, SBX crossover

¹ Daca da, exista variante asemănătoare care sa nu necesite cunoasterea frontului Pareto real?
2. Mutation: BitFlip mutation, Nonuniform mutation, Polynomial mutation, Swap mutation, Uniform mutation

3. Selection: Best Solution Selection, Binary tournament, PESA2 Selection, Ranking and Crowding Selection, Worst Solution Selection

All operators are implemented in jMetal (http://jmetal.sourceforge.net/index.html)


6. Evolutionary magic square solver: Given a 10x10 grid with a small 3x3 square in it, you must arrange the numbers 1-100 on the grid such that all horizontal, vertical, diagonal sums are equal (505), and have a small 3x3 square forms a solution for 1-9 [19, 20]. Analyze the problem using small mutation rate vs. larger mutation rate. The solution must allow a mutation step size modified on-line. Problem description, applicability and software implementation. Conclusions and Results interpretation.

7. (a) Solve the 8-queens problem\(^2\) through backtracking. Measure the time required for obtaining the first (and all solutions).
(b) Solve the 8-queens problem through Evolutionary Algorithm provided and make the same measures. How big is the phenotype space for the eight queens’ problem? How big is the genotype space?
(c) Extend the problem with two (or three) types of pieces on the board (queens & horses).

8. (a) Solve the 0-1 Knapsack problem through greedy programming technique. Measure the time required for obtaining the first solution.
(b) Solve the 0-1 knapsack problem through Evolutionary Algorithm provided and make the same measures. [21]

   b. User guide
   c. Examples of optimization models

10. (a) Solve the Travelling Salesman Problem through greedy programming technique. Measure the time required for obtaining the first solution. “Given a list of cities and their pair wise distances, the task is to find a shortest possible tour that visits each city exactly once.”
(b) Solve the Travelling Salesman Problem using Genetic Algorithms and make the same measures.

\(^2\) For problems 7 (8-queens) and 10 (Traveling Salesman) the program should be parameterized and flexible allowing to select for Mutation operator between (Insert Mutation, Swap mutation, Inversion mutation, Scramble mutation) and to select for Crossover operators between (Order 1 crossover, Partially matched crossover (PMX), Cycle crossover).
11. (a) Realize a small tutorial related to installing and using \textit{jMetal} under Linux operating system, using NetBeans or Eclipse IDE. Prepare a more detailed user guide. 
(b) Solve the \textbf{University timetable Scheduling} and \textbf{Job Shop Problems} using Genetic Algorithms and present some comparisons when you are using different parameters (number of generation, crossover probabilities, mutation probabilities, etc). Solve the problem keeping in mind the constraints: \textit{students may wish to have no more than two classes in a row, while their lecturers may be more concerned with having whole days free for conducting research. Meanwhile, the main goal of the university management might be to make room utilization more efficient, or to cut down the amount of movement around or between the buildings.}

12. Implementing an \textbf{ant-colony based algorithm} for solving a problem of exploring a graph (e.g. \textbf{Travelling Salesman Problem}). Discuss the methodology of the algorithm; Theoretical and software implementation. 
“Behaviour of insects that live in the community (ants and bees) are the source of inspiration for the development of optimization methods (in order to solve difficult combinatorial optimization problems such as the \textbf{Travelling Salesman Problem, planning activities, etc}) and distributed control (e.g. adaptive routing in communication systems). The ability of these communities to solve problems is based on chemical communication via pheromones: on the route map to search for food, ants leave "trace" for identifying the route that led to finding food. The artificial systems implement pheromones action by "rewarding" best configurations and evaporation of pheromones by "forgetting / abandoning" some configurations.

13. Implementing a PSO (Particle Swarm Optimization) algorithm for solving continuous / or discrete optimization problems (ex. \textit{Multi-Objective Optimizations for a Superscalar Architecture with Selective Value Prediction, PSATSim}). 
"Particle Swarm Optimization - PSO" technique of is inspired by social behaviour (particularly related to the transmission and sharing of information) of living beings such as flocks of birds, swarms of bees or schools of fish. The artificial search process is provided by a set of "particles" whose movement is characterized by a "speed" that change over time depending on the characteristics of the entire system. These algorithms enable quick finding optimum but have difficulty in avoiding local minima. Population is called \textit{swarm}. Individuals are called particles. These particles fly through the search space following the best particle at that time. The position of a particle is given by the current values of its parameters. Each particle tries to approach the best particles. To achieve this, its parameters are modified. In order to change the position, it is considering both the global best particle (leader) of the moment and the best position he had in its history (personal best). After this change, the particle will have a new position and should be evaluated again. After all particles have been evaluated, a new leader is selected, each particle's personal best is updated and the process is restarted. In multi-objective algorithms can be more leaders at some point of time.

14. Implementation of a \textbf{genetic algorithm} to solve the problem of \textbf{Register Allocation: a discrete optimization problem} (mapping a large number of variables (let say \textit{n}) to small number of registers of the processor (let say \textit{k} \leq 32)). Hint: This application is similar with Graph Coloring Problem. Thus, a program has a set \textit{V} of \textit{n} variables and only a small number \textit{k} of registers is available on a CPU. Two variables not used at the same time can use the same register. Two variables used at the same time are conflicting variables. If the same register is used for both, the program speed is penalized and additional RAM / Cache access operations are needed. Objective: assign \textit{k} registers to the variables so as to minimize the number of conflicting variables using the same register! It is a NP-Hard problem.
15. (a) Realize a small tutorial related to Genetic Programming (representation, operators, example of applications solved, etc). Presents some differences face to Genetic Algorithms.

(b) Implementation of a genetic algorithm to solve the problem [26]: designing a tin\(^3\) with the following restriction: Consider a cylindrical tin, with only two parameters: diameter \(d\) and height \(h\) (obviously, can be considered and other parameters such as thickness, material properties, shape, but only two parameters are sufficient to illustrate working with genetic algorithms). Let’s consider that the tin’s volume must be at least 300 ml and our goal is to minimize the material used to produce the tin. In other words, it must be minimized the value of objective function:

\[
f(d, h) = c \left( \frac{\pi d^2}{2} + \pi dh \right)
\]

where \(c\) — represents the tin material cost per square centimeter (\(\text{cm}^2\)) and the expression from bracket represents the tin’s total area (lateral and 2 \(\times\) basis area). The restriction related to volume (\(\geq 300\) ml) it is stated with the help of \(g\) function (the cylinder volume formula):

\[
g(d, h) = \frac{\pi d^2 h}{4} \geq 300.
\]

The parameters \(d\) and \(h\) may vary in some well define ranges:

\[
d_{\text{min}} \leq d \leq d_{\text{max}} \quad \text{and} \quad h_{\text{min}} \leq h \leq h_{\text{max}}
\]

16. (a) Realize a small tutorial in which you describe comparative the following search optimization techniques: Hill-Climbing, Simulated Annealing, Tabu Search and Branch & Bound implementing on a specific problem.

(b) Implementing the Branch & Bound algorithm for solving a problem of exploring a graph (e.g. Travelling Salesman Problem).

17. (a) Realize a small tutorial related to Evolutionary Programming (representation, operators, example of applications – solved, etc). Presents some differences face to Genetic Algorithms.

(b) Solve the Vehicle Routing Problem with Tabu Search Heuristics [27] and Genetic Algorithms (“Vehicle Routing Optimization Problem with Time-windows and its Solution by Genetic Algorithm”).

18. Developing an Automatic Design Space Exploration for a reduced superscalar architecture (see the PSATSim simulator dedicated to PowerPC architecture), in order to analyze from multi-objectiv point of view (performance, energy consumption) using non-Pareto optimization techniques (weighted-sum method, VEGA algorithm, lexicographical order).

19. Implementing an Automatic Design Space Exploration for a reduced superscalar architecture (see the PSATSim simulator dedicated to PowerPC architecture), in order to analyze from multi-objectiv point of view (performance, energy consumption) using Pareto optimization techniques: NSGA-II (Non-dominated Sorting Genetic Algorithm) [2], SPEA2 (Improving the strength Pareto evolutionary algorithm) [3].

---

\(^3\) Cutie de conserve
20. Developing an Automatic Design Space Exploration for a reduced superscalar architecture (see the PSATSim simulator dedicated to PowerPC architecture), in order to analyze from multi-objective point of view (performance, energy consumption) using bio-inspired optimization techniques: SMOPSO (Simple Multi-Objective Particle Swarm Optimizer) [29], SMPSO (Speed-constrained Multi-objective PSO) [14].

21. Multi-Objective Optimization in Gait Planning of Biped Robot Using Genetic Algorithm and Particle Swarm Optimization Algorithm. Describe the representation model of the genetic algorithms' solution; apply mono-/ multiobjective optimization algorithm and develop a C# application for integrating these.

22. Fuzzy control of mechanical vibrating systems. Describe the representation model of the genetic algorithms' solution; apply mono-/ multiobjective optimization algorithm and develop a C# application for integrating these.

23. Design of Optimal Linear Suspension for Quarter Car with Human Model using Genetic Algorithms (4DOF). Describe the representation model of the genetic algorithms' solution; apply mono-/ multiobjective optimization algorithm and develop a C# application for integrating these.

24. Multi-objective Suspension Optimization of a 5-DOF Vehicle Vibration Model Excited by Random Road Profile. Describe the representation model of the genetic algorithms' solution; apply mono-/ multiobjective optimization algorithm and develop a C# application for integrating these.

25. Design Optimization of Quarter-car Models with Passive and Semi-active Suspensions under Random Road Excitation. Describe the representation model of the genetic algorithms' solution; apply mono-/ multiobjective optimization algorithm and develop a C# application for integrating these.

26. Spare Parts Inventory Model for Auto Mobile Sector Using Genetic Algorithm. A Fast Genetic Algorithm (FGA) for Spare Part Inventory Management Problem. Describe the representation model of the genetic algorithms' solution; apply mono-/ multiobjective optimization algorithm and develop a C# application for integrating these.

27. An Effective Decision-Based Genetic Algorithm Approach to Multiobjective Portfolio Optimization Problem. Describe the representation model of the genetic algorithms' solution; apply mono-/ multiobjective optimization algorithm and develop a C# application for integrating these.

28. Optimizing Inventory Using Genetic Algorithm for Efficient Supply Chain Management. Application of Genetic Algorithms and Visual Simulation in a Real-Case Production Optimization. Using genetic algorithms and simulation as decision support in marketing strategies and long-term production planning. Describe the representation model of the genetic algorithms' solution; apply mono-/ multiobjective optimization algorithm and develop a C# application for integrating these.


32. a) Hybrid Evolutionary Multi-Objective Optimization of Machining Parameters. b) Multi-Objective Optimization of Machining Parameters During Dry Turning of AISI 304 Austenitic Stainless Steel Using Grey Relational Analysis. c) Multi-Objective Optimization of Turning Process during Machining of AlMg1SiCu Using Non-Dominated Sorted Genetic Algorithm. Describe the representation model of the genetic algorithms' solution (Pareto techniques); apply mono- / multiobjective optimization algorithm and develop a C# application for integrating these.

33. a) Optimization of the characteristic parameters in milling using the PSO evaluation technique. b) Overview of PSO for Optimizing Process Parameters of Machining c) Cutting Parameters Optimization by Using Particle Swarm Optimization (PSO). Describe the representation model of the genetic algorithms' solution (swarm-based techniques); apply mono- / multiobjective optimization algorithm and develop a C# application for integrating these.

34. a) Optimization in Strategy Games: Using Genetic Algorithms to Optimize City Development in FreeCiv. b) Generating emergent team strategies in football simulation videogames via genetic algorithms. c) Automatic Video Game Level Generation

35. a) Multicriteria Inventory ABC Classification in an Automobile Rubber Components Manufacturing Industry; b) Multicriteria ABC inventory classification using acceptability analysis Describe the representation model of solution, optimization / classification algorithm and develop a C# application for integrating these.

36. a) Simulating annealing (SA) algorithm to solve the Green Vehicle Routing problem. Limitations. b) Multi-objective optimization model for a Green Vehicle Routing problem. Describe the representation model of the (SA + genetic algorithms’ solution; apply mono- / multiobjective optimization algorithm and develop a C# application for integrating these.

37. Metaoptimization (NSGA-II + SPEA / SMPSO) for multi-objective optimization of Green Vehicle Routing problem. Describe the representation model of the solution; apply mono- / multiobjective optimization algorithm and develop a C# application for integrating these.


