Artificial Intelligence and Knowledge Engineering Laboratory

Task 1. Genetic Algorithms (*final version*)

Authors: M. Paradowski, M.Przewoźniczek, M. Piasecki

Deadline: 4th laboratory classes

Task Objectives

Getting familiar with genetic algorithms metaheuristics in practice through individual implementation.

Subtasks

- Get familiar with genetic algorithms metaheuristics
- Choose one of the problems to solve:

Problem 1:
Finding the shortest path in a graph with weights assigned to arcs. The graph must defined in a file as a parameter to the task. We assume that if there is at most one arc between any two nodes. The graph is connected, i.e. there is a path between any two nodes. Any node can be visited no more than two times.

Problem 2
Finding the best covering of a piece of material by rectangular elements to be cut off it. The size of the material sheet, as well as the number of elements and their sizes must be read from a file. The maximal number of shapes is given in such a way that the sum of all the element areas is larger than the area of the material.

Problem 3
Finding a way in a labyrinth. The labyrinth is defined in an input file. It is described by a gird of square blocks, such that some of them are marked as walls, the rest are empty spaces.

- Build genetic algorithm. Define: individual (binary coding must be used), fitness function, crossover operator, mutation operator and selection operator
- Implement genetic algorithm
- Create user interface for visualisation of the best solutions found by the algorithm: in each population or the final ones.
- Test the influence of:
    - different parameters on genetic algorithm efficiency and effectiveness (eg. probability of mutation, crossover, selection operator, population size, generations number etc.)
    - and the problem sizes.
- Create report in which shortly explain all your decisions and provide analysis of the obtained results, your observations and findings.

- Create graphs presenting the changes of fitness function value during algorithm run (fitness value of best individual, population average, standard deviation or the worst individual).
- Compare your genetic algorithm to other, non-evolutionary optimization method (choose one: brute force, gradient method, non-gradient method, random search, hill climbing). Note the result quality, execution time, number of fitness function evaluations.
- Present most interesting results
- Discuss the results
- The report should contain all the above points

Additional information

Genetic algorithm is a method which imitates the natural evolution process through selection pressure and natural selection. In order to use the GA, the potential solution (individual), the way it changes (mutation), mating (crossover) and fitness function must be defined. The work schema of genetic algorithm might be presented as follows:

```
begin
t:=0;
initialise(pop_0);
evaluate(pop_0);
while (not stop_condition) do
begin
pop_{t+1} := selection(pop_t);
pop_{t+1} := crossover(pop_{t+1});
pop_{t+1} := mutation(pop_{t+1});
evaluate(pop_{t+1});
t:=t+1;
end
end
```

At the beginning the method initializes (usually at random) the solutions (individuals) population. Next all individuals are evaluated (for every individual the fitness function value is computed). Next the stop condition is checked (it might be based on best found solution quality, computation time, number of fitness function evaluations etc.). The individuals for the next population are chosen (with the use of roulette wheel method or tournament method) before crossover and mutation. Next all individuals are rated and the next iteration is started by check of stop criteria.

Task rating

2 points – building genetic algorithm model
2 points – genetic algorithm implementation
2 points – creation of proper UI for the visualisation of the selected results
2 points – testing the influence of crossover and mutation on the results outputted by method
2 points – testing the influence of the population size on the results outputted by method

Attention:

1. While testing it is necessary to take into account the results of at least 10 independent runs and averaging the results
2. For doing the exercise only for easy 1-D function gives only 50% of points for each subtask

Bibliography

1. Mitchell M. An Introduction to Genetic Algorithms. The MIT Press, 1998 (accessible via Google Books, almost complete)
https://books.google.pl/books?id=0eznlz0TF-IC&lpg=PP1&dq=mitchell%20evolutionary%20algorithms&hl=pl&pg=PP1#v=onepage&q&f=false

2. Goldberg D *Genetic algorithms in search, optimization and machine learning*

3. Michalewicz Z. *Genetic Algorithms* + *Data* Structures = *Evolution Programs*
In the library or partially accessible at
https://books.google.pl/books?id=FfXrCAAAQBAJ&lpg=PR2&dq=genetic%20algorithms&hl=pl&pg=PR2#v=onepage&q=genetic%20algorithms&f=false

4. Michalewicz Z., Fogel D.B. *How to Solve It: Modern Heuristics*

5. Man K.F., Tang K.S and Kwong S. Genetic Algorithms. Concepts and Designs Springer, 1999 (*Accessible from the WUS&T university network:*
http://link.springer.com/book/10.1007/978-1-4471-0577-0)