Report in pdf

Times New Roman 14pt, 1.0 spacing

Send to: algorithms_itmo@mail.ru

Email topic: *Name and surname, Academic group, Task# X*

A separate email for each task!

Use any programming language you want. You can use existing implementations. The numerical results and plots should be informative and correct.

Title page:

FEDERAL STATE AUTONOMOUS EDUCATIONAL INSTITUTION

OF HIGHER EDUCATION

ITMO UNIVERSITY

Report

on the practical task No. X

“Task title”

Performed by

Name and surname

Academic group

Accepted by

Dr Petr Chunaev

St. Petersburg

2020

Goal

Indicate the goal of your work

Formulation of the problem

Formulate the problem within the task

Brief theoretical part

Provide with brief theoretical information related to the task, for example, definitions, descriptions of algorithms, methodological approaches to solving the assigned tasks (at most 2 pages)

Results

Present the results of solving the assigned problems, including graphs and tables, as well as a brief discussion of the results obtained (at most 4 pages)

Conclusions

Make conclusions on the results obtained and on the achievement of the goal of your work

Appendix

Present the listings of programs written for performing the task, with comments

Task 1. Experimental time complexity analysis

Goal

Experimental study of the time complexity of different algorithms

Problems and methods

For each n from 1 to 2000, measure the average computer execution time (using timestamps) of programs implementing the algorithms and functions below for five runs. Plot the data obtained showing the average execution time as a function of n. Conduct the theoretical analysis of the time complexity of the algorithms in question and compare the empirical and theoretical time complexities.

I. *Generate an n-dimensional random vector* $\boldsymbol{v} = [v_1, v_2, \ldots, v_n]$ *with non-negative elements. For* $\boldsymbol{v}$*, implement the following calculations and algorithms:*

1) $f(\boldsymbol{v}) = const$ *(constant function);*
2) $f(\boldsymbol{v}) = \sum_{k=1}^{n} v_k$ *(the sum of elements);*
3) $f(\boldsymbol{v}) = \prod_{k=1}^{n} v_k$ *(the product of elements);*
4) *supposing that the elements of* $\boldsymbol{v}$ *are the coefficients of a polynomial* P *of degree* $n-1$*, calculate the value* $P(1.5)$ *by a direct calculation of* $P(x) = \sum_{k=1}^{n} v_k x^{k-1}$ *(i.e. evaluating each term one by one) and by Horner's method by representing the polynomial as* $P(x) = v_1 + x\big(v_2 + x(v_3 + \cdots)\big)$*;*
5) *Bubble Sort of the elements of* $\boldsymbol{v}$*;*
6) *Quick Sort of the elements of* $\boldsymbol{v}$*;*
7) *Timsort of the elements of* $\boldsymbol{v}$*.*

II. *Generate random matrices* A *and* B *of size* $n \times n$ *with non-negative elements. Find the usual matrix product for* A *and* B*.*

III. *Describe the data structures and design techniques used within the algorithms.*

Task 2. Algorithms for unconstrained nonlinear optimization. Direct methods

Goal

The use of direct methods (one-dimensional methods of exhaustive search, dichotomy, golden section search; multidimensional methods of exhaustive search, Gauss, Nelder-Mead) in the tasks of unconstrained nonlinear

Problems and methods

I. *Use the one-dimensional methods of exhaustive search, dichotomy and golden section search to find an approximate (with precision* $\varepsilon = 0.001$*) solution* $x: f(x) \to min$ *for the following functions and domains:*

1. $f(x) = x^3, x \in [0, 1]$;
2. $f(x) = |x - 0.2|, x \in [0, 1]$;
3. $f(x) = x\, sin\frac{1}{x}, x \in [0.01, 1]$.

Calculate the number of f*-calculations and the number of iterations performed in each method and analyze the results. Explain differences (if any) in the results obtained.*

II. *Generate random numbers* $\alpha \in (0,1)$ *and* $\beta \in (0,1)$. *Furthermore, generate the noisy data* $\{x_k, y_k\}$, *where* $k = 0, \dots, 100$, *according to the following rule:*

$$y_k = \alpha x_k + \beta + \delta_k, \quad x_k = \frac{k}{100},$$

where $\delta_k \sim N(0,1)$ *are values of a random variable with standard normal distribution. Approximate the data by the following linear and rational functions:*

1. $F(x, a, b) = ax + b$ *(linear approximant),*
2. $F(x, a, b) = \frac{a}{1+bx}$ *(rational approximant),*

by means of least squares through the numerical minimization (with precision $\varepsilon = 0.001$*) of the following function:*

$$D(a, b) = \sum_{k=0}^{100} (F(x_k, a, b) - y_k)^2.$$

To solve the minimization problem, use the methods of exhaustive search, Gauss and Nelder-Mead. If necessary, set the initial approximations and other parameters of the methods. Visualize the data and the approximants obtained in a plot ***separately for each type of approximant****. Analyze the results obtained (in terms of number of iterations, precision, number of function evaluations, etc.).*

Task 3. Algorithms for unconstrained nonlinear optimization. First- and second-order methods

Goal

The use of first- and second-order methods (Gradient Descent, Conjugate Gradient Descent, Newton's method and Levenberg-Marquardt algorithm) in the tasks of unconstrained nonlinear optimization

Problems and methods

Generate random numbers $\alpha \in (0,1)$ *and* $\beta \in (0,1)$. *Furthermore, generate the noisy data* $\{x_k, y_k\}$, *where* $k = 0, \dots, 100$, *according to the following rule:*

$$y_k = \alpha x_k + \beta + \delta_k, \quad x_k = \frac{k}{100},$$

where $\delta_k \sim N(0,1)$ *are values of a random variable with standard normal distribution. Approximate the data by the following linear and rational functions:*

1. $F(x, a, b) = ax + b$ *(linear approximant),*
2. $F(x, a, b) = \frac{a}{1+bx}$ *(rational approximant),*

by means of least squares through the numerical minimization (with precision $\varepsilon = 0.001$*) of the following function:*

$$D(a, b) = \sum_{k=0}^{100} (F(x_k, a, b) - y_k)^2.$$

To solve the minimization problem, use the methods of Gradient Descent, Conjugate Gradient Descent, Newton's method and Levenberg-Marquardt algorithm. If necessary, set the initial approximations and other parameters of the methods. Visualize the data and the approximants obtained ***separately for each type of approximant****. Analyze the results obtained (in terms of number of iterations, precision, number of function evaluations, etc.) and compare them with those from Task 2 for the same dataset.*

Task 4. Algorithms for unconstrained nonlinear optimization. Stochastic and metaheuristic algorithms

Goal

The use of stochastic and metaheuristic algorithms (Simulated Annealing, Differential Evolution, Particle Swarm Optimization) in the tasks of unconstrained nonlinear optimization and the experimental comparison of them with Nelder-Mead and Levenberg-Marquardt algorithms

Problems and methods

Generate the noisy data (x_k, y_k), *where* $k = 0, \dots, 1000$, *according to the rule:*

$$y_k = \begin{cases} -100 + \delta_k, & f(x_k) < -100, \\ f(x_k) + \delta_k, & -100 \le f(x_k) \le 100, \\ 100 + \delta_k, & f(x_k) > 100, \end{cases} \qquad x_k = \frac{3k}{1000},$$

$$f(x) = \frac{1}{x^2 - 3x + 2},$$

where $\delta_k \sim N(0,1)$ *are values of a random variable with standard normal distribution. Approximate the data by the rational function*

$$F(x, a, b, c, d) = \frac{ax + b}{x^2 + cx + d}$$

by means of least squares through the numerical minimization of the following function:

$$D(a, b, c, d) = \sum_{k=0}^{1000} (F(x_k, a, b, c, d) - y_k)^2.$$

To solve the minimization problem, use Nelder-Mead algorithm, Levenberg-Marquardt algorithm and ***at least two*** *of the methods among Simulated Annealing, Differential Evolution and Particle Swarm Optimization. If necessary, set the initial approximations and other parameters of the methods. Use* $\varepsilon = 0.001$ *as the precision; at most 1000 iterations are allowed. Visualize the data and the approximants obtained* ***in a single plot****. Analyze and compare the results obtained (in terms of number of iterations, precision, number of function evaluations, etc.).*

Task 5. Algorithms on graphs. Introduction to graphs and basic algorithms on graphs

Goal

The use of different representations of graphs and basic algorithms on graphs (Depth-first search and Breadth-first search)

Problems and methods

I. *Generate a random adjacency matrix for a simple undirected unweighted graph with 100 vertices and 200 edges (note that the matrix should be symmetric and contain only 0s and 1s as elements). Transfer the matrix into an adjacency list. Visualize the graph and print several rows of the adjacency matrix and the adjacency list. Which purposes is each representation more convenient for?*

II. *Use Depth-first search to find connected components of the graph and Breadth-first search to find a shortest path between two random vertices. Analyse the results obtained.*

III. *Describe the data structures and design techniques used within the algorithms.*

Task 6. Algorithms on graphs. Path search algorithms on weighted graphs

Goal

The use of path search algorithms on weighted graphs (Dijkstra's, A and Bellman-Ford algorithms)*

Problems and methods

I. *Generate a random adjacency matrix for a simple undirected weighted graph of 100 vertices and 500 edges with assigned random positive integer weights (note that the matrix should be symmetric and contain only 0s and weights as elements). Use Dijkstra's and Bellman-Ford algorithms to find shortest paths between a random starting vertex and other vertices. Measure the time required to find the paths for each algorithm. Repeat the experiment 10 times for the same starting vertex and calculate the average time required for the paths search of each algorithm. Analyse the results obtained.*

II. *Generate a 10x10 cell grid with 30 obstacle cells. Choose two random non-obstacle cells and find a shortest path between them using A* algorithm. Repeat the experiment 5 times with different random pair of cells. Analyse the results obtained.*

III. *Describe the data structures and design techniques used within the algorithms.*

Task 7. Algorithms on graphs. Tools for network analysis

Goal

The use of the network analysis software Gephi

Problems and methods

1. *Download and install Gephi from https://gephi.org/.*
2. *Choose a network dataset from https://snap.stanford.edu/data/ with number of nodes at most 10,000. You are free to choose the network nature and type (un/weighted, un/directed).*
3. *Change the format of the dataset for that accepted by Gephi (.csv, .xls, .edges, etc.), if necessary.*
4. *Upload and process the dataset in Gephi. Check if the parameters of import and data are correct.*
5. *Obtain a graph layout of two different types.*
6. *Calculate available network measures in Statistics provided by Gephi.*
7. *Analyze the results for the network chosen.*

While performing the work, screenshot the main steps you are doing and insert in the report.

Task 8 (Extended). Practical analysis of advanced algorithms

Goal

Practical analysis of advanced algorithms

Book: *Thomas H. Cormen Charles E. Leiserson Ronald L. Rivest Clifford Stein Introduction to Algorithms Third Edition, 2009 (or other editions).*

Sections:

- *I Foundations*
  - *4 Divide-and-Conquer*
  - *5 Probabilistic Analysis and Randomized Algorithms*
- *VI Graph Algorithms*
  - *23 Minimum Spanning Trees*
  - *25 All-Pairs Shortest Paths*
  - *26 Maximum Flow*
- *IV Advanced Design and Analysis Techniques*
  - *15 Dynamic Programming*
  - *16 Greedy Algorithms*
- *VII Selected Topics*

Task for the students non-experienced in algorithm analysis:

I. *Choose* **two** *algorithms (interesting to you and not considered in the course) from the above-mentioned book sections.*

II. *Analyse the chosen algorithms in terms of time and space complexity, design technique used, etc. Implement the algorithms and produce several experiments. Analyse the results.*

Task for the students well-experienced in algorithm analysis:

I. *Choose an algorithm (interesting to you and not considered in the course) from the above-mentioned book sections.*

II. *Choose an algorithm interesting to you and proposed at most 10 years ago in a research paper for solving a certain practical problem (including optimization algorithms, graph algorithms, etc.).*

III. *Analyse the chosen algorithms in terms of time and space complexity, design technique used, etc. Implement the algorithms (or use the existing ones from the research paper) and produce several experiments. Your experiments should differ of those in the research paper. Analyse the results.*

Further reading (the book):

II Sorting and Order Statistics

III Data Structures

V Advanced Data Structures

VII Selected Topics

34 NP-Completeness