COURSE DESCRIPTION CARD - SYLLABUS

Course name
Combinatorial Optimization

Course

Field of study	Year/Semester
Bioinformatics	2/3
Area of study (specialization)	Profile of study
	general academic
Level of study	Course offered in
First-cycle studies	Polish
Form of study	Requirements
full-time	compulsory

Number of hours

Lecture	Laboratory classes	Other (e.g. online)
30	15	
Tutorials	Projects/seminars	

Number of credit points

4

Lecturers

Responsible for the course/lecturer:
Maciej Drozdowski
email: Maciej.Drozdowski@cs.put.poznan.pl
tel: 616652981
Faculty of Computing and Telecommunications
Piotrowo 2, 60-965 Poznań

Responsible for the course/lecturer:

Prerequisites

A student beginning this subject of study should have basic understanding of discrete mathematics (set theory, logic, graph theory), methods of algorithm design, basic programming structures, abstract data types (e.g. lists, stacks, queues, arbitrary graphs), typical algorithms (e.g. sorting, search in data structures), also basic knowledge on the computational complexity of algorithms and problems.
The student should be able to design basic algorithms and code them, to recognize basic discrete structures, to estimate computational complexity of algorithms, as well as acquire information from the indicated sources.
The student should understand the necessity of expanding his/her competences and be ready to undertake cooperation in a team. As far as social competences are considered, the student must be honest, responsible, persevering, curious, creative, respectful to other people.

Course objective

Introduction into basic problems of combinatorial optimization and the methods of solving them. In

particular:
1. acquiring ground understanding on optimizing problems with discrete nature,
2. demonstrating solvability barrier arising from exponential computational complexity of algorithms and computational hardness of problems and to stimulate understanding consequences of this barrier,
3. developing a skill of recognizing hard combinatorial optimization problems,
4. familiarizing with the methodology of analyzing and practically solving of computationally hard optimization tasks for problems with discrete nature.

Course-related learning outcomes

Knowledge

1. ordered and theoretically grounded general knowledge on key issues of computer science, the issues of the current subject
2. knowledge on important directions and developments of computing, and related areas
3. knowing basic methods, techniques and tools applied in the process of solving combonatorial optimization problems mainly of engeenering type, solving simple cases of analyzing computational complexity of algorithms and combinatorial problems

Skills

1. designing and conducting simple experiments in combinatorial optimization, in particular computer measurements and simulations, analyzing the obtained results and drawing conclusions
2. apply analytical and experimental methods to solve combinatorial optimization problems
3. estimating computational complexity of algorithms and problems
4. designing and coding algorithms using at least one popular tool

Social competences

1. understanding that knowledge and skills in computer science quickly change and deprecate
2. understanding the meaning of knwoledge in solving engineering problems, knowing examples of engineering problems leading to social issues

Methods for verifying learning outcomes and assessment criteria

Learning outcomes presented above are verified as follows:
Formative assessment:
a) lectures:
- based on answers to question asked and open problems posed during the lectures,
b) labs:
- evaluation of the correctness of the programs solving the assigned combinatorial optimization problems
- evaluation of student’s knowledge necessary to prepare, and carry out the lab tasks
Total assessment:
a) lectures:
- based on answers to question in a written test,
b) labs:
- monitoring students activities during classes,

- evaluation of reports on the method and computer program solving the assigned combinatorial optimization problems
Additional elements cover:
- punctuality: additional points for providing solutions (programs) and reports on time
- efficiency (time, quality) of the solutions delivered by the student programs
- ability to work in a team solving a lab assignment
- recommendations improving the teaching process.

Programme content

The lecture covers the following topics: Computational complexity of optimization problems: NP-hardness. The notion of approximation algorithms, examples of approximation algorithms. Hardness of approximation. Practical solving of hard combinatorial optimization problems. Algorithm selection problem. Computationally easy combinatorial optimization problems: Shortest paths in graphs: Dijkstra's algorithm, DAG algorithm, all-pair shortest paths algorithm. Transitive closure of a binary relation: Floyd-Warshall algorithm. Network flows and related problems: maximum flow problem, Dinic algorithm. flows with minimum arc flow, minimum cost flows, applications of max flow problem in solving scheduling problems and graph partitioning. Matching in bipartite graphs. Greedy algorithms with examples, e.g. Kruskal and Prim algorithms for minimum spanning tree, scheduling examples, Huffman coding. The notion of a matroid. Graph coloring problem: formulation, applications, algorithms. Packing and cutting: formulation, applications, bin packing problem, algorithms for bin packing.
During the lab-classes students solve NP-hard combinatorial optimization problems. It is required to design and code two algorithms solving the assigned problem: a fast method (e.g. simple greedy algorithm) and a metod of improved quality solutions running in longer time (e.g. some metaheuristic, type to student's discretion).

Teaching methods

Lecture: multimedia presentation, illustrated with examples given on the board.
Labs: practical solving combinatorial optimization problems by coding their solutions, conducting computational experiments, discussion on the chosen methods, team work.

Bibliography

Basic
1. J. Błażewicz, Złożoność obliczeniowa problemów kombinatorycznych, WNT, W-wa, 1988
2. W. Lipski, Kombinatoryka dla programistów, WNT, W-wa, 1982
3. M.R.Garey, D.S.Johnson, Computers and intractability: A guide to the theory of NP-completeness, W.H.Freeman, San Francisco, 1979
4. W.Cook, W.Cunningham, W.Pulleyblank, A.Schrijver, Combinatorial optimization, Wiley &#38;#38; Sons, 1998
5. M.Sysło, N.Deo, J.Kowalik, Algorytmy optymalizacji dyskretnej z programami w języku Pascal, PWN, Warszawa, 1993

6. T.Cormen, C.Leiserson, R.Rivest, C.Stein, Wprowadzenie do algorytmów, WNT, Warszawa, 2005
7. M.Kubale (redaktor), Optymalizacja dyskretna modele i metody kolorowania grafów, WNT, Warszawa, 2003.

Additional
1. J. Błażewicz, K. Ecker, E.Pesch, G. Schmidt, J. Węglarz, Scheduling Computer and Manufacturing Processes, Springer, Berlin, New York, 2001
2. J.Błazewicz, W.Cellary, R.Słowinski, J.Weglarz, Badania operacyjne dla informatyków, WNT, W-wa, 1983
3. L.Banachowski, A.Kreczmar, Elementy analizy algorytmów, WNT, W-wa, 1989
4. A.V.Aho, J.E.Hopcroft, J.D.Ullman, Projektowanie i analiza algorytmów komputerowych, PWN, W-wa, 1983
5. K.Manuszewski, Grafy Algorytmicznie trudne do kolorowania, praca doktorska, WETI, Gdańsk, 1997
6. M.Drozdowski, D.Kowalski, J.Mizgajski, D.Mokwa, G.Pawlak, Mind the gap: a heuristic study of subway tours, Journal of Heuristics vol.20, Issue 5, October 2014, pp 561-587, DOI 10.1007/s10732-014-9252-3
7. J.Marszałkowski, D.Mokwa, M.Drozdowski, Ł.Rusiecki, H.Narożny, Fast algorithms for online construction of web tag clouds, Engineering Applications of Artificial Intelligence, vol. 64 (2017) pp. 378-390 DOI: 10.1016/j.engappai.2017.06.023
8. J.Wawrzyniak, M.Drozdowski, É.Sanlaville, Selecting Algorithms for Large Berth Allocation Problems, European Journal of Operational Research, Volume 283, Issue 3, 16 June 2020, Pages 844-862, https://doi.org/10.1016/j.ejor.2019.11.055

Breakdown of average student's workload

	Hours	ECTS
Total workload	100	4,0
Classes requiring direct contact with the teacher	45	2,0
Student's own work (literature studies, preparation for laboratory classes/tutorials, preparation for test, lab. report preparation) [1]	55	2,0

[1] delete or add other activities as appropriate