COURSE DESCRIPTION CARD - SYLLABUS

Course name
Scheduling in Theory and Practice [S1Inf1>SZAD]

Course

Field of study Computing	Year/Semester 4/7
Area of study (specialization) –	Profile of study general academic
Level of study first-cycle	Course offered in polish
Form of study full-time	Requirements elective

Number of hours

Lecture 30	Laboratory classes 30	Other (e.g. online) 0
Tutorials 0	Projects/seminars 0	

Number of credit points

4,00

Coordinators

dr Maciej Machowiak
maciej.machowiak@put.poznan.pl

prof. dr hab. inż. Małgorzata Sterna
malgorzata.sterna@put.poznan.pl

Lecturers

Prerequisites

Students starting this course should have a basic knowledge of algorithms and data structures, computational complexity, as well as programming in any high-level languages. Students should be able to analyze practical cases, solve combinatorial problems, implement algorithms dedicated to such problems, evaluate their quality and efficiency, as well as to acquire information from the indicated sources. Students must present ability to work independently and in a team, responsibility for this work, as well as ability to communicate with others.

Course objective

Teach students fundamental knowledge of scheduling theory including basic single, parallel and dedicated machine models, present exemplary scheduling algorithms based on various algorithmic techniques involving among others graph theory and geometry, show the methodology of analyzing and solving scheduling problems using achievements of complexity theory. Develop students" ability to solve practical scheduling problems starting from formal modelling a real case, through selecting a proper algorithmic approach and designing an algorithm, to evaluating its quality and efficiency in computational experiments. Develop student"s ability to team work during laboratories.

Course-related learning outcomes

Knowledge:
Upon completion of the course the student:
1. has the general knowledge of fundamentals of scheduling theory, particularly concerning dedicated machines, i.e. shop systems;
2. knows classical scheduling algorithms solving basic single machine, parallel machine and two-machine shop models;
3. is familiar with exemplary applications of various algorithmic techniques and methods of computational complexity theory to solve scheduling problems.

Skills:
Upon completion of the course the student:
1. can propose a proper algorithm, particularly a heuristic one, to solve a given scheduling problem;
2. is able to implement and evaluate the proposed algorithms from various viewpoints;
3. can participate in discussions on various ideas to solve a scheduling problem and on their evaluation;
4. can design and conduct computational experiments allowing to evaluate simple scheduling algorithms;
5. is able to adjust approaches known from the literature to the specificity of a considered scheduling problem.

Social competences:
Upon completion of the course the student:
1. is aware of development of new algorithmic techniques applicable to solve practical scheduling problems;
2. is aware that research results significantly support solving real practical cases.

Methods for verifying learning outcomes and assessment criteria

Learning outcomes presented above are verified as follows:

Learning outcomes gained during lectures are verified based on the results of 90-minute written test at the end of a teaching period. The test contains dozens of open-ended and close-ended questions. To formulate the answers, awarded with various numbers of points, the students must recall the knowledge or apply some techniques or methods presented during the lectures. Positive grade is obtained by acquiring at least 50% of the maximum number of points. Thresholds for particular grades are increased with 10%.

Learning outcomes gained during laboratories are verified based on the results of solving three tasks. Completing each task requires designing, implementing and testing in computational experiments an algorithm solving a simple scheduling problem, preparing some auxiliary procedures, as well as preparing an optional written report. Positive grade is obtained only after completing all three tasks. The grade depends on: the position in ranking of algorithms' efficiency expressed in terms of the quality of solutions and computational time, activity during laboratories, and preparing optional written reports presenting the proposed methods.

Programme content

Lectures cover the following topics:
1. General formulation of scheduling problem: basic task and machine parameters, classical objective functions and their practical interpretation.
2. Classification of scheduling problems, three-field notation and its relation to practical problems.
3. Classification of scheduling problems in the view of the computational complexity, proving intractability of

exemplary scheduling problems.
4. Selected single machine problems: list algorithms, Jackson's algorithm, Horn"s algorithm, Smith's rule, Hodgson's algorithm.
5. Selected parallel machine problems: McNaughton's algorithm, Hu's algorithm, list algorithms, exemplary approximation algorithms, dynamic programming methods, examples of applying graph theory to solve some scheduling problems.
6. General formulation of shop scheduling problems and examples of their applications in solving real world cases.
7. Using graph models for efficient representation of scheduling problems.
8. Open shop problem: Gonzalez-Sahni's algorithm and intractability of multi-machine models, an example of applying mathematical programming to solve scheduling problems.
9. Flow shop problem: Johnson's algorithm, heuristic algorithms for multi-machine cases (aggregation heuristic), geometric method.
10. Job shop problem: Jackson's algorithm, generalized geometric method, list algorithms, constraint propagation method.
11. Non-classical shop models inspired by practical applications: setup times, transportation times, buffers, flexible systems, multiprocessor tasks.
12. Fundamentals of online scheduling: basic notations and techniques.
13. An exemplary case study: from modelling, through determining the computational complexity, designing various types of algorithms to validation in computational experiments.

During the laboratories students solve 3 tasks, concerning given scheduling problems. Each task requires:
1. to design individually, implement and validate in computational experiments a simple heuristic algorithm for a given scheduling problem (competition for the best algorithm in a laboratory group);
2. to design individually and implement the procedure generating test instances and the procedure verifying the correctness of obtained schedules;
3. to perform in a laboratory group computational experiments for instances generated by particular students, to validate the correctness of all algorithms designed by all group members and to discuss obtained results;
4. to prepare an optional written report.

Teaching methods

1. Lectures: multimedia presentations of programme contents with numerous examples.

2. Laboratories illustrating programme contents presented during lectures with given examples: analysis and solving a simple scheduling case, designing and implementing algorithms, planning and conducting computational experiments, composing reports (optional), discussions and negotating with collegues common testing environment and rules of evaluating algorithms.

Bibliography

Basic
1. Handbook on scheduling: from theory to practice, J. Błażewicz, K. Ecker, E. Pesch, M.Sterna, G. Schmidt, J. Węglarz, Springer, Cham, 2019.
2. Planning and scheduling in manufacturing and services, M. Pinedo, Springer, New York, 2007.
3. Late work scheduling in shop systems, M. Sterna, Rozprawy nr 405, Wydawnictwo PP, Poznań, 2006.
4. Badania operacyjne dla informatyków, J. Błażewicz, W. Cellary, R. Słowiński, J. Węglarz, WNT, Warszawa, 1983.

Additional
1. Scheduling: theory, algorithms, and systems, M. Pinedo, Springer, New York, 2016.
2. Scheduling algorithms, P. Brucker, Springer, Berlin, 2007.
3. Handbook of scheduling: algorithms, models, and performance analysis, J.Y.-T. Leung (eds.), CRC Press, Boca Raton, 2004.
4. Operations Research. A Practical Introduction, M.W. Carter, C.C. Price, CRC Press, Boca Raton, 2001.
5. Handbook of combinatorial optimization, D.-Z. Du, P.M. Pardalos (eds.), Kluwer Academic Publishers, Boston, 1998.
6. The disjunctive graph machine representation of the job shop scheduling problem, J. Błażewicz, E. Pesch, M. Sterna, European Journal of Operational Research 127/2 (2000), 317-331 (https://doi.org/

10.1016/S0377-2217(99)00486-5)
7. Open shop scheduling problems with late work criteria, J. Błażewicz, E. Pesch, M. Sterna, F. Werner, Discrete Applied Mathematics 134 (2004), 1-24 (https://doi.org/10.1016/S0166-218X(03)00339-1)
8. Scheduling on parallel identical machines with late work criterion. Offline and online cases. X. Chen, M. Sterna, X. Han, J. Błażewicz, Journal of Scheduling 19/6 (2016), 729-736 (https://doi.org/10.1007/s10951-015-0464-7)
9. Late and early work scheduling: A survey, M. Sterna, Omega (2021), 102453 (https://doi.org/10.1016/j.omega.2021.102453)
10. Polynomial time approximation scheme for two parallel machines scheduling with a common due date to maximize early work, M. Sterna, K. Czerniachowska, Journal of Optimization Theory and Applications 174/3 (2017), 927-944, (https://doi.org/10.1007/s10957-017-1147-7)

Breakdown of average student's workload

	Hours	ECTS
Total workload	110	4,00
Classes requiring direct contact with the teacher	60	2,00
Student's own work (literature studies, preparation for laboratory classes/ tutorials, preparation for tests/exam, project preparation)	50	2,00