## Course Outline

**Course Name:** Approximation Algorithms  
**Course Code:** 202-5711

### Course Information

- **Course Type:** Option
- **Credit:** 4.0

### Course Objectives

Approximation algorithms are one of the most important and central topics in computer science, under the assumption that it is impossible to solve NP-hard problems in polynomial time.

- **Objective:** How can we deal with hard problems in polynomial time?  
- **Heuristic techniques:** These techniques provide an intuitive tool to deal with difficult optimization problems.  
- **Approximation algorithms:** Focus on finding the closest solution to the optimum feasible in polynomial time for the worst input.

- **Objective:** How can we use tools from the field of experimental research, such as linear programming or semidefinite programming, to turn combinatorial problems into continuous problems without changing much of the optimum?

### Course Content

- How can we easily cover graphs with at most twice the number of vertices compared to the minimum?  
- Why is it believed that there is no better approximation than 2 for this problem?  
- How much time will a tourist take to visit all the cities in his list, and how can he find a path between them that is not much longer than the shortest possible?  
- How can we use tools from the field of experimental research, such as linear programming or semidefinite programming, to turn combinatorial problems into continuous problems without changing much of the optimum?

### Course Audience

- The course is intended for students in their 4th year, Master’s and Doctoral programs of all fields, with appropriate algorithmic and mathematical background.

### Assessment

- **Assignments:** 40%
- **Midterm Exam:** 60%
Approximation algorithms is one of the most important and exciting areas of theoretical computer science. Under the widely accepted assumption that NP-hard problems cannot be solved in polynomial time, approximation algorithms provide a straightforward answer to the question “how can we cope with hard problems in polynomial time”?

While heuristics may give us intuitive tools to cope with hard optimization problems, in approximation algorithms we try to find the solution which is closest to the optimum for the worst possible input. In this course, we will focus on using various algorithmic techniques when designing approximation algorithms, and analyzing the ratio between the solution found by the algorithm and the best possible solution (in the worst case).

Among the many interesting questions we will examine:

- How can we (easily) find a vertex cover with at most twice the number of vertices as in the minimum vertex cover?
- Why do we believe that we cannot obtain a better approximation than 2 for this problem?
- How long will it take a travelling salesman to pass through all the cities on his list, and how can he (efficiently) find a path through them which is not much longer than the shortest such path?
- How can we use tools from operations research, such as linear and semidefinite programming, to turn hard combinatorial problems into easy continuous problems, without changing the optimum value too much?
- How can these same tools help us find better approximation than what we can find using all known combinatorial approaches?

The course is open to third year undergraduates, as well as graduate students from all fields, with the appropriate algorithmic and mathematical background.

Grades:
Homework - 40%
Take-home final - 60%

Course books: