Practice Questions and Reading Material

The final exam is cumulative and will include problems relating to all the lectures. Please go over the two first study guides, as well as this one. The current study guide includes material corresponding only to lectures 20 to 26.

The chapters and exercises from the books are provided as supportive references for the material covered during the lectures.

Lecture 20: The class of NP problems - Examples of NP complete problems

Reading material: Chapter 8.1 from DPV, Chapters 34.2, 34.5 from CLRS2
Practice questions:

- What is the satisfiability problem? What is the running time of the best known algorithm for this problem in the worst case? What is the running time of checking whether a candidate solution is truly solving the problem or not?
- Are there versions of the general satisfiability problem that can be solved in polynomial time? Describe at least two of them.
- What is the traveling salesman problem? What is a dynamic programming approach for the traveling salesman approach? What is the running time of this approach?
- Show that any optimization problem can be reduced to a search problem. Show that any search problem can be reduced to an optimization problem. Why do we typically prefer to work with search problems when studying the computational complexity of algorithms?
- What is an Eulerian tour? When does a graph have an Eulerian tour?
- What is the Rudrata cycle/path problem? Is there a polynomial time algorithm for this problem?
- What is the independent set problem? Describe a dynamic programming solution for computing an independent set on trees. What is the running time of this solution? Is there a polynomial time algorithm for this problem on general graphs?
- What is the vertex cover problem? What is the clique problem? Are there polynomial time algorithms for these problems?

Lecture 21: Additional examples of NP complete problems - Reductions

Reading material: Chapters 8.1, 8.2, 8.3 from DPV, Chapters 34.5, 34.3 from CLRS2
Practice questions:

- What is the minimum cut problem? Is there a polynomial time algorithm for this problem?
- Consider the following approach for computing a minimum cut: run Kruskal's algorithm on an unweighted graph and remove the last edge of the resulting minimum spanning tree to define a cut (randomize the selection of the edges among multiple equivalent choices). Show that the probability of this cut being the minimum cut is at least $\frac{1}{n^2}$. What is the running time of the randomized approach that computes the minimum cut with high probability through Kruskal's algorithm?
- What is the balanced cut problem? Is there a polynomial time algorithm for this problem?

- What is the knapsack problem? Is there a polynomial time algorithm for this problem? What is the subset sum problem and how does it relate to the knapsack problem?
- What is the class of NP problems? What is the class of P problems? What is the relation between these two classes of problems? For instance, is P = NP?
- What does it mean that you can reduce a search problem A to a search problem B? What do you need to do in order to provide such a reduction? (you can provide a drawing to explain your answer)
- What is the class of NP-compete problems? What is the class of NP-hard problems?
- Show that the general satisfiability (SAT) problem reduces to the 3-SAT problem.

Lecture 22: Examples of reductions between NP complete problems

Reading material: Chapter 8.3 from DPV, Chapter 34.3, 34.4 from CLRS2
Practice questions:

- When is a problem in the class of co-NP problems? Provide an example of a co-NP problem.
- Show that the 3-SAT problem reduces to the Independent Set problem.
- Show that the Independent Set problem reduces to the Vertex cover problem.
- Show that the Independent Set problem reduces to the Clique problem.
- Show that the Circuit SAT problem reduces to the SAT problem. What is the importance of this reduction?

Lecture 23: Intelligent Exhaustive Search, Intro to Approximation Algorithms

Reading material: Chapters 9.1, 9.2 from DPV, Chapter 35.Intro from CLRS2
Practice questions:

- What is the idea behind backtracking search in order to solve NP-complete problems?
- Describe a general framework for backtracking search.
- Describe the application of backtracking search to the satisfiability problem, i.e., which sub-problems are expanded at each iteration and which branching variable is considered?
- Describe the general branch-and-bound approach for optimization problems.
- Describe an application of branch-and-bound to the traveling salesman problem.
- How is the approximation ratio of an approximation algorithm computed?
- Provide an approximation algorithm for the vertex cover problem. What approximation ratio does it achieve?

Lecture 24: Approximation Algorithms, Local Search Heuristics

Reading material: Chapters 9.2, 9.3 from DPV, Chapter 35.1-3 from CLRS2
Practice questions:

- When does a distance function satisfy metric properties?
- What is the k-clustering NP-complete problem? Provide a simple approximation scheme for the k-clustering problem. What is the approximation ratio that it achieves. Prove it.

- Provide an approximation algorithm for the traveling salesman problem given that the underlying graph has edge weights that satisfy metric properties. What is the approximation ratio for this algorithm? Prove it.
- Describe the general local search framework.
- Provide a local search approach for the traveling salesman problem.
- Provide a local search strategy for the graph partitioning problem.
- How can you deal with local optimal in the context of local search?

Lectures 25/26: (Integer) Linear Programming and Maximum Flow (minimum-cut) problems on Networks

Reading material: Chapters 7.1.1, 7.2 from DPV, Chapters 29.1, 29.2, 26.2 from CLRS2
Practice questions:

- What kind of problems can be resolved using a linear programming approach?
- You may be provided a description of such a problem and asked to reformulate it as a linear program (i.e., to define the objective function and the constraints).
- You may also be asked to visualize 2D linear constraints of a linear program and solve the corresponding problem graphically.
- What is the idea behind the simplex algorithm? How and why does it work? What is the running time of the simplex algorithm in the worst case?
- Why are maximization and minimization challenges equivalent as linear programs?
- Why can constraints be expressed either through equations or inequalities in linear programs?
- Why is it not important whether variables are restricted to be non-negative or if they are unrestricted?
- What is the standard form of a linear program?
- What is the difference between integer linear programming and general linear programming?
- Is linear programming in P or NP? Is integer linear programming in P or NP?
- Give an example of a flow in network problem. Reformulate this problem as a linear program, i.e., define the objective function and the linear constraints.
- Describe an approach for solving maximum flow problems in networks. What is the running time of the approach?
- What is the relationship between maximum flow and graph cuts? In other words: what does the maximum flow, minimum cut theorem specify. Prove it.

Related exercises from DPV:
Chapter 7: 7.1, 7.4, 7.5, 7.6, 7.7, 7.9, 7.10, 7.12, 7.17, 7.18, 7.19, 7.21, 7.28a,b
Chapter 8: 8.1, 8.2, 8.3, 8.4a, 8.6, 8.10, 8.15
Chapter 9: 9.1, 9.2, 9.3, 9.4, 9.5