CS301- Algorithms
Fall 2020-2021

Project Guidelines

You will carry out project work as a group. This project work will include both theoretical and practical aspects formulated around a hard problem. Here, by hard we mean a problem which is known to be computationally hard, such as NP-hard, PSPACE-hard, etc. Your work around this problem can be considered as divided into three parts as follows:

1. The problem
    a. The definition of the problem
    b. Proving the hardness of the problem

2. The algorithm
    a. Finding a heuristic algorithm for the problem, or (if you like) designing a new heuristic algorithm by yourselves
    b. Analysing the worst case time complexity of the heuristic algorithm
    c. Proving the correctness of the algorithm
    d. If the algorithm is an approximation algorithm, a proof for the ratio bound.

3. The implementation
    a. Finding an implementation of the algorithms, or (if you like) implementing the algorithm by yourselves
    b. Testing the correctness of the implementation by applying some test cases designed by yourselves
    c. Testing the performance of the implementation by applying some test cases designed by yourselves
    d. If the algorithm is an approximation algorithm, an experimental analysis for the ratio bound of the implementation in practice.

All this work will be documented in the form of a project report and will be presented as a team at the end of the semester.

The groups will consist of 4 people. If you cannot form a group of 4, we will put you in a group.

Each group will pick a project from the problem pool. We will announce the pool of problems on SUCourse+.

You are expected to prepare your project report and present your work as a team at the end of the semester.

Your project report must have the following sections:

1. Problem Description

Describe your problem both as intuitively and formally. If possible talk about the applications (where this problem) might be used in practice. State the hardness of your problem in the form of a theorem. For example, give a theorem claiming that your problem is NP-complete, or NP-hard, or PSPACE-complete, PSPACE-hard, etc. For the proof of this theorem, you can simply cite (refer to) an appropriate source in the literature which gives this proof, or you can give an explicit proof in your report. In the case you give an explicit proof and this proof is not a novel proof suggested by your group (which is ok for your report), you must still give the citation to the original paper/book from which you got this proof.

2. Algorithm Description

Find a heuristic or an approximation algorithm for your problem and explain the algorithm in detail. If you want, you can also design an algorithm yourself. State the approach of this algorithm to your problem by explaining the steps of the algorithm. Also give a pseudo-code of the algorithm. If the algorithm follows an algorithm design technique, e.g. divide-and-conquer, dynamic programing, etc., you need to mention this and explain why you think the algorithm is using this technique.

In addition, if the algorithm is an approximation algorithm, you need to state and show the proof of the ratio bound. The proof need not be a novel proof that your team put forward, although this is perfectly okay. You can just include a proof that already exists in the literature, by citing the appropriate source.

3. Algorithm Analysis

In this part you need to analyze both the correctness and the complexity of your algorithm in detail.

- Claim and show that the algorithm works correctly, possibly in the form of a theorem.
- For the complexity analysis, consider worst case time complexity. You can use the techniques we have seen in the lectures, including for example, the amortized analysis. If possible, try to give a tight upper bound by using Θ.
- Optionally, you can also consider the complexity of the algorithm for resources other than time, e.g. the space complexity.

4. Experimental Analysis of The Performance

In the experimental analysis part of your project, you are expected to analyze the performance of the implementation of the algorithm experimentally. You may implement the algorithm yourself, or you can simply use an implementation that you find available on the internet.

The complexity results presented in Section 3 are theoretical results. The worst case time complexity of the algorithm may not be displayed in practice. You will assess the practical time complexity of the algorithm by performing performance tests. We expect you to use the statistical methods which you covered in the lectures for this part. Furthermore, we want you to visualize your results in graphs (for example: a running time graph). You need to fit a line for the measurement values.

In addition, if your algorithm is an approximation algorithm, you need to consider the approximation performance of the algorithm in practice as well. Although you will have a theoretical ratio bound given in Section 2, this ratio bound might be better in practice. Therefore, you will need to design and perform tests to assess the performance of the algorithm in terms of the approximation to the optimal values. Note that, in order to carry out these tests you will also need to use an implementation of a brute-force algorithm (either implemented by your team or found from the internet) to find the optimal value. Since the problems are NP-hard (and since we do not know the answer of the "P vs. NP" question yet), we currently need to use exponential time algorithms only. Therefore, the runs to find the optimal values will typically be restricted to small input sizes only.

5. Experimental Analysis of the Correctness

The correctness results given in Section 3 show that the algorithm is correct. However, there can be errors introduced into the implementation. In other words, there can be coding errors. For this part of work, you will need to perform testing of the implementation to assess the correctness of the implementation of the algorithm. We want you to use some testing methods for your algorithm that is covered in the lectures for this part.

6. Discussion

Discuss your results in detail. Are there any defects in your algorithm? Is there any inconsistency between your theoretical analysis and experimental analysis?

Submission

On SUCourse+, for each group, only ONE person will submit a single zip file that contains your project report (in PDF format), presentation and codes. Your zip file name must be in the following format:

CS301 Project Group X.zip

where X is your group number which you can find here.