CS4234: Optimisation Algorithms Mini Project

October 22 – November 14, 2018

V1.2: Steven Halim

Preliminaries

This mini project spans 3+ weeks from (start of) Week 10 to Wednesday of Week 13. It worth 15% of the course weightage (3x of the normal PS). It focuses mostly on things that are difficult, or perhaps impossible, to be examined on paper for just 2 hours, which is the experimentation and empirical analysis of Stochastic Local Search (SLS) techniques on real (NP-)hard COPs.

To reduce your own project workload and my own grading workload in the busiest last month of the semester, I will reduce the number of student submissions (up to 26) by a factor of ≈ 3 by forcing most of you to work in a *team of three*[1]. As 26 % 3 is not 0, on ideal scenario there will only be 9 teams. 8 with 3 members and 1 with 2 members. The only team with 2 members will get a boost of 1% bonus point (still max at 15%). After some deliberation, this is Steven's intended format to minimize the total presentation time later on the last lecture slot. Therefore, Steven will prevent anyone from going solo (the workload will be too heavy and will cause problem with presentation time later). Note that team of size 4 is **strictly not allowed**.

Please inform Steven latest by Thursday, 25 October 2018, i.e. one day after the official start of this project on your team member grouping (one member email/inform me and mention the other two teammates). Teammates for pairs or triples can be across CS4234 tutorial groups as everyone will present their findings at the same time (during last lecture on Wednesday, 14 November 2018). Students without project group will be 'randomly grouped' by this deadline...

To-Do

Use *any* techniques that you have learned so far in CS4234 (or will learn in the last few weeks of CS4234, or any hacking techniques) to *get the best possible results* on two (NP-)hard Combinatorial Optimization Problems (COPs). As you will work in a team of three, you can either split the workload by having 2 members do one COP and 1 other member to do the other COP and share the successful (or unsuccessful) techniques that have been tried, or all members can attack both COPs, and simply present the best solutions (and combined experimentation reports) found by any team member at the end.

The first COP for the third AY is still the same as the first two AYs: Travelling-Salesman-Problem (No-Repeat version, i.e. NR-TSP, not 100% metric but almost...) that is available at Kattis. The second COP is 'new', but you have actually seen it before. For the first COP, Steven has lots of research data from his own PhD days: 2004-2009 and seniors works in 2016-2017. For the second COP, everyone (including Steven) will be clueless at the beginning and we will all gradually learn something in the next few weeks.

COP 1: `Travelling-Salesman-Problem` (TSP) (7%)

Assoc Prof Per Austrin from KTH Royal Institute of Technology, Sweden, whom I have met several times in ACM ICPC World Finals and helped as (external) technical committee for my ACM ICPC Singapore 2015, has set up this problem: `https://nus.kattis.com/problems/tsp` which is very suitable for our CS4234 training purposes. If you read the problem description carefully, it is clearly the No Repeat TSP that we discussed back in CS4234 Lecture 4.

[1]This is an NP-hard optimization problem, resembling `Partition-Into-Triangles` problem.

However, it is not perfectly Metric TSP because although we use Euclidean distances, we have to *round them to nearest integers*, i.e. we may have a triangle with side lengths of $a = 1.4$, $b = 1.4$, $c = 2.6$ (which satisfies triangle inequality as $a + b \geq c$) but if the side lengths are rounded to nearest integers, we may have $a = 1$, $b = 1$, and $c = 3$ (which does *not* satisfy triangle inequality anymore as $a+b < c$). This round to nearest integer operation is called as the *nint(x)* operation in the famous (to TSP researchers) TSP Benchmark Library called: TSPLIB: `http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/` that Steven used during his PhD days. More detailed documentation of TSPLIB format, especially the EUC_2D format, can be found in: `http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/tsp95.pdf`. Albeit it is not perfectly Metric TSP by definition, running 2- (or the harder to implement 1.5-, if you choose to do so) approximation algorithm for Metric TSP should still give a reasonable result.

Now `https://nus.kattis.com/problems/tsp` provides a convenient platform for Steven to have CS4234 students experiment on various SLS techniques on TSP. We will hear the first few preliminary techniques during Lecture 9 on Week 10 that will be expanded further in the subsequent lectures. You are given only 2 second per (secret) TSP instance. There are 50 TSP instances of up to $N \leq 1\,000$ points (vertices). You are given a certain scoring function that measures your tour quality of each TSP instance against its (known) optimal value (maybe after running the optimal algorithm for hours). As of 18 October 2016, getting 42.0/50 points already put you in the top 10, but one year later, at 24 October 2017 you need 45.7/50 to be in top 10. By 22 October 2018, you need 46.6/50 to be in top 10 with one CS4234 senior Tan Jun An occupying current rank 3 slot with 47.1/50 points `https://open.kattis.com/problems/tsp/statistics` (I used the less powerful open version in the first two years). Steven's PhD code from $\approx$ 10 years ago is currently at $\approx$ 41.5/50 (I do not bother optimizing further). Can you be close to these benchmark results or do better? Sign up a free account at Kattis (`https://nus.kattis.com`) if you have not done so and test your local search ideas.

Code of conduct for this Mini Project: Steven is aware that by simply Goggling 'Kattis tsp' will yield interesting hits of past public results, like this report by `http://papers.philipshield.com/tsp.pdf` or this public GitHub repository `https://github.com/estan/tsp` (I already tried resubmitting this verbatim, it currently get $\approx$ 36/50) that may give you some initial local search ideas and working :O implementation ideas with reasonably high scores. As it is impossible for Steven to police this, I will let you peruse those *current* publicly available material in your quest to get as close as possible to 50/50. The rule of NOT posting your own project to public domain throughout this semester as with our other PSes remains.

The scoring scheme for this COP 1: TSP is therefore:

1. Your team best score at Kattis (maximum 50) when ranked against the other 8 teams in CS4234 (3%, with the best group (hopefully within top 10 @ Kattis) taking all 3% and the next ranked group gets 0.2% less, i.e. 2.8% and so on (as we will only have exactly 9 teams, the last one will get at least 1.4 for this component). Each group has to show briefly the Kattis submission (just mention the submission id, Steven can verify) that generates that best score for verification purpose (and to see if you copy too much (or all?) from the few available public resources :(...). As we will use `https://nus.kattis.com/problems/tsp` this time (registration key is mentioned via separate channel), Steven will have access to all your Kattis submissions and all those plagiarism flag alerts, so never submit anyone else code verbatim (or with trivial modifications) :O... Note that submissions to the open version `https://open.kattis.com/problems/tsp` will **NOT** be tracked this time.

2. Your group presentation about your experimentations on TSP @ Kattis on Wednesday, 14 November 2018 (3%, categorical grading: average: 2%, good: 3%, bad: 1%). Steven is more interested in your learning/experimentation journey on various SLS techniques on attacking the TSP rather than just the final result above. Therefore, you have to also record negative results and should be able to give some explanation on why such (failed) ideas did not work.

3. Your no-more than 2 pages of report about TSP experimentations (the written from of your presentation above), for Steven to re-read after your presentation to give final grade (1%, categorical grading: OK: 1% or poor: 0.5%).

COP 2: `Min-Weight-Vertex-Cover` (MWVC), PS1++ (8%)

For the second COP this S1 AY18/19, Steven will extend PS1 (Art Gallery) subtask D. Steven has re-opened PS1 and added a new subtask E which is *currently* subtask D with just one modification to the verification algorithm. Instead of accepting solution that produces 2-OPT (or 100% off) from known optimal or best known values across all test cases, it is now set to *currently* only accept solution that produces output not more than **30% off** from known optimal or from best known values across all test cases (this is called **benchmark 1**). As soon as your program produces result that is 30.0....1% or more for at least one test case, it will be automatically flagged as wrong answer.

Assuming that a few of the best students can pass this threshold soon, Steven will lower the current benchmark of **30%** to a lower number and/or add new test cases as new subtask F (this is called **benchmark 2**) (likely announced during Wednesday lectures of Week 11).

Ultimately, by Wednesday of Week 12, i.e. Wednesday, 07 November 2018, 2pm (after Week 12 lecture), each team is supposed to submit **ONE** secret test case. You can add whatever 'graph signature' that your team can think of so that your team's code can recognize it and output the optimal (or best known) answer for your own test case in $O(1)$. Steven will then install these 9 secret test cases from 9 different groups (on top of Steven's test cases) as the new last subtask G for everyone to attack until Wednesday of Week 13.

The scoring scheme for this COP 2: MWVC is therefore:

1. Your group's ability to pass benchmark 1 (the initial 30% off) and/or benchmark 2 (the lower than 30% off) (1%, categorical grading: 1% pass both benchmark 1 and 2, 0.5% pass only the lower benchmark 1, 0% fail).

2. Your group's creatively crafted test case in PS1 format with your best known (or even optimal, if you can show it) answer. Supply a certificate (which vertices/corners have to be covered/guarded by a policeman) to get that best known (or even optimal) value. Details of test case and certificate submission format will be announced nearing Week 12 (1%, categorical grading: 1% good, interesting, and challenging test case, 0.5% trivial test case :(, 0% no submission).

3. Your team total answers at Mooshak when ranked against the other 8 teams in CS4234 (2%, with the best group with the lowest total answers (unlikely to have all optimal (minimum) answers for all test cases :O) taking all 2% and the next ranked group gets 0.2% less, i.e. 1.8% and so on (as we will only have exactly 9 teams, the last one will get at least 0.4 for this component). As all submissions at Mooshak are recorded, Steven will have access to all your submissions thus Steven will know if your code is legit or not.

4. Your group presentation about your experimentations on MWVC problem on Wednesday, 14 November 2018 (3%, categorical grading: average: 2%, good: 3%, bad: 1%). Steven is more interested in your learning/experimentation journey on various SLS techniques on attacking the MWVC problem rather than just the final result above. Therefore as with TSP report, you should also record negative results and provide explanations.

5. Your no-more than 2 pages of report about MWVC problem experimentations (the written from of your presentation above), for Steven to re-read after your presentation to give final grade (1%, categorical grading: OK: 1% or poor: 0.5%).