Final Year Project Report

Full Unit - Plan

Metaheuristics

Luke Sell

A report submitted in part fulfilment of the degree of

MSci (Hons) in Computer Science

Supervisor: Eduard Eiben

Department of Computer Science
Royal Holloway, University of London

October 3, 2021

Declaration

This report has been prepared on the basis of my own work. Where other published and unpublished source materials have been used, these have been acknowledged.

Word Count: 1854

Student Name: Luke Sell

Date of Submission: October 3, 2021

Table of Contents

Abstract

Introduction and Background

A Metaheuristic is a general type of heuristic that is useful in quickly finding near ideal solutions to a range of optimisation problems, and thus does not try to use the properties of a specific problem to modify how they search for a solution, but instead follows a general strategy for all problems. This is different in comparison to standard heuristics which are designed to solve specific problems and are not usable in general outside the scope of that problem. Metaheuristics are therefore able to work efficiently even without knowing anything about the problem and is also better at dealing with local minima than standard heuristics. Metaheuristics are most useful when finding the most optimal solution through an exhaustive search would take too long comparatively, for example, the travelling salesman problem, which is NP hard and has a search space that grows exponentially with the increasing size of the problem. There are different types of Metaheuristics with varying strategies used that each have their own advantages and disadvantages relative to each other and can be grouped by classifications based on how they search for a solution to a problem.

Aims and Objectives

The overall aim of this project is to understand the different strategies used by various metaheuristics in searching for solutions to optimisation problems, to design and implement examples of each of these metaheuristics to solve a wide range of these problems and compare the efficiency of each of these different algorithms by benchmarking them against common problems. To do this I will have to research these metaheuristics, explain how they work differently to each other to find solutions and how I can create each of these to solve example problems. I will also need to be able to demonstrate my implementations to a user through the use of a GUI that allows for each algorithm to be tested on a chosen example problem, with a way of benchmarking all of them against different problems to compare the efficiency and usefulness of each.

Overview of Early Deliverables

Proof of Concept Programs

To begin I will design and implemented the tabu search metaheuristic, this will require me to first set up an initial code structure and a basic user interface. I will be writing my code in C++17 and using the SDL 2 library for GUI handling, I will also be using catch 2 as my testing framework, once the code is working I will have to thoroughly test it on example problems to get a better understanding of how it works.

Reports

I will need to write reports on computational complexity, local search and tabu search, this will involve researching these topics using the sources I have found already and describing how they relate to the overall project. Computational complexity will be important in explaining the usefulness of metaheuristics in general as the problems they will be used on are NP hard, meaning the search space grows exponentially with larger problems, finding the best solution would be infeasible, so understanding this will allow me to motivate the use of metaheuristics instead to solve the problem. I will then begin research on the first two metaheuristics and will describe how they work and design the algorithms in pseudocode so that I can implement them later, I will also note their key properties and how this modifies the way they search for a solution, which will be useful later when I need to compare the individual metaheuristics against each other.

Overview of Final Deliverables

The Program

I will continue with implementing the other metaheuristics and testing them, if I have time I will extend this by implementing even more heuristics on other problems in extension to the pseudo boolean optimisation problem. I will also implement a GUI for the metaheuristics to test them, this will involve using the SDL 2 library, I will first create draft designs of what I expect this to look like, before coding it and then testing it with users to make sure it functions properly and intuitively. I will finally benchmark and compare the different metaheuristics by running them many times on different sized problems and recording the times in a log, which I will then use for my comparison.

The Report

I will continue my research with reports on iterated local search, genetic algorithms and memetic algorithms, this will further my understanding of different types of metaheuristics and allow me to design and implement versions of them and finally benchmark them on example problems for a comparison on the efficiency of each individual metaheuristic.

Timeline

Report on the basics of computational complexity from 08/10/2021 to 17/10/2021

Report on local search from 19/10/2021 to 27/10/2021

Report on tabu search from 30/10/2021 to 11/11/2021

Implementation of tabu search from 19/11/2021 to 30/11/2021

Interim submission on 03/12/2021

Preparation for presentation from 04/12/2021 to 05/12/2021

Report on iterated local search from 06/12/2021 to 12/12/2021

Report on genetic algorithms from 13/12/2021 to 20/12/2021

Report on memetic algorithms from 21/12/2021 to 28/12/2021

Draft GUI designs from 29/12/2021 to 30/12/2021

Implementation of GUI from 02/01/2022 to 10/01/2022

Testing of GUI from 11/01/2022 to 12/01/2022

Implementation of iterated local search from 13/01/2022 to 23/01/2022

Implementation of genetic algorithm from 25/01/2022 to 04/02/2022

Implementation of memetic algorithm from 06/02/2022 to 17/02/2022

Implementation of other metaheuristics from 19/02/2022 to 26/02/2022

Testing and benchmarking the algorithms from 01/03/2022 to 04/03/2022

Comparing the algorithms from 08/03/2022 to 16/03/2022

Proof reading the report and generating the code documentation and UML from 18/03/2022 to 23/03/2022

Final submission on 25/03/2022

Preparation for demonstration from 02/04/2022 to 18/04/2022

Risk Assessment

If the algorithms are too slow it might be difficult to benchmark and test them on large sets of example problems, this is very unlikely to be an issue as the metaheuristics should be very fast even on large sets and problems with large search spaces, but it is still important to make sure these algorithms are tested thoroughly, therefore I shall schedule plenty of time for this specific purpose.

I may have problems understanding some of the concepts needed for writing the reports, therefore I will make sure I have a wide range of different research sources that cover each report so that I can learn about each metaheuristic and cite them in my reports, I do not think this is likely to be a problem but it will still be important to have many different sources of varying types when writing the reports.

The GUI might not be intuitive enough for the user, might not display correctly or might not even function as intended, there is a high chance of this happening in some way, but it is not as important to mitigate as other potential risks as the program should hopefully still be mostly usable for the user, however I will try to mitigate this by doing sufficient user testing to observe any issues with the UX so that they can be fixed and testing the program on multiple computers to check that the resolution the GUI is displayed at is generally compatible with most modern computers that may be used by the user.

I might not be able to sufficiently and accurately benchmark the different metaheuristic implementations for comparison if I can not test them on enough problems of varying large random search spaces, this is important in proving the usefulness of the algorithms, but likely wont be a problem as I should be able to find many example problems to test the metaheuristics on, if it does happen however I will mitigate it by creating additional problems to use for testing that can help me give a good overview in comparison of where each metaheuristic succeeds or fails against the other algorithms.

It is very important that the program itself is runnable on a different computer, it is possible that due to hardware and software differences that some parts of the program may work differently or not work at all when run by another user, therefore to mitigate this risk I should thoroughly test my program works in all ways exactly as expected on different hardware and software combinations to find and fix any issues before submission of my project, the most likely issue might be not having up to date language distributions for C++, which I could solve by statically linking these into the compiled executable of the program.

Bibliography

[1] Metaheuristics First Edition 2016 by Patrick Siarry

This book provides a detailed overview of all the metaheuristics I will writing about in my reports and explains how they are useful, this will be helpful in gaining a basic understanding of each individual metaheuristic before I do any further research specific to the report I am writing and will most likely be the main source I am using for this project.

[2] Essentials of Metaheuristics Second Edition 2013 by Sean Luke

This book covers many of the metaheuristics I will be implementing for the project, explaining them and giving good examples, it should be useful in understanding how to design the algorithms I will be writing for the project.

[3] Metaheuristics in Combinatorial Optimization: Overview and Conceptual Comparison 2001 by Christian Blum and Andrea Roli and published in the research journal ACM Computing Surveys Volume 35 Number 3 2003 pages 268 to 308

This research paper compares the metaheuristics I will be using in my project and will be useful in explaining the differences between them.

[4] Metaheuristics 2015 by Fred Glover and Kenneth Sorensen and published on `www.scholarpedia.org/article/Metaheuristics`

This web page provides a brief introduction to metaheuristics and gives an overview and comparison of them which will be useful in first understanding them before progressing my research further onto specific classifications of metaheuristics.

[5] Metaheuristic Optimization 2011 by Xin She Yang and published on `www.scholarpedia.org/article/Metaheuristic_Optimization`

This web page also provides a brief overview of each metaheuristic which will also be useful for gaining a base understanding before any research on specific algorithms.