Lancaster University

A Heuristic population based method to solve the assymetrical TSP with time constraints

Arnaud Da Silva

36471977

Supervisor

Ahmed Kheiri

A thesis submitted in fulfillment of the requirements
for the degree of Master of Science Business Analytics

in the

Lancaster University Management School
Department of Management Science

September 2024

Declaration of Authorship

I, **AHMED**, hereby declare that this thesis entitled, **"Title"**, is all my own work, except as indicated in the text.

The report has been not accepted for any degree and it is not being submitted currently in candidature for any degree or other reward.

Signed: _____

Date: _____

Abstract

(TODO) Give a short (1 page) overview of the work. This should summarise (not advertise) your research project. After reading the abstract the reader should know what problem you are tackling, the techniques you are using, the results you have achieved.

Acknowledgements

(TODO) Thanking anyone who has helped you in any way

Contents

List of Figures

List of Tables

Abbreviations

AK Ahmed Kheiri

MS Management Science

Dedicate this to someone here.

Chapter 1

Introduction

(TODO) Set the scenes. Explain why you are doing this work and why the problem being solved is difficult. Most importantly you should clearly explain what the aims and objectives of your work are.

(TODO) Structure of the thesis. Academic publications produced (if any), including any achievements/highlights

Chapter 2

Literature Review

2.1 What is optimisation?

2.1.1 Definition and Concept

Explain the concept of optimization in the context of operations research and decision-making.

2.1.2 Exact method vs Heuristic method

Discuss different types of optimization problems such as linear programming, integer programming, combinatorial optimization, and heuristics.

2.1.3 Some day to day optimisation problems

2.2 Population based heuristic

2.3 Dynamic programming

2.4 The different routing problems

2.5 The evolution of air transport in the last decades

2.5.1 Overall stats

2.5.2 The logistic and supply constraints

2.5.3 Day to day use of optimisation in air transport

(TODO) Present a survey of your main approach and an overview of the approaches proposed previously for solving the problem dealt with in this work

(TODO) Identify the practical and research motivation of this work and the literature gaps

(TODO) How convincing is the authors' argument? (Critical response - comparisons with other research, strengths or weaknesses but in relation to your research)

Chapter 3

Problem Description

3.1 Overview

3.2 Instances Description

(TODO) Present and explain your problem AHA in detail Assumptions should be stated clearly

Chapter 4

Solving the 14 instances using a heuristic

(TODO) Describe implementation details

Chapter 5

Results and performance

(TODO) Present the results and discuss any differences between the findings and your initial predictions/hypothesis

(TODO) Interpret your experimental results - do not just present lots of data and expect the reader to understand it. Evaluate what you have achieved against the aims and objectives you outlined in the introduction

Chapter 6

Conclusion

(TODO) Explain what conclusions you have come to as a result of doing this work. Lessons learnt and what would you do different next time. Please summarise the key recommendations at the end of this section, in no more than 5 bullet points.

6.1 Summary of Work

6.2 Critics

6.3 Future Work

(TODO) The References section should include a full list of references. Avoid having a list of web sites. Examiners may mark you down very heavily if your references are mainly web sites.

Chapter 7

Progress and next steps

7.1 Coding

7.1.1 class data preprocessing

Take the instance and create all the needed attributes and methods

7.1.1.1 The class

```
def __init__(self,instance_path):
    self.instance_path=instance_path

    self.info, self.flights = self.read_file(f_name=self.instance_path)

    self.number_of_areas,self.starting_airport=int(self.info[0][0]),self.info[0][1]


    self.flights_by_day_dict = self.flights_by_day(flight_list=self.flights)

    self.flights_by_day_dict=self.redistribute_day_zero(data=self.flights_by_day_dict,number_o


    self.flights_by_day_dict=self.remove_duplicate(flights_by_day=self.flights_by_day_dict)

    self.list_days= self.flights_by_day_dict.keys()
```

```
    self.airports_by_area = self.get_airports_by_areas()
    self.area_by_airport=self.invert_dict(original_dict=self.airports_by_area)


    self.starting_area=self.associated_area_to_airport(airport=self.starting_airport)
    self.list_airports=self.get_list_of_airports()
    self.list_areas=list(self.airports_by_area.keys())

    self.areas_connections_by_day=self.possible_flights_from_zone_to_zone_specific_day()
```

7.1.1.2 What needs to be done

- For the big instances, it takes a lot of time to preprocess: solutions are directly eliminated because it runs out of time
- The issue is because: self.redistribute day zero and self.remove duplicate
- Should we improve the time complexity of these 2 functions to avoid this problem? Can we consider we only focus on the execution time of the heuristic algorithm?

7.1.2 class heuristic operators

- Here we only define the classic low level heuristic operator like swap, reverse..

7.1.3 class heuristics

7.1.3.1 The class

```
class heuristics:
def __init__(self, data_preprocessing_class):
    self.data = data_preprocessing_class

    self.starting_airport = self.data.starting_airport
    self.starting_area=self.data.starting_area
    self.total_cost = 0
```

7.1.3.2 Errors and trials

- Tried to solve this problem using graphs: unsucessfull
- I have tried to find LLH: unsucessfull because I did not have all the constraints of the problems
- Works: for each instance I find the whole feasible area solutions: an area solution is considered feasible if all the areas are visited, starting=ending and at least it exists a flight between these areas

7.1.3.3 What needs to be done

Redefine the exact scope of the dissertation: we want solve the KIWI problem using a heuristic. The question is which one. Do we want to compare it with the Local Search results? If yes I'll need the code

What Yaros did: Single point optimisation and RL

Nested problems: first you have to find a feasible area solution and then in this feasible area solution you can find the optimal solution
Problem: I don't really see known names of heuristics on my algo
Question: how can we handle these two operations knowing that when we check if an area solution is feasible we check that there is at least one flight but we can also incorporate to pick the good flight at this change.

What I plan: use some kind of GA.

I think we can try to use a population based algorithm

Target: first draft of the dissertation Friday 16th of August -¿ Is it okay for Ahmed?
Goal: all the lecture review written - 80% of the analysis done and my heuristic works - then time to do final comparison