OÉ Gaillimh
NUI Galway

Semester II Examinations 2019/2020

Exam Code(s)	1MAI1
Exam(s)	MSc in Computer Science (Artificial Intelligence)
Module Code(s)	CT5141
Module(s)	**Optimisation**
Discipline	School of Computer Science
Paper No.	1
External Examiner	Prof. Pier Luca Lanzi
Internal Examiner(s)	Prof. Michael Madden
	Dr. James McDermott *
Programme Coordinator(s)	Dr. Michael Schukat

Instructions

Answer any 4 questions. All are worth equal marks.
You may answer either: in a Word document or similar, and then `CT5141_Optimisation_Answer_Sheet.docx` is suggested; or on paper, uploading a scan of the pages.
This is an **open-book** exam: you may use textbooks, notes, and **existing resources** on the internet.
You may **not communicate** with anyone, in person, via phone, internet, or otherwise. You may **not post questions** on internet sites or elsewhere during the exam.

Duration	2 Hours exam plus 30 minutes for upload
Number of pages	5 (including this page)
Discipline	Computer Science

Requirements

Release in Exam Venue	Yes ☒ No ☐
Release to Library	Yes ☒ No ☐

Question 1: Formulating problems

Suppose we are asked to choose the location for a new police station on an island. There are no existing police stations. We wish to minimise the time spent travelling by police to reported crimes. There are 10 villages on the island, with populations $p_i, i \in \{1, \ldots, 10\}$ and reported crime rates of $r_i, i \in \{1, \ldots, 10\}$ per person. Travel time is proportional to Euclidean distance (we ignore roads). All crimes are assumed to occur only in villages. The new police station does not necessarily have to be placed inside a village.

(a) What is the search space in this problem? What form will our solutions be in? **[5]**

(b) Formulate a suitable objective function. **[5]**

(c) Describe another optimisation problem (e.g. one we have studied in class) that this problem is similar to. **[5]**

(d) Is there a danger that our optimisation algorithm will give us a solution not on the island, i.e. in the sea? If not, why not? If so, what steps can we take to avoid the problem? **[5]**

(e) Suppose the government decides that the new police station must be in one of the villages. How would this change your approach? **[5]**

Question 2: Black-Box Optimisation

(a) What does *black-box* mean, in the context of optimisation? **[5]**

(b) Define *basins* (also known as *basins of attraction*) in the context of optimisation. **[5]**

(c) What results can we expect when we use a black-box optimisation method on a non-black box problem and why? **[5]**

(d) Define the terms *exploration* and *exploitation* and explain what they mean in the context of EITHER the *late-acceptance hill-climbing* OR the *simulated annealing* algorithm. Explain how its exploration-exploitation behaviour changes over time and why this is desirable. **[10]**

Question 3: Representations

(a) Define a *neighbour* function suitable for a black-box search in the search space $\mathbb{R}^n$. You may define it in mathematical notation, Python, or pseudo-code. Include a hyperparameter δ (`delta`) representing the step-size. **[5]**

(b) The `2-opt` neighbour function is suitable for permutation spaces. To illustrate how it works, give three possible offspring of the following permutation: `123456789`. **[5]**

(c) Suppose we are given an objective function f defined over a search space X, where X consists of one categorical decision variable with approximately 5 billion possible values. How could we optimise f? **[5]**

(d) Computers represent everything, ultimately, in binary. So why don't we use a bitstring GA for every optimisation problem? **[10]**

Question 4: Constructive Heuristics

(a) Given the following data for a Knapsack problem, give an example of (i) a feasible but non-optimal solution; (ii) an infeasible solution. **[5]**

Table 1: Item values and weights. **Weight limit**: 30

item	**value**	**weight**
a	10	10
b	5	8
c	4	8
d	8	2
e	9	4
f	9	6
g	7	11
h	7	10

(b) Here is a *deterministic* constructive algorithm to create feasible solutions for the Knapsack problem: order the items by *efficiency* and then take items in order of decreasing efficiency until the weight limit would be exceeded. Define efficiency, and show the working of this algorithm on the data of part (a). **[5]**

(c) Compared to the deterministic algorithm in part (b), what is the advantage of using a non-deterministic constructive algorithm? **[5]**

(d) Recall that we studied a Covid-19 Policy Response problem and solved it using IP. In that problem, we have to choose a set of policy responses to implement, such as closing grocery shops, or implementing improved hygiene. Each policy response gives some numerical reduction in R, the basic reproduction number, whose value is initially 2.5. Each policy response has a cost in dollars. We wish to achieve $R \leq 1$ at minimal cost. How can we see this as a Knapsack problem? **[10]**

Question 5: Linear and integer programming

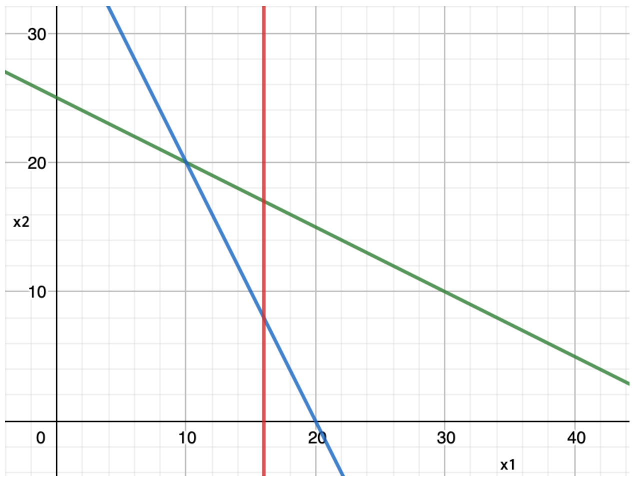
(a) In the context of linear programming, define the terms *unbounded*, *infeasible*, and *degenerate*. **[5]**

(b) Do linear programming problems suffer from *local optima*? Why or why not? **[5]**

(c) Suppose your client is a manufacturer with two products, x_1 and x_2. The below model will help to decide how much of each product to manufacture per day. Both variables are continuous and non-negative.

$$\begin{aligned} \text{Maximise profit } & 2x_1 + 2x_2 & \\ \text{Such that } & 3x_1 + 6x_2 \leq 150 & \text{(Resource 1)} \\ & 4x_1 + 2x_2 \leq 80 & \text{(Resource 2)} \\ & x_1 \leq 16 & \text{(Demand)} \end{aligned}$$

The figure below shows a partial graphical solution. Identify all the corner points of the feasible area, and identify which is the optimum. Calculate the profit at the optimum. Say which constraints are binding. (You do not need to draw any diagram in your Answer Sheet.) **[10]**



(d) Suppose the variables are also constrained to be integer-valued. Illustrate how the *branch-and-bound* algorithm will run on this problem. **[5]**