Stellenbosch University

Department of Industrial Engineering

Optimisation (Eng) 774/874

Post-block Assignment 1 — Due on 20 August 2021 at 23:55

(On single solution-based and population-based metaheuristics)

Instructions

This assignment covers those parts of Chapters 2 and 3 of the textbook by E-G Talbi, titled *Metaheuristics: From design to implementation*, considered during the lectures of the module Optimisation (Eng) 774/874. Kindly complete this assignment and submit it as a single electronic submission in PDF format on SUNLearn by the above date.

Optimisation (Eng) 774 students should attempt Questions 1 and 2 only, while all three questions should be attempted by Optimisation (Eng) 874 students.

Rubrics are provided after the assignment questions according to which the assignment will be assessed. Please pay careful attention to the requirements laid out in these rubrics.

Late submission (*i.e.* later than the required submission time mentioned above) will be penalised by 20% per day or part thereof. A final cut-off per assignment will apply after four days, and no assignments will be accepted after this cut-off (*i.e.* a 100% penalty will be applied). Please work independently and ensure that the assignment you submit contains your own work.

Assignment Questions

(1) Implement, in a computer language of your choice, the method of local search for solving the following instance of the knapsack problem.

$$\begin{array}{ll} \text{Maximise} & z = 8s_1 + 11s_2 + 9s_3 + 12s_4 + 14s_5 + 10s_6 + 6s_7 + 7s_8 + 13s_9 \\ \text{such that} & s_1 + 2s_2 + 3s_3 + 2s_4 + 3s_5 + 4s_6 + s_7 + 5s_8 + 3s_9 \;\leq\; 16, \\ & \qquad s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8, s_9, s_{10} \;\in\; \{0, 1\}. \end{array}$$

Encode candidate solutions as binary vectors of the form $[s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8, s_9]$, employ single-bit complement moves and use the method of best improvement for neighbour selection.

Submit the following assignment elements:

(a) A carefully commented print-out of your program source code, and

(b) the output produced by your program, containing the same formation as that tabulated during the lecture on Local Search (formatted according to your preference), when applied to the following initial conditions:

i. The vector $\boldsymbol{s}^{(0)} = [0, 1, 1, 1, 0, 0, 1, 1, 1]$,

ii. The vector $\boldsymbol{s}^{(0)} = [0, 1, 0, 1, 0, 0, 0, 1, 0]$.

(2) Implement, in a computer language of your choice, a genetic algorithm for solving a 6-city instance of the TSP with distance matrix

$$\begin{array}{c c} & \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array} & \begin{bmatrix} 0 & 32 & 39 & 42 & 29 & 35 \\ 32 & 0 & 36 & 27 & 41 & 25 \\ 39 & 36 & 0 & 28 & 33 & 40 \\ 42 & 27 & 28 & 0 & 27 & 38 \\ 29 & 41 & 33 & 27 & 0 & 26 \\ 35 & 25 & 40 & 38 & 26 & 0 \end{bmatrix} \end{array}.$$

Encode candidate solutions as permutation vectors of length 6, maintain a population of eight candidate solutions per population, with the initial population generated randomly. Apply 2-point crossover to three pairs of parents per generation, selected by means of tournament selection with a tournament size of three. Also apply mutation to one randomly selected offspring solution, by randomly swapping two of its entries during each generation. Finally, implement an elitism replacement strategy whereby the best eight individuals from the combined parent and mutated offspring populations survive to the next generation, and adopt as stopping criterion algorithmic termination once eight generations have elapsed without having observed an improvement in the incumbent.

Submit the following assignment elements:

(a) A carefully commented print-out of your program source code, and

(b) the output produced by your program for one random population initialisation. This output must contain the following for each generation:

i. The generation number,

ii. the entire population of individuals,

iii. the fitness value of each individual in the population,

iv. an indication of which parent pairs were subjected to crossover,

v. where the cut points were for each crossover operation,

vi. an indication of which offspring solution was subjected to mutation, and

vii. the mean population fitness.

(3) (a) Adapt your program of Question (1) above so as to perform a random walk of length a hundred (feasible) candidate solutions through the fitness landscape of the knapsack problem instance in Question (1) above in which a (feasible) neighbouring solution of any quality is accepted during the walk. Save the objective function values achieved by each candidate solution encountered along this walk and use these data to compute the random walk correlation function value $r(1)$. Also use the latter value to approximate the normalised correlation length ξ of the fitness landscape. Finally, explain how you estimate $\mathrm{diam}(G)$ in the latter calculation.

(b) Let $\mathcal{U}$ be the set of all 36 candidate solutions to the knapsack problem instance in Question (1) above in which exactly two items are packed into the knapsack. Use your program of Question (1) to compute the corresponding set $\mathcal{O}$ of local optima at which the method of local search terminates upon being launched from each of the solutions in the set $\mathcal{U}$. Submit a list of each initial solution in the set $\mathcal{U}$ (together with its objective function value), followed by the length of the corresponding descent walk launched from that initial solution, as well as the local optimum in $\mathcal{O}$ at which the walk terminates (together with its objective function value). Finally, also compute the correlation indicator value of LLM($\mathcal{U}$).

(c) Compute the relative variation ΔAmp of the amplitude between the starting random population $\mathcal{U}$ and the population of local optima $\mathcal{O}$, as well as the value of the average value Gap($\mathcal{O}$) of the relative gaps between the objective function values of solutions in $\mathcal{O}$ and that of the best known solution $\boldsymbol{s}^*$.

(d) Use your results in (a)–(c) above to pronounce, based on the fitness landscape properties, how hard you expect it to be to solve the knapsack problem instance in Question (1) approximately.

Assessment Rubric for Questions 1 and 2

Criterion A: Commenting of program to make it readible/interpretable (Mark awarded: A Marks)

1 Mark	2 Marks	3 Marks	4 Marks	5 Marks
No or virtually no commenting; basically impossible for a marker to figure out the functionality of the different program parts	Existing but poor commenting; leaving the marker to figure out the functionality of the various program parts; although possible to do this, it is very difficult	A serious attempt at commenting, but leaving much to be desired; the onus is on the marker to figure out the working of the program	A good attempt made at commenting, but still leaving the marker to figure out the functionality of some program parts	Excellent, clear comments making it easy to read and interpret the entire implementation

Criterion B: Modularity of independent program parts/procedures/functions (Mark awarded: B Marks)

1 Mark	2 Marks	3 Marks	4 Marks	5 Marks
Little or no modularity; programming structure resembling a bowl of spaghetti; almost impossible to verify various metaheuristic component implementations	Existing but poor modularity, making it difficult to verify various metaheuristic component implementations	An attempt at functional modularity, making it possible to verify some metaheuristic component implementations, but leaves much to be desired terms of sensibility of the scoping of various functions/procedures	Acceptable functional modularity of implementation, but could have been better in terms of facilitating easy verification	Very clear functional modularity of implementation, with sensible scoping of various functions/procedures, making it easy to verify metaheuristic component implementations

Criterion C: Perceived integrity and adherence to specified elements of metaheuristic implementation (Mark awarded: C Marks)

1 Mark	2 Marks	3 Marks	4 Marks	5 Marks
Very serious questions about program integrity and significant non-adherence to metaheuristic component specification	Very serious questions about program integrity or significant non-adherence to metaheuristic component specification (but not both)	No serious questions about implementation integrity, but average coding efficiency of various metaheuristic components, adherence to metaheuristic component specification	No questions about program integrity, acceptable coding efficiency of various metaheuristic components, adherence to metaheuristic component specification	Crisp, high-quality and efficient coding of various metaheuristic components, adherence to metaheuristic component specification, implementation integrity beyond question

Criterion D: Credibility of program numerical results output (Mark awarded: D Marks)

1 Mark	2 Marks	3 Marks	4 Marks	5 Marks
Numerical results fundamentally flawed (Question 1) or seriously dubious (Question 2)	Serious numerical errors (Question 1) or some results doubtful (Question 2)	Acceptable numerical results with few serious errors (Question 1) or few unexpected individuals (Question 2)	Good numerical results, which are virtually error-free (Question 1) or with high-quality incumbent (Question 2)	Excellent, error free numerical results (Question 1) or conforming narrowly with expected results and with an incumbent very close to optimal (Question 2)

Final mark awarded for question (out of 40): $A + 2B + 3C + 2D$

Assessment Rubric for Question 3

Question 3(a) (Mark awarded: A' Marks)				
1 Mark	**2 Marks**	**3 Marks**	**4 Marks**	**5 Marks**
No real attempt at computing $r(1)$ or ξ values	Flawed attempt at computing the values of $r(1)$ and ξ	Acceptable attempt at computing the values of $r(1)$ and ξ, but with errors	Correct calculation of $r(1)$ value, but problematic calculation of ξ, either because of a poor approximate value of $\mathrm{diam}(G)$ or a good value of this parameter but this value being poorly motivated	Excellent attempt at computing $r(1)$ and ξ, including a good motivation for the choice of a value for $\mathrm{diam}(G)$

Question 3(b) (Mark awarded: B' Marks)				
1 Mark	**2 Marks**	**3 Marks**	**4 Marks**	**5 Marks**
No real attempt at computing descent path lengths or $\mathrm{LLM}(\mathcal{U})$ value	Flawed attempt at computing descent path lengths and $\mathrm{LLM}(\mathcal{U})$ value	Acceptable attempt at computing descent path lengths and $\mathrm{LLM}(\mathcal{U})$ value, but with multiple errors	Correct calculation of $\mathrm{LLM}(\mathcal{U})$ value, but based on incorrect descent path lengths	Flawless computation of descent path lengths and value of $\mathrm{LLM}(\mathcal{U})$

Question 3(c) (Mark awarded: C' Marks)				
1 Mark	**2 Marks**	**3 Marks**	**4 Marks**	**5 Marks**
No real attempt at computing either of the values of $\Delta\mathrm{Amp}$ or $\mathrm{Gap}(\mathcal{O})$	Attempt at computing one of the values $\Delta\mathrm{Amp}$ or $\mathrm{Gap}(\mathcal{O})$, but with errors in the other	Acceptable attempt at computing both the $\Delta\mathrm{Amp}$ value and the $\mathrm{Gap}(\mathcal{O})$ value, but with some errors	Flawless computation of $\Delta\mathrm{Amp}$ value or $\mathrm{Gap}(\mathcal{O})$ value, but not both	Flawless computation of both $\Delta\mathrm{Amp}$ and $\mathrm{Gap}(\mathcal{O})$ values

Question 3(d) (Mark awarded: D' Marks)				
1 Mark	**2 Marks**	**3 Marks**	**4 Marks**	**5 Marks**
No real attempt at interpreting the results of parts (a)–(c)	Interpretation of the results of parts (a)–(c) unclear or only incorrect	Interpretation of the results of parts (a)–(c) unclear or only partly correct	Interpretation of the results of parts (a)–(c) of question correct, but interpretation lacking insight	Crisp interpretation of the results of parts (a)–(c) of question, and interpretation demonstrating insight

Final mark awarded for question (out of 40): $3A' + 2B' + C' + 2D'$