FINAL PROJECT ASSIGNMENT

MIS41260 - Metaheuristics

UCD Michael Smurfit
Graduate Business School

Authors

Apoorva Kulkarni (20200244)
Siva Thirumavalavan (20200235)

Assessment Submission Form

Student Name	Apoorva Kulkarni, Siva Thirumavalavan
Student Number	20200244, 20200235
Assessment Title	Final Project Assignment
Module Code	MIS41260
Module Title	Decision Model and Metaheuristics
Module Co-ordinator	Prof. Rafael Marti
Tutor (if applicable)	
Date Submitted	06-08-2021
Date Received	
Grade/Mark	

A SIGNED COPY OF THIS FORM MUST ACCOMPANY ALL SUBMISSIONS FOR ASSESSMENT.

STUDENTS SHOULD KEEP A COPY OF ALL WORK SUBMITTED.

Procedures for Submission and Late Submission

Ensure that you have checked the school's procedures for the submission of assessments.
Note: There are penalties for the late submission of assessments. For further information please see the University's ***Policy on Late Submission of Coursework***, (http://www.ucd.ie/registrar/)

Plagiarism: the unacknowledged inclusion of another person's writings or ideas or works, in any formally presented work (including essays, examinations, projects, laboratory reports or presentations). The penalties associated with plagiarism designed to impose sanctions that reflect the seriousness of University's commitment to academic integrity. Ensure that you have read the University's ***Briefing for Students on Academic Integrity and Plagiarism and the UCD Plagiarism Statement, Plagiarism Policy and Procedures,*** (http://www.ucd.ie/registrar/)

Declaration of Authorship

I declare that all material in this assessment is my own work except where there exists a clearly stated acknowledgement and appropriate reference to the work of others.

Signed: Date: 06-08-2021

Contents

List of figures

1. Abstract

The Maximum Diversity Problem refers to the selection of a subset of m elements from a set of n elements such that the sum of distances between individual elements in the subset is maximum. This problem, introduced by Glover, Hersh and McMillian, has been studied using different methodologies mentioned in this report. The purpose of this report is to compare the working of Tabu search algorithm (built in R) and GRASP (using excel) with the 5 instances provided. We have experimented with different tabu tenure parameters and documented corresponding diversity and run time. Results show that Tabu Search (built in R) achieves both better results and much shorter computational times with respect to those reported for GRASP (using Excel).

2. Problem Description

Here we are considering finding the best solution for a Maximum Diversity Problem (MDP). An MDP refers to the selection of a subset of m elements from a set of n elements such that the sum of distances between individual elements in the subset is maximum.

Let $x_i = 1$ if element $i \in N$ belongs to the solution M, $x_i = 0$ otherwise. The MDP can be formulated as follows:

$$\max z = \frac{1}{2}\sum_{i\in N}\sum_{j\in N} d_{ij}x_i x_j$$

$$\sum_{i\in N} x_i = m$$

$$x_i \in \{0,1\} \quad i \in N$$

Where d_{ij} is the distance between two elements i and j such that $i \neq j$.

The problem of maximizing diversity deals with selecting a set of elements from some larger collection such that the selected elements exhibit the greatest variety of characteristics (Kuo et al., 1993).

Despite its fundamental importance, the maximum diversity problem is largely unexplored in the literature; research on maximizing diversity and related subjects has been sparse as compared to other optimisation problems. We come across a variety of applications for the Maximum Diversity Problem some of which include combating diseases by creating programs with more diverse line of defense, maintaining social balance by promoting diverse ethnicities, maximizing product diversity to enhance product design amongst others (Martí et al., 2011).

Here we aim to select 40 elements from a set of 500 elements such that sum of the individual distances between the elements of our subset is maximized to ensure maximum diversity in the selection.

3. Methods to solve MDP

3.1 SA – Simulated Annealing

Simulated Annealing is a probabilistic approach for estimating the global optimum of a given function. It is a metaheuristic to approximate global optimization in a large search space for an optimization problem and is frequently used when the search space is discrete.

3.2 GRASP – Greedy Randomized Adaptive Search Procedure

GRASP is a metaheuristic method that is widely used to solve combinatorial optimization issues. Iterations are generally made up of repeated builds of a greedy randomized solution and subsequent iterative refinements to it via a local search.

3.3 TS – Tabu Search

Tabu search is a metaheuristic search approach that employs local search methods often employed in mathematics optimization. By loosening its core rule, Tabu search improves the performance of local search. First, if no improving move is offered at each phase, deteriorating movements might be accepted (like when the search is stuck at a strict local minimum). Prohibitions are also implemented to dissuade the search from returning to previously visited solutions. It uses memory structures that stores the visited solutions or user-supplied sets of rules. If a possible solution has been visited earlier within a specific short-term time, or if it has violated a rule, it is marked as "tabu" (forbidden) so that the algorithm does not examine it again.

3.4 VNS – Variable Neighborhood Search

Variable Neighborhood Search (VNS) is a metaheuristic approach for addressing a collection of combinatorial and global optimization problems. It investigates remote neighborhoods of the present incumbent solution and progresses to a new one if and only if an improvement is made. To go from solutions in the neighborhood to local optima, the local search approach is used frequently. VNS was created for approximating solutions to discrete and continuous optimization issues, and it is aimed at tackling linear program problems, integer program problems, mixed integer program problems, nonlinear program problems etc.

3.5 SS – Scatter Search

Scatter search (SS) is an evolutionary or population-based approach for exploring solution space by developing a collection of reference solutions stored in a reference set (RefSet). The use of four techniques causes the reference set to evolve: subset creation, update, combination, and improvement, where the first two have standard implementations but the final two must be created for each unique situation.

3.6 MA – Memetic Algorithm

Memetic algorithm is an extension on the standard genetic algorithm. It employs a local search strategy to limit the possibility of early convergence. Memetic algorithms are one of the most recent and rapidly expanding fields of research in evolutionary computation. MA is frequently used to describe a combination of evolutionary or other population-based method with distinct individual learning or local improvement procedures for issue solving.

4. TABU Search Implementation

We implemented a straightforward and efficient Tabu Search algorithm. We have one primary parameter - Tabu Tenure, and two secondary parameters - Iteration Count and Stopping Criteria.

For each instance, we start by generating a random solution and then calculating the diversity. We refer to it as the current best solution. We check for the neighborhood of each element of the current solution and evaluate possible moves for a given number of iterations. If we find a move that improves our current solution, we apply it and update our best solution. If no such move exists, we perform the best of the available moves, even if it reduces the current diversity. The performed move is saved in a Tabu list. In subsequent iterations, we check to see if a specific move is stored in the tabu list and, if so, we refrain from performing it. This is to ensure that we do not return to a previously visited neighborhood. The Tabu Tenure is the number of iterations during which we will not repeat a performed move. We also monitor the improvements to our solution, and if no progress is made after a certain number of iterations, we stop the process and move on to the next element in the solution space – which is defined by the Stopping Criteria.

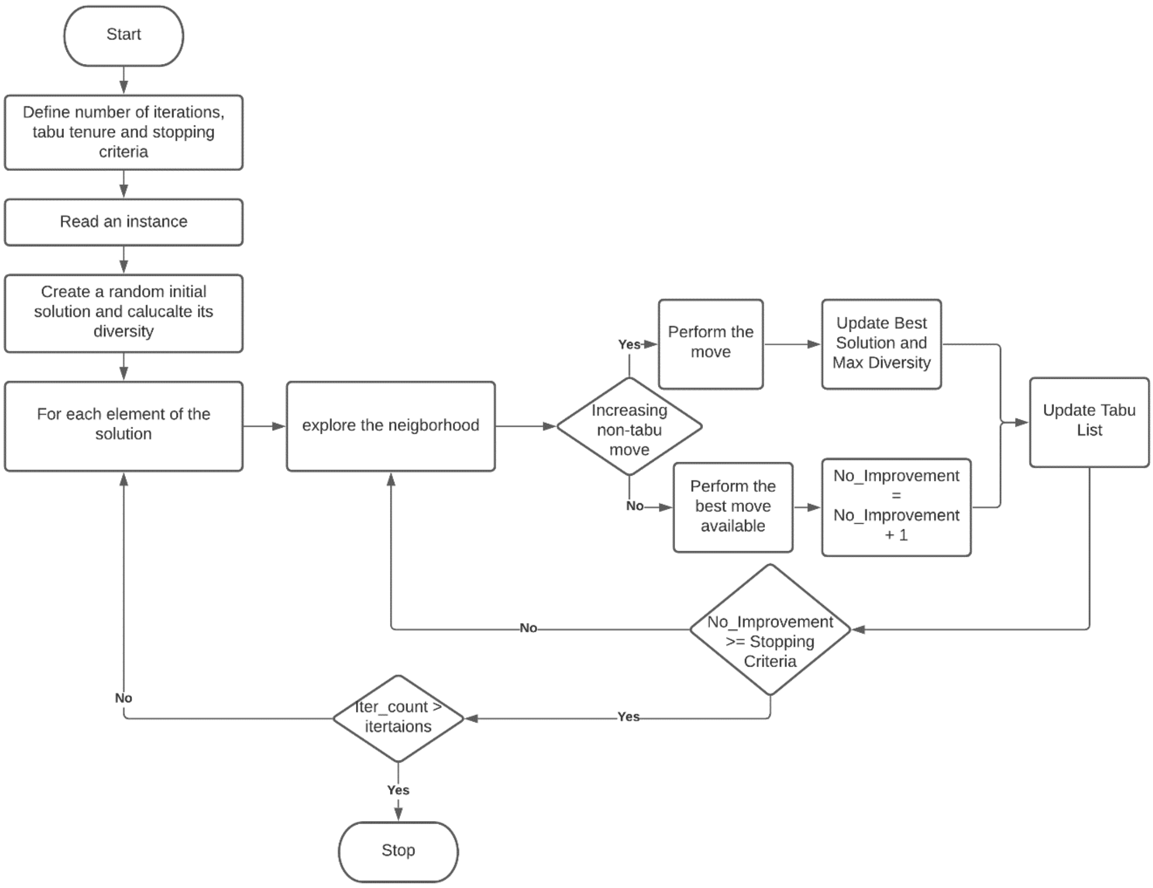

Figure 1: Flow Diagram

5. Empirical Comparison

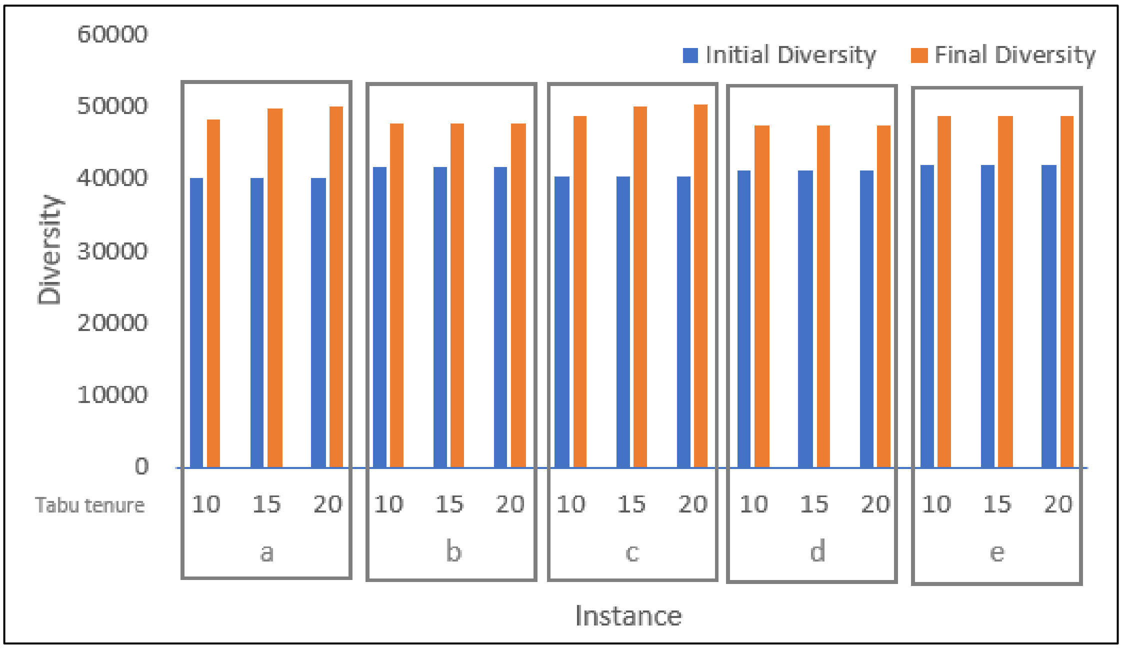

Figure 2: Tabu Search results

We built our Tabu search algorithm in R. For each instance, we fixed an initial solution and ran the Tabu search algorithm for different Tabu tenures (5, 10, and 15) to see how they affected the improvement. We ran the algorithm for 20 iterations and gave the stopping criteria as 5000 (Sub-iterations). According to the findings, increasing the tenure leads to an increase in diversity but at the expense of run time.

The experiment was performed in an Intel Core i7-8550U CPU @ 1.80GHz with 8 GB of RAM and Windows 10 v20H2 64 bits OS.

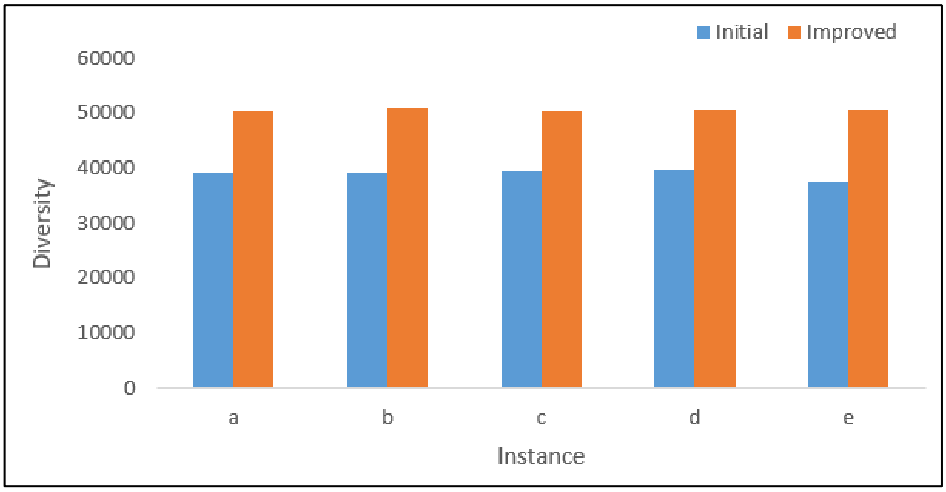

Figure 3: GRASP results

We used VBA to create the GRASP algorithm. During the construction phase, we create an initial solution. We pick one element at a time and add it to the current partial solution until it has 40 elements. As the new element's contribution to the objective's value is partly unknown, the algorithm computes a lower and upper estimate, generates a random value uniformly distributed between them, and selects the element for which this value is maximum. We restrict the new element's selection to a Restricted Candidate List (RCL). The GRASP equation is given by

$$c^{min} \leq c(e) \leq c^{min} + \alpha(c^{max} - c^{min})$$

Where α is the RCL parameter with a value between 0 and 1. 0 represents a pure greedy construction, while 1 represents a pure random construction. We set alpha to 0.5 to avoid being too greedy or too random.

Instance	Initial diversity	Improved diversity
a	39051	50337
b	39146	51043
c	39346	50425
d	39817	50532
e	37596	50658

Table 1: GRASP results

While GRASP produced results similar to Tabu search, the runtime of Tabu search was significantly faster. Tabu also allows for a trade-off between runtime and diversity by adjusting Tabu Tenure and Stopping Criteria.

6. Conclusion

Our goal was to implement Tabu search on a set of five instances. We implemented the algorithm in R and compared the results to GRASP in terms of final diversity and runtime.

As we increase the tabu tenure, the diversity increases but at the cost of an increased runtime (for most of the instances). But even with the increased run time, we were able to achieve better results in a shorter time compared to GRASP.

7. References

Aringhieri, R., Cordone, R. and Melzani, Y. (2007). Tabu Search versus GRASP for the maximum diversity problem. *4OR*, 6(1), pp.45–60.

Duarte, A. and Marti, R. (2007). *Tabu search and GRASP for the maximum diversity problem | Request PDF*. [online] ResearchGate. Available at: https://www.researchgate.net/publication/222817653_Tabu_search_and_GRASP_for_the_maximum_diversity_problem.

Kuo, C.C., Glover, F. and Dhir, K. (1993). *Download Limit Exceeded*. [online] citeseerx.ist.psu.edu. Available at: https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.418.1678&rep=rep1&type=pdf [Accessed 30 Jul. 2021].

Martí, R., Gallego, M., Duarte, A. and Pardo, E.G. (2011). Heuristics and metaheuristics for the maximum diversity problem. *Journal of Heuristics*, 19(4), pp.591–615.

Wikipedia (2020). *Greedy randomized adaptive search procedure*. [online] Wikipedia. Available at: https://en.wikipedia.org/wiki/Greedy_randomized_adaptive_search_procedure [Accessed 6 Aug. 2021].