# Question 1

## Tabu Search Setup

A tabu search algorithm was developed to solve the quadratic assignment problem (QAP). The program first parses the flow and distance data, then initializes the following: a candidates list via a priority queue of pairs of costs and department layouts, and a tabu list via a map of department layouts and tabu tenures. The stopping criterion for the program is iterations, and for each iteration, it performs the tabu search. It randomly selects five candidate solutions from the entire neighbourhood and evaluates each layout to determine associated costs. The costs are computed by multiplying the appropriate flows and distances for each department based on their positions in the layout. These costs and layouts are then pushed to the priority queue, which automatically sorts by lowest cost from the top. Next, the tabu list is updated, where all candidates in the tabu list have their tenures reduced by one, and if a tenure reaches zero, that candidate is removed. Then, any candidate solutions that appear in the tabu list are discarded, since tabu search skips previously visited solutions in the list to escape local optima. Now, the candidate at the top of the priority queue is the next best solution, so add this to the tabu list with a tabu tenure of five. Lastly, the program checks if this solution is better than the current best solution, and if so, replace the best layout and best cost solutions with this one. The costs and candidate list are reset, and the loop repeats until the termination criteria is met.

## Tabu Search Results

The program was run various times with iterations, a candidate list of size , and a tabu tenure of . The initial solution was random every time, the neighbourhood was the entire neighbourhood, and no aspiration criteria was used. A sample result is shown below:

|  |  |
| --- | --- |
| **Layout** | **Cost** |
| 17 4 20 7 6  19 15 8 5 3  2 11 16 1 13  14 18 12 10 9 | 2,792 |

The obtained results varied from as low as to as high as , depicted in the next section on the tabu search modifications. Overall, the results appear to be reasonable and relatively close to the optimal solution of . This problem is NP-hard, and as such, does not have a polynomial time solution. Since there are 20 locations and the problem is encoded as a permutation setup, the total amount of solutions is 20 factorial. This amounts to a solution space of an order of magnitude of . Thus, with such a large solution space, one could only achieve the optimal solution with a different candidate selection strategy or modifying the tabu setup parameters such as the termination criteria or tabu list size.

## Tabu Search Modifications

The initial starting point was randomized and the tabu search was executed ten times:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Run** | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| **Cost** | 2,794 | 2,908 | 2,878 | 2,858 | 2,850 | 2,830 | 2,880 | 2,870 | 2,850 | 2,836 |

The tabu list size was changed twice: once to be smaller and once to be larger:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Smaller** | 3,090 | 3,076 | 3,146 | 3,062 | 3,072 | 3,074 | 3,026 | 3,046 | 3,048 | 3,012 |
| **Larger** | 2,776 | 2,768 | 2,788 | 2,810 | 2,770 | 2,784 | 2,699 | 2,780 | 2,776 | 2,806 |

The tabu list size was changed to be a dynamic one, chosen randomly between a range:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Cost** | 2,654 | 2,678 | 2,688 | 2,610 | 2,680 | 2,680 | 2,670 | 2,610 | 2,708 | 2,654 |

Two aspiration criteria were added, as stated in the question, for two experiments:

|  |  |  |
| --- | --- | --- |
| **Best Solution So Far** | 2,784 | 2,780 |
| **Best Solution in Neighbourhood** | 3,102 | 3,000 |

Only a section of the neighbourhood was used rather than the whole neighbourhood:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Cost** | 2,688 | 2,666 | 2,672 | 2,676 | 2,718 | 2,636 | 2,684 | 2,692 | 2,646 | 2,702 |

Lastly, a frequency based tabu list was added to encourage search to diversify:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Cost** | 3,070 | 3,052 | 3,050 | 3,066 | 3,078 | 3,106 | 3,094 | 3,056 | 3,050 | 3,090 |

Analyzing all the modifications above, the trend for my tabu search appears to be that all intensification techniques improve the solution, whereas diversification techniques worsen it. In particular, increasing the size of the tabu list, using a dynamic tabu list size, and only using a section of the neighbourhood as opposed to the whole neighbourhood, all resulted in improved cost values. As shown, the initial cost range of roughly improved to around . Conversely, decreasing the size of the tabu list and utilizing a frequency based tabu list led to increased cost values. The cost from the initial range of worsened to roughly . The outlying result is the aspiration criteria for choosing the best solution in the neighbourhood, as this is an intensification technique, however, it significantly increased the cost.

As mentioned, the behaviour of my tabu search appears to be that intensifying improves it while diversifying worsens it. Using the domain and scope of the problem, this is attributed to the fact that the problem has such a large solution space. Even with iterations, it is unable to reach the most optimal solution of , however, it does get close, with the best solutions being in the range. To further emphasize, the solution space is 20 factorial, which results in a massive search space, and with only a limited number of iterations, it would take many attempts before eventually attaining the most optimal solution. As such, intensification seems to help my tabu search because it allows my program to exploit small portions of the search space, which is necessary for a problem of this size. It is almost as if the program needs to first get lucky to attain a favourable arrangement, and then focus in on that, only making minor modifications to reach the optimal solution. The solution space is too large for diversification relative to the number of iterations, so it is desirable to initially reach a suitable arrangement, and then home in on that specific setup.