DISCRETE OPTIMIZATION

Megan Bryant
mrbryant@email.wm.edu

Department of Mathematics, The College of William and Mary, P.O. Box 8795, Williamsburg, VA 23187

Executive Summary

The files you will need to implement the simulated annealing code from the Numerical Recipes text may be downloaded from Dr. Rex Kincaid's homepage: http://www.math.wm.edu/rrkinc.

A copy of this section of Numerical Recipes is available at: http://www.nrbook.com/a/bookcpdf.php.

A copy of 'Optimization by Simulated Annealing: An Experimental Evaluation; Part 1, Graph Partitioning' by Johnson et. al. is available at: http://vis.lbl.gov/ aragon/pubs/annealing-pt1.pdf and http://vis.lbl.gov/ aragon/pubs/annealing-pt2.pdf.

1

Consider the minimum cost spanning tree problem pictured in Figure 4 of the Glover article from Interfaces. Using the starting solution given perform four iterations of tabu search with a tabu list of length three containing dropped edges only and an aspiration criterion of the best cost yet found.

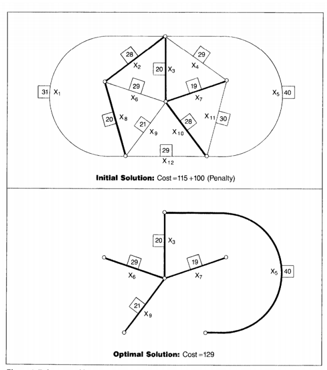


Figure 4: Relevance of longer-term memory and diversification minimum-cost tree problem. Added constraints are $X_9 \leqslant X_7$, $X_3 + X_7 \leqslant 2X_5$. (Unit violation penalty = 50.)

Tabu Search Parameters

Starting Solution:	(0,1,1,0,0,0,1,1,0,0,1,0,0)
Constraints:	$x_9 \leq x_7$
	$x_3 + x_7 \leq 2x_5$
Objective(w/Constraints):	$115 + 2 \times 50 = 215$
Move:	Edge Swap (Drop/Add)
Aspiration Criteria:	Best Yet Seen Objective

Starting Solution: (0,1,1,0,0,0,1,1,0,1,0,0)
Objective(w/Constraints): 215

Table 1: Iteration 1

Iteration 1	1	2	3	4	5	6	7	8	9	10	11	12	Obj	C1	C2	Delta
Starting	0	1	1	0	0	0	1	1	0	1	0	0	115	115	215	100
drop 2, add 1	1	0	1	0	0	0	1	1	0	1	0	0	118	118	218	103
drop 2 add 4	0	0	1	1	0	0	1	1	0	1	0	0	116	116	216	101
drop 2 add 5	0	0	1	0	1	0	1	1	0	1	0	0	127	127	**127**	**12**
drop 2 add 6	0	0	1	0	0	1	1	1	0	1	0	0	116	116	216	101
drop 2 add 9	0	0	1	0	0	0	1	1	1	1	0	0	108	108	208	93
drop 2 add 11	0	0	1	0	0	0	1	1	0	1	1	0	117	117	217	102
drop 2 add 12	0	0	1	0	0	0	1	1	0	1	0	1	116	116	216	101
drop 3, add 1	1	1	0	0	0	0	1	1	0	1	0	0	126	126	176	61
drop 3 add 4	0	1	0	1	0	0	1	1	0	1	0	0	124	124	174	59
drop 3 add 5	0	1	0	0	1	0	1	1	0	1	0	0	135	135	135	20
drop 3 add 6	0	1	0	0	0	1	1	1	0	1	0	0	124	124	174	59
drop 3 add 9	0	1	0	0	0	0	1	1	1	1	0	0	116	116	166	51
drop 3 add 11	0	1	0	0	0	0	1	1	0	1	1	0	125	125	175	60
drop 3 add 12	0	1	0	0	0	0	1	1	0	1	0	1	124	124	174	59
drop 7, add 1	1	1	1	0	0	0	0	1	0	1	0	0	127	127	177	62
drop 7 add 4	0	1	1	1	0	0	0	1	0	1	0	0	125	125	175	60
drop 7 add 5	0	1	1	0	1	0	0	1	0	1	0	0	136	136	136	21
drop 7 add 6	0	1	1	0	0	1	0	1	0	1	0	0	125	125	175	60
drop 7 add 9	0	1	1	0	0	0	0	1	1	1	0	0	117	167	217	102
drop 7 add 11	0	1	1	0	0	0	0	1	0	1	1	0	126	126	176	61
drop 7 add 12	0	1	1	0	0	0	0	1	0	1	0	1	125	125	175	60
drop 8, add 1	1	1	1	0	0	0	1	0	0	1	0	0	126	126	226	111
drop 8 add 4	0	1	1	1	0	0	1	0	0	1	0	0	124	124	224	109
drop 8 add 5	0	1	1	0	1	0	1	0	0	1	0	0	135	135	135	20
drop 8 add 6	0	1	1	0	0	1	1	0	0	1	0	0	124	124	224	109
drop 8 add 9	0	1	1	0	0	0	1	0	1	1	0	0	116	116	216	101
drop 8 add 11	0	1	1	0	0	0	1	0	0	1	1	0	125	125	225	110
drop 8 add 12	0	1	1	0	0	0	1	0	0	1	0	1	124	124	224	109
drop 10, add 1	1	1	1	0	0	0	1	1	0	0	0	0	118	118	218	103
drop 10 add 4	0	1	1	1	0	0	1	1	0	0	0	0	116	116	216	101
drop 10 add 5	0	1	1	0	1	0	1	1	0	0	0	0	127	127	**127**	**12**
drop 10 add 6	0	1	1	0	0	1	1	1	0	0	0	0	116	116	216	101
drop 10 add 9	0	1	1	0	0	0	1	1	1	0	0	0	108	108	208	93
drop 10 add 11	0	1	1	0	0	0	1	1	0	0	1	0	117	117	217	102
drop 10 add 12	0	1	1	0	0	0	1	1	0	0	0	1	116	116	216	101

Tabu List

Add	Drop
10	5

Table 2: Table 2

Iteration 2	1	2	3	4	5	6	7	8	9	10	11	12	Obj	C1	C2	Delta
Starting	0	1	1	0	1	0	1	1	0	0	0	0	127	127	127	0
Drop 2, add 1	1	0	1	0	1	0	1	1	0	0	0	0	130	130	130	3
drop 2 add 4	0	0	1	1	1	0	1	1	0	0	0	0	128	128	128	1
drop 2 add 6	0	0	1	0	1	1	1	1	0	0	0	0	128	128	128	1
drop 2 add 9	0	0	1	0	1	0	1	1	1	0	0	0	120	120	**120**	**-7**
drop 2 add 11	0	0	1	0	1	0	1	1	0	0	1	0	129	129	129	2
drop 2 add 12	0	0	1	0	1	0	1	1	0	0	0	1	128	128	128	1
Drop 3, add 1	1	1	0	0	1	0	1	1	0	0	0	0	138	138	138	11
drop 3 add 4	0	1	0	1	1	0	1	1	0	0	0	0	136	136	136	9
drop 3 add 6	0	1	0	0	1	1	1	1	0	0	0	0	136	136	136	9
drop 3 add 9	0	1	0	0	1	0	1	1	1	0	0	0	128	128	128	1
drop 3 add 11	0	1	0	0	1	0	1	1	0	0	1	0	137	137	137	10
drop 3 add 12	0	1	0	0	1	0	1	1	0	0	0	1	136	136	136	9
Drop 7, add 1	1	1	1	0	1	0	0	1	0	0	0	0	139	139	139	12
drop 7 add 4	0	1	1	1	1	0	0	1	0	0	0	0	137	137	137	10
drop 7 add 6	0	1	1	0	1	1	0	1	0	0	0	0	137	137	137	10
drop 7 add 9	0	1	1	0	1	0	0	1	1	0	0	0	129	179	179	52
drop 7 add 11	0	1	1	0	1	0	0	1	0	0	1	0	138	138	138	11
drop 7 add 12	0	1	1	0	1	0	0	1	0	0	0	1	137	137	137	10
Drop 8, add 1	1	1	1	0	1	0	1	0	0	0	0	0	138	138	138	11
drop 8 add 4	0	1	1	1	1	0	1	0	0	0	0	0	136	136	136	9
drop 8 add 6	0	1	1	0	1	1	1	0	0	0	0	0	136	136	136	9
drop 8 add 9	0	1	1	0	1	0	1	0	1	0	0	0	128	128	128	1
drop 8 add 11	0	1	1	0	1	0	1	0	0	0	1	0	137	137	137	10
drop 8 add 12	0	1	1	0	1	0	1	0	0	0	0	1	136	136	136	9
Aspiration																
drop 5 add 1	1	1	1	0	0	0	1	1	0	0	0	0	118	118	218	91
drop 5 add 4	0	1	1	1	0	0	1	1	0	0	0	0	116	116	216	89
drop 5 add 6	0	1	1	0	0	1	1	1	0	0	0	0	116	116	216	89
drop 5 add 9	0	1	1	0	0	0	1	1	1	0	0	0	108	108	208	81
drop 5 add 10	0	1	1	0	0	0	1	1	0	1	0	0	115	115	215	88
drop 5 add 11	0	1	1	0	0	0	1	1	0	0	1	0	117	117	217	90
drop 5 add 12	0	1	1	0	0	0	1	1	0	0	0	1	116	116	216	89
drop 3 add 10	0	1	0	0	1	0	1	1	0	1	0	0	135	135	135	8
drop 7 add 10	0	1	1	0	1	0	0	1	0	1	0	0	136	136	136	9
drop 8 add 10	0	1	1	0	1	0	1	0	0	1	0	0	135	135	135	8
drop 2 add 10	0	0	1	0	1	0	1	1	0	1	0	0	127	127	127	0

Tabu List

Add	Drop
10, 2	5, 9

Table 3: Iteration 3

Iteration 3	1	2	3	4	5	6	7	8	9	10	11	12	Obj	C1	C2	Delta
Starting	0	0	1	0	1	0	1	1	1	0	0	0	120	120	120	0
drop 3 add 1	1	0	0	0	1	0	1	1	1	0	0	0	131	131	131	11
drop 3 add 4	0	0	0	1	1	0	1	1	1	0	0	0	129	129	129	9
drop 3 add 6	0	0	0	0	1	1	1	1	1	0	0	0	129	129	129	9
drop 3 add 11	0	0	0	0	1	0	1	1	1	0	1	0	130	130	130	10
drop 3 add 12	0	0	0	0	1	0	1	1	1	0	0	1	129	129	129	9
Drop 7, add 1	1	0	1	0	1	0	0	1	1	0	0	0	132	182	182	62
drop 7 add 4	0	0	1	1	1	0	0	1	1	0	0	0	130	180	180	60
drop 7 add 6	0	0	1	0	1	1	0	1	1	0	0	0	130	180	180	60
drop 7 add 11	0	0	1	0	1	0	0	1	1	0	1	0	131	181	181	61
drop 7 add 12	0	0	1	0	1	0	0	1	1	0	0	1	130	180	180	60
Drop 8, add 1	1	0	1	0	1	0	1	0	1	0	0	0	131	131	131	11
drop 8 add 4	0	0	1	1	1	0	1	0	1	0	0	0	129	129	129	9
drop 8 add 6	0	0	1	0	1	1	1	0	1	0	0	0	129	129	129	9
drop 8 add 11	0	0	1	0	1	0	1	0	1	0	1	0	130	130	130	10
drop 8 add 12	0	0	1	0	1	0	1	0	1	0	0	1	129	129	129	9
Aspiration																
drop 3 add 2	0	1	0	0	1	0	1	1	1	0	0	0	128	128	**128**	**8**
drop 3 add 10	0	0	0	0	1	0	1	1	1	1	0	0	128	128	128	8
drop 7 add 2	0	1	1	0	1	0	0	1	1	0	0	0	129	179	179	59
drop 7 add 10	0	0	1	0	1	0	0	1	1	1	0	0	129	179	179	59
drop 8 add 2	0	1	1	0	1	0	1	0	1	0	0	0	128	128	128	8
drop 8 add 10	0	0	1	0	1	0	1	0	1	0	1	0	130	130	130	10
drop 5 add 1	1	0	1	0	0	0	1	1	1	0	0	0	111	111	211	91
drop 5 add 4	0	0	1	1	0	0	1	1	1	0	0	0	109	109	209	89
drop 5 add 6	0	0	1	0	0	1	1	1	1	0	0	0	109	109	209	89
drop 5 add 10	0	0	1	0	0	0	1	1	1	1	0	0	108	108	208	88
drop 5 add 11	0	0	1	0	0	0	1	1	1	0	1	0	110	110	210	90
drop 5 add 12	0	0	1	0	0	0	1	1	1	0	0	1	109	109	209	89
drop 9 add 1	1	0	1	0	1	0	1	1	0	0	0	0	130	130	130	10
drop 9 add 4	0	0	1	1	1	0	1	1	0	0	0	0	128	128	128	8
drop 9 add 6	0	0	1	0	1	1	1	1	0	0	0	0	128	128	128	8
drop 9 add 10	0	0	1	0	1	0	1	1	0	1	0	0	127	127	127	7
drop 9 add 11	0	0	1	0	1	0	1	1	0	0	1	0	129	129	129	9
drop 9 add 12	0	0	1	0	1	0	1	1	0	0	0	1	128	128	128	8

Tabu List

Add	Drop
10, 3	5, 9, 2

We see that using the Tabu Search methodology that we obtain a best scene objective value of 120 in Iteration 2 with a solution of (0,0,1,0,1,0,1,1,1,0,0,0). In Iteration 3, since there are no new best objective values to choose from as we have reached a local optimum, we can instead choose a tabu move meeting the aspiration criteria. In this case, we will choose the move of dropping 3 and adding two, which will result in a new, never seen objective value of 128.

2

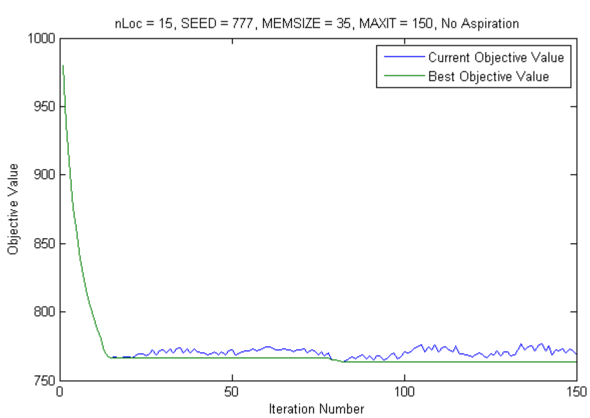
Download the tabu search code for the P-median problem and a 100×100 *shortest path distance 'matrix spd100.unfrm' from 'rrkinc/homework/'. The C code generates an initial median solution randomly using Parks random number generator and a subroutine called CONFIG.*
Run the following 4 computational experiments.

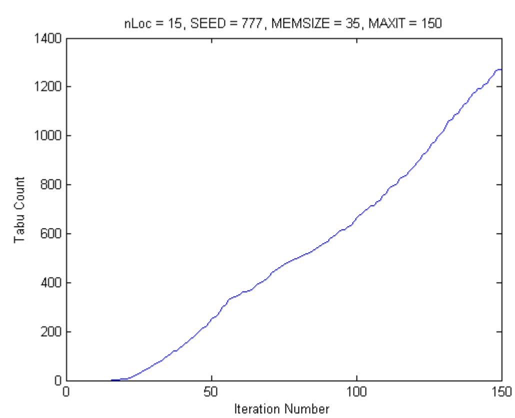
1.) $nLoc = 15, SEED = 777, MEMSIZE = 35$, and $MAXIT = 150$.

The following results were recorded:

Best Objective Value:	762.95
Iteration Number of Best Objective:	82
Path:	51 5 34 96 48 26 13 49 64 86 79 71 31 78 83
Short Term Memory Count:	1273
Aspiration Count:	1

Figure 1: Objective Values and Tabu Counts versus Iteration Number



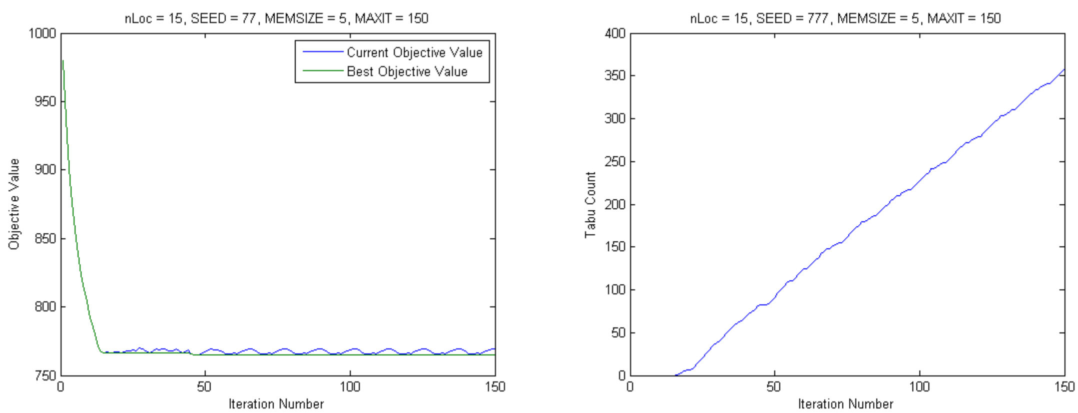


2.) $nLoc = 15, SEED = 777, MEMSIZE = 5$, and $MAXIT = 150$.

The following results were recorded:

Best Objective Value:	764.702
Iteration Number of Best Objective:	47
Path:	51 11 34 96 48 26 13 49 64 31 79 67 1 78 83
Short Term Memory Count:	359
Aspiration Count:	1

Figure 2: Objective Values and Tabu Counts versus Iteration Number

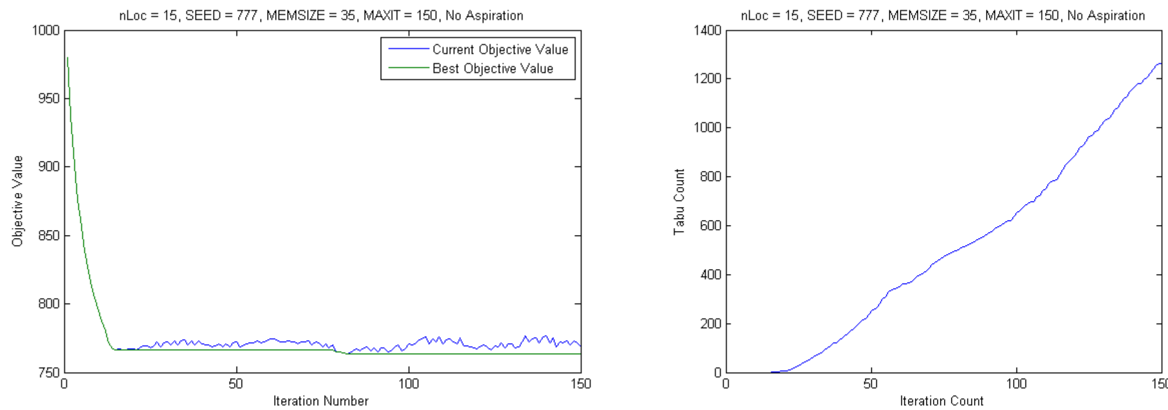


3.) $nLoc = 15, SEED = 777, MEMSIZE = 35$, $MAXIT = 150$, and remove the lines of code that check the aspiration criterion.

The following results were recorded:

Best Objective Value:	763.815
Iteration Number of Best Objective:	82
Path:	51 5 34 96 50 26 13 49 64 86 79 71 31 78 83
Short Term Memory Count:	1263
Aspiration Count:	0

Figure 3: Objective Values and Tabu Counts versus Iteration Number

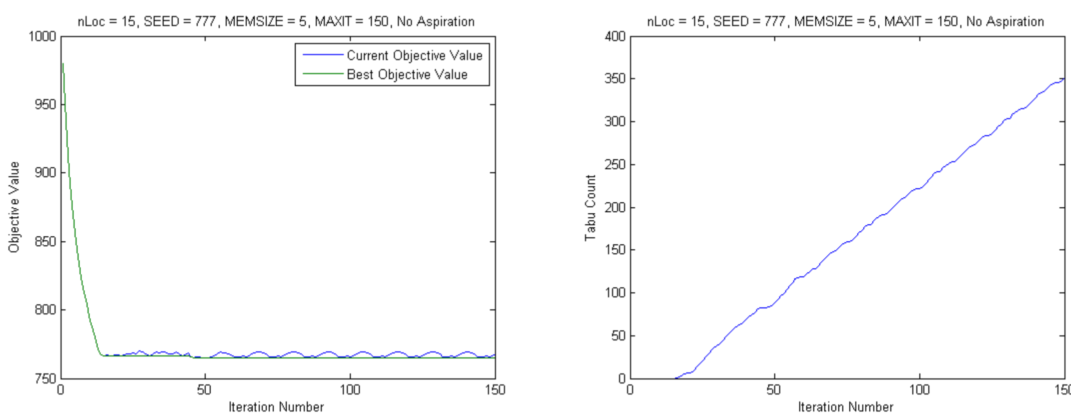


4.) $nLoc = 15, SEED = 777, MEMSIZE = 5$, $MAXIT = 150$, and remove the lines of code that check the aspiration criterion.

The following results were recorded:

Best Objective Value:	764.702
Iteration Number of Best Objective:	50
Path:	51 11 34 96 48 26 13 49 64 31 79 67 1 78 83
Short Term Memory Count:	352
Aspiration Count:	0

Figure 4: Objective Values and Tabu Counts versus Iteration Number

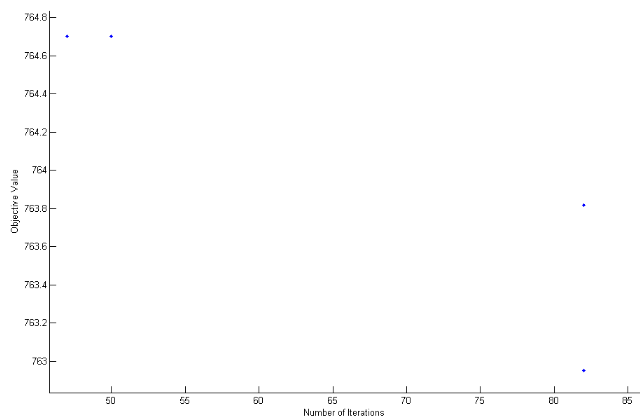


We see in Figure 5 that there was a decrease in objective value that corresponded to an increase in iteration number. These more minimal objective values occurred when the memory size parameter MEMSIZE was set to 35 as opposed to when it was set to 5 for the higher objective values.

Furthermore, the most minimal objective was obtained when the aspiration criteria was included.

In fact, when the parameters were held constant, more minimal objective values were obtained when the aspiration criteria was included. This fact, along with the aforementioned effect of varying the memory size parameter, leads us to conclude that in order to obtain better quality results, memory size should be increased and aspiration criteria should be included.

Figure 5: Comparison Plot



When the MEMSIZE =35, it took 80 iterations until the first tabu move met aspiration criteria. This is 33 iterations longer than when the MEMSIZE =5 and the first tabu move met the aspiration criteria at 47. Only one move met the aspiration criteria in either run.

Table 4: First Local Optimum

Run	Iteration	First Local Optimum
1	15	766.725
2	15	766.725
3	15	766.725
4	15	766.725

Table 4 shows us that for each run, the first local optimum that is found occurs at iteration 15 with a value of 766.725. Again, this is because all four runs are conducted with the same seed value. Table 4 also shows that all four runs reached the local optimum at the same time and since the first move is added to the tabu list at this time, the memory size or the aspiration criteria hasn't yet come into play. As a result, Table 5 the height of the peak, 767.099, out of the local optimum is the same for each run.

Table 5: Height of Peak out of First Local Optimum

Run	Iteration	Height of Peak
1	16	767.099
2	16	767.099
3	16	767.099
4	16	767.099

It is here that the four runs begin to differ significantly. Table6 shows that the largest objective value after the first local optimum appears at iteration 140 when the memory size is 35 regardless of whether or not aspiration criteria is being used. However, when the memory size is 5, the next highest objective value occurs at iteration 27.

Table 6: Largest Objective Value After First Local Optimum

Run	Iteration	Objective Value
1	140	776.888
2	27	769.815
3	140	776.888
4	27	769.815

When the MEMSIZE =35, it took 80 iterations until the first tabu move met aspiration criteria. This is 33 iterations longer than when the MEMSIZE =5 and the first tabu move met the aspiration criteria at 47. This is right after each run encountered the next improved objective value, as see in Table 7.

Table 7: Next Improved Objective Value after First

Run	Iteration	Objective Value
1	79	764.996
2	45	765.657
3	79	764.996
4	45	765.657

Overall, Table 8 shows that the three best solutions encountered by each run varied. Run one with memory size 35 and aspiration found the best objective value of 762, .95 at iteration 82. However, it did take it nearly twice as long to find its best objective than memory size 5 and aspiration, which found its best objective at iteration 47 and value of 764.702. We see that when aspiration was employed, the implementation performed better as both run 1 and run 2 at least matched the best results of runs 3 and 4.

Table 8: Best Three Objective Values

Run	Iteration	Best
1	79	764.996
	80	764.131
	82	762.95
2	45	765.657
	46	765.108
	47	764.702
3	80	764.996
	81	764.277
	82	763.815
4	45	765.657
	46	765.108
	50	764.702

So, we see that by increasing the memory size, a better objective value is achieved. However, this is results in an increase amount of iterations until the best objective value is found. Depending on computing resources available, it may be wise to use a lower memory size. However, if twice the computational time isn't issue, it increasing the memory size can result in a better objective value.

3

Find as good as solutions as you can for the 150 node data set ('spd150.galvao') for the $p-$ medianproblem with $p = 15, 20$ *and* 25. *The optimal objective values are* $7390, 6454$ *and* 5875, *respectively. The tabu search code ('tabu3.c') is setup to solve this problem. Comment on the parameter values (including SEED choices) that led to your best solutions.*

1.) *p= 15, optimal = 7390*

Table 9: Parameters when P = 15

Parameter	Value
nLoc	15
SEED	1
MEMSIZE	75
MAXIT	250
nRow	150

The optimal objective value of 7390 was achieved at iteration number 202. Two moves met aspiration criteria: one at iteration 16 and one at iteration 17. 878 moves were made tabu overall. The MEMSIZE was increased to 75 so that we would obtain the optimal value, as we saw in the previous experiment that an increased memory size led to a better objective value. The number of iterations was increased to give the program the appropriate amount of time to locate the optimal objective value.

2.) *p= 20, optimal 6454*

Table 10: Parameters when P = 20, optimal = 6454

Parameter	Value
nLoc	20
SEED	1
MEMSIZE	75
MAXIT	350
nRow	150

The optimal objective value of 6454 was achieved at iteration number 314. Two moves met aspiration criteria: one at iteration 313 and one at iteration 314. 1519 moves were made tabu overall. As with $p = 15$, the MEMSIZE was increased to 75. MAXINT was further increased to 350 in order to accommodate the increase in memory size.

3.) *p= 25, optimal = 5875*

Table 11: Parameters when P = 25

Parameter	Value
nLoc	25
SEED	1
MEMSIZE	75
MAXIT	1600

The optimal objective value of 5875 was achieved at iteration number 1559. Four moves met aspiration criteria: one at iteration 17, one at iteration 38, one at 1516, and one more at 1559. MEMSIZE was varied from 35 to 150, but the best results were obtained when MEMSIZE was kept to 75. However, the increase in p to 25 necessitated a dramatic increase in the number of iterations required; MAXIT was increased to 1600.