Second report on the Tabu Search Algorithm

Fanny Kalinowski, Julien Molinier, Robin Lambert, Maxime Leras

24 Avril 2020

Sommaire

1 Introduction

Tabu Search is an algorithm created in 1986. It uses a deterministic heuristic on local search methods. The goal is to take a potential solution, that's to stay a local optimum, and to check its immediate neighbors in the hope of finding an improved solution. This time, some improvements of the Tabu Search algorithm have been made. This report presents the results and the conclusions of two improvements : the Random-based Tabu Duration and the Frequency-based Tabu Duration. At the end, the members of the team present their personal learning from this work. [1]

For the 100 cities dataset where distance between a and b is different to the one between b and a, we used the full dataset and not only half.

2 Reference results on 50 and 100 cities with constant Tabu duration

2.1 The two-best constant Tabu durations

In the first report, the two best constant Tabu durations we observed on the 50 cities problem were 20 and 80. They will be used in the following runs.

2.2 Run of the algorithm

The algorithm was run 10 times with two different configurations.
Configuration_1 of the first version :
Data set : 50 cities
Nb_iterations: 10 000
Duration_tabou: 20

[1] https://en.wikipedia.org/wiki/Tabu_search

Run	**Best solution fitness**	**Iteration of the best solution**	**Nb_Local_minima**
1	5693	340	3063
2	5649	323	2594
3	5649	1756	2592
4	5715	1691	2542
5	5658	325	2272
6	5649	533	2601
7	5766	781	2547
8	5756	122	1907
9	5644	603	2443
10	5851	52	2568

Table 1: Parameters found with the first version configuration at each run for Duration_Tabou equals to 20

Configuration_2 of the first version :
Data set : 50 cities
Nb_iterations: 10 000
Duration_tabou: 80

Run	**Best solution fitness**	**Iteration of the best solution**	**Nb_Local_minima**
1	5685	6890	2314
2	5655	261	2295
3	5644	214	2306
4	5645	8000	2269
5	5655	3449	2265
6	5649	2015	2200
7	5649	5226	2255
8	5649	9327	2270
9	5649	7228	2252
10	5649	3185	2293

Table 2: Parameters found with the first version configuration at each run for Duration_Tabou equals to 80

Configuration_3 of the second version :
Data set : 100 cities
Nb_iterations: 100 000
Duration_tabou: 20

Run	Best solution fitness	Iteration of the best solution	Nb_Local_minima
1	8115	328	44002
2	8151	424	44203
3	8063	380	44301
4	8346	227	44304
5	8193	376	44135
6	8351	132	44129
7	8295	1000	44099
8	8227	377	44622
9	8308	771	44002
10	8224	861	44170

Table 3: Parameters found with the second version configuration at each run for Duration_Tabou equals to 20

Configuration_4 of the second version :
Data set : 100 cities
Nb_iterations: 100 000
Duration_tabou: 80

Run	Best solution fitness	Iteration of the best solution	Nb_Local_minima
1	8422	1915	44079
2	8233	582	44030
3	8352	549	44163
4	8109	612	44128
5	8179	106	44609
6	8321	541	44053
7	8176	869	44153
8	8005	456	44142
9	8055	449	44074
10	8182	186	44166

Table 4: Parameters found with the second version configuration at each run for Duration_Tabou equals to 80

	Configuration_1	**Configuration_2**	**Configuration_3**	**Configuration_4**
Best solution fitness	5703	5652,9	8227,3	8204,4
Iteration of the best solution	652,6	4578,5	487,6	626,5
Nb_Local_minima	5644	2271,9	44 196,7	44 157

Table 5: Average of the parameters for the four configurations

For 10 runs in each configuration, we can see that the fitness is better with a Tabu duration of 80, for both 50 and 100 cities samples. However, the average iteration where the best fitness solution was found is clearly the lowest with a Tabu duration of 20, which means that the solution was found quicker than with 80, again for both cities samples.

What is quite interesting is that the best solution was found around the same iteration with a Tabu duration of 20 for both samples for each run (652,6 against 487,6). It is even a little bit faster for the 100 cities problem, but it's about the same rough size, in comparison of the 10 000 and 100 000 iterations respectively taken to run the algorithm.

The number of local minimum visited are really important for the 100 000 iterations runs, of course. As expected, more local minimum were visited with a tabu duration of 80 for the 100 cities problem, than with the tabu duration of 20.

What we can say so far is that the 100 cities problem is a really complex one, and the best solution found was found only one time (8005 km). Let's see if the random-based tabu duration increase the results when it comes to fitness, but also performance-wise.

3 Random based tabu duration RTD

3.1 Running results

Run	Best solution fitness	Iteration of the best solution	Nb_Local_minima
1	5720	67	2664
2	5715	49	2561
3	5716	40	2384
4	5654	50	2580
5	5715	52	2629
6	5874	84	2653
7	5771	143	2402
8	5654	138	2496
9	5851	579	2432
10	5764	76	2429

Table 6: Results with RTD, alpha = 1, 10000 iterations and 50 cities

10 4 4 6 6 0 8 0 6 8
7 5 5 7 4 4 9 7 4 4
4 4 10 0 10 0 5 4 4 4
0 4 4 4 4 4 4 7 6 0
5 8 5 4 4 4 4 4 0 4

(Mean Tabu duration by city table 6)

Run	Best solution fitness	Iteration of the best solution	Nb_Local_minima
1	5644	3073	2249
2	5644	3489	2207
3	5644	9839	2184
4	5644	855	2240
5	5644	3241	2157
6	5644	651	2248
7	5644	7706	2184
8	5644	949	2232
9	5644	5027	2228
10	5644	4030	2238

Table 7: Results with RTD, alpha = 5, 10000 iterations and 50 cities

38 18 18 18 18 18 18 18 18 18
18 18 18 18 18 18 18 18 18 18
18 18 18 18 18 18 18 18 18 18
18 18 18 18 18 18 18 18 24 34
23 27 38 18 18 18 18 18 37 18

(Mean Tabu duration by city table 7)

Run	Best solution fitness	Iteration of the best solution	Nb_Local_minima
1	8349	351	24285
2	8319	137	24405
3	8413	97	23071
4	8652	56723	24592
5	8279	397	23167
6	8747	107	23713
7	8215	338	24323
8	8254	208	24384
9	8182	1176	23738
10	8287	710	24372

Table 8: Results with RTD, alpha = 1, 100000 iterations and 100 cities

10 10 10 10 10 10 10 10 7 8
10 8 9 11 9 9 10 11 0 8
9 12 7 10 12 10 10 13 0 9
7 10 10 10 10 10 8 9 8 8
10 10 11 9 9 10 10 10 10 10
10 10 7 10 10 5 10 0 0 10
14 12 9 9 10 10 10 10 10 12
13 10 10 10 10 12 0 8 0 0
12 9 12 10.5 13 13 11 0 10 0
13 14 10 10 10 10 14 10 10 10

(Mean Tabu duration by city table 8)

Run	Best solution fitness	Iteration of the best solution	Nb_Local_minima
1	7910	15848	22190
2	7949	74429	23207
3	8040	13322	20414
4	8247	1077	20931
5	8322	62656	19201
6	7974	42390	22730
7	8048	1900	20979
8	7944	2379	22143
9	8079	24546	21896
10	8175	18612	21571

Table 9: Results with RTD, alpha = 5, 100000 iterations_ and 100 cities

50 50 50 50 50 50 50 50 50 49
50 50 50 50 50 50 50 51 51 50
49 50 50 50 50 50 50 51 51 50
49 50 50 50 51 55 50 50 50 50
49 50 50 50 51 50 50 50 48 46
51 51 46 50 50 51 49 50 50 50
49 50 50 51 51 51 50 50 51 50
50 50 50 50 49 49 46 48 50 50
50 50 52 51 50 49 49 50 50 50
50 49 50 50 50 50 50 50 50 50

(Mean Tabu duration by city table 9)

	Configuration_6	**Configuration_7**	**Configuration_8**	**Configuration_9**
Best solution fitness	5743,4	5644	8369,7	8060,8
Iteration of the best solution	127.8	3886	6024,6	25715,9
Nb_Local_minima	2523	2216,7	24004,8	21526,2

Table 10: Average of the parameters for the four configurations

We can observe that the solution is better for the configurations 7 and 9, that's to say when alpha = 5 for the 50 and the 100 set of cities.

Let's calculate the bounds. When alpha = 1, lower_bound = 158 and upper_bound = 474. Consequently, the tabu duration is included in S1 = [158:474]. There are 316 values in this set, so the tabu duration can take 316 random values.

When alpha = 5, lower_bound = 790 and upper_bound = 2 371. The tabu duration is included in S2 = [790:2 371]. There are 1581 values in this set, so the tabu duration can take 1581 random values.

Regarding S1 and S2, the probability to pick a high random value for the tabu_duration is higher in S2 than in S1. The more high tabu_duration is, the more visited are the neigbors of the solution because it prohibits to reeuse the same neighbors in the solution.

So using alpha = 5 raise the probability to find a neighbor more intersting than the best solution. It increases the size of the duration tabou, and the number of local minima visited is less than when alpha = 1.

The randomness isn't predictable, we can't really make assured conclusions. Nevertheless, we can observe that using the randomness, the best solution is improved comparing to the configuration in the previous part. This is due to the fact that 20 and 80 are not even included in S1 and S2. Every random value picked from those two sets will produce a higher tabu duration that 20 or 80. So, more neighbors are explored. This is why the best solution is improved. Nevertheless, with less local minima visited, alpha = 5 allows to find a better solution because the moves are more valuable.

4 Frequency-based tabu duration FTD

4.1 Run the algorithm for the realease FTD1

Run	Best solution fitness	Iteration of the best solution	Nb_Local_minima
1	5649	8100	2525
2	5773	154	2425
3	5645	2015	2478
4	5771	358	2495
5	5768	2716	2364
6	5839	1023	2513
7	5715	1434	2421
8	5839	698	2493
9	5654	8093	2562
10	5768	2555	2236

Table 11: Results with FTD1, alpha = 1, 10 000 iterations_ and 50 cities

```
10 17 10 10  0  10 15 16 11 11
11 11 11 10 10 14 13 14 12 11
11 11 13 14 11 11 12 10 12 12
12 12 12 12 10 12 12 11 12 11
11 10 10 10 10 10 16 16 10 16
```

(Mean Tabu Duration by city table 11)

Above, the matrix shows the mean tabu duration for each city during one of the runs with FTD1 where there are 50 cities and alpha is 1. We can see that some cities weren't visited enough times to have a Tabu duration changed so the mean value is 10 (same as default constant). We can also see that on 10 000 iterations, one city hasn't been visited once so the mean value is 0.

Run	Best solution fitness	Iteration of the best solution	Nb_Local_minima
1	5768	8725	2273
2	5750	7432	2369
3	5644	9855	2402
4	5644	140	2320
5	5644	174	2349
6	5644	6833	2351
7	5786	8073	2293
8	5723	3714	2389
9	5763	7582	2177
10	5644	9468	2370

Table 12: Results with FTD1, alpha = 5, 10 000 iterations_ and 50 cities

```
10 29 16 12 13 14 19 19 16 16
16 16 15 11 16 19 19 19 18 18
19 10 15 21 21 18 21 20 10 10
10 12 16 23 24 23 19 12 10 10
 0  10 10 11 16 16 32 32 10 34
```

(Mean Tabu Duration by city table 12)

Run	Best solution fitness	Iteration of the best solution	Nb_Local_minima
1	8502	64479	23907
2	8468	87330	17992
3	8089	51183	22372
4	8096	80233	23466
5	8356	83545	23729
6	8311	6760	22647
7	8310	94709	22311
8	8359	74478	22969
9	8287	59608	23871
10	7994	59341	24091

Table 13: Results with FTD1, alpha = 1, 100 000 iterations_ and 100 cities

```
11 33 10 10 10 10 10  0  23 23
19 23 25 26 26 15 10 10  0  10
11 18 17 17 18 10 10 10 10 13
13 16 17 16 15 14 13 13 11 10
10 24 25 19 18 15 11 11 16 17
15 17 19 10 13 13 10 10 17 17
19 16 15 15 17 16 10 10 10 10
10 10 10 10 10 10 14 14 14 12
19 19 14 10 10 10 10 10 10 10
10 10 15 15 10 10 11 31 33 32
```

(Mean Tabu Duration by each city table 13)

```
41 63 14 24 33 64 65 60 13 12
13 47 46 40 45 42 42 11 11 14
14 44 45 10 10 58 55 14 17 21
44 45 17 14 26 29 30 24 16 19
32 43 41 24 31 12 44 47 54 54
19 19 17 19 24 24 24 22 25 28
23 17 19 20 19 22 28 10 30 31
26 23 23 27 23 15 10 10 18 17
24 29 10 28 24 11 12 17 26 25
27 18 48 45 20 13 26 42 41 58
```

(Mean Tabu Duration by city table 14)

Run	Best solution fitness	Iteration of the best solution	Nb_Local_minima
1	8703	91598	22382
2	8045	74418	19044
3	8217	11256	20928
4	8329	92618	23365
5	8323	92495	21745
6	8068	61046	21696
7	7910	77713	22646
8	8039	56475	22773
9	8061	47198	22537
10	8035	8688	22771

Table 14: Results with FTD1, alpha = 5, 100 000 iterations_ and 100 cities

	Configuration_11	Configuration_12	Configuration_13	Configuration_14
Best solution fitness	5 742,1	5 701	8 277,2	8 173
Iteration of the best solution	2 714.6	6 199.6	66 166,6	61 350,5
Nb_Local_minima	2 451.2	2 329,3	22 735,5	21 988,7

Table 15: Average of the results for the four runs with FTD1

4.2 Algorithm performances for FTD1

For both cities sizes, we can see that having a greater alpha gives better results. This is due to the fact that the effect of changing the tabu duration depending on frequency is much more visible when alpha is big.

Regarding the best solution fitness average, we observe a little increase in performances when alpha is greater. But when we look at the fitness one by one, optimal solutions are much more present when alpha is 5 than when alpha is equal to 1. For example, the optimal solution (5644km) for 50 cities is found 0 times with alpha = 1 but 5 times with alpha = 5. The same thing can be noticed for the 100 cities problem where a solution of 7910km was found with alpha = 5.

In the mean tabu duration for each city matrix, we can see that when alpha increases, the tabu duration for cities that were often visited is much higher.

For 50 cities, we can see that the number of iterations before finding the best solution doubles when alpha is multiplied by 5. This is because we are increasing the tabu duration too fast for the problem size and many city switches are prohibited for too long.

This problem disapears for the 100 cities problem because alpha = 5 is still adapted for this problem size. A greater alpha would probably create the same problem as with 50 cities.

4.3 Run the algorithm for the release FTD2

Run	Best solution fitness	Iteration of the best solution	Nb_Local_minima
1	5645	179	2414
2	5763	9779	2256
3	5811	9853	2378
4	5763	55	2250
5	5649	9728	2560
6	5645	45	2416
7	5649	8467	2294
8	5715	6360	2356
9	5644	4067	2494
10	5795	7583	2339

Table 16: Results with FTD2, alpha = 1, 10 000 iterations_ and 50 cities

16 15 15 14 14 16 17 16 17 15
14 11 12 16 17 16 16 17 12 14
12 13 19 22 22 22 25 24 18 11
11 11 11 12 14 14 15 15 11 20

(Mean Tabu Duration by city table 16)

Run	Best solution fitness	Iteration of the best solution	Nb_Local_minima
1	5644	4261	2267
2	5644	7975	2219
3	5645	663	2231
4	5655	8013	2277
5	5645	367	2249
6	5644	106	2267
7	5644	2390	2290
8	5644	7043	2251
9	5644	3771	2316
10	5644	2702	2320

Table 17: Results with FTD2, alpha = 5, 10 000 iterations_ and 50 cities

11 42 32 36 35 35 36 40 40 39
38 40 38 38 35 37 37 37 35 34
24 21 31 36 38 39 36 35 34 30
25 27 40 45 46 46 44 40 45 29
29 28 24 24 27 27 24 24 11 35

(Mean Tabu Duration by city table 17)

14 33 44 11 11 16 16 19 26 26
12 12 13 43 44 46 13 12 12 13
13 0 12 12 12 12 12 12 11 11
11 11 11 13 13 14 11 28 32 36
36 38 28 27 32 31 34 31 23 27
27 27 26 25 19 20 30 32 27 29
18 18 15 14 12 31 30 32 11 13
13 13 11 25 33 31 32 25 26 11
26 38 37 39 37 13 13 13 0 11
11 12 12 18 19 23 17 14 14 27

(Mean Tabu Duration by city table 18)

Run	Best solution fitness	Iteration of the best solution	Nb_Local_minima
1	8319	38314	23490
2	8279	45937	21257
3	7944	92926	20264
4	8585	68202	22017
5	8323	27327	23669
6	8246	98642	22213
7	8248	51592	20030
8	8292	78529	22483
9	8410	29825	23301
10	8358	63426	18636

Table 18: Results with FTD2, alpha = 1, 100 000 iterations_ and 100 cities

Run	Best solution fitness	Iteration of the best solution	Nb_Local_minima
1	8054	53160	21517
2	7949	98104	20949
3	7911	84982	20983
4	7956	99565	21208
5	8256	58593	19764
6	7955	94970	19999
7	7959	95487	18694
8	8183	13473	19997
9	7948	79910	19707
10	8189	88352	19564

Table 19: Results with FTD2, alpha = 5, 100 000 iterations_ and 100 cities

68	112	102	98	105	96	100	119	102	94
83	92	99	102	107	107	107	86	79	72
66	66	64	68	48	38	40	31	44	103
94	89	96	71	87	89	89	95	99	62
32	60	61	61	63	40	32	30	34	46
49	51	50	34	26	21	24	27	31	31
28	30	33	37	38	39	40	41	44	41
43	43	34	27	23	23	31	37	43	58
85	86	84	86	75	74	60	54	54	49
52	48	45	60	62	63	67	72	89	82

(Mean Tabu Duration by city table 19)

	Configuration_1	Configuration_2	Configuration_3	Configuration_4
Best solution fitness	5707,9	5645,3	8300,4	8036
Iteration of the best solution	5611,6	3729,1	59472	76659,6
Nb_Local_minima	2375,7	2268,7	21736	20238,2

Table 20: Average of the parameters for the four configurations

4.4 Algorithm performances for FTD2

For all configurations, it is intersting to see that with alpha equals to 5, the solution is better when it comes to fitness. We can explain this by saying that the bigger alpha is, the longer the tabu duration for a move bewteen two cities is, so a move is considered as Tabu for a longer time with alpha equals to 5. This mean that more solutions should be explored and that the algorithm is more able to find the best solution (this is verified for both samples).

What is trickier to explain is the part where the best solution is found quickier with alpha = 1 for 100 cities, while it is the contrary for 50 cities. Maybe this can be due to the fact that the 100 cities problem is way more complex (plus the distance between two cities is different depending on the way you travel...). For the 100 city problem, we're doing 100 000 iterations, so maybe a big enough alpha will quickly banned moves containing a city : we're exploring more and more, and banning every city nearby going slowly to the solution.

4.5 Comparing FTD1 and FTD2

Comparing to the FTD1, we're banning moves a longer time for the 100 city problem with alpha equals to 5 : we're setting a tabu duration for a move by taking in consideration the number of times we went onto the city we chose first. Thus, we can ban this move a way longer time because there's 100 possibilites of choosing this city in another move, which will quickly increase the number of times we visited the city, so the tabu duration. This can explain why we're exploring an area completely, and waiting until the last moment to explore another part, quickly being considered Tabu for a long time. Globally, increasing the alpha increases the fitness.

5 Best solution

The best solution encountered is :
0-17-61-86-14-62-85-96-66-12-48-20-74-81-84-13-11-3-31-8-25-73-19-77-2-32-9-26-91-16-71-69-37-53-72-49-45-55-18-36-27-92-76-94-58-75-57-88-1-22-34-28-43-99-38-95-54-39-42-24-23-41-6-33-40-21-4-60-10-51-44-15-30-83-87-65-97-93-5-78-52-98-46-56-35-63-90-80-79-29-47-64-89-50-82-67-70-7-68-59 : 7910 km

6 Personnal learnings

Julien Molinier During this project I have learned a new way to find the best solution in a large dimension problem with a lot of variables. I had never heard about Tabu algorithm before so it was quite interesting to discover. I found really interesting the idea to force an algorithm to explore another way even if it won't improve the best solution immediatly.

Fanny Kalinowski I had already heard about the Tabu search algorithm but it was the first time I had to work on its implementation. First of all, it was not that easy to understand it. But this work was necessary to understand the results of all the runs and draw some conclusions. Playing with the parameters, understanding various implementations (RTD, FTD) put the light on the complexity and all the angles this algorithm can be turned. Also, this part reflects how important it is to elaborate a strategy to solve a given problem.

Robin Lambert My personal learning from this work is the usage of a very powerful algorithm. It was really interesting to see how the parameters impacts the results, and trying to extract a general interpretation from a full set of data. This second part was really relevant, and shows me even more the importance of machine learning to solve problems completely impossible to solve by hand. The frequency based Tabu duration was especially efficient and impressive.

Maxime Leras During this project, I learned a way to find multi parameters maximum. It was something I had already done with multi parameters functions but we were able to find maximum or minimum just with mathematics as all the solutions were following the definition of the function. This time there was no function to define this multi dimensional space so we had to travel through different solutions and finding the best way to do it. On another side, because there were very long executions of the program to repeat, I learned to write bash scripts to do this job. Trying to understand a code I didn't write was interesting as well.