

ColoTe Problem

A MEMETIC APPROACH

TEAM 35

COMETA, DAVIDE
CAPACCIO, LUCA
DEDOMINICI, CHRISTOPHER
MORALES, CARLOS
SAVA, CLAUDIO

Problem description

Asking users to share the internet connection, offering some reward for it.

How to cover all the dumpsters minimizing the costs (reward).

Optimization Problem

Objective function : The goal of the project is to **minimize the total amount of the reward.**

Constraint (1) : **Every dumpster must be visited.**

Constraint (2) : The total amount of requests can not **exceed the number of people in the cell.**

Domain : The number of people must be **an integer number.**

Designing a solution

Steps:

1. Select a proper metaheuristic for the problem.

The GRASP option

A pure GRASP metaheuristic search algorithm states the following:

```
procedure GRASP(Max_Iterations,Seed)
1  Read_Input();
2  for  $k = 1, \dots, \text{Max\_Iterations}$  do
3      Solution  $\leftarrow$  Greedy_Randomized_Construction(Seed);
4      Solution  $\leftarrow$  Local_Search(Solution);
5      Update_Solution(Solution,Best_Solution);
6  end;
7  return Best_Solution;
end GRASP.
```

Source: “GRASP: Greedy Randomized Adaptive Search Procedures”. Resende, Mauricio et al.

“Greedy randomized construction”:

By a greedy randomized adaptive procedure.

Taking moves randomly from a structure that order them by “quality”.


“Local search”:

A local search method it is applied to the trial solution.

Either a “best improving strategy” or a “first improving strategy”.

Designing a solution

Steps:

1. Select a proper metaheuristic for the problem.  **GRASP.**
2. Define a “Greedy randomized construction” of possible solutions.
3. Define a local search method to find a better solution.

A greedy randomized solutions generator

A structure of moves ordered by “quality”:

A multimap of all possible moves, ordered by their cost/task ratio:

The cost/task ratio of the move m_{ijmtx} , that transfers x users of type m from cell i to j at timestamp t , is $\frac{c_{ijmt}}{n_m}$, where c_{ijmt} is the cost of moving from i to j an user of m type at t , and n_m the quantity of tasks that is able to do a user of type m .

Randomization:

Given by the shuffle of the order in which cell's demands are satisfied.

Greedy:




Taking from the multimap, for each cell, sequentially and myopically, the most efficient moves until its demand is satisfied.

Adaptive:

At each iteration, the move chosen is a function of the moves previously chosen.

Designing a solution

Steps:

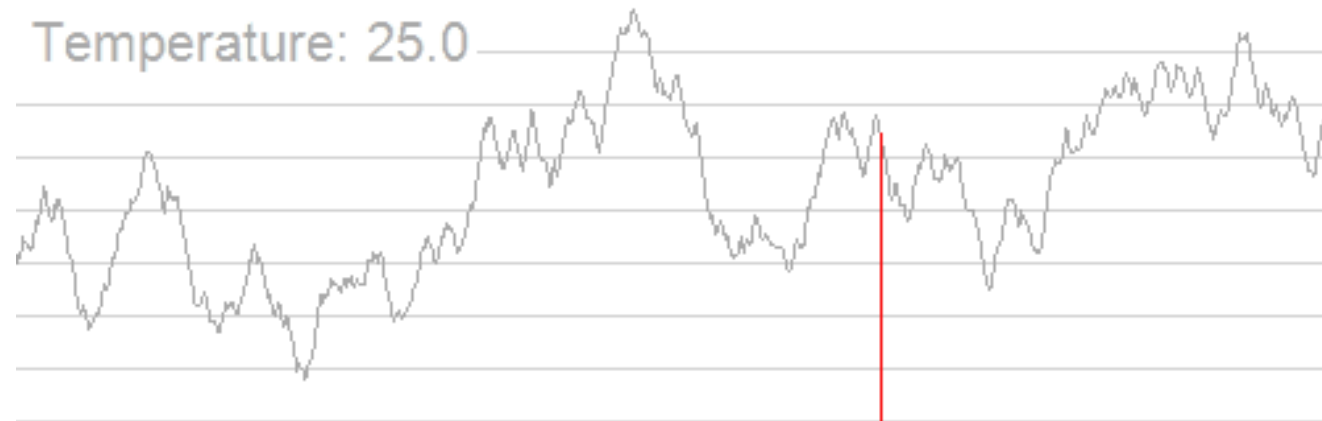
1. Select a proper metaheuristic for the problem.  **GRASP.** 
2. Define a “Greedy randomized construction” of possible solutions. 
3. Define a local search method to find a better solution.

Local search method

- At the beginning of our analysis, we observed that using this costs structure and a classical local heuristic for the local search phase, such as Steepest Descent or First Improvement, could lead to a local minimum.
- In order to escape from this possible local minimum, we decided to replace those classical heuristics with the Simulated Annealing metaheuristic.

Memetic approach – SA research

- A neighborhood based metaheuristic.
- Inspired in the controlled annealing of metals for improving its characteristics.
- The worsening solutions are stochastically accepted with a probability affected inversely by the “temperature of the system”.
- The initial temperature is the maximum and during the simulation the system is constantly cooling, making worsening solution less probable to be accepted.



Source:
https://en.wikipedia.org/wiki/Simulated_annealing

Memetic approach – SA research

Our SA implementation:

- Probability of accept worsening solutions:

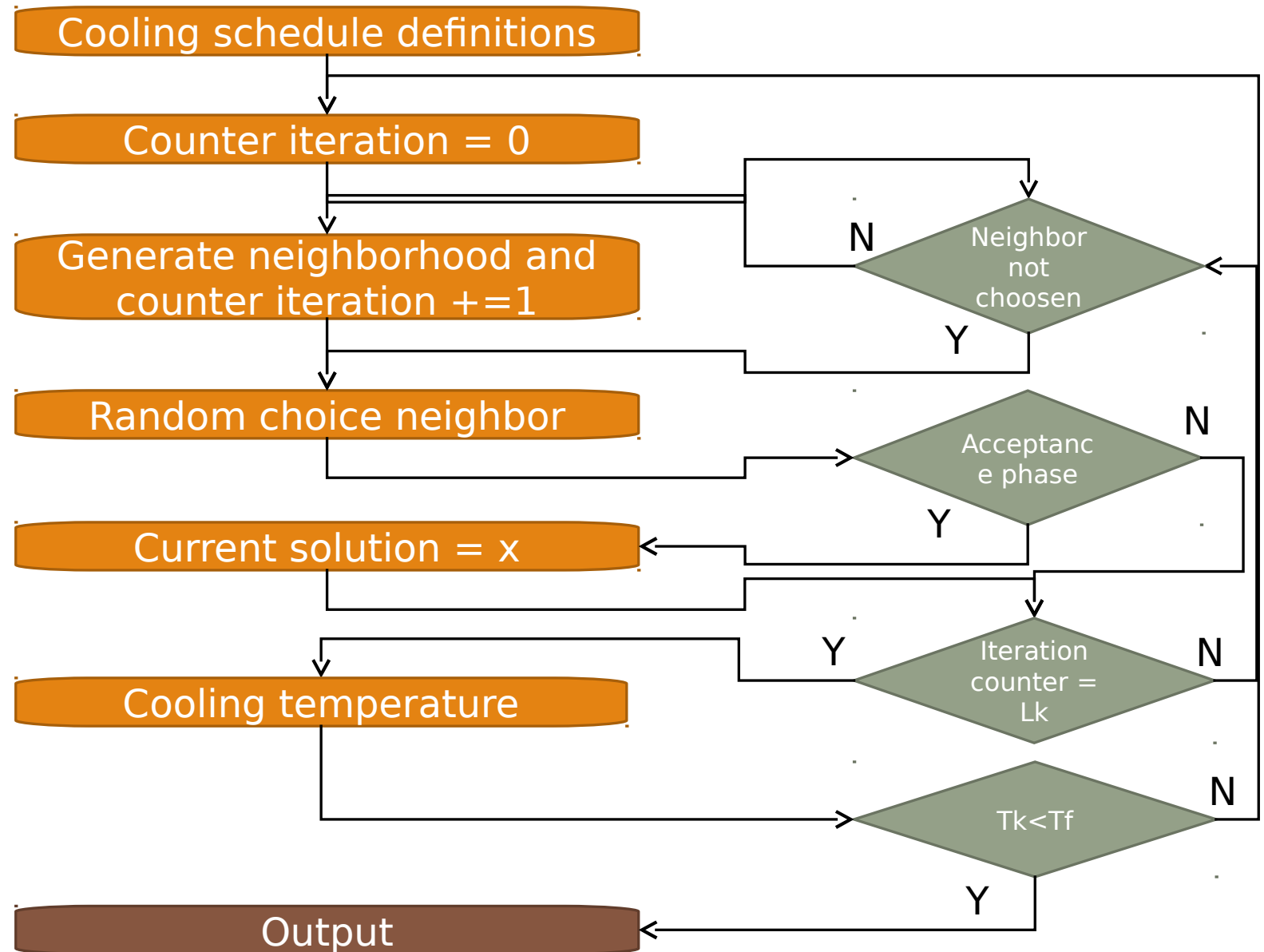
It is used the classic simulated annealing enunciation (Kitpatrick et al.).

$p = P(v, v', t) = e^{-\left(\frac{v-v'}{t}\right)}$, where v and v' are the current solution and the candidate energies respectively, and t the system temperature.

- Temperature of the system:




$t = T(i) = \alpha^{\lfloor \frac{i}{L} \rfloor} \times T_0$, where i is the iteration number, α the cooling parameter, L the so called plateau, and T_0 the initial temperature.

Memetic approach – SA research



Designing a solution

Steps:

1. Select a proper metaheuristic for the problem.  **GRASP.**
2. Define a “Greedy randomized construction” of possible solutions. 
3. Define a local search method to find a better solution.  **Simulated Annealing.**
4. Define the neighborhood generation.

Neighborhood generation

Basic idea:





- Select a random subset of cells.
- For each cell of the subset delete all its moves.
- Solve again the cell's demands, considering this time also expensive moves, getting a new neighbor.

Worsening strategy

The neighbor solution can be better or worse than the current one because when it selects the moves to solve the demand the algorithm sometimes it's forced to skip the best ones.

Designing a solution

Steps:

1. Select a proper metaheuristic for the problem.  **GRASP.**
2. Define a “Greedy randomized construction” of possible solutions. 
3. Define a local search method to find a better solution.  **Simulated Annealing.**
4. Define the neighborhood generation. 

Testing the solution

Testing machine specifics:

- CPU: Intel Core i7-4710HQ (2,5GHz)
- RAM: 8GB
- 4 Cores
- Operating system: Ubuntu 64bit

Instance	OF	Gap
Co_30_1_NT_0	1055	<pre>Input file: ./input/Co_30_1_NT_0.txt Co_30_1_NT_0 Output file: ./output/solFile_Co_30_1_NT_0.csv Solving... Solution is feasible Gap is 1.3449</pre>
Co_30_1_NT_8	2324	<pre>Input file: ./input/Co_30_1_NT_8.txt Co_30_1_NT_8 Output file: ./output/solFile_Co_30_1_NT_8.csv Solving... Solution is feasible Gap is 1.5734</pre>
Co_100_1_T_7	5291	<pre>Input file: ./input/Co_100_1_T_7.txt Co_100_1_T_7 Output file: ./output/solFile_Co_100_1_T_7.csv Solving... Solution is feasible Gap is 1.4573</pre>
		<pre>Number of instances is 120 Mean gap is: .23202666666666666666</pre>

Tuning the search

- Adjust Simulated Annealing parameters.
- Refine worsening probability.
- Define an adaptive percentage subset of cells to be solved in a different way for the neighborhood generation.

Tuning the search - Adjust Simulated Annealing parameters

Initial temperature definition:

At the beginning of the procedure, each worsening variation of the objective function had to be accepted in order to escape from possible local minimum.

To do that it had to result $\exp(-\Delta f/T_0) \approx 1 \quad \forall \Delta f > 0$ and so $(-\Delta f/T_0) \approx 0$. The choice was performed as follows:

1. Considering the initial objective function.
2. Assuming a maximum objective function variation value: $\Delta f_{\max} = |f_0/2|$.
3. Assigning $T_0 = 10 * \Delta f_{\max}$.

Final temperature definition:

As stopping condition of the whole SA procedure we decided to define a final temperature to be reached. The definition criteria was as follows:

1. Considering the initial objective function.
2. Establishing a reference value for the minimum objective function variation: $\Delta f_{\min} = 0.001 * |f_0/2|$.
3. Assigning $\Delta f = 0.1 * \Delta f_{\min}$
4. Calculating $TF = \Delta f / 0.69$

Tuning the search - Adjust Simulated Annealing parameters

α (cooling parameter) :

We observed that a high value(i.e. 0.99) made the cooling process too slow, allowing only small “Plateau” value to fulfill time constraints, while a small cooling parameter admit high “Plateau” values in order to restore energy balance condition. We decided for 0.5.

“Plateau” value:

According to the previous consideration about the cooling parameter, we decided to assign:

$$L = 15 * |\text{Neighborhood}|.$$

Testing the tuned solution

Testing machine specifics:

- CPU: Intel Core i7-4710HQ (2,5GHz)
- RAM: 8GB
- 4 Cores
- Operating system: Ubuntu 64bit

Before

```
Input file: ./input/Co_30_1_NT_0.txt
Co_30_1_NT_0
Output file: ./output/solFile_Co_30_1_NT_0.csv
Solving...
Solution is feasible
Gap is 1.3449
```

```
Input file: ./input/Co_30_1_NT_8.txt
Co_30_1_NT_8
Output file: ./output/solFile_Co_30_1_NT_8.csv
Solving...
Solution is feasible
Gap is 1.5734
```

```
Input file: ./input/Co_100_1_T_7.txt
Co_100_1_T_7
Output file: ./output/solFile_Co_100_1_T_7.csv
Solving...
Solution is feasible
Gap is 1.4573
```

```
Number of instances is 120
Mean gap is:
.23202666666666666666
```

After

```
Input file: ./input/Co_30_1_NT_0.txt
Co_30_1_NT_0
Output file: ./output/solFile_Co_30_1_NT_0.csv
Solving...

Best objective funtion:1050
Solution is feasible
Gap is 0.8646
```

```
Input file: ./input/Co_30_1_NT_8.txt
Co_30_1_NT_8
Output file: ./output/solFile_Co_30_1_NT_8.csv
Solving...

Best objective funtion:2296
Solution is feasible
Gap is 0.3497
```

```
Input file: ./input/Co_100_1_T_7.txt
Co_100_1_T_7
Output file: ./output/solFile_Co_100_1_T_7.csv
Solving...

Best objective funtion:5247
Solution is feasible
Gap is 0.6136
```

```
Number of instances is 120
Mean gap is:
.15201583333333333333
```

Conclusion and learned lessons

Saying we met the desired gap threshold, we consider the main objective was accomplished.

Drawbacks:

- If total number of tasks required are equal or very close to the ones available to solve them, the metaheuristic is not capable to find a solution without wasting tasks.
- The metaheuristic is not able to find a uniform degree of rebuilt as we noticed from some tests;

Advantage:

- Our metaheuristic provides higher performances with 20 timesteps instances because of the higher number of users and a greater number of possible combinations for choosing moves to create a solution.
- The combination of GRASP, Simulated Annealing and the costs multimap allow us to obtain good results in a short time. Nevertheless the tuning phase has been fundamental in order to improve gap results.

Complete results

1	Co_100_1_NT_0.txt	4.89104	4467	613	365	418
2	Co_100_1_NT_1.txt	4.89264	4539	638	357	346
3	Co_100_1_NT_2.txt	4.89102	4291	629	434	253
4	Co_100_1_NT_3.txt	4.89134	4108	610	288	387
5	Co_100_1_NT_4.txt	4.89197	5273	954	390	370
6	Co_100_1_NT_5.txt	4.89271	3817	537	281	316
7	Co_100_1_NT_6.txt	4.89182	4478	855	335	316
8	Co_100_1_NT_7.txt	4.89264	4851	564	354	398
9	Co_100_1_NT_8.txt	4.8925	4810	575	339	470
10	Co_100_1_NT_9.txt	4.89149	6511	920	566	402
11	Co_100_1_T_0.txt	4.89078	4287	505	335	474
12	Co_100_1_T_1.txt	4.89278	4848	476	333	416
13	Co_100_1_T_2.txt	4.89024	4672	471	435	305
14	Co_100_1_T_3.txt	4.89185	4733	633	386	314
15	Co_100_1_T_4.txt	4.89277	5365	720	363	466
16	Co_100_1_T_5.txt	4.89112	3930	529	303	304
17	Co_100_1_T_6.txt	4.89153	4798	729	320	368
18	Co_100_1_T_7.txt	4.89100	5247	625	376	363
19	Co_100_1_T_8.txt	4.89164	5027	439	323	526
20	Co_100_1_T_9.txt	4.89072	6758	1047	429	451

21	Co_100_20_NT_0.txt	4.89227	3626	2500	114	91
22	Co_100_20_NT_1.txt	4.89294	3203	2942	29	54
23	Co_100_20_NT_2.txt	4.89407	2196	2116	10	20
24	Co_100_20_NT_3.txt	4.89546	2183	1633	43	123
25	Co_100_20_NT_4.txt	4.89273	2169	2091	37	1
26	Co_100_20_NT_5.txt	4.89596	2614	2310	39	57
27	Co_100_20_NT_6.txt	4.89229	2016	1763	42	37
28	Co_100_20_NT_7.txt	4.89387	2481	2240	59	35
29	Co_100_20_NT_8.txt	4.89178	3027	2961	33	0
30	Co_100_20_NT_9.txt	4.8921	2552	2409	33	25
31	Co_100_20_T_0.txt	4.89427	3663	2336	136	131
32	Co_100_20_T_1.txt	4.89199	3164	2948	47	40
33	Co_100_20_T_2.txt	4.89347	2207	1958	47	48
34	Co_100_20_T_3.txt	4.89417	2195	1639	64	107
35	Co_100_20_T_4.txt	4.89273	2174	2007	40	27
36	Co_100_20_T_5.txt	4.89439	2610	2297	83	32
37	Co_100_20_T_6.txt	4.89319	2007	1801	11	45
38	Co_100_20_T_7.txt	4.89246	2486	2281	22	46
39	Co_100_20_T_8.txt	4.89181	3034	2704	97	43
40	Co_100_20_T_9.txt	4.89288	2552	2279	71	43

Complete results (2)

41	Co_300_20_NT_0.txt	4.91017	7371	6425	146	87
42	Co_300_20_NT_1.txt	4.89819	7039	6236	114	122
43	Co_300_20_NT_10.txt	4.91176	8102	7652	28	74
44	Co_300_20_NT_11.txt	4.90301	7638	7099	68	94
45	Co_300_20_NT_12.txt	4.90148	8194	7244	122	124
46	Co_300_20_NT_13.txt	4.90379	7582	6905	98	118
47	Co_300_20_NT_14.txt	4.91367	7682	6566	143	105
48	Co_300_20_NT_15.txt	4.91726	8546	7758	19	93
49	Co_300_20_NT_16.txt	4.9006	7132	6718	107	64
50	Co_300_20_NT_17.txt	4.9061	7195	6427	136	123
51	Co_300_20_NT_18.txt	4.89979	8076	7170	187	105
52	Co_300_20_NT_19.txt	4.90086	7557	6782	118	100
53	Co_300_20_NT_2.txt	4.90918	7657	6869	66	110
54	Co_300_20_NT_3.txt	4.9027	7168	6350	178	131
55	Co_300_20_NT_4.txt	4.89941	8038	7420	63	106
56	Co_300_20_NT_5.txt	4.90023	6320	5105	212	192
57	Co_300_20_NT_6.txt	4.89634	6998	6768	59	35
58	Co_300_20_NT_7.txt	4.90282	6381	6115	61	43
59	Co_300_20_NT_8.txt	4.9047	7020	6503	75	102
60	Co_300_20_NT_9.txt	4.89604	7183	6792	109	57

61	Co_300_20_T_0.txt	4.90123	7391	6371	134	113
62	Co_300_20_T_1.txt	4.90013	7051	6198	106	140
63	Co_300_20_T_10.txt	4.90973	8102	7552	78	74
64	Co_300_20_T_11.txt	4.91206	7662	6838	125	143
65	Co_300_20_T_12.txt	4.90634	8224	7105	133	163
66	Co_300_20_T_13.txt	4.90779	7607	6434	146	243
67	Co_300_20_T_14.txt	4.9212	7647	6606	93	125
68	Co_300_20_T_15.txt	4.90829	8590	7562	96	107
69	Co_300_20_T_16.txt	4.89633	7143	6434	228	78
70	Co_300_20_T_17.txt	4.90232	7234	6279	132	175
71	Co_300_20_T_18.txt	4.91903	8049	7158	145	137
72	Co_300_20_T_19.txt	4.90177	7511	6903	53	103
73	Co_300_20_T_2.txt	4.89958	7677	6688	116	137
74	Co_300_20_T_3.txt	4.90671	7182	5947	267	206
75	Co_300_20_T_4.txt	4.90989	8055	7348	72	124
76	Co_300_20_T_5.txt	4.91439	6359	5035	214	214
77	Co_300_20_T_6.txt	4.90512	6995	6641	85	60
78	Co_300_20_T_7.txt	4.90095	6390	5854	94	108
79	Co_300_20_T_8.txt	4.90393	7026	6336	118	129
80	Co_300_20_T_9.txt	4.89734	7184	6765	139	46

Complete results (3)

81	Co_30_1_NT_0.txt	4.89045	1050	189	27	93
82	Co_30_1_NT_1.txt	4.89111	1757	248	111	171
83	Co_30_1_NT_2.txt	4.89051	2350	325	77	160
84	Co_30_1_NT_3.txt	4.89031	2106	398	151	99
85	Co_30_1_NT_4.txt	4.89156	1482	172	194	108
86	Co_30_1_NT_5.txt	4.89299	3018	232	238	165
87	Co_30_1_NT_6.txt	4.89172	1624	191	81	193
88	Co_30_1_NT_7.txt	4.89115	1035	219	142	61
89	Co_30_1_NT_8.txt	4.89324	2296	291	259	117
90	Co_30_1_NT_9.txt	4.89328	1578	229	144	113
91	Co_30_1_T_0.txt	4.89014	1109	135	33	107
92	Co_30_1_T_1.txt	4.89165	1798	172	137	179
93	Co_30_1_T_2.txt	4.89066	2448	242	87	181
94	Co_30_1_T_3.txt	4.89183	2075	373	135	118
95	Co_30_1_T_4.txt	4.89283	1551	171	211	97
96	Co_30_1_T_5.txt	4.89013	2899	296	179	183
97	Co_30_1_T_6.txt	4.89115	1598	160	116	180
98	Co_30_1_T_7.txt	4.89165	1396	178	101	102
99	Co_30_1_T_8.txt	4.89304	2023	268	299	98
100	Co_30_1_T_9.txt	4.89323	1830	398	43	124

101	Co_30_20_NT_0.txt	4.89065	720	661	29	0
102	Co_30_20_NT_1.txt	4.89038	901	878	1	5
103	Co_30_20_NT_2.txt	4.8916	872	872	0	0
104	Co_30_20_NT_3.txt	4.89209	457	409	24	0
105	Co_30_20_NT_4.txt	4.89082	706	558	1	45
106	Co_30_20_NT_5.txt	4.89145	827	827	0	0
107	Co_30_20_NT_6.txt	4.8904	437	437	0	0
108	Co_30_20_NT_7.txt	4.89251	984	585	1	57
109	Co_30_20_NT_8.txt	4.89166	938	746	0	62
110	Co_30_20_NT_9.txt	4.89182	1132	829	47	39
111	Co_30_20_T_0.txt	4.89201	719	636	40	1
112	Co_30_20_T_1.txt	4.89069	895	895	0	0
113	Co_30_20_T_2.txt	4.8906	872	857	0	5
114	Co_30_20_T_3.txt	4.89266	457	405	26	0
115	Co_30_20_T_4.txt	4.89171	721	558	1	45
116	Co_30_20_T_5.txt	4.89224	827	817	5	0
117	Co_30_20_T_6.txt	4.89186	437	433	2	0
118	Co_30_20_T_7.txt	4.89249	991	584	24	42
119	Co_30_20_T_8.txt	4.89321	933	566	39	96
120	Co_30_20_T_9.txt	4.89169	1143	873	58	17



Thank you for your attention!