Task 1 All One Problem (12.02.2016)

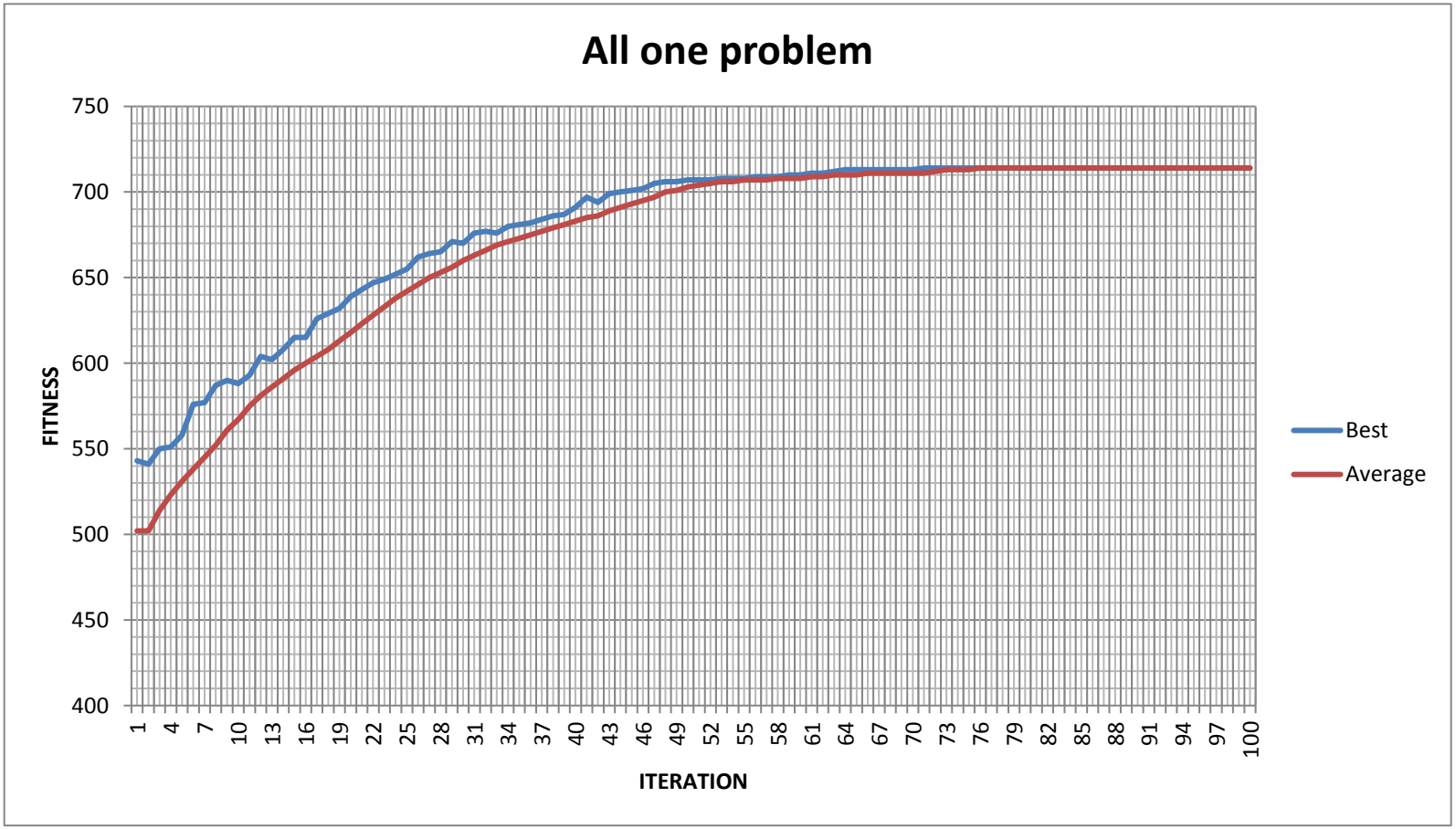
Student –Polina Rachkovskaya

We have the initial population with 100 binary chromosomes with 1000 genes. The fitness is the number of "1" is good, "0" is bad. We need to get gene which consists all "1".

Algorithm:

1. Create 100 binary-chromosomes at random where each chromosome contains 1000 gens.
2. Select 2 chromosomes using Roulette method.
3. Create a child chromosome by a one-point-crossover.
4. Repeat steps 2-3, until we get new population.
5. Repeat 4. until the fitness value does not change any more.

Results:



The best fitness chromosome in 1st generation is 543 and in last 100th - generation 714.
Since 76 iteration, there is no improvement of chromosomes because we achieved the best result.

Iteration	1			25			50			75			100		
	Fitness	Probability	Choice	Fitness	Probability	Choice	Fitness	Probability	Choice	Fitness	Probability	Choice	Fitness	Probability	Choice
1	543	0,010815656	3	648	0,010161199	2	706	0,010063574	2	714	0,01000827	3	714	0,01	0
2	533	0,010616472	2	647	0,010145518	0	705	0,01004932	0	714	0,01000827	2	714	0,01	0
3	532	0,010596554	3	645	0,010114157	3	705	0,01004932	3	714	0,01000827	2	714	0,01	0
4	532	0,010596554	0	644	0,010098476	5	705	0,01004932	0	714	0,01000827	1	714	0,01	0
5	529	0,010536799	3	644	0,010098476	2	705	0,01004932	3	714	0,01000827	2	714	0,01	0
6	527	0,010496962	2	644	0,010098476	0	705	0,01004932	0	714	0,01000827	0	714	0,01	0
7	526	0,010477044	1	643	0,010082795	1	705	0,01004932	2	714	0,01000827	0	714	0,01	0
8	525	0,010457126	1	643	0,010082795	2	704	0,010035066	0	714	0,01000827	1	714	0,01	0
9	525	0,010457126	1	643	0,010082795	3	704	0,010035066	0	714	0,01000827	0	714	0,01	0
10	525	0,010457126	2	643	0,010082795	0	704	0,010035066	0	714	0,01000827	0	714	0,01	0
11	524	0,010437207	0	643	0,010082795	3	704	0,010035066	4	714	0,01000827	0	714	0,01	0
12	523	0,010417289	2	643	0,010082795	0	704	0,010035066	0	714	0,01000827	2	714	0,01	0
13	522	0,010397371	0	642	0,010067114	3	704	0,010035066	2	714	0,01000827	0	714	0,01	0
14	522	0,010397371	0	642	0,010067114	2	704	0,010035066	0	714	0,01000827	0	714	0,01	0
15	522	0,010397371	2	642	0,010067114	0	704	0,010035066	3	714	0,01000827	2	714	0,01	0
16	520	0,010357534	1	642	0,010067114	0	703	0,010020811	0	714	0,01000827	0	714	0,01	0
17	519	0,010337616	3	642	0,010067114	2	703	0,010020811	0	714	0,01000827	1	714	0,01	0
18	517	0,010297779	2	641	0,010051433	0	703	0,010020811	0	714	0,01000827	2	714	0,01	0
19	516	0,010277861	1	641	0,010051433	3	703	0,010020811	0	714	0,01000827	1	714	0,01	0
20	515	0,010257942	0	641	0,010051433	1	703	0,010020811	5	714	0,01000827	2	714	0,01	0
21	514	0,010238024	1	641	0,010051433	0	703	0,010020811	0	714	0,01000827	2	714	0,01	0
22	514	0,010238024	2	641	0,010051433	2	703	0,010020811	0	714	0,01000827	0	714	0,01	0
23	514	0,010238024	2	641	0,010051433	0	703	0,010020811	2	714	0,01000827	1	714	0,01	0
24	513	0,010218106	2	640	0,010035752	1	703	0,010020811	0	714	0,01000827	0	714	0,01	0
25	513	0,010218106	3	640	0,010035752	1	703	0,010020811	0	714	0,01000827	2	714	0,01	0
26	512	0,010198187	2	640	0,010035752	0	703	0,010020811	4	714	0,01000827	0	714	0,01	0
27	512	0,010198187	0	640	0,010035752	0	703	0,010020811	0	714	0,01000827	0	714	0,01	0
28	511	0,010178269	2	640	0,010035752	2	703	0,010020811	0	714	0,01000827	2	714	0,01	0
29	511	0,010178269	0	640	0,010035752	0	703	0,010020811	1	714	0,01000827	0	714	0,01	0
30	510	0,010158351	0	640	0,010035752	0	703	0,010020811	0	714	0,01000827	0	714	0,01	0
31	510	0,010158351	1	639	0,010020072	3	702	0,010006557	3	714	0,01000827	3	714	0,01	0
32	510	0,010158351	2	639	0,010020072	0	702	0,010006557	3	714	0,01000827	2	714	0,01	0
33	510	0,010158351	0	639	0,010020072	0	702	0,010006557	2	714	0,01000827	0	714	0,01	0
34	508	0,010118514	0	639	0,010020072	5	702	0,010006557	0	714	0,01000827	2	714	0,01	0
35	508	0,010118514	0	639	0,010020072	0	702	0,010006557	0	714	0,01000827	0	714	0,01	0

36	508	0,010118514	1	639	0,010020072	0	702	0,010006557	0	714	0,01000827	1	714	0,01	0
37	508	0,010118514	2	639	0,010020072	0	702	0,010006557	1	714	0,01000827	0	714	0,01	0
38	508	0,010118514	0	639	0,010020072	3	702	0,010006557	2	714	0,01000827	0	714	0,01	0
39	507	0,010098596	2	639	0,010020072	0	702	0,010006557	2	714	0,01000827	2	714	0,01	0
40	506	0,010078677	1	638	0,010004391	2	702	0,010006557	0	714	0,01000827	0	714	0,01	0
41	505	0,010058759	0	638	0,010004391	0	702	0,010006557	3	714	0,01000827	1	714	0,01	0
42	505	0,010058759	3	638	0,010004391	0	702	0,010006557	0	713	0,009994253	0	714	0,01	0
43	504	0,010038841	0	638	0,010004391	3	702	0,010006557	0	713	0,009994253	0	714	0,01	0
44	504	0,010038841	0	638	0,010004391	0	702	0,010006557	0	713	0,009994253	1	714	0,01	0
45	504	0,010038841	0	638	0,010004391	1	702	0,010006557	3	713	0,009994253	2	714	0,01	0
46	503	0,010018922	2	638	0,010004391	1	702	0,010006557	0	713	0,009994253	2	714	0,01	0
47	503	0,010018922	0	638	0,010004391	0	702	0,010006557	4	713	0,009994253	0	714	0,01	0
48	502	0,009999004	0	638	0,010004391	1	702	0,010006557	0	713	0,009994253	1	714	0,01	0
49	501	0,009979086	3	638	0,010004391	0	702	0,010006557	2	713	0,009994253	0	714	0,01	0
50	501	0,009979086	0	637	0,00998871	0	701	0,009992303	0	713	0,009994253	0	714	0,01	0
51	501	0,009979086	0	637	0,00998871	2	701	0,009992303	0	713	0,009994253	0	714	0,01	0
52	500	0,009959167	0	637	0,00998871	0	701	0,009992303	0	713	0,009994253	2	714	0,01	0
53	500	0,009959167	3	637	0,00998871	1	701	0,009992303	1	713	0,009994253	2	714	0,01	0
54	499	0,009939249	0	637	0,00998871	0	701	0,009992303	0	713	0,009994253	2	714	0,01	0
55	498	0,009919331	0	637	0,00998871	4	701	0,009992303	3	713	0,009994253	0	714	0,01	0
56	498	0,009919331	0	636	0,009973029	0	701	0,009992303	0	713	0,009994253	0	714	0,01	0
57	498	0,009919331	2	636	0,009973029	1	701	0,009992303	3	713	0,009994253	2	714	0,01	0
58	497	0,009899412	1	636	0,009973029	3	701	0,009992303	2	713	0,009994253	0	714	0,01	0
59	497	0,009899412	0	636	0,009973029	0	701	0,009992303	0	713	0,009994253	1	714	0,01	0
60	496	0,009879494	2	636	0,009973029	0	701	0,009992303	0	713	0,009994253	2	714	0,01	0
61	496	0,009879494	0	636	0,009973029	0	701	0,009992303	0	713	0,009994253	0	714	0,01	0
62	496	0,009879494	0	636	0,009973029	0	701	0,009992303	2	713	0,009994253	0	714	0,01	0
63	496	0,009879494	0	636	0,009973029	1	701	0,009992303	2	713	0,009994253	2	714	0,01	0
64	495	0,009859576	2	636	0,009973029	0	700	0,009978048	0	713	0,009994253	0	714	0,01	0
65	495	0,009859576	0	636	0,009973029	0	700	0,009978048	1	713	0,009994253	0	714	0,01	0
66	495	0,009859576	0	636	0,009973029	0	700	0,009978048	0	713	0,009994253	2	714	0,01	0
67	495	0,009859576	0	635	0,009957348	0	700	0,009978048	0	713	0,009994253	0	714	0,01	0
68	494	0,009839657	3	635	0,009957348	3	700	0,009978048	1	713	0,009994253	2	714	0,01	0
69	493	0,009819739	0	635	0,009957348	0	700	0,009978048	0	713	0,009994253	0	714	0,01	0
70	493	0,009819739	0	635	0,009957348	1	700	0,009978048	2	713	0,009994253	1	714	0,01	0
71	493	0,009819739	0	635	0,009957348	0	700	0,009978048	0	713	0,009994253	2	714	0,01	0

72	493	0,009819739	1	635	0,009957348	2	700	0,009978048	0	713	0,009994253	3	714	0,01	0
73	492	0,009799821	0	635	0,009957348	2	700	0,009978048	0	713	0,009994253	0	714	0,01	0
74	492	0,009799821	3	635	0,009957348	0	700	0,009978048	2	713	0,009994253	3	714	0,01	0
75	492	0,009799821	0	635	0,009957348	1	700	0,009978048	2	713	0,009994253	0	714	0,01	0
76	492	0,009799821	0	635	0,009957348	0	700	0,009978048	0	713	0,009994253	0	714	0,01	0
77	492	0,009799821	2	634	0,009941667	0	700	0,009978048	4	713	0,009994253	2	714	0,01	0
78	492	0,009799821	0	634	0,009941667	4	700	0,009978048	0	713	0,009994253	0	714	0,01	0
79	491	0,009779902	0	634	0,009941667	0	700	0,009978048	0	713	0,009994253	2	714	0,01	0
80	491	0,009779902	0	634	0,009941667	1	700	0,009978048	3	713	0,009994253	3	714	0,01	0
81	490	0,009759984	2	634	0,009941667	2	700	0,009978048	0	713	0,009994253	0	714	0,01	0
82	490	0,009759984	0	634	0,009941667	0	700	0,009978048	2	713	0,009994253	1	714	0,01	0
83	488	0,009720147	3	634	0,009941667	0	700	0,009978048	0	713	0,009994253	0	714	0,01	0
84	487	0,009700229	0	634	0,009941667	0	700	0,009978048	0	713	0,009994253	4	714	0,01	0
85	487	0,009700229	3	634	0,009941667	2	700	0,009978048	1	713	0,009994253	0	714	0,01	0
86	487	0,009700229	0	634	0,009941667	0	700	0,009978048	0	713	0,009994253	1	714	0,01	0
87	486	0,009680311	1	634	0,009941667	2	700	0,009978048	0	713	0,009994253	0	714	0,01	0
88	486	0,009680311	0	634	0,009941667	0	700	0,009978048	0	713	0,009994253	3	714	0,01	0
89	486	0,009680311	0	634	0,009941667	3	700	0,009978048	1	713	0,009994253	0	714	0,01	0
90	482	0,009600637	3	634	0,009941667	1	699	0,009963794	0	713	0,009994253	2	714	0,01	0
91	481	0,009580719	2	634	0,009941667	0	699	0,009963794	0	713	0,009994253	0	714	0,01	0
92	481	0,009580719	0	633	0,009925986	0	699	0,009963794	2	713	0,009994253	2	714	0,01	0
93	480	0,009560801	0	633	0,009925986	1	699	0,009963794	0	713	0,009994253	0	714	0,01	0
94	480	0,009560801	2	633	0,009925986	0	699	0,009963794	4	713	0,009994253	2	714	0,01	0
95	479	0,009540882	0	633	0,009925986	1	699	0,009963794	0	713	0,009994253	0	714	0,01	0
96	479	0,009540882	2	633	0,009925986	0	699	0,009963794	2	713	0,009994253	3	714	0,01	0
97	473	0,009421372	0	633	0,009925986	1	699	0,009963794	3	713	0,009994253	0	714	0,01	0
98	471	0,009381536	2	633	0,009925986	2	699	0,009963794	0	713	0,009994253	1	714	0,01	0
99	463	0,009222189	1	633	0,009925986	0	699	0,009963794	1	713	0,009994253	0	714	0,01	0
100	461	0,009182352	0	633	0,009925986	2	699	0,009963794	0	713	0,009994253	3	714	0,01	0

Code:

```
const chromosome = 100,
gen = 1000,
itterations = 100;
let generation = generateGeneration(chromosome, gen),
itterationNum = 0;
for (let i = 0; ++i < itterations;) {
generation = generation.sort((gen1, gen2) => {
return fitness(gen2) - fitness(gen1);
});
readOut(generation);
generation = nextGeneration(generation);
}
readOut(generation);
return;
function generateGeneration(count, size) {
let generation = [];
for (let i = -1; ++i < count;) {
let gen = [];
for (let j = -1; ++j < size;) {
gen.push(getRandomBinary());
}
generation.push(gen);
}
return generation;
}
function fitness(gen) {
let count = 0;
for (let i = -1; ++i < gen.length;) if (gen[i]) count++;
return count;
}
function nextGeneration(parentGeneration) {
let average = 0, allFitness=0,currFitness=0,probability==0,newGeneration = [],currProbability;
let firstIndex=0,secondIndex =0;
for (let i = 0; i < generation.length; i++) allFitness +=fitness(generation[i]);
for (let i = 0; ++i < chromosome;) {
let probability_1 = getRandomFloat(0, 1),
probability_2 = getRandomFloat(0, 1);
currFitness=fitness(generation[i]);
currProbability =currFitness/allFitness;
if (probability_1 >= currProbability) firstIndex = i;
if (probability_2 >= currProbability) secondIndex = i;
document.write(firstIndex + " " + secondIndex + " | ");
newGeneration.push(...crossover(parentGeneration[firstIndex],
parentGeneration[secondIndex]));
}
return newGeneration;
}
function crossover(firstParent, secondParent) {
```

```
let crossIndex = getRandomInt(1, firstParent.length);
let firstPartFirst = firstParent.slice(0, crossIndex),
secondPartFirst = firstParent.slice(crossIndex, firstParent.length),
firstPartSecond = secondParent.slice(0, crossIndex),
secondPartSecond = secondParent.slice(crossIndex, secondParent.length);
return [firstPartFirst.concat(secondPartSecond),
firstPartSecond.concat(secondPartFirst)];
}
function getRandomBinary(min, max) {
return Math.round(Math.random());
}
function getRandomInt(min, max) {
return Math.floor(Math.random() * (max - min)) + min;
}
function getRandomFloat(min, max) {
return (Math.random() * (max - min)) + min;
}
function bestCh (generation) {
let best = fitness(generation[0]); ;
  for (let i = 1; i < generation.length; i++) {
        if (fitness(generation[i])>best) best = fitness(generation[i]);
  }
return best;
}
function readOut(generation) {
let average = 0, allFitness=0,currFitness=0,probability==0;
for (let i = 0; i < generation.length; i++) allFitness +=fitness(generation[i]);
for (let i = 0; i < generation.length; i++){
currFitness=fitness(generation[i]);
probability=currFitness/allFitness;
document.write(currFitness + "|" + probability + '<br>');
}
averageCh = Math.floor(allsum / generation.length);
document.write("Average "+averageCh + '<br>');
document.write("Best" + bestCh(generation)+'<br>');
}
```