Application of Swarm Intelligence Solving the Shortest Route Problem

Gord Fedoriw, Corey Fehr, Merle Good, and Shawn Keown

Table Of Contents

Table Of Contents 2

Table Of Figures and Graphs 2

General purpose and direction of project 3

Implementation details and problems encountered 4

Figure 1. Initial Map Design 5

Figure 2. Modified Map Design 6

Final Results and Implications 8

Figure 3. Simple Map With Default Parameters 8

Figure 4. Results of 2 Sets of 100 Trials on Simple Map 1 9

Figure 5. BIG Map Example 9

Appendix A. Raw Data 11

Bibliography 13

Table Of Figures and Graphs

Figure 1. Initial Map Design 5

Figure 2. Modified Map Design 6

Figure 3. Simple Map With Default Parameters 8

Figure 4. Results of 2 Sets of 100 Trials on Simple Map 1 9

Figure 5. BIG Map Example 9

General purpose and direction of project

The idea behind our project is *Swarm Intelligence*, which is based on real-life observations of social animals (usually insects). For the purposes of this project, we have concentrated on research that has been done using ants. Marco Dorigo and his associates use ant behaviour as a principle tool in their book "Swarm Intelligence, From Natural to Artificial Systems" which was the primary source of research [1]. The ways that ants choose the paths that they travel between the nest and the food sources are of particular interest.

Dorigo mentions a few methods of solving problems that are difficult to efficiently resolve with traditional mathematical models. For example, the traveling salesperson and network load-balancing problems are large, difficult problems to solve because of their status as NP-Hard problems. Instead of following Dorigo's lead and using known algorithms to solve one of these difficult problems, we decided to start from scratch and use what we have learned about swarm intelligence and social insects to design our own algorithm to solve a somewhat simpler problem. Not only would this be a much more interesting pursuit but we would also learn more about swarm intelligence techniques. We realized that ants have a very reliable and effective method of choosing the shortest path to their food source. With this knowledge, we decided to attempt to create an application that would use an ant-like method of solving the shortest route problem.

Ants have an interesting method of transporting food to their nest. When they walk to a food source, they emit a substance known as *pheromone*. Ant pheromone is a very strong stimulant and when an ant senses pheromone, it greatly increases the probability that the ant will follow the trail of the pheromone. The level of pheromone (or rather, the amount of pheromone that has been left) on a certain path indicates the number of ants that have taken that path recently. The stronger the pheromone level, the more likely an ant is to take that route. When an ant is searching for food, it will be likely to take the most popular path.

Although the pheromone level does increase the chances that ant will follow a particular trail, it is not a guarantee that another ant will take that trail. Sometimes an ant will choose to take a path that has a relatively low pheromone level. This strange phenomenon as it turns out, is very important when it comes to adapting to a changing environment. If a new food source comes available that is closer to the nest than the previously known food source, then the fact that some ants will choose not to follow the pheromone means that this new food source can be found by a stray ant. The stray ants will establish new pheromone trails and the other ants will sense it. More ants can travel between the new food source and the nest in the time it takes ants to travel between the older food source and the nest. The pheromone level of the new path will increase and will draw more and more ants to the new food source and this means that less ants will be going to the old food source. While the new path increases in pheromone levels, the old path's pheromone level will be reduced. The reason for this is that pheromone is a time sensitive substance that dissipates over time. The randomness of the ant's choice of paths and the dissipation of pheromone are amazing features in the ant's method of finding the shortest route.

In designing our model, we considered these features. In order to model our design, we had to decide which features were relevant to our situation and which could be omitted. Although we knew about how the ant's method worked, there were still some things that we had little knowledge about. We were unsure how powerful the pheromone was. How did the level of pheromone correspond to the probability that an ant would take a particular path? We were also unsure about the dispersal of ants. In other words, how often would a new ant leave the nest in search of the goal (food source) and how many ants would we have to use before a particular route became a 'popular' route? Would an initial level of pheromone on a particular path effect the ant's behaviour? The length of time that it would take an ant to travel a certain distance was

another question we were interested in. Finally, we were interested in discovering how altering these different variables would affect the outcome.

Implementation details and problems encountered

Once the decision to create an Ant System Java applet to solve shortest route problem was decided, it was decided that we would create the algorithm ourselves. There is a TSP (Traveling Salesperson) algorithm given in Dorigo's book that was being used as part of our reference [1]. We could have initially trimmed the TSP algorithm down to an instance capable of solving the shortest route problem; we believed that we would benefit more by taking the knowledge gained by preparing a presentation on Swarm Intelligence and applying it to the implementation of our own algorithm.

Our initial design included the notions of City, Map, Route, Ants, and an Ant Factory. The design called for us to construct Maps out of Cities interconnected using Routes. It was these Maps that the threaded Ants would travel to determine the shortest route between an initial City and an end City. The design called for us to set up an AntGenerator at the initial City, which would create all of the Ants that would travel through the Map in succession. We provided for an Ant creation frequency factor which, if used correctly, would allow prior Ants to better update the pheromone trails on paths before latter Ants would have to make the decision of which Route to take. All Ants would start from the initial City and make one pass through the Map to the finish City at which point they would be killed. We allowed for variable inputs to the algorithm of the number of Ants that would be used in any given instance. At any given City, an Ant would have to make a choice of which departing Route to take out of the City. This decision was calculated with a nice linear probability that was purely based on the number of Ants that had previously taken each Route. The Map would need to initially indicate that one Ant had previously taken each Route to prevent the algorithm from assigning a zero percent probability to any given Route. We had included a Weighting Factor into the algorithm, which was supposed to be multiplied by the higher of the Route's pheromone level to give that Route a larger percentage and thus a higher probability of being traveled. This weighting factor, it was hoped, would cause the algorithm to converge on the correct result quicker.

For example, if there are two routes, A & B, leaving a given city, then the chance of taking route A is: $A \div (A+B)$

Now, taking the Weighting Factor into account the respective probabilities for A & B would be:

Route A (higher concentration) then: $(A \times weight) \div ((A \times weight) + B)$

Route B (lower concentration) then: $(B \div ((A \times weight) + B))$

We made the decision to include yet another variable into the algorithm as well. There was not enough disparity in the length of the Routes to actually get a proper time-based pheromone update rule, so we decided to include a linear factor to lengthen the Routes. Seeing as we had the Ants entering a Route, sleeping for the length of the Route, and updating the pheromone level on the Route, we decided we could get better results at subsequent routes by increasing disparity in the time Ants spent sleeping on their Routes. Rather than having the Ants sleep for the route length, it was decided that they would sleep for (*routelength* x *routemultiplier)*. All Maps in the system would be hard-coded into the system to prevent further complexity in areas other than the Swarm Intelligence aspect of our project. Furthermore, all Maps would follow a certain format to allow for greater control over result collection.

All Maps would have the form similar to this:

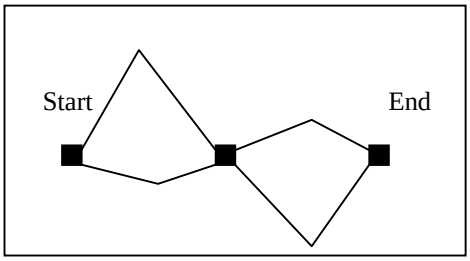

Figure 1. Initial Map Design

- All Routes would have a start City, an end City, and a mid-point.
- All Cities along the path would have two Routes leaving, both leading to the same City.

Generally, in nature, the level of pheromone on a particular path will dissipate over a variable amount of time allowing new paths to become dominant in changing environments. Since the Maps that we are using are static and never change, this brought into question the need for the dissipation of pheromone. Since this is a feature that is essentially used to adapt to a changing environment and the Maps were not going to change while the application was running, we decided not to implement the dissipation effect.

At this point, it was decided that the algorithm was complete and we were ready to implement it and test the results. Just for an overview, let us look at what this algorithm includes.

- Routes contain "Ants that traveled" statistics that are used for route selection. Ants choose a Route to travel then add themselves to the Route's count **after sleeping**.
- Ants sleep on the chosen Route for (*routelength* x *routemultiplier*).
- Ants choose which Route to take via a linear probability function.
- The most prevalent Route leaving a City has a better chance of being chosen because of the use of a linear weighting factor.
- Ants are generated continuously at a regular time interval that is specified by the Ant creation frequency.
- Ants make one pass, traveling from start to end before dying.
- All Routes are initially assigned a pheromone level of 1 to prevent a zero probability for any given Route.
- All Maps are static hard-coded examples with definite correct answers; consequently, we can easily compare this with the solution that the simulation discovers.
- A pheromone level on a path does not dissipate.

Once the algorithm was implemented as designed, we found that the results were not quite what we had envisioned. All results collected from the initial runs of the system seemed to be purely random. With further digging, it was discovered that all Ants being sent off from the start City to the end City would be facing the exact same choice among Routes. Since the pheromone level was not being updated until after the Ant's sleep time had expired, all Ants would approach the choice of Routes and face the exact same probabilities as all other Ants.

Consider an example with two Routes, A & B, leaving the start City each initialized to 1 for a pheromone level. Assume that the AntGenerator would send off new Ants at a frequency of 2 time units until 200 Ants have been sent. Assuming this, the AntGenerator would send off all Ants within 400 time units. Now assume that route A has a length of 500, and route B has a length of 1000. At the time that the first ant is sent off it will face the choice between A and B with a 50-50 probability that it will choose either route. Once the first Ant has chosen a route it will sleep for the designated time. Let us assume the Ant chose route A; therefore, the Ant will

sleep for 500 units assuming a sleep factor of 1. This first ant will not update the level of pheromone on Route A until after the sleep of 500 units has completed. Now assume 2 units of time have passed and the AntGenerator has created another Ant to send. This second Ant will now have the choice between Route A and Route B, each of which, indicate one Ant's prior travel. The second Ant will choose a Route based upon this 50-50 probability again. In fact, every one of the 200 Ants will face this exact same probability *before* the first Ant wakes-up and updates the pheromone level on Route A.

To solve this problem the algorithm was modified such that an Ant updates the pheromone level on a path *before* entering its sleep. Once this modification was made, the prior obvious random nature no longer appeared. Unfortunately, other questions had arisen that needed to be addressed.

It was assumed that we would be operating under the basic Map structure we had designed which was fine. What we did not realize was that, thinking about other types of Maps could help us to visualize the operation of our algorithm solving the problem at hand, which was in fact the Shortest Route Problem. A severe miscalculation that was apparent in our results was that the algorithm was not finding the shortest route in the Map. Instead, the algorithm was finding the shortest route from the current City to the next City. At this point, it was realized that we had violated one of the fundamental rules of building a good model. We had over-simplified it. Up until now, we had not been including any type of positive reinforcement for a shorter path. In nature and actual observations of real Ants colonies, the best path is reinforced by the fact that it is the shorter and quicker path to travel. Because the shorter path is quicker to travel, ants reaching the end will be more likely to choose the same path on the way back to the nest, which increases the amount of pheromone on the trail. It was decided that since we were trying to solve the over-all shortest route problem and not the *local* shortest route problem that we should modify the algorithm. Map rules were changed as we introduced a new style of Map along with the addition of Ants returning to the start City once reaching the End City.

An Example of the newly allowed Maps

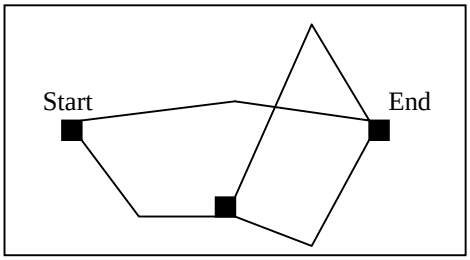

Figure 2. Modified Map Design

With the newly modified algorithm, we seemed to be on the right track as the Ants were traveling from the start City to the end City, but the new results did not seem quite right as the Ants traveled back from the end City to the start City. Actually, it seemed that the Ants were choosing their way back based on the numbers of Ants that had both traveled down a Route, as well as the number of Ants currently sleeping on the Route. We then came to realize that this was not correct.

For example, if there are two Routes entering the end City where one Route is twice as long as the other. Now assume that there is an Ant choosing the Route to take back, the Ant is not necessarily going to choose with correct probability. If there are pheromone levels of 50 on both Routes then the ant will choose with a 50-50 probability even if all of the Ants on the long Route are currently still sleeping on the Route while the shorter Route has actually had all 50 Ants travel the whole path. So, in fact, the choice the Ant was to make should have a probability

of 50-1 rather than 50-50. This problem was a direct result of the addition of Ants traveling back to the start City.

The fix for this problem was rather simple. All that needed to be done was to add another pheromone level to each Route, so now the Routes contained a start pheromone level and an end pheromone level. Then the only thing left was to change the way in which the Ants traveled their chosen Route. Instead of just updating the pheromone level on the path then sleeping, the Ant would now update the start pheromone level on the path, sleep, and then update the end pheromone level while the Ant was traveling from the start City to the end City. Conversely, when the Ant was traveling from the end City to the start City, they would update the end pheromone level, sleep, and finally update the start pheromone level once they had chosen a Route to take. Now, when an Ant was choosing a Route to take going forward, it would use the start pheromone level, and while it was choosing to go back, it would use the end pheromone levels of the Routes.

At this point, we were sitting with a fairly stable algorithm providing encouraging results. One main disappointment we were left with was the narrow margin of victory for the winning Routes. We were at a loss for ideas and decided to swallow our pride and see what the experts before us had done to achieve better results. We decided to look into how the Ants in the TSP algorithm of Dorigo's book [1] were choosing their Routes. What we found was quite helpful and interesting. It seems that, as a minor note, the initial value of 1 that we were using to initialize all of the Routes was a correct assumption, but that varying this value can give drastically different results. If you start with a low initial value, it does not take many Ants traveling on another path to make the probabilities swing wildly. However, using large numbers for initial value would not allow very large swings in the probabilities that are used to choose the Routes. Once the importance of this factor was realized, we decided to incorporate a modifiable initial value into the model. Secondly, while we had used a linear weighting factor applied to only the strongest Route as a method to speed up convergence it appeared that Dorigo had another idea. He used an exponential model for what we had called weighting factor in our model. The weighting factor that Dorigo used was applied to *all* routes, not just the most dominant one (as we had been doing).

Dorigo's Route probability calculation assuming two routes A & B was:

Probability for Route A = (*A*^ *weight*) ÷ ((*A*^ *weight*) + (B^ *weight*))

Probability for Route B = (*B*^ *weight*) ÷ ((*A*^ *weight*) + (B^ *weight*))

This greatly improved our algorithm and allowed us to see some very encouraging results. The very last things that we decided to change were more of programming issues rather than algorithm issues but they do have an impact on the situation. One of the last things we changed was the random number generation, which was running off one instance for all random decisions made. This resulted in somewhat nonrandom results. We solved this by giving each City its own random number generator and found that our results became much more reliable. A final change we made to the algorithm was done to magnify the differences in the lengths of the Routes. Originally, we were extending all of the Routes by a certain linear sleep factor to make all of the Ants spend that much longer on the Route they were on. After some investigation we found that this linear extension really did not have the effect that we required, which was to make a longer Route take comparably longer than a shorter Route. The resolution of this problem came about by using an exponential model for the sleep factor rather than the linear one. The only problem with doing this was that some of the Routes had large lengths to begin with and putting them to a large power results in some speed and memory issues for route lengths. It was decided to leave these as known problems that could be corrected later.

Our decision to create our own algorithm to solve this very intriguing problem enabled us to definitely learn volumes more than we would have if we had merely implemented a previously

designed algorithm. Furthermore, we believe the algorithm exhibits very good results considering the sheer lack of information required to be stored and 'remembered' by the agents.

Final Results and Implications

We felt that the best approach to quantifying our results in a somewhat reasonable manner was to run a number of trials and explore the resulting numbers. We ran one hundred separate trials of the simple map (see Figure 3.) using the default values for all parameters. We then ran one hundred more trials on the same map, but with an increased Ant Sleep Factor of

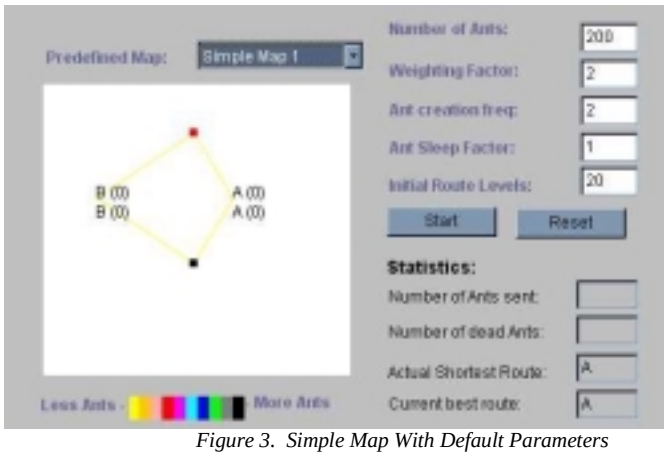

Figure 3. Simple Map With Default Parameters

1.5. This should provide some concrete connection between the size of the routes and the reliability of the results.

We recorded the number of Ants that took route A and Route B over each set of the hundred trials and computed the ratio between A and B. This ratio provides a key insight into the expressions of the system and the degree to which our results meet our expectations. As can be seen in Figure 4, as the ratio of route lengths increase our results become increasingly more reliable. The blue diamond line in the graph represents the base values of Simple Map 1 where the distance of Route B is 1.55 times longer than Route A. The pink square line indicates the same 100 trials but where Route B is 1.93 times longer than Route A.

Even with this slight change, the results are noticeably improved. The red line on the graph indicates the dividing line between correct and incorrect overall results. Here we can see that the slight increase in relative route length has resulted in approximately half as many incorrect results being calculated. With the base values, our calculations resulted in 33% of the readings being completely wrong. After increasing the difference between the route lengths, the resulting percentage of error dropped dramatically to 18%.

The change in the error rate was not the only one that was experienced. The results themselves were more correct. The mean for our initial set of trials was approximately 2.80 times as many ants took the shortest route as compared to the longer route. After applying the modified sleep factor that average rose to 3.58. Given that all of the trials used 200 Ants (going both ways), we can actually calculate the average number to take the shortest route in both cases. This small modification in route length resulted in an average of 30 Ants moving over to the correct route. On the extremes, the ratios moved from 11.12 to 14.38 for our best results and from 0.08 to 0.17 for our worst.

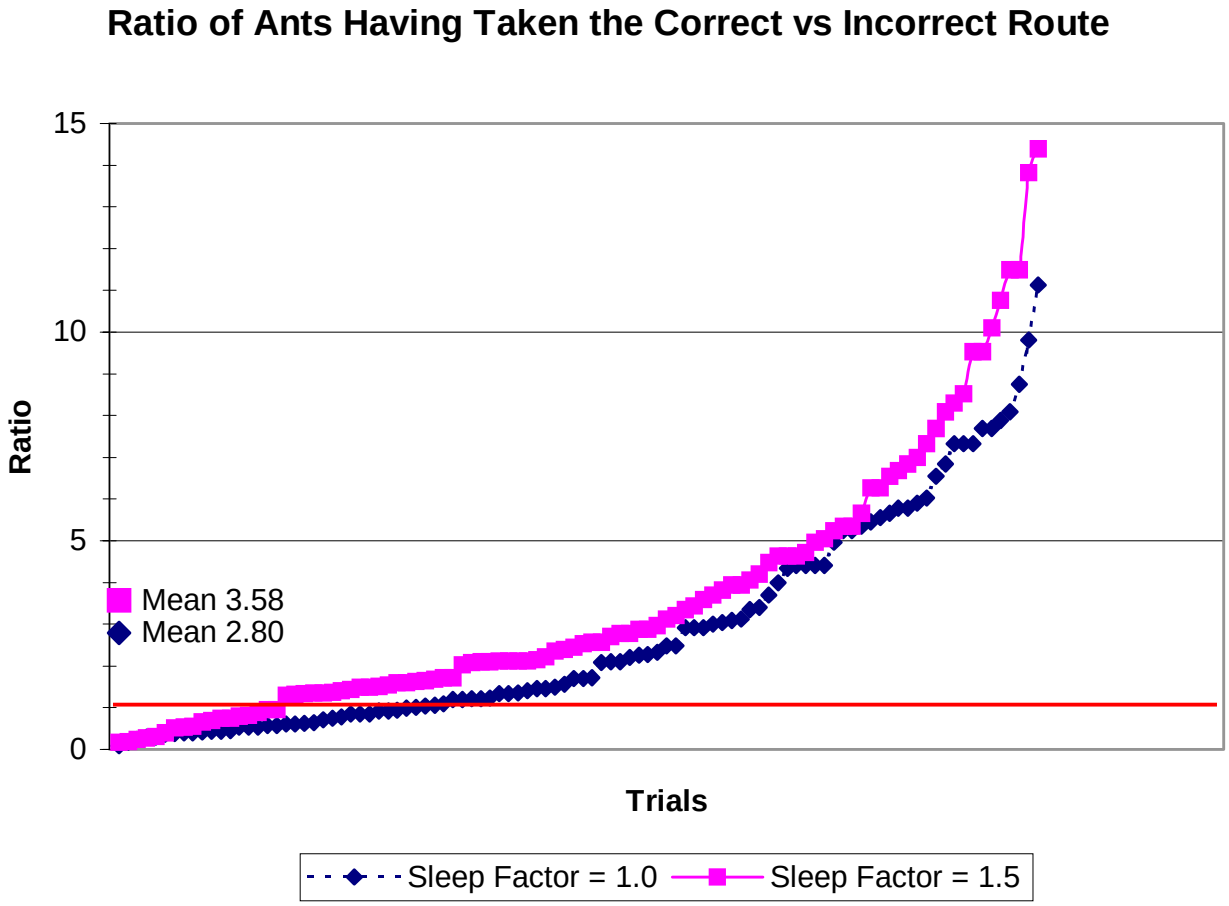

Figure 4. Results of 2 Sets of 100 Trials on Simple Map 1

In both cases, there was a marked improvement.

The data that we have presented here is only on the simpliest of maps and of course, results are likely to vary as the maps are scaled up. We did do some preliminary testing on a much larger map (see Figure 5) and had promising results. For the most part, we did receive the correct shortest route, however the number of Ants used needed to be dramatically increased

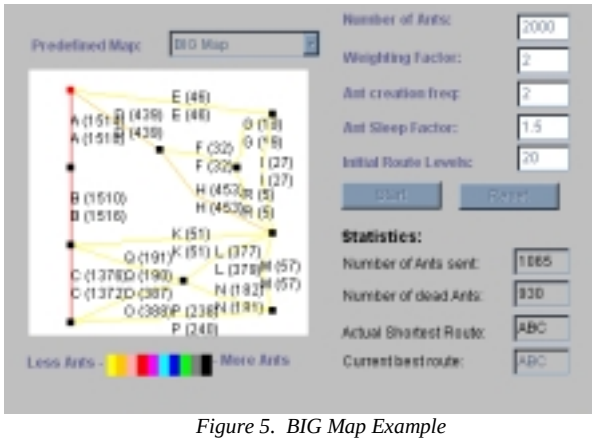

Figure 5. BIG Map Example

as the size of the map increased. This is largely due to the continuous stream of Ants that is required to get good results.

Our results could be a little misleading however, and so should be taken with caution. Only one factor was changed during our trials and although we saw improvements in our results further study would be needed to draw concrete and irrefutable results. We have also noted that despite our attempts to properly generate random numbers, there are certain times that the results are definitely skewed by a non-random generation of choices. In the worst case that we noticed, before the first Ant had even started on the route back, an imbalance of 5 to 1 was in place in favour of the worst route. As mentioned earlier this imbalance can have a drastic impact on our results. This is probably one of the greatest limitations with our model.

If anything our tests have pointed out what a huge can of worms (or ants if you will) we have opened. Although we present no concrete data here, our experiments with the factors show that particular values can drastically alter the results obtained. Given the small scale of our project and our limited time frame it is beyond the scope of our project to do the analysis that this program truly deserves. It is also important to point out that to do a detailed analysis would likely require much more statistical analysis. This project would be more in line with that of a graduate student and could provide a good starting point for some kind of further analysis.

Although we have only scratched the surface of this complex problem, we feel that we have made excellent progress in understanding the situation. Our applet has proved to be just the first step in our pursuit of Swarm Intelligence, but has definitely provided us with a firm foundation from which further investigation can hopefully take place. You can obtain a copy of our applet at: www.cpsc.ucalgary.ca/~keown/CPSC533/TermProject.html.

Appendix A. Raw Data

Sleep Fator = 1.0				Sleep Factor = 1.5			
Trial	Ants Taken Correct Route	Ants Taken Incorrect Route	Ratio	Trial	Ants Taken Correct Route	Ants Taken Incorrect Route	Ratio
1	33	367	0.0899183	1	57	343	0.1661808
2	58	342	0.1695906	2	64	336	0.1904762
3	76	324	0.2345679	3	76	324	0.2345679
4	91	309	0.2944984	4	85	315	0.2698413
5	92	308	0.2987013	5	94	306	0.3071895
6	108	292	0.369863	6	114	286	0.3986014
7	112	288	0.3888889	7	136	264	0.5151515
8	113	287	0.3937282	8	139	261	0.532567
9	114	286	0.3986014	9	142	258	0.5503876
10	117	283	0.4134276	10	160	240	0.6666667
11	121	279	0.4336918	11	165	235	0.7021277
12	122	278	0.4388489	12	172	228	0.754386
13	126	274	0.459854	13	172	228	0.754386
14	140	260	0.5384615	14	178	222	0.8018018
15	141	259	0.5444015	15	180	220	0.8181818
16	141	259	0.5444015	16	188	212	0.8867925
17	145	255	0.5686275	17	195	205	0.9512195
18	147	253	0.5810277	18	196	204	0.9607843
19	151	249	0.6064257	19	226	174	1.2988506
20	151	249	0.6064257	20	228	172	1.3255814
21	153	247	0.6194332	21	229	171	1.3391813
22	156	244	0.6393443	22	230	170	1.3529412
23	167	233	0.7167382	23	230	170	1.3529412
24	170	230	0.7391304	24	231	169	1.3668639
25	175	225	0.7777778	25	234	166	1.4096386
26	183	217	0.843318	26	236	164	1.4390244
27	183	217	0.843318	27	240	160	1.5
28	184	216	0.8518519	28	240	160	1.5
29	191	209	0.9138756	29	241	159	1.5157233
30	192	208	0.9230769	30	243	157	1.5477707
31	194	206	0.9417476	31	246	154	1.5974026
32	199	201	0.9900498	32	246	154	1.5974026
33	200	200	1	33	248	152	1.6315789
34	204	196	1.0408163	34	249	151	1.6490066
35	206	194	1.0618557	35	251	149	1.6845638
36	209	191	1.0942408	36	253	147	1.7210884
37	218	182	1.1978022	37	253	147	1.7210884
38	218	182	1.1978022	38	268	132	2.030303
39	219	181	1.2099448	39	270	130	2.0769231
40	220	180	1.2222222	40	271	129	2.1007752
41	221	179	1.2346369	41	271	129	2.1007752
42	229	171	1.3391813	42	272	128	2.125
43	229	171	1.3391813	43	272	128	2.125
44	230	170	1.3529412	44	272	128	2.125
45	234	166	1.4096386	45	272	128	2.125
46	237	163	1.4539877	46	273	127	2.1496063
47	237	163	1.4539877	47	276	124	2.2258065
48	240	160	1.5	48	281	119	2.3613445
49	244	156	1.5641026	49	282	118	2.3898305
50	252	148	1.7027027	50	284	116	2.4482759
51	252	148	1.7027027	51	287	113	2.539823
52	253	147	1.7210884	52	288	112	2.5714286
53	270	130	2.0769231	53	288	112	2.5714286
54	271	129	2.1007752	54	292	108	2.7037037
55	271	129	2.1007752	55	294	106	2.7735849
56	275	125	2.2	56	294	106	2.7735849
57	277	123	2.2520325	57	297	103	2.8834951
58	278	122	2.2786885	58	297	103	2.8834951
59	280	120	2.3333333	59	299	101	2.960396
60	285	115	2.4782609	60	303	97	3.1237113
61	285	115	2.4782609	61	305	95	3.2105263
62	298	102	2.9215686	62	308	92	3.3478261
63	298	102	2.9215686	63	310	90	3.4444444
64	298	102	2.9215686	64	313	87	3.5977011
65	300	100	3	65	315	85	3.7058824

Sleep Fator = 1.0				Sleep Factor = 1.5			
Trial	Ants Taken Correct Route	Ants Taken Incorrect Route	Ratio	Trial	Ants Taken Correct Route	Ants Taken Incorrect Route	Ratio
66	301	99	3.040404	66	317	83	3.8192771
67	302	98	3.0816327	67	319	81	3.9382716
68	303	97	3.1237113	68	319	81	3.9382716
69	308	92	3.3478261	69	321	79	4.0632911
70	309	91	3.3956044	70	323	77	4.1948052
71	315	85	3.7058824	71	327	73	4.4794521
72	320	80	4	72	329	71	4.6338028
73	325	75	4.3333333	73	329	71	4.6338028
74	326	74	4.4054054	74	329	71	4.6338028
75	326	74	4.4054054	75	330	70	4.7142857
76	326	74	4.4054054	76	333	67	4.9701493
77	326	74	4.4054054	77	334	66	5.0606061
78	333	67	4.9701493	78	336	64	5.25
79	336	64	5.25	79	337	63	5.3492063
80	336	64	5.25	80	337	63	5.3492063
81	337	63	5.3492063	81	340	60	5.6666667
82	338	62	5.4516129	82	345	55	6.2727273
83	339	61	5.557377	83	345	55	6.2727273
84	340	60	5.6666667	84	347	53	6.5471698
85	341	59	5.779661	85	348	52	6.6923077
86	341	59	5.779661	86	349	51	6.8431373
87	342	58	5.8965517	87	350	50	7
88	343	57	6.0175439	88	352	48	7.3333333
89	347	53	6.5471698	89	354	46	7.6956522
90	349	51	6.8431373	90	356	44	8.0909091
91	352	48	7.3333333	91	357	43	8.3023256
92	352	48	7.3333333	92	358	42	8.5238095
93	352	48	7.3333333	93	362	38	9.5263158
94	354	46	7.6956522	94	362	38	9.5263158
95	354	46	7.6956522	95	364	36	10.111111
96	355	45	7.8888889	96	366	34	10.764706
97	356	44	8.0909091	97	368	32	11.5
98	359	41	8.7560976	98	368	32	11.5
99	363	37	9.8108108	99	373	27	13.814815
100	367	33	11.121212	100	374	26	14.384615
Averages	244.57	155.43	2.8010418	**Averages**	271.79	128.21	3.578835
	Length A	**Length B**	**Ratio B : A**		**Length A**	**Length B**	**Ratio B : A**
	116	180	1.5517241		1249	2414	1.9327462

Bibliography

[1] Eric Bonabeau, Marco Dorigo, Guy Theraulaz. *Swarm Intelligence from Natural to Artificial Systems*. Oxford University Press, 1999.