ATG Press Research Kit

Something New to the Traveling Salesman Problem

Executive Summary

January 2008

MOTIVATIONS

The Coordinated Brute Force method, as introduced in our Appetizer "Globe Optimizer," has a unique property: it is originally an exact solution to hard computational problems, but it can be easily scaled back to a heuristic method only by decreasing the amount of the required space (memory). So it is very desirable to see the performance of this method when it is scaled back as a heuristic one. It is what this research kit is about. To this end, we selected the TSP as it is a strong NP-C problem without any approximation scheme [1], and it would be interesting to see the approximation rate of the proposed heuristic method in some practical samples of the TSP. These approximation rates along with the consumed computational resources are a good reference in the evaluation of this method, especially its applicability in practical applications.

IMPLEMENTATION

We know that as we decrease the size of the method required space, the probability of producing a more inexact solution is increased. Also, we know that the required space of the method is proportional to the width and the length of the space tape [2]. As the length of the tape is equal to the length of the shortest path, we have no control over it. In contrast, we can fix the width of the tape to a desired value without affecting the basis of the method. To this end, we developed two different versions of the proposed method. At the first version, the width of the tape was fixed to k, and at the second, the width was fixed to $n \times k$, where n is the number of cities and k is an arbitrary positive integer.

The most limited form of the space tape is the first version with $k = 1$, and its subsequent form with $k = 2$. In terms of accuracy, these forms are the most inaccurate forms of the method, but in terms of speed, they are the fastest forms. These two forms when used for benchmarking, can give us two key consequences. Firstly, they show how much reliable is the method in nature by seeing the results in its weakest form. Secondly, how much fast the method can run by seeing the calculation time of its fastest form.

Therefore, we used the first version for benchmarking purpose, and the second version to show a more accurate and sophisticated form of the method to the user of this research kit. We selected the sample problems from TSPLIB, where the exact solutions of the samples are known, and therefore, it is possible to calculate the approximation rates. We selected 24 samples, 14 symmetric and 10 asymmetric. The average size (number of cities) of asymmetric samples in TSPLIB is smaller than the symmetric ones. It is due to the more complication in the calculation of exact solution in asymmetric TSP.

RESULTS

We run the first version of the method on an Intel Pentium M 750 machine (6050 MIPS), with 533 MHz front side bus and 1 GB of RAM. We set the k to 1 and 2, respectively. The following tables show the results including the name of the sample problem at TSPLIB, the length of the exact solution, the length of the resulted shortest paths for $k = 1$ and 2, the approximation rates, and the consumed CPU time in seconds.

Sample name	Exact solution	Solution for *k* =1	Approximation rate for *k* = 1	Consumed CPU time	Solution for *k* =2	Approximation rate for *k* = 2	Consumed CPU time
a280	2,579	3,425	1.32	52 s	3,403	1.31	110 s
berlin52	7,542	8,344	1.10	2 s	7,955	1.05	4 s
ch150	6,528	8,035	1.23	17 s	7,854	1.20	33 s
d2103	80,450	102,640	1.27	1,410 s	100,008	1.24	3,798 s
eil101	629	816	1.29	1 s	780	1.24	3 s
fnl4461	182,566	248,721	1.36	26,376 s	246,078	1.34	58,125 s
kroA200	29,363	37,278	1.27	121 s	36,302	1.23	345 s
kroB200	29,437	36,732	1.24	123 s	37,368	1.27	336 s
pcb1173	56,892	78,063	1.37	708 s	77,701	1.36	1851 s
rat195	2,323	2,965	1.27	10 s	2,709	1.16	18 s
rat783	8,806	11,870	1.34	2,424 s	11,699	1.32	5,490 s
st70	675	868	1.28	1 s	799	1.18	1 s
ts225	126,643	147,961	1.16	1,095 s	142,659	1.12	2,010 s
tsp225	3,919	4,969	1.26	25 s	4,874	1.24	46 s

Table 1- Results for symmetric samples

Sample name	Exact solution	Solution for *k* =1	Approximation rate for *k* = 1	Consumed CPU time	Solution for *k* =2	Approximation rate for *k* = 2	Consumed CPU time
br17	39	47	1.20	1 s	45	1.15	1 s
ft53	6,905	8,344	1.20	2 s	8,040	1.16	4 s
ftv33	1,286	1,497	1.16	1 s	1,406	1.09	1 s
ftv38	1,530	1,757	1.14	1 s	1,633	1.06	1 s
ftv70	1,950	2,298	1.17	2 s	2,330	1.19	2 s
ftv170	2,755	3,589	1.30	8 s	3,382	1.22	15 s
kro124	36,230	41,928	1.15	29 s	40,988	1.13	64 s
rbg323	1,326	1,673	1.26	35 s	1,615	1.21	57 s
rbg403	2,465	3,253	1.32	76 s	3,160	1.28	163 s
rbg443	2,720	3,783	1.39	120 s	3,713	1.36	245 s

Table 2- Results for asymmetric samples

The interesting point here is that the Coordinated Brute Force method treats both of the symmetric & asymmetric samples in a same way without any lack of performance. It is one of the great achievements of this method.

SAMPLE PATHS

It may be of interest to see the generated paths by the method. So we have included the result files for all samples in this research kit. These files include the shortest path along with time stamps which show the consumed CPU time. But here, we just bring up the generated paths for two samples: ts225 as a symmetric and ftv170 as an asymmetric sample. The followings are the generated paths for ts225 and ftv170 for *k* = 1.

Start of tour (ts225):
221, 99, 100, 200, 199, 198, 197, 196, 75, 175, 174, 173, 172, 171, 50, 150, 149, 148, 147, 146, 25, 24, 23, 22, 21, 20, 19, 141, 142, 143, 144, 145, 44, 166, 167, 168, 169, 170, 69, 191, 192, 193, 194, 195, 94, 216, 217, 218, 219, 220, 119, 120, 121, 122, 123, 124, 125, 225, 224, 223, 222, 98, 97, 96, 95, 93, 92, 91, 90, 89, 88, 211, 212, 213, 214, 215, 118, 117, 116, 115, 114, 113, 112, 111, 110, 109, 108, 107, 210, 209, 208, 207, 206, 82, 185, 184, 183, 182, 181, 57, 160, 159, 158, 157, 156, 32, 135, 134, 133, 132, 131, 7, 8, 9, 10, 11, 12, 13, 136, 137, 138, 139, 140, 38, 161, 162, 163, 164, 165, 63, 186, 187, 188, 189, 190, 87, 86, 85, 84, 83, 81, 80, 79, 106, 105, 104, 103, 102, 101, 205, 204, 203, 202, 201, 76, 180, 78, 77, 179, 178, 177, 176, 51, 155, 154, 153, 152, 151, 26, 130, 129, 29, 30, 31, 33, 34, 35, 36, 37, 39, 40, 41, 42, 43, 45, 46, 47, 48, 73, 74, 72, 71, 70, 68, 67, 66, 65, 64, 62, 61, 60, 59, 58, 56, 55, 54, 53, 52, 27, 128, 28, 127, 126, 1, 2, 3, 4, 5, 6, 14, 15, 16, 17, 18, 49, 221
Length: 147961

Start of tour (ftv170):
103, 104, 164, 100, 96, 97, 98, 166, 105, 115, 110, 109, 167, 94, 93, 92, 155, 90, 91, 89, 154, 88, 86, 87, 84, 85, 72, 61, 50, 171, 74, 78, 79, 83, 80, 81, 82, 1, 2, 3, 4, 5, 6, 170, 112, 113, 133, 134, 135, 132, 114, 165, 128, 117, 118, 119, 120, 121, 122, 125, 129, 131, 136, 139, 140, 141, 142, 7, 8, 9, 15, 14, 18, 19, 20, 21, 22, 33, 159, 37, 158, 34, 35, 157, 41, 46, 47, 48, 49, 52, 53, 54, 44, 56, 55, 59, 58, 63, 62, 69, 68, 168, 71, 70, 67, 64, 65, 57, 43, 42, 156, 40, 39, 36, 32, 31, 29, 30, 23, 24, 27, 25, 16, 26, 151, 161, 152, 153, 143, 144, 148, 138, 137, 130, 111, 108, 107, 106, 99, 102, 101, 163, 124, 123, 147, 146, 145, 149, 150, 162, 10, 11, 77, 75, 76, 12, 13, 28, 160, 17, 38, 45, 60, 51, 169, 73, 66, 95, 116, 127, 126, 103
Length: 3589

THE PACKAGE

In addition to this executive summary, this research kit contains a research report which includes the technical details of the two implementations. The most important part of this report is a section which illustrates how the proposed method should be calibrated in accordance with our requirements and computational resources.

The implementations are two separate Borland (CodeGear) Delphi projects which can be compiled by Delphi 5 or higher versions for Win32. Borland also ships a free version of its Delphi 10 for Win32 under the name *Turbo Delphi Explorer for Win32*. The proposed implementations have a very simple interface and can directly accept TSPLIB files as their inputs. Their outputs are simple text files including the complete tours and time stamps. We have also included 59 output files of the first implementation and 22 output files of the second one in this research kit. The source codes are completely documented so they can be easily converted and reused under different platforms.

REFERENCES

[1] C. H. Papadimitriou, “*Computational Complexity*,” First Ed. Addison Wesley, 1994.
[2] F. Laleh, “Globe Optimizer,” *ATG Press Appetizer*, Vol. 1, pp. 18-24, 2005.