REU Report: Simulated Annealing Optimization

by Le Wang
Advisor: Linda Schmidt

U.G. 99-5

Simulated Annealing Optimization

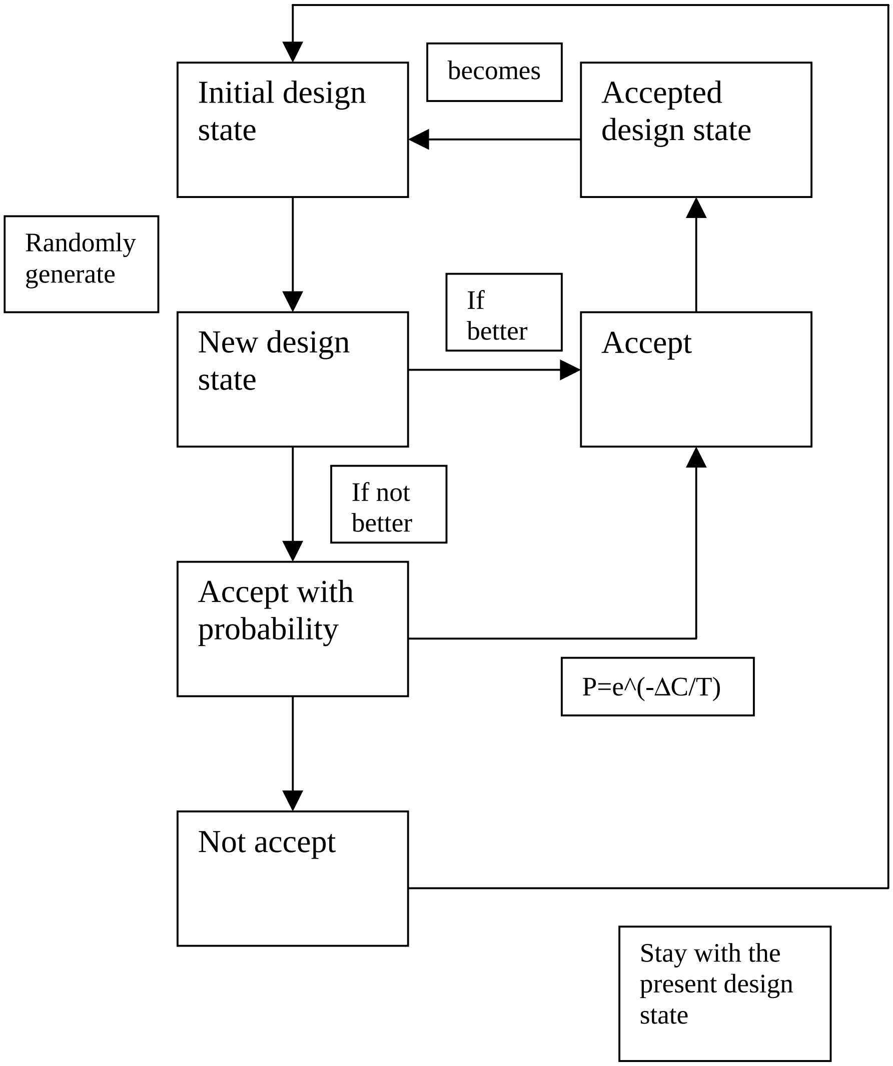
Research Experience for Undergraduates
Le Wang
8/5/99

Introduction

The process of annealing is to heat a piece of stressed metal to a certain temperature and letting it cool down slowly to relieve the stress in the metal. When the stressed metal is heated, the molecules become more energetic and start to vibrate, and as the molecules cool down, they form a more stabilized structure, therefore relieve the stress in the metal. The process of simulated annealing process is an algorithm that requires a long and excessive search time, therefore the goal of this project is to shorten the search time of the algorithm but still give good results. This project uses the concept of statistical process control to find where a non-productive search start, so it can decrease the search time. The project also focuses on a couple variations of the Detection of Productive Search algorithm to find which one is the better algorithm.

The Process of Simulated Annealing

The simulated annealing process is an optimization algorithm. It starts at an initial design state, and then randomly generates a new design state. The process compares the new design state with the initial design state to see if it is a better design. If better, it will accept this new design state and this new design state becomes the initial design state. If not better, it will accept it with a probability, $p=e^{(-\Delta C/T)}$, if not accepted it will go to the initial design state and start the process again, and this iteration will continue until a close to optimal design state is found. The probability changes as temperature changes, it starts at a high probability, meaning the process will accept a worse design state many times in the beginning to make sure that the design solution is not at a local optimal solution. And as temperature decrease as the process continues, the probability also decreases, so the simulated annealing process will accept little or no worse solutions to get close to the optimal design state. Here is a flow chart of the simulated annealing process.



The extensive search process

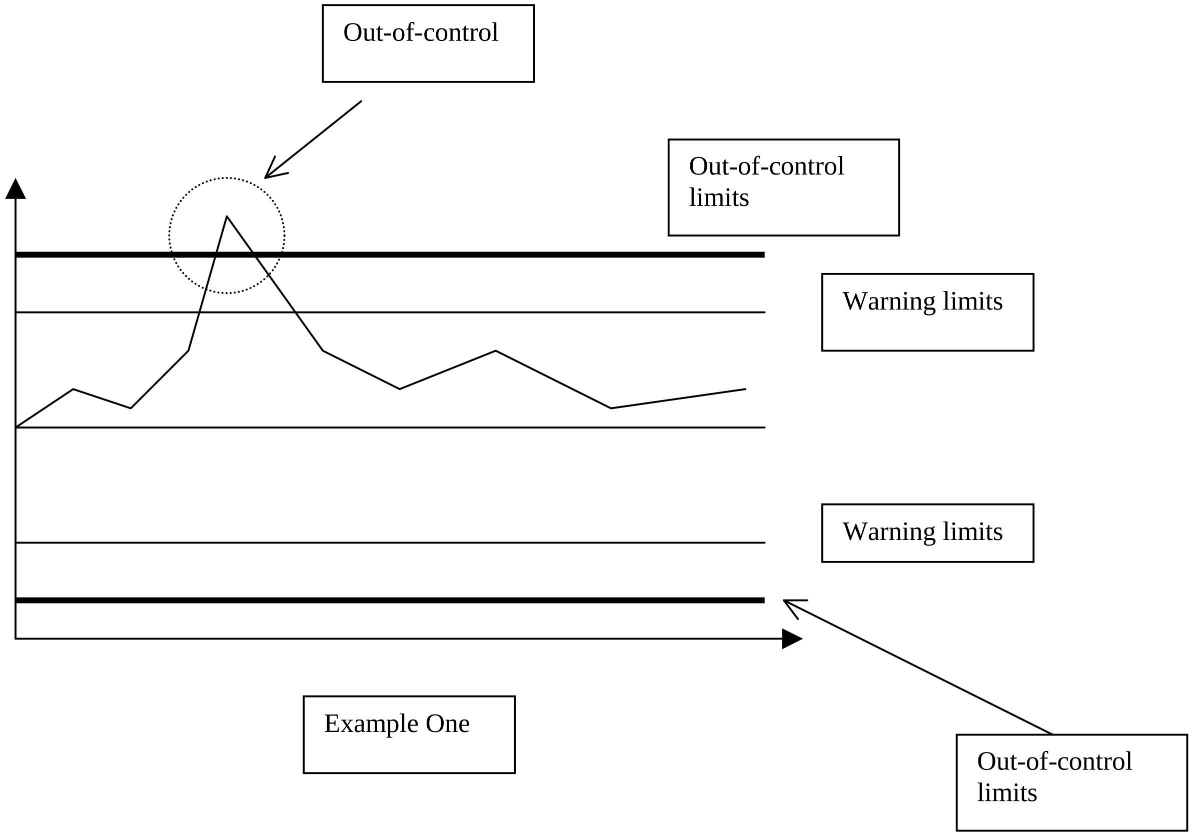
The problem with the simulated annealing process is the wasted search time a certain temperature. This is because of the probability of accepting a worse design state which causes the algorithm to find a close to optimal solution and the search time beyond this close to optimal design state is non-productive search. In order to shorten the search time and still gives good result, we need to find where the non-productive search starts and then stop the searching.

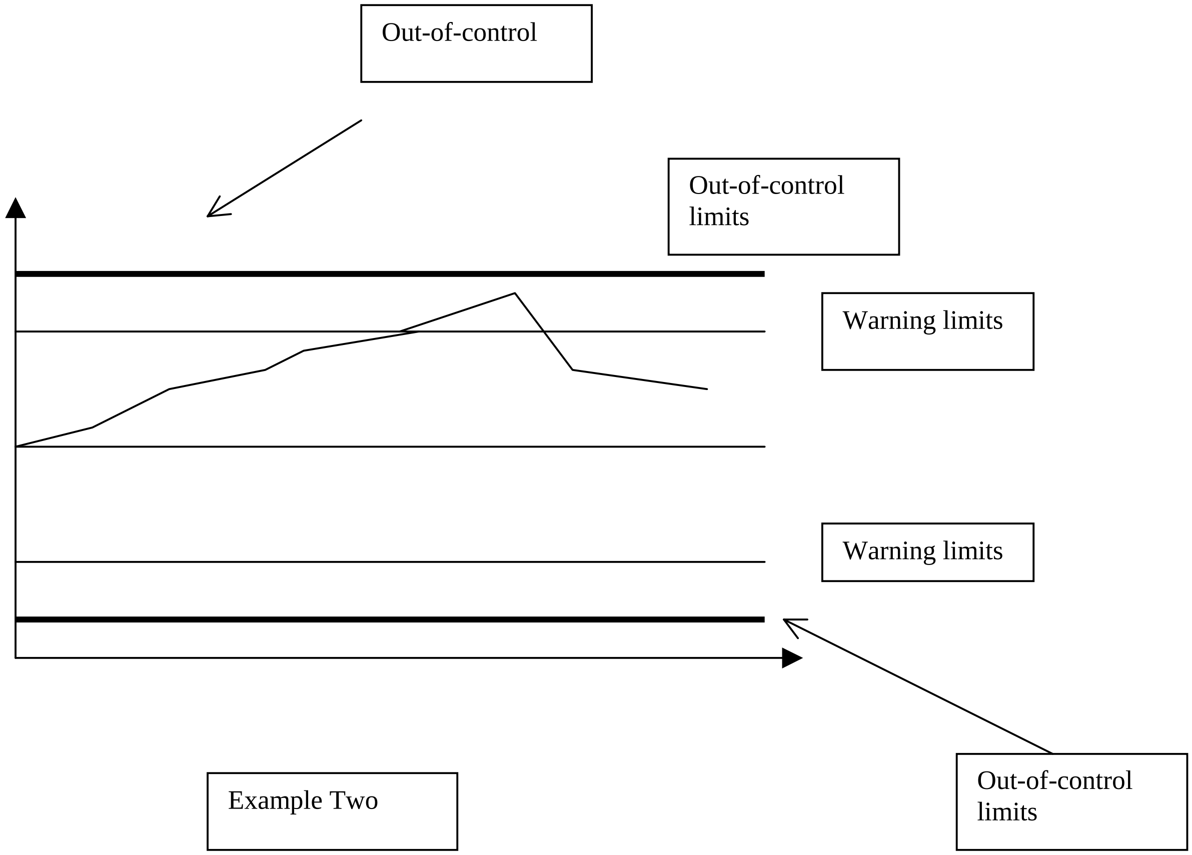
How Detection of Productive Search works

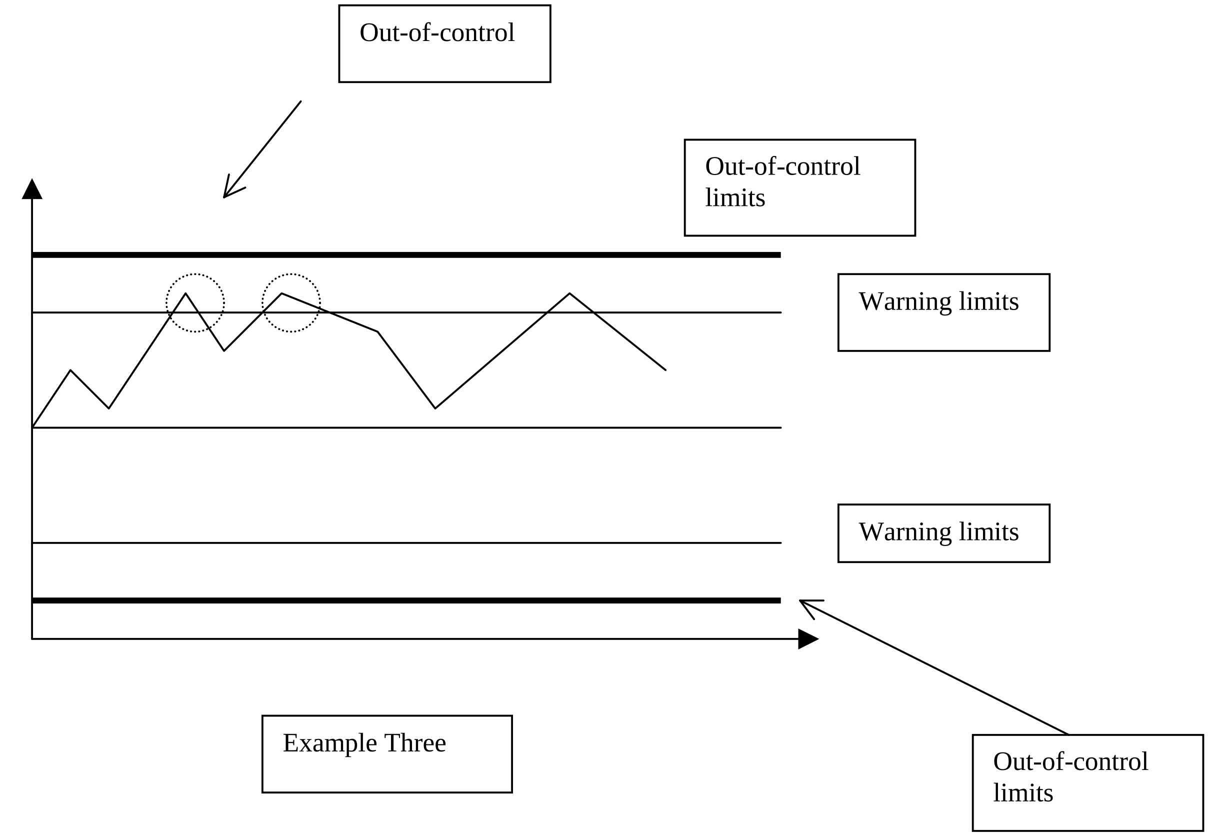
The Dectection of Productive Search algorithm uses the statistical process control, SPC, concept to find where is productive search is. The statistical process control concept is a way of detecting if a process is giving out in-control or out-of-control results. In the simulated annealing process, the out-of-control region of the SPC is actually the productive search part of the process, because at the out-of-control region the sample data (design states) change a lot, and at that point the algorithm is still searching for the close to optimal design states. At the in-control region, the sample data do not change very often, at that point the algorithm is not finding better design states, therefore maybe already at the close to optimal design states, so should stop the search there.   There are three ways of finding when the sample data are in the out-of-control region.

1. One Sample is out of action limits.
2. Two out of three successive sample data outside of warning limits.
3. A run of six consecutive sample data up or down.

The action limits is the approximate overall data average plus and minus 3 approximate standard deviations. The warning limits is the approximate overall data average plus and minus 2 approximate standard deviations. The reason that it is approximate is because that we update the average and standard deviations as the process continues.


Out-of-control
Out-of-control limits
Warning limits
Warning limits
Example One
Out-of-control limits


Out-of-control
Out-of-control limits
Warning limits
Warning limits
Example Two
Out-of-control limits

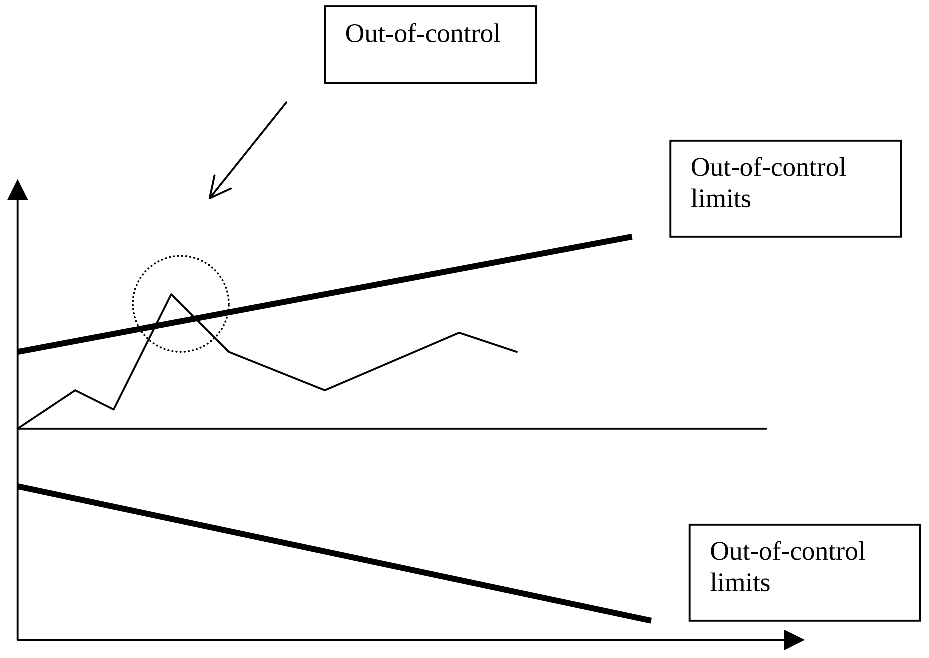


To Test the DPS using the Traveling Salesman Problem

The Traveling Salesman Problem asks to determine the minimum-length of a closed tour by which a salesman will visit each city only one time. We use 100 cities to test the algorithm. We first randomly generate a combination of cities (initial design state), then randomly changes the combination (new design state). We can see if that makes a better tour length and if not, according to the probability at the time we can determine if to accept the design state or not.

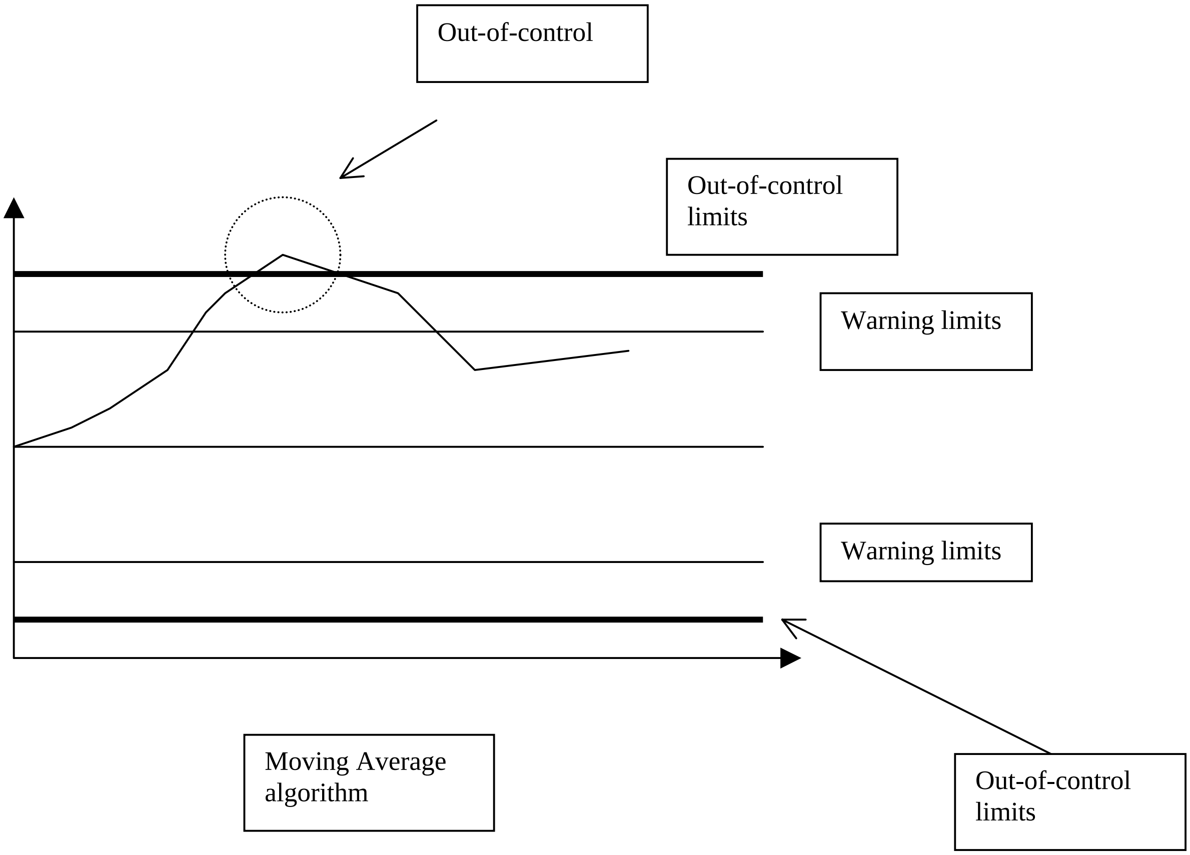
Results

Previous results (1) have shown that when comparing DPS to other algorithms the iterations and average iterations per temperature of DPS is lower than other algorithm, which tells us that DPS is not searching as long as other algorithms. The tour length data are also lower for DPS, so DPS is still giving good tour length results with less searching iterations. This concludes that DPS solved some of the excessive search time of the process. The main focus of this project is to compare DPS to other very similar algorithms such as Cumulative Sum, and Moving Average. They are variations of DPS, so maybe they will give better results or less search time. The Cumulative Sum algorithm's out-of-control limits are not parallel to the approximate average, this algorithm is for processes that does not have independent data samples.



Cumulative Sum algorithm

Another algorithm similar to DPS is the Moving Average algorithm. The graph of algorithm is the same, but it is not the samples that are drawn in the graph, it is the average of every 5 samples.



From the data we collected, we applied the analysis of variance technique to the data sets, and we found that DPS and Moving Average are giving almost the same tour length results, but as iteration limits goes higher, DPS would give longer iteration than Moving Average. But we found there are some problems with the Cumulative Sum algorithm, it's results remains almost constant as iteration limits changes. We are trying to print out some processing data of the Cumulative Sum to find out why is it's results are almost constant as iteration limit changes.

Data

	Cumulative Sum		DPS		Moving Averages	
Iter/Temp	**1000**		**1000**		**1000**	
Seed	**Iterations**	**Tour Length**	**Iterations**	**Tour Length**	**Iterations**	**Tour Length**
0	110748	31785	306634	27356	238352	29331
1	113717	31079	237841	28819	177928	32494
2	113694	29062	249969	26482	256848	32326
3	85885	33459	239476	28665	236948	26975
4	149106	33704	273512	29122	170179	28145
5	115731	29794	229509	28504	234539	29354
6	196052	29447	405016	24688	151815	31163
7	125243	31430	428890	28724	223075	28468
8	123040	30019	360994	27110	241363	27613
9	93349	34244	413061	26354	153591	26787
10	201901	33459	279807	32535	173155	32769
11	152952	30275	329454	29787	286467	27606
12	127230	30753	317625	26650	197981	25563
13	165989	33512	245598	29604	233616	26822
14	98736	31707	223535	28514	216942	29097
15	147421	31802	238911	27185	316732	26563
16	171002	30282	235034	28918	205333	29258
17	182058	29851	211037	27425	243300	28061
18	105059	30348	292916	27141	217007	30325
19	77936	31431	275063	29025	202916	30388
20	123019	28538	221175	28920	239830	32811
21	119451	32475	213069	33724	272046	27294
22	192600	33139	230729	27111	269385	27385
23	135488	31475	227018	26626	273412	32357
24	132700	35505	146365	33054	182466	27752
25	319008	29438	203782	28448	203875	26711
26	127335	30119	258951	33435	213664	28593
27	154875	34057	213213	30262	261701	31899
28	196223	32279	237673	30805	243513	28355
29	153523	30400	160075	28999	221930	28780
30	94706	34906	284426	27831	199266	28984
AVG.	**142122**	**31606**	**264205**	**28768**	**224490**	**29033**
STD. DEV	**47362**	**1840**	**67523**	**2158**	**39107**	**2086**

This is the data for iteration limit of 1000 iterations per temperature.

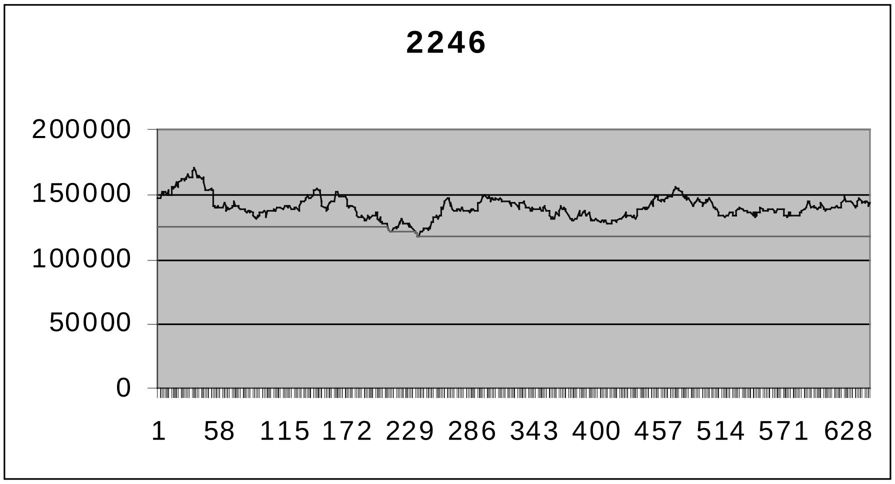
	Cumulative Sum		DPS		Moving Averages	
Iter/Temp	2500		2500		2500	
Seed	Iterations	Tour Length	Iterations	Tour Length	Iterations	Tour Length
0	72263	29692	350800	29686	294908	27733
1	59989	32066	393731	27282	318012	26261
2	164108	30672	258139	28306	334976	29836
3	85885	33459	336807	29464	251797	30271
4	156791	30402	343916	30320	328356	26087
5	107626	29137	392660	31217	333254	27027
6	123124	32066	320964	26903	369389	28228
7	167111	32131	432983	26241	395273	25368
8	90845	34007	359355	26613	265426	32544
9	120127	31171	338898	25331	323972	26400
10	114591	30627	330597	31032	294908	27733
11	91694	33121	385665	28975	299491	28691
12	135216	29751	404472	27770	372147	27991
13	305372	31442	414025	29551	291836	27877
14	198717	31920	287068	27312	286166	30474
15	93567	29759	432979	26466	355474	27473
16	174274	34717	360546	27528	333527	30363
17	140073	34146	334160	26958	403694	28868
18	122845	32521	282302	24750	276936	27838
19	92053	30551	478266	29598	343775	27013
20	123019	28538	329351	28324	388256	29399
21	235014	29045			317463	27536
22	99321	29207				
23	143697	29147				
24	233339	34465				
25	86927	31668				
26	143771	35689				
27	150980	29459				
28	123498	30928				
29	192561	30601				
30	106991	33806				
AVG.	**137271**	**31481**	**360366**	**28077**	**326320**	**28228**
STD. DEV	**53518**	**1945**	**54521**	**1796**	**42096**	**1715**

This is the data for iteration limit of 2500 iterations per temperature.

	Cumulative Sum		DPS		Moving Averages	
Iter/Temp	5000		5000		5000	
Seed	Iterations	Tour Length	Iterations	Tour Length	Iterations	Tour Length
0	72263	29692	294908	27733	362516	28066
1	59989	32066	318012	26261	347425	28861
2	164108	30672	251797	30271	405002	27547
3	85885	33459	328356	26087	424530	27022
4	156791	30402	333254	27027	322049	27179

5	107626	29137	369389	28228	358783	28504
6	123124	32066	395273	25368	405016	24688
7	167111	32131	265426	32544	428890	28724
8	90845	34007	323972	26400	360994	27110
9	120127	31171	294908	27733	413061	26354
10	114591	30627	299491	28691	512221	27188
11	91694	33121	372147	27991	322622	27407
12	135216	29751	291836	27877	377484	29810
13	333895	31448	286166	30474	385706	27825
14	198717	31920	355474	27473	431386	27766
15	93567	29759	333527	30363	357250	27786
16	174274	34717	403694	28868	377499	28812
17	140073	34146	276936	27838	439525	29273
18	122845	32521	343775	27013	359508	28590
19	92053	30551	388256	29399	419151	26390
20	123019	28538	317463	27536	360536	26048
21	235479	29108				
22	99321	29207				
23	143697	29147				
24	233339	34465				
25	86927	31668				
26	143771	35689				
27	150980	29459				
28	123498	30928				
29	193033	30459				
30	89920	32356				
AVG.	**137670**	**31432**	**325908**	**28151**	**389103**	**27664**
STD. DEV	**57094**	**1904**	**43091**	**1718**	**44926**	**1202**

This is the data for iteration limit of 5000 iterations per temperature.



Here is the graph of the processing data for Cumulative Sum algorithm

Conclusions and Future Work

The DPS algorithm uses SPC to fix the problem of non-productive search. The Moving Average and DPS gave the same tour length results but iterations of DPS is more than the Moving Average as iteration limit increases. So Moving Average may be a better algorithm for the process. Future work includes finding out why Cumulative Sum is giving different results, and compares it with the DPS and Moving Average.

Acknowledgement

I like to thank the National Science Foundation, University of Maryland, and Institute of System Research for giving me this great opportunity to participate in the Research Experience for Undergraduates program. I also like to thank Dr.Schmidt, Jitendra Chandna, Eddie Chang, and everyone at the DAT Lab for supporting me through out the summer program.

References

1. Szykman, S. (1997), “Improving the Efficiency of Simulated Annealing Optimization Through Detection of Productive Search,” *Proceedings of the 1997 ASME Design Engineering Technical Conferences.* Paper no. DETC97/DAC-3980. Sacramento, CA, September.

2. Kirkpatrick, S., C. D. Gelatt, Jr., and M. P. Vecchi (1983), “Optimization by Simulated Annealing,” Science, 220 (4598):671-679.