Simulated Annealing: Minimum-energy Configuration of Particles in a Circle

C. Carissimo: 10890661[a)]

(Dated: 17 December 2017)

This report will investigate Simulated Annealing as a method for finding the lowest energy configuration of particles on a 2-dimensional surface bounded by a circle. Simulated annealing will be implemented using Python 2.7. Different implementations of simulated annealing are tested to determine the best convergence. An optimal method is then proposed based on the findings of the experimentation.

Keywords: Simulated Annealing, Low-energy Configurations

I. INTRODUCTION

This report will investigate Simulated Annealing as a method for finding the lowest energy configuration of particles on a 2-dimensional surface bounded by a circle. Simulated annealing will be implemented using Python 2.7, and the code can be found in the appendix. Simulated annealing is an optimization algorithm based on the Metropolis-Hastings algorithm, and a physical procedure known as annealing used to reduce imperfections in crystalline materials.

It is necessary to mention, that the implemented simulated annealing algorithm did not converge to the optimal solution consistently, and for a number of particles of 12 and above it rarely found the optimal configuration of particles.

II. SIMULATED ANNEALING

Simulated annealing applies a stochastic optimization process based on the Metropolis-Hastings algorithm to find optimal solutions to a problem. The structure of simulated annealing is as follows:

1. Generate a random solution to your problem;
2. Evaluate the "quality" of the solution;
3. Generate another solution by adjusting/perturbing the first solution;
4. Evaluate the "quality" of this new solution;
5. If the new solution is better switch to it;
6. Otherwise, switch to it with a probability e^{dE*T};
7. Repeat from step 3.

where dE is the difference in quality between the two compared solutions, and T is the temperature of the system. This is the basic structure of the Metropolis-Hastings algorithm. The temperature T of the system is a distinguishing factor of Simulated Annealing. T is varied as the algorithm is iterated. Since we are multiplying $dE * T$ in our exponential function, T will be reduced progressively as our algorithm iterates. This progressive reduction of temperature is called the cooling schedule.

A. Particles on a Circle

Simulated Annealing will be applied specifically to finding the optimum configuration of particles on a circle. In this case optimum means the lowest energy configuration. Essentially this is a minimization problem. For small amounts of particles this problem can be approached with minimization techniques that involve evaluating one by one the setups of the particles, however for large numbers of particles such minimization techniques become computationally very expensive, and eventually in feasible in the computational time available to us given the clock speeds of consumer computers[1]. Simulated Annealing applies a trial and error approach, characteristic also of optimization algorithms like Evolutionary Algorithms. Solutions are randomly generated, tested, and used if better than the currently available ones. An advantage of using simulated annealing to find the lowest energy configuration of particles on a circle, with respect to using an evolutionary algorithm, is the pressure the algorithm exerts on our computer memory. The details of the used algorithm will be discussed in the methods section, but in a nutshell: evolutionary algorithms require populations of solutions to optimize, say for example 100 different configurations of particles. Along with that we would also need to save all the relevant calculations on our memory to be accessed at any given time. With simulated annealing we will be simultaneously considering only two particle configurations, thus requiring less memory.

III. METHOD

The experiments that were conducted for this report were all targeted at improving the speed and convergence of the algorithm such that it was able to, on the one side, arrive at the optimal configuration of particles, and on the other, do so in as short a time as possible.

[a)] Master Computational Science, University of Amsterdam.

The first implementation of the algorithm generated an entire new set of random points on a circle every iteration and then decided whether or not to switch to this entirely new configuration. This implementation can not even be called simulated annealing, because one of the main principles of simulated annealing is that the generated new solutions should be random neighbours of the current solutions, such that there is a constructed correlation between solutions rather than serial correlation. Needless to say this implementation failed to ever find the optimal configuration of particles.

The second implementation of the algorithm, rather than perturbing all points, randomly perturbed a random subset of points when generating a new solution. This implementation of the algorithm also did not produce satisfactory results. The likelihood of the algorithm randomly picking a point that was actually well placed remained very high such that producing an optimal solution and maintaining it was very unlikely.

The final implementation of the algorithm, as it should be, generated a new set of points by perturbing a single point every iteration, where the perturbed point is chosen based on the individual energy contributions of all points to the system. The perturbed point is chosen to be the highest contributor to the energy of the system. This implementation produced the most intersting results, that best approximated the optimal solutions. As we will see in the report, the algorithm was consistently able to place points along the edge of the circle (partly also due to the coding procedure), but has a hard time arranging points in the center.

The following experiments were conducted to better understand the performance of the last algorithm and try and find ways to improve it further. The results of these experiments are reported in the results section.

1. Energy of a random configuration of particles, for $6 < N < 20$ particles.

2. Minimum energy configurations for $6 < N < 20$ particles.

IV. RESULTS

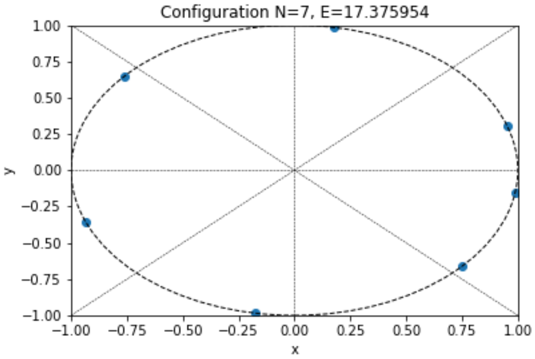
The results section will post the results for several experiments that were conducted to test the efficacy of the Simulated Annealing algorithm developed for the assignment.

Most runs of the simulated annealing algorithm for $6 < N < 12$ produced optimal configurations of particles with one particle in the middle. For $N < 12$ the optimal energy configuration should have all particles spread along the edge of the circle[1]. The results are thus surprising, and clearly wrong. A possible reason for these results is discussed in the next section.

FIG. 1. Average energy of random configurations of particles, by number of particles.

N	Mean	Lower Bound	Upper Bound
7	35.824752	35.531345	36.118158
8	47.351093	47.022425	47.679760
9	61.017882	60.682397	61.353368
10	76.279698	75.798568	76.760827
11	93.482579	92.970627	93.994532
12	111.796341	111.305193	112.287489
13	132.138823	131.588747	132.688899
14	154.159835	153.493565	154.826106
15	177.708179	177.022403	178.393955
16	203.563290	202.813142	204.313437
17	230.800165	229.917279	231.683051
18	259.220926	258.296224	260.145629
19	290.080319	289.311408	290.849230
20	322.774266	321.651231	323.897300

FIG. 2. Minimum energy configuration of 7 particles after 10 runs of the simulated annealing algorithm.

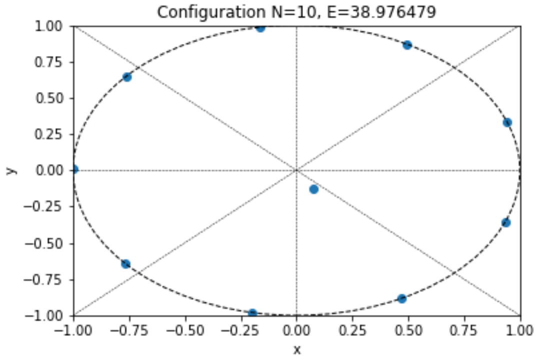


7.png

The experiment with for 7 particles did produce a lowest configuration where all the particles are around the edges, as can be seen in figure 2. However, all other experiments with higher numbers of particles yielded configurations with one particle approaching the middle, for example figure 3.

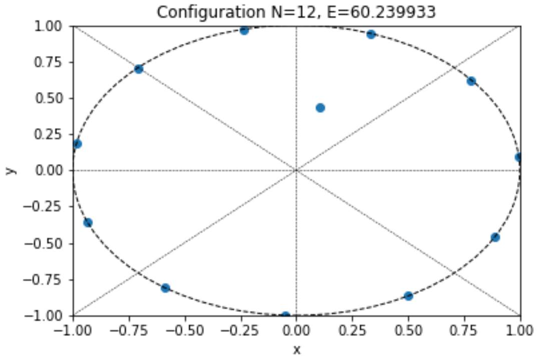
The experiment for 12 particles also had a particle in the middle for the minimum energy configuration, as can

FIG. 3. Minimum energy configuration of 10 particles after 10 runs of the simulated annealing algorithm.



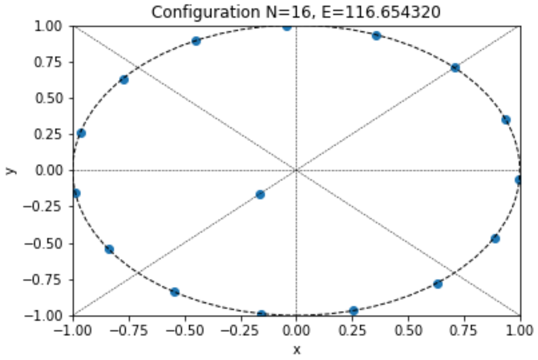
10.png

FIG. 4. Minimum energy configuration of 12 particles after 10 runs of the simulated annealing algorithm.



12.png

FIG. 5. Minimum energy configuration of 16 particles after 10 runs of the simulated annealing algorithm.



16.png

be seen in figure 4.

When the simulated annealing algorithm was run for 16 particles, only one particle could be found in the middle.

V. DISCUSSION

It is unfortunate that the annealing algorithm appeared to be malfunctioning, or simply was not optimized to properly yield a minimum energy configuration. Several drawbacks of the annealing approach that was implemented will be discussed here.

First, the final iteration of the algorithm prioritized particles that were contributing most to the energy of the system. This meant that particles that were closer to other particles with respect to all the others were selected for perturbation, after which a random perturbation generated a new set of x and y values for that point. The annealing algorithm was very quickly able to place particles along the boundaries of the circle. My assumption is that, once all but one particle was placed at the edge of the circle, the remaining particle was left in a tricky spot. This is because , if it was placed on the edge of the circle, it was likely to be closer to its two neighbouring particles on the circle than any of the other particles, thus leading the algorithm to have a higher chance of picking it again for perturbation. This would lead the particle away from the edges of the circle.

Second, is a potential issue arising from the manner in which particles were treated if accidentally placed outside the range of the circle. If a perturbation generated for a particle lead it to be outside the circle, the particle was placed on the edge of the circle such that the angle between the position of the perturbed particle and the center of the circle was preserved. This of course made it very likely for a particle to be placed along the edge of the circle, perhaps too likely. I am not entirely sure how this could have led the simulated annealing algorithm to yield the results it has, but a tentative explanation follows. The explained setup made it more likely for a particle to end up on the edge of the circle, and subsequently if it was already on the edge of the circle the particle was likely to stay there. This likelihood is determined by the size of the perturbation and the position of the particles. Different methods were tried to perturb particles. The final utilized method, for lack of a method that appeared to perform better, was to randomly add two numbers independently drawn from a uniform distribution over the interval $[-1, 1]$ from the x and y coordinates.

Two different scenarios can be discussed in relation to this problem. The first scenario has a number of particles between $7 \leq N \leq 11$. In this scenario, particles should be on the circles circumference for the lowest energy configuration[1]. To make space for other particles along the ring, particles must be perturbed either along the circumference or jumbled back towards the center to initiate a beneficial rearrangement. In this scenario a higher likelihood of staying on the circumference may have inhibited the system from undergoing a beneficial rearrangement, because it was very likely that a particle would be moved along the circumference, on which it was also likely that there were other neighbouring particles, which was likely to lead the resulting energy of the system to be worse than the initial state, thus resulting in a low probability of switching to this state.

The second scenario has a number of particles between $16 \leq N \leq 20$. In this scenario, the minimum energy configuration of the particles should have between two particles somewhere close to the center of the circle[1]. This means that more than one particle must be perturbed away from the circumference and stay there over the course of iteration. This is again made harder given a higher likelihood of ending up along the circumference.

VI. CONCLUSION

In conclusion, it was clear that to write an algorithm that optimizes the configuration of particles on a circle a smart algorithmic design is required. Particles must somehow be perturbed within the circle's confines, whilst still being a random perturbation. The particles that are picked for perturbation must on the one hand be selected on the basis of their contribution to the total energy of the system, and on the other particles that are not necessarily contributing much energy to the system in a given state must also be selected for perturbation such that they can make spaces attractive for other particles to be in.

However, it was also clear how powerful the simulated annealing algorithm can be. If properly programmed to optimize a system and if given enough computational time it is capable of arranging many the particles in such a way as to minimize the energy of the system. If a problem is understood in enough depth such that it can be expressed as a combinatorial optimization problem, simulated annealing is a worthwhile method to consider.

[1] Wille, L. T., Vennik, J. (1985). Electrostatic energy minimization by simulated annealing. Journal of Physics A: Mathematical and General, 18(17), L1113.