Simulated Annealing

Prasangsha Ganguly

February 1, 2021

Introduction

In materials, we want them in minimum energy state. The process of obtaining such a minimum energy material is called annealing, which means controlled cooling.

Recall that we associated *Hill climbing* with "exploitation" or exploitation of the gradient. Where, we generate the neighbors of a node and choose the best out of it and move to it if the objective of the node is better than the current node. On the other hand there is *Random Walk* which is another extreme. It is associated with "exploration". This is actually taking a step in a random direction, without any consideration of the objective values of the neighbors.

Our goal is to have a judicious mix of **exploitation** and **exploration**. Because exploration will take us beyond the local optima and exploitation essentially take us to the optima. To do that, we will explore a non-deterministic or stochastic method. We would want to make a move with a certain probability. This is controlled in such a manner that if the move is a good move, then we are going to make the move with higher probability and if the move is a bad move, the probability is less.

Say, c is the current node and n is the next node with $Eval(c)$ and $Eval(n)$ denote the objective values of the nodes. Now, previously for tabu search, we generated all the neighbors of c, then select the best neighbor and then we choose to move to n or not.

Stochastic Hill Climbing

In this case, we are not going to generate all the neighbors of c. Instead, we are going to generate one random neighbor and decide whether to move to it or not. Either we move to n from c or we remain at c. When we stay at c, we generate another random neighbor and again decide whether to move to it or not. We want decide based on the probability and the probability is governed by whether the move is good or bad.

We will associate the term,

$$\Delta E = Eval(n) - Eval(c)$$

and we consider a maximization problem say. So, this ΔE is something we want our probability of move to be influenced by. So, should we make a move from c to n, it depends on ΔE. For a maximization problem,

- If ΔE is $(+)ve$, we would more likely make the move with higher probability.
- if ΔE is $(-)ve$, which means n is worse than c, then we would still want to allow the move to escape local optima. This time, we will allow the move with lower probability.

So, the function we will be using for computing the probability of move from c to n we denote as $P(c, n)$ depends on this ΔE. The $P(c, n)$ also depend on another factor which actually controls how the ΔE affects $P(c, n)$. Hence, we want two things: the ΔE should influence the probability of the move and secondly, we want to control how ΔE affects the probability $P(c, n)$.

The function has a domain $(-\infty, \infty)$. because we don't have any restriction on how big or small ΔE can be. The range of the function is $[0, 1]$ because the output of the function is a probability. We can use the sigmoid function

$$P(c, n) = \frac{1}{1 + e^{-\Delta E}} \tag{1}$$

Now, as mentioned before, we want another parameter to control how much ΔE influence this probability. Because, we want to place this somewhere in-between the spectrum with one end on Hill Climbing where, if ΔE is $(+)ve$, we accept the move with probability 1, and when the ΔE is $(-)ve$, we reject the move straightway. On the other hand, there is random walk, the other extreme, which doesn't care about whether ΔE is $(+)ve$ or $(-)ve$, it picks up a random neighbor out of all the neighbors and either move to it or not move to it with probability 0.5. Hence we modify the Eqn(1) as,

$$P(c, n) = \frac{1}{1 + e^{-\frac{\Delta E}{T}}} \tag{2}$$

This gives us the simple algorithm

Algorithm 1: Stochastic Hill Climbing

```
begin
1   while True do
2       n ← Random Neighbor(c);
3       Evaluate ΔE = Eval(n) − Eval(c) ;
4       Move with probability P(c, n) = 1/(1+e^{−ΔE/T});
```

Note that, if we have a minimization problem, we have to modify the formula as,

$$P(c, n) = \frac{1}{1 + e^{\Delta E/T}} \tag{3}$$

Simulated Annealing

Simulated Annealing algorithm is an improvement over the simple stochastic hill climbing algorithm. Let us first inspect the effect of ΔE on the probability. Let us assume that $T = 10$. Let us also assume that $Eval(c) = 107$. We have a maximization problem. Hence, objectives more than 107 are better and less than 107 are worse.

Table 1: Effect of ΔE

$Eval(n)$	$-\Delta E$	$e^{-\Delta E/T}$	P
80	27	14.88	0.06
100	7	2.01	0.33
107	0	1.0	0.5
120	-13	0.27	0.78
150	-43	0.01	0.99

The table 1 shows how ΔE affects the probability of moving from c to n. If $Eval(n) > Eval(c)$, then the ΔE is $(+)ve$, $-\Delta E$ is $(-)ve$ and the probability is high. Otherwise, the probability is low. Otherwise, the probability is low. Remember, all these are for a fixed $T = 10$.

Now, let us explore what is the effect of T on the probability. Let us fix $Eval(n) = 120$. Hence, $-\Delta E = -13$. Now, we vary T.

Table 2: Effect of T

T	$e^{-13/T}$	P
1	0.00002	1.0
5	0.074	0.93
10	0.27	0.78
20	0.52	0.66
50	0.77	0.56
1000	0.9999	0.5

From the Table 2, we see at very low temperature T, the probability is 1. This is similar to Hill Climbing in a sense that if it sees a better move, it will make the move. As the temperature increases and at very high temperature, the behavior is like a random walk.

So, if we want more randomness, we will keep high temperature. As we are approaching high temperature, irrespective of the value of ΔE, it will be 0.5 essentially. That is either moves with probability 0.5 or don't move with probability 0.5. If you want to explore more, keep the temperature high. If we want to follow the gradient, keep the temperature low and let it behave like hill climbing in a sense that it only moves to a better spot.

If we construct such table for all the $Eval(n)$ we will see similar behavior. That is, at high temperature, we will have a random walk like behavior with probability 0.5. At low

temperature, it will behave like hill climbing. Note that, at low temperature, it doesn't always move with probability 1. It moves with probability 1 only if $Eval(n)$ is better than $Eval(c)$. If $Eval(n)$ is worse than $Eval(c)$, i.e., for example the case $Eval(n) = 80$ and $Eval(c) = 107$, then the low temperature will result in probability 0.

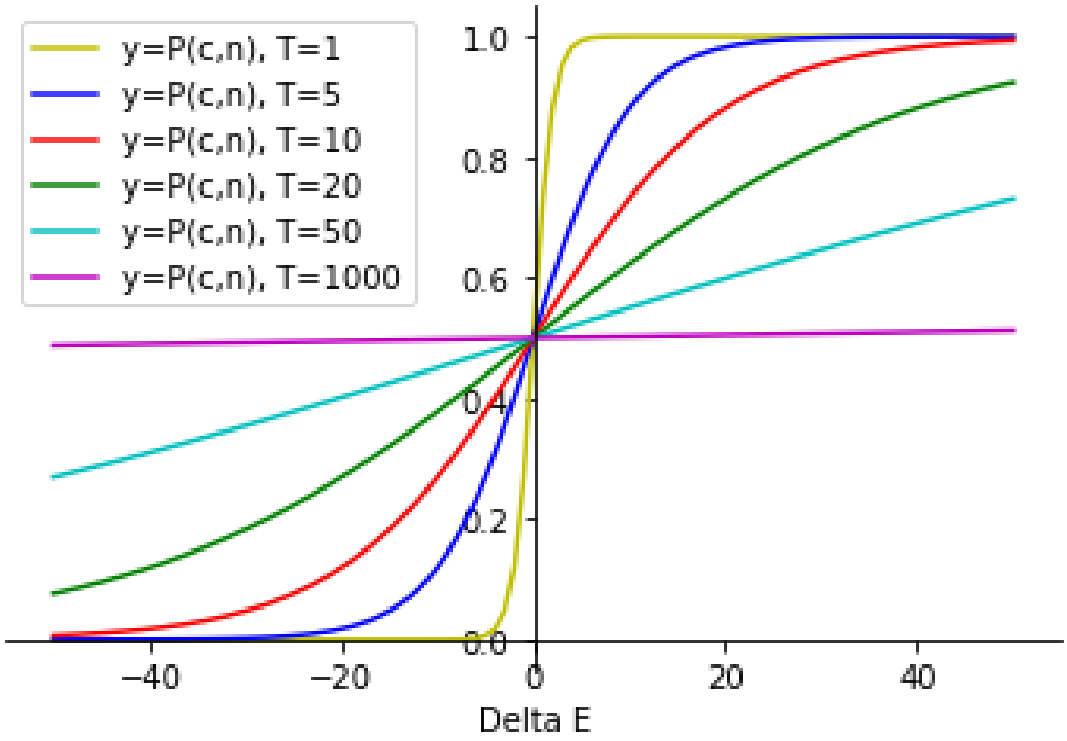

Figure 1: Sigmoid function for different T

In the Fig 1, the probability $P(c, n)$ is plotted on the $y - axis$ and ΔE is plotted on the $x - axis$. For different T, the sigmoid functions are plotted. The curve gets flatter as the temperature increases. If the temperature is low, it looks like a step function. At high temperature, the curve is a flat straight line.

Instead of making a choice of T, we follow what is done in physical world. Which is to say, we will cool down the system gradually, and hope that it will settle into an optimal state essentially.

Initially we set T to very high. Then we lower the temperature gradually. Very often the algorithm has two loops. There is one inner loop where we generate a random neighbor, evaluate ΔE, and move to the neighbor with probability $P(c, n)$. Then in the outer loop,

we decrease T.

Algorithm 2: Simulated Annealing

```
begin
1   T ← Very high value ;
2   while T is not too low do
3       while Some number of iteration do
4           n ← Random Neighbor(c);
5           Evaluate ΔE = Eval(n) − Eval(c) ;
6           Move with probability P(c, n) = 1/(1+e^{−ΔE/T});
7       T decreases by some monotonically decreasing function ;
```

To implement a probabilistic move from c to n, we check if $P(c, n)$ is greater than a threshold then make the move else not. This threshold rather than being a constant , we can randomly choose a number in $[0, 1]$ and if the threshold $< P(c, n)$ or not. If so, then make a move else not.