Simulated Annealing

Applications in Clustering

David Atlas

November 18, 2018

Introduction to Simulated Annealing

Simulated Annealing

Simulated annealing is an combinatorial optimization technique that randomly searches a state space, employing heuristics to find approximate solutions. The rest of this presentation will assume the optimzation problem faced is a minimization problem.

Metallurgy Annealing

Metallurgy Annealing

- Its name is derived from the physical annealing process, in which metal is heated and then cooled.
- While cooling, the metal generally moves towards lower energy states. At temperature τ, the probability of the energy increase of magnitude ΔE is modeled by $f(\Delta E) = \mathrm{e}^{\frac{-\Delta \mathrm{E}}{\mathrm{k}\tau}}$, where k is Boltzmann's constant.
- During this process, the atoms in the metal are heated such that movement is not restricted, and the slow cooling allows the metal to settle in a low energy state.

Intuition

1. Start at a random point
2. Pick a nearby point
3. If it is a better point, switch to it. Otherwise, switch to it with some probability.
4. Continue this process for some set of iterations.

This explores the space, while generally moving towards better solutions, but not getting stuck in local minima.
Additionally, the probability of moving to a worse solution decreases as the number of iterations increases. Thus, we are open to exploring the space more at the beginning, and eventually converge on a good state.

Use Cases

Any of the following reasons may warrant using simulated annealing.

1. Discrete state spaces
2. Large, combinatorial problems.
3. Numerous local minima
4. Approximate solution is suitable

Comparisons

Descent Algorithms

- Descent alogrithms employ heuristics around a function's first and second derivatives to make educated guesses as to the next candidate solution. Simulated annealing does not involve the derivatives of a function, but searches over similar good solutions to find the next candidate solution.
- Additionally, discrete functions are not continuous, and so do not have derivatives, making the implementation of descent algorithms a challenge.

Introduction to Simulated Annealing

Minimization Problem

Let Θ be the state space, and f be an objective function to be optimized.
Find $\hat{\theta} = \mathrm{argmin}_{\theta \in \Theta} \mathrm{f}(\theta)$.

Iterations

- We run the algorithm in stages, indexed by j.
- Each stage has iterations, indexed by t.
- The length of each stage is denoted m_j
- Each stage has a temperature τ_j

Acceptance Probability

- We define the acceptance probability as the probability of a move from state $\theta^{(t)}$ to state θ^*.
- We can use any PDF for this, but we usually use the Boltzmann distribution.

Boltzmann Distribution

$$P(\theta^{(t)}, \theta^*, \tau_j) = \min\left(1, \mathrm{e}^{\frac{\mathrm{f}(\theta^{(\mathrm{t})}) - \mathrm{f}(\theta^*)}{\tau_{\mathrm{j}}}}\right)$$

Neighborhood Function

- The neighborhood function is used to generate candidate solutions. There is no standard neighborhood function to apply in all situations.
- Generally, $\theta^{(t)}$ is selected from some neighborhood of the θ^* with uniform probability.
- For combinatorial problems, we simply permute one element of $\theta^{(t)}$.

Parameters

Annealing Schedule

- τ is decreasing in j.
- $\tau_j = \alpha(\tau_{j-1}), 0 < \alpha < 1$

Number of Iterations

- m is increasing in j
- $m_j = \beta(m_{j-1})$

The Algorithm

Algorithm 1 SimulatedAnnealing(f, Θ, α, β, ϵ, m_0)

Choose $\theta^{(0)} \in \Theta$
$\tau_j = \infty$
$m_j = m_0$
while $\tau_j > \epsilon$ **do**
  for $t = 0$ to m_j **do**
    $\theta^* = \text{Neighbor}(\theta^{(t)})$
    Draw $u \sim U(0, 1)$
    if $u \leq \min(1, \mathrm{e}^{\frac{\mathrm{f}(\theta^{(\mathrm{t})}) - \mathrm{f}(\theta^*)}{\tau_\mathrm{j}}})$ **then**
      $\theta^{(t)} = \theta^*$
    end if
  end for
  $\tau_j = \alpha(\tau_j)$
  $m_j = \beta(m_j)$
end while