Simulated Annealing

Simulated annealing improves the greedy local search in the following way. Assume the task is to minimize a function $f(\mathbf{x})$.

- At each iteration we choose a point randomly from the neighborhood of the current point. Let $\mathbf{x}$ denote the current point and let $\mathbf{y}$ be the randomly chosen neighbor.
- If $f(\mathbf{y}) < f(\mathbf{x})$, that is, a better point is found, then we accept it and $\mathbf{y}$ will be the next position.
- If $\mathbf{y}$ is not better or even worse than $\mathbf{x}$, we still accept it with a certain probability. This probability is computed as follows. Define a quantity ΔE by

  $$\Delta E = f(\mathbf{y}) - f(\mathbf{x}).$$

  This shows by how much $\mathbf{y}$ is worse than $\mathbf{x}$. The notation is based on a physical analogy in which the function value represents the energy of a state in the state space and then ΔE is the energy difference. With this the acceptance probability is defined as

  $$\text{Prob}_{accept} = \mathrm{e}^{-\frac{\Delta E}{kT}}$$

  where k is a constant and T is a parameter called temperature (these come again from the physical analogy, where k is called the Boltzmann constant, but this playes no role in the optimization, here we can simply take $k = 1$). Thus, the new point $\mathbf{y}$ is accepted with probability Prob_{accept} or we stay in $\mathbf{x}$ with probability $1 - \text{Prob}_{accept}$.
- As the iteration proceeds the temperature is gradually decreased. Thus, instead of a constant T, we use a so called *cooling schedule*, which is a

sequence T_n, such that in the n^{th} iteration T_n is used as the temperature parameter. The cooling schedule is chosen such that $T_n \to 0$ holds. A typical choice is the logarithmic cooling schedule:

$$T_n = \frac{a}{b + \log n}$$

where a, b are positive constants.

- As a stopping rule we can again say that if no improvement was obtained over a certain number of iterations than the algorithm stops. Or, we can continue the iterations until the temperature drops below a small $\epsilon > 0$, that is, the system "freezes".

Interpretation of the Algorithm

An essential feature of simulated annealing that it can climb out from a local minimum, since it can accept worse neighbors as the next step. Such an acceptance happens with a probability that is smaller if the neighbor's quality is worse. That is, with larger ΔE the acceptance probability gets smaller. At the same time, with decreasing temperature *all* acceptance probabilities get smaller. This means, initially the system is "hot", makes more random movement, it can relatively easily jump into worse states, too. On the other hand, over time it "cools down", so that worsening moves are accepted less and less frequently. Finally, the system gets "frozen".

This process can be interpreted such that initially, when the achieved quality is not yet too good, we are willing to accept worse positions in order to be able to climb out of local minimums, but later, with improving quality we tend to reject the worsening moves more and more frequently. These tendencies can be balanced and controlled by the choice of the cooling schedule. If it is chosen carefully, then convergence to a *global* optimum can be guaranteed (proven mathematically).

Advantages of Simulated Annealing

- Improves greedy local search by avoiding getting trapped in a local optimum.
- With appropriately chosen cooling schedule the algorithm converges to the global optimum.

Disadvantages of Simulated Annealing

- Convergence to the optimum can be slow.
- There is no general way of estimating the number of iterations needed for a given problem, so we cannot easily guarantee that the result is within a certain error bound of the global optimum.

Exercise

Assume that in a Simulated Annealing algorithm we define the neighborhood of an n-dimensional binary vector $\mathbf{x}$ as follows. Let $w(\mathbf{x})$ denote the number of 1 bits in $\mathbf{x}$. The binary vector $\mathbf{y}$ is a neighbor of $\mathbf{x}$ if and only if $|w(\mathbf{x}) - w(\mathbf{y})| = \alpha \min\{w(\mathbf{x}), w(\mathbf{y})\}$ holds, where α is a constant that we can choose. We want to choose the value of α, such that the state space becomes connected, that is, any vector can be reached from any other one through some sequence of moves, going from neighbor to neighbor. Which of the following is correct?

1. If $\alpha = 1$, then the state space will be connected.
2. If $\alpha \geq 2$, then the state space will be connected.
3. The state space will be connected for any $\alpha > 0$.
4. No matter how α is chosen, this state space will never be connected.
5. None of the above.