5. Simulated Annealing

5.1 Basic Concepts
Simulated Annealing: Part 1

Outline

- Introduction
- Real Annealing and Simulated Annealing
- Metropolis Algorithm
- Template of SA
- A Simple Example
- References
Introduction
What Is Simulated Annealing?

- **Simulated Annealing (SA)**
  - SA is applied to solve optimization problems
  - SA is a **stochastic algorithm**
  - SA is escaping from local optima by allowing worsening moves
  - SA is a **memoryless algorithm**, the algorithm does not use any information gathered during the search
  - SA is applied for both **combinatorial and continuous** optimization problems
  - SA is simple and easy to implement.
  - SA is motivated by the physical annealing process
Simulated Annealing: Part 1

SA vs Greedy Algorithms: Ball on terrain example

Initial position of the ball

Simulated Annealing explores more. Chooses this move with a small probability (Hill Climbing)

Greedy Algorithm gets stuck here!
Locally Optimum Solution.

Upon a large no. of iterations, SA converges to this solution.
History

- Numerical simulation of annealing, Metropolis et al. 1953.

Simulated Annealing: Part 1

History

- SA for combinatorial problems
  - Kirkpatrick et. al, 1986
  - Cerny, 1985


Simulated Annealing: Part 1

History

- Originally, the use of simulated annealing in combinatorial optimization
- In the 1980s, SA had a major impact on the field of heuristic search for its simplicity and efficiency in solving combinatorial optimization problems.
- Then, it has been extended to deal with continuous optimization problems.
- SA was inspired by an analogy between the physical annealing process of solids and the problem of solving large combinatorial optimization problems.
Simulated Annealing: Part 1

Applications

- Basic problems
  - Traveling Salesman Problem
  - Graph partitioning
  - Matching prob.
  - Quadratic Assignment
  - Linear Arrangement
  - Scheduling
  - ....
Applications

- **Engineering problem**
  - VLSI: Placement, routing…
  - Facilities layout
  - Image processing
  - Code design
  - Biology
  - Physics
  - …. 
Real Annealing and Simulated Annealing
Simulated Annealing: Part 1

Real Annealing Technique

- **Annealing Technique** is known as a thermal process for obtaining low-energy state of a solid in a heat bath.

- The process consists of the following two steps:
  - **Increasing temperature**: Increase the temperature of the heat bath to a maximum value at which the solid melts.
  - **Decreasing temperature**: Decrease carefully the temperature of the heat bath until the particles arrange themselves in the ground state of the solid.
Real Annealing Technique

- In the **liquid phase** all **particles** arrange themselves randomly, whereas in the ground state of the solid, the particles are arranged in a highly structured lattice, for which the corresponding energy is minimal.

- The **ground state** of the solid is obtained only if:
  - the maximum value of the temperature is sufficiently high and
  - the cooling is done sufficiently slow.

- Strong solid are grown from careful and slow cooling.
Simulated Annealing: Part 1

Real Annealing Technique

- **Metastable states**
  - If the initial temperature is not sufficiently high or a fast cooling is applied, *metastable states* (imperfections) are obtained.

- **Quenching**
  - The process that leads to metastable states is called *quenching*

- **Thermal equilibrium**
  - If the *lowering of the temperature* is done sufficiently slow, the solid can reach *thermal equilibrium* at each temperature.
Real Annealing and Simulated Annealing

- The analogy between the physical system and the optimization problem.

<table>
<thead>
<tr>
<th>Physical System</th>
<th>Optimization Problem</th>
</tr>
</thead>
<tbody>
<tr>
<td>System state</td>
<td>Solution</td>
</tr>
<tr>
<td>Molecular positions</td>
<td>Decision variables</td>
</tr>
<tr>
<td>Energy</td>
<td>Objective function</td>
</tr>
<tr>
<td>Minimizing energy</td>
<td>Minimizing cost</td>
</tr>
<tr>
<td>Ground state</td>
<td>Global optimal solution</td>
</tr>
<tr>
<td>Metastable state</td>
<td>Local optimum</td>
</tr>
<tr>
<td>Quenching</td>
<td>Local search</td>
</tr>
<tr>
<td>Temperature</td>
<td>Control parameter T</td>
</tr>
<tr>
<td>Real annealing</td>
<td>Simulated annealing</td>
</tr>
</tbody>
</table>
Real Annealing and Simulated Annealing

- The objective function of the problem is analogous to the energy state of the system.
- A solution of the optimization problem corresponds to a system state.
- The decision variables associated with a solution of the problem are analogous to the molecular positions.
- The global optimum corresponds to the ground state of the system.
- Finding a local minimum implies that a metastable state has been reached.
Metropolis Algorithm
In 1958 Metropolis et al. introduced a simple algorithm for simulating the evolution of a solid in a heat bath to \textit{thermal equilibrium}.

Their algorithm is based on Monte Carlo techniques, and generates a sequence of states of the solid in the following way.

Given a current state $i$ of the solid with energy $E_i$, a subsequent state $j$ is generated by applying a \textit{perturbation mechanism} that transforms the current state into a next state by a small distortion, for instance, by a displacement of a single particle.
Simulated Annealing: Part 1

Metropolis Algorithm

- The energy of the next state is $E_j$. (Ej)
- If the energy difference, $E_j - E_i$, is less than or equal to 0, the state $j$ is accepted as the current state.
- If the energy difference is greater than 0, then state $j$ is accepted with a probability given by

$$\exp\left(\frac{E_i - E_j}{k_B T}\right)$$

- where 'T' denotes the temperature of the heat bath and
- $k_B$ a physical constant known as the Boltzmann constant.
Metropolis Algorithm

- The acceptance rule described above is known as the Metropolis criterion (Metropolis rule) and the algorithm that goes with it is known as the Metropolis algorithm.
- In the Metropolis algorithm thermal equilibrium is achieved by generating a large number of transitions at a given temperature value.
Template of SA
Template of SA

- On the basis of a given, the system is subjected to an elementary,
- if this modification causes a decrease in the objective function of the system, it is accepted;
- if it causes an increase $\Delta E$ of the objective function, it is also accepted, but with a probability

$$e^{\frac{-\Delta E}{T}}$$
Template of SA

- By repeatedly observing this Metropolis rule of acceptance, a sequence of configurations is generated.
- With this formalism in place, it is possible to show that, when the chain is of infinite length (in practical consideration, of “sufficient” length. . . ), the system can reach (in practical consideration, can approach) thermodynamic balance (Equilibrium State) at the temperature considered.
Simulated Annealing: Part 1

Template of SA

- At high temperature, $e^{-\frac{\Delta E}{T}}$ is close to 1,
  - therefore the majority of the moves are accepted and the algorithm becomes equivalent to a simple random walk in the configuration space.

- At low temperature, $e^{-\frac{\Delta E}{T}}$ is close to 0,
  - therefore the majority of the moves increasing energy is refused.

- At an intermediate temperature,
  - the algorithm intermittently authorizes the transformations that degrade the objective function
Simulated Annealing: Part 1

Template of SA

- SA can be viewed as a sequence of Metropolis algorithms, evaluated at decreasing values of the temperature.
Simulated Annealing: Part 1

Template of SA

- From an initial solution, SA proceeds in several iterations.
- At each iteration, a random neighbor is generated.
- Moves that improve the cost function are always accepted.
- Otherwise, the neighbor is selected with a given probability that depends on the current temperature and the amount of degradation $\Delta E$ of the objective function.
- $\Delta E$ represents the difference in the objective value (energy) between the current solution and the generated neighboring solution.
Template of SA

- The higher the temperature, the more significant the probability of accepting a worst move.
- At a given temperature, the lower the increase of the objective function, the more significant the probability of accepting the move.
Simulated Annealing: Part 1

Template of SA

- As the algorithm progresses, the probability that such moves are accepted decreases.
The acceptance probability function, in general, the Boltzmann distribution:

\[ P(\Delta E, T) = e^{-\frac{f(s')-f(s)}{T}} \]

- It uses a control parameter, called temperature, to determine the probability of accepting nonimproving solutions.
- At a particular level of temperature, many trials are explored.
- Once an equilibrium state is reached, the temperature is gradually decreased according to a cooling schedule such that few nonimproving solutions are accepted at the end of the search.
Simulated Annealing: Part 1

Start

Initialize Solution and Temperature

Generate a Random Neighbor

Better solution? N

Accept Neighbor Solution

Equilibrium Condition? N

Update Temperature

Cooling Enough? N

Should we accept? N

Y

End
Simulated Annealing: Part 1

Template of SA

Input: Cooling schedule.
$s = s_0$ ; /* Generation of the initial solution */
$T = T_{max}$ ; /* Starting temperature */

Repeat
   Repeat /* At a fixed temperature */
      Generate a random neighbor $s'$ ;
      $\Delta E = f(s') - f(s)$ ;
      If $\Delta E \leq 0$ Then $s = s'$ /* Accept the neighbor solution */
      Else Accept $s'$ with a probability $e^{\frac{-\Delta E}{T}}$ ;
   Until Equilibrium condition
      /* e.g. a given number of iterations executed at each temperature $T$ */
      $T = g(T)$ ; /* Temperature update */
   Until Stopping criteria satisfied /* e.g. $T < T_{min}$ */

Output: Best solution found.
Simulated Annealing: Part 1

Inhomogeneous vs. Homogeneous Algorithm

- **SA has two variants:**
  - **Homogeneous variant**
    - Previous algorithm is the homogeneous variant
    - $T$ is kept constant in the inner loop and is only decreased in the outer loop
  - **Inhomogeneous variant**
    - There is only one loop
    - $T$ is decreased each time in the loop, but only very slightly
Simulated Annealing: Part 1

Inhomogeneous variant

Start

Initialize Solution and Temperature

Generate a Random Neighbor

Better solution?

N

Should we accept?

N

Accept Neighbor Solution

Y

Update Temperature

N

Cooling Enough?

Y

End

N
Simulated Annealing: Part 1

Inhomogeneous variant

**Input:** Cooling schedule.
\[ s = s_0 \quad \text{/* Generation of the initial solution */} \]
\[ T = T_{\text{max}} \quad \text{/* Starting temperature */} \]

**Repeat**
- Generate a random neighbor \( s' \);
- \[ \Delta E = f(s') - f(s) \];
- **If** \( \Delta E \leq 0 \) **Then** \( s = s' \) /* Accept the neighbor solution */
- **Else** Accept \( s' \) with a probability \( e^{\frac{-\Delta E}{T}} \);
- \[ T = g(T) \quad \text{/* Temperature update */} \]

**Until** Stopping criteria satisfied /* e.g. \( T < T_{\text{min}} \) */

**Output:** Best solution found.
The cooling schedule defines for each step of the algorithm $i$ the temperature $T_i$.

It has a great impact on the success of the SA optimization algorithm.

The parameters to consider in defining a cooling schedule are:
- the starting temperature,
- the equilibrium state,
- a cooling function, and
- the final temperature that defines the stopping criteria.
Template of SA

- Main components of SA:
  - Acceptance Function
  - Initial Temperature
  - Equilibrium State
  - Cooling Function
  - Stopping Condition
A Simple Example
Let us maximize the continuous function
\[ f(x) = x^3 - 60x^2 + 900x + 100. \]
A solution \( x \) is represented as a string of 5 bits.
The neighborhood consists in flipping randomly a bit.
The initial solution is 10011 (\( x = 19, f(x) = 2399 \))
Testing two sceneries:
- First scenario: initial temperature \( T_0 \) equal to 500.
- Second scenario: initial temperature \( T_0 \) equal to 100.

Cooling: \( T = 0.9 \cdot T \)
A Simple Example

- In addition to the current solution, the best solution found since the beginning of the search is stored.
- Few parameters control the progress of the search, which are:
  - The temperature
  - The number of iterations performed at each temperature
Simulated Annealing: Part 1

A Simple Example

- First Scenario $T = 500$ and Initial Solution (10011)

<table>
<thead>
<tr>
<th>$T$</th>
<th>Move</th>
<th>Solution</th>
<th>$f$</th>
<th>$\Delta f$</th>
<th>Move?</th>
<th>New Neighbor Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>500</td>
<td>1</td>
<td>00011</td>
<td>2287</td>
<td>112</td>
<td>Yes</td>
<td>00011</td>
</tr>
<tr>
<td>450</td>
<td>3</td>
<td>00111</td>
<td>3803</td>
<td>&lt;0</td>
<td>Yes</td>
<td>00111</td>
</tr>
<tr>
<td>405</td>
<td>5</td>
<td>00110</td>
<td>3556</td>
<td>247</td>
<td>Yes</td>
<td>00110</td>
</tr>
<tr>
<td>364.5</td>
<td>2</td>
<td>01110</td>
<td>3684</td>
<td>&lt;0</td>
<td>Yes</td>
<td>01110</td>
</tr>
<tr>
<td>328</td>
<td>4</td>
<td>01100</td>
<td>3998</td>
<td>&lt;0</td>
<td>Yes</td>
<td>01100</td>
</tr>
<tr>
<td>295.2</td>
<td>3</td>
<td>01000</td>
<td>3972</td>
<td>16</td>
<td>Yes</td>
<td>01000</td>
</tr>
<tr>
<td>265.7</td>
<td>4</td>
<td>01010</td>
<td>4100</td>
<td>&lt;0</td>
<td>Yes</td>
<td>01010</td>
</tr>
<tr>
<td>239.1</td>
<td>5</td>
<td>01011</td>
<td>4071</td>
<td>29</td>
<td>Yes</td>
<td>01011</td>
</tr>
<tr>
<td>215.2</td>
<td>1</td>
<td>11011</td>
<td>343</td>
<td>3728</td>
<td>No</td>
<td>01011</td>
</tr>
</tbody>
</table>
Simulated Annealing: Part 1

A Simple Example

- Second Scenario: $T = 100$ and Initial Solution (10011).

<table>
<thead>
<tr>
<th>$T$</th>
<th>Move</th>
<th>Solution</th>
<th>$f$</th>
<th>$\Delta f$</th>
<th>Move?</th>
<th>New Neighbor Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>1</td>
<td>00011</td>
<td>2287</td>
<td>112</td>
<td>No</td>
<td>10011</td>
</tr>
<tr>
<td>90</td>
<td>3</td>
<td>10111</td>
<td>1227</td>
<td>1172</td>
<td>No</td>
<td>10011</td>
</tr>
<tr>
<td>81</td>
<td>5</td>
<td>10010</td>
<td>2692</td>
<td>&lt; 0</td>
<td>Yes</td>
<td>10010</td>
</tr>
<tr>
<td>72.9</td>
<td>2</td>
<td>11010</td>
<td>516</td>
<td>2176</td>
<td>No</td>
<td>10010</td>
</tr>
<tr>
<td>65.6</td>
<td>4</td>
<td>10000</td>
<td>3236</td>
<td>&lt; 0</td>
<td>Yes</td>
<td>10000</td>
</tr>
<tr>
<td>59</td>
<td>3</td>
<td>10100</td>
<td>2100</td>
<td>1136</td>
<td>Yes</td>
<td>10000</td>
</tr>
</tbody>
</table>

- When Temperature is not High Enough, Algorithm Gets Stuck
References
Simulated Annealing: Part 1

References

The End