

Annealing Simulation Algorithm Part

December 3, 2019

1 Introduction

Simulated annealing (SA) algorithm is another local search strategy we are going to use for this project. the simulated annealing algorithm is not guaranteed to get the optimal solution. However, it is useful to speed up convergence at the beginning of the process with a high temperature, and when it cools down, the probability of optimal solution increases. As it closes to 0, it has the highest probability to get the optimal solutions [1]. In this case, when T is large, it is easy to accept neighboring solutions even it is worse than current solution, but it will be strict when T is getting small. For Simulated annealing algorithm, it is important to know how to update temperature and what should be stop criterion. Sometimes, it is also needed to restart the algorithm.

2 General Procedure

Overview :

The Annealing simulation algorithm follows a basic procedure:

1. Set the initial temperature and a cooling rate.
2. Create a random initial solution.
3. Begin looping till the stop condition is met. Usually either the system temperature is cool enough or the good solution is found.
4. Inside of the loop, we select the neighbor by making small change to the current solution.
5. Then we decide whether we accept the neighbor solution or not.
6. Finally we decrease the temperature and continue the looping.

3 Psudocode

Heuristics

The maximum iteration number is the $\frac{initialTemperature}{coolingDegrees}$. A common acceptance method is always to accept the better solution and accept the worse solution with a probability of $P(accept) \leftarrow \exp(\frac{e-e'}{T})$, where T is the current temperature, e is the cost of current solutio and e' is the cost of the candidate solution.

When the temperature decreased, the T becomes smaller and the probability of acceptance decrease as well. Therefore the acceptance criteria is getting more stringent.

Restarting the procedures sometimes can help to improve outcomes.

Pseudocode

Pseudocode :

Algorithm 1 Simulated Annealing

Input: ProblemSize, $iterations_{max}$, $temp_{max}$
Output: S_{best}

function SIMULATEDANNEALING(ProblemSize)
 $S_{current} \leftarrow S_{createInitialSolution(ProblemSize)}$
 $S_{best} \leftarrow S_{current}$
4: **for** ($i = 1$ to $iterations_{max}$) **do**
 $S_i \leftarrow createNeighborSolution(S_{current})$
 $temp_{current} \leftarrow calculateTemperature(i, temp_{max})$
 if $cost(S_i) \leq cost(S_{current})$ **then**
8: $S_{current} \leftarrow S_i$
 if $cost(S_i) \leq cost(S_{best})$ **then**
 $S_{best} \leftarrow S_i$
 end if
12: **elseif** $Exp(\frac{CostS_{current}-CostS_i}{temp_{current}}) > Rand()$
 $S_{current} \leftarrow S_i$
 end if
 end for
16: **return** S_{best}
end function

4 TSP Implementation and design

The choice of parameters includes initial temperature (T_{max})

5 Complexities

Simulated Annealing goes through temperature change $O(\log n)$ steps. For each temperature change, there is a search space $O(n)$ to attempt till accept the new temperature. The rejection for a temperature change cost $O(1)$. If a change is accepted, the average path reversal cost $O(n)$.

Therefore the total time complexity is $T_n = O((n^2 + n)\log n)$.

Since There are more steps take place at lower temperature with constant rejection, the $O(n\log)$ is not negligible compare to $O(n^2\log)$, we still need to keep it[citation1].

6 Empirical evaluation

Strength and Weakness :