Simulated Annealing (SA) is an effective and general form of optimization.

Basically the general idea of the argument is to start with some greedy solution and randomly generate neighbors of the solution by just changing up the order of some nodes and checking to see if the solution is better and if it doesn’t have an outright better cost than the current solution than we take that new solution based on a calculated probability.

Pseudocode: from <https://en.wikipedia.org/wiki/Simulated_annealing>

SimulatedAnneal:

Let *s* = *s*0

For *k* = 0 through *k*max (exclusive):

*T* ← temperature( *(k+1)*/*k*max )

Pick a random neighbour, *s*new ← neighbour(*s*)

If *P*(*E*(*s*), *E*(*s*new), *T*) ≥ random(0, 1):

*s* ← *s*new

Output: the final state *s*

Create an initial state *s*0 of the problem that we will attempt to improve over time. The greedy algorithm to get an initial solution by choosing the closest neighbor. Produce a neighbor of a state by conservatively altering a given state, each state is defined as a permutation of cities to be visited neighbors of a state are the set of permutations produced by reversing the order of any two successive cities. Finding a neighboring state using the 2 opt process, <https://en.wikipedia.org/wiki/2-opt>, Calculate the total cost of the current path, update if necessary.

The annealing schedule is defined by the call temperature(*r*), which should yield the temperature to use, given the fraction *r* of the time budget that has been expended so far. Calculates the probability of making the transition from current state to a potential new state depends on the total cost of the current and new path and the current temperature value.