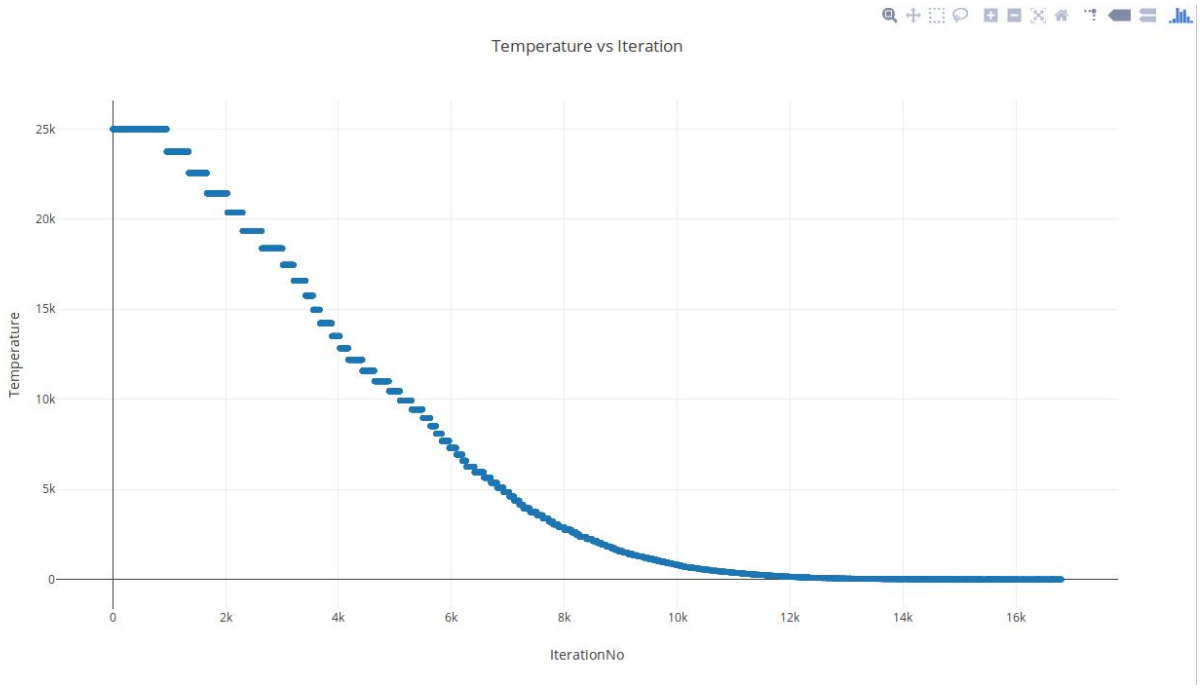
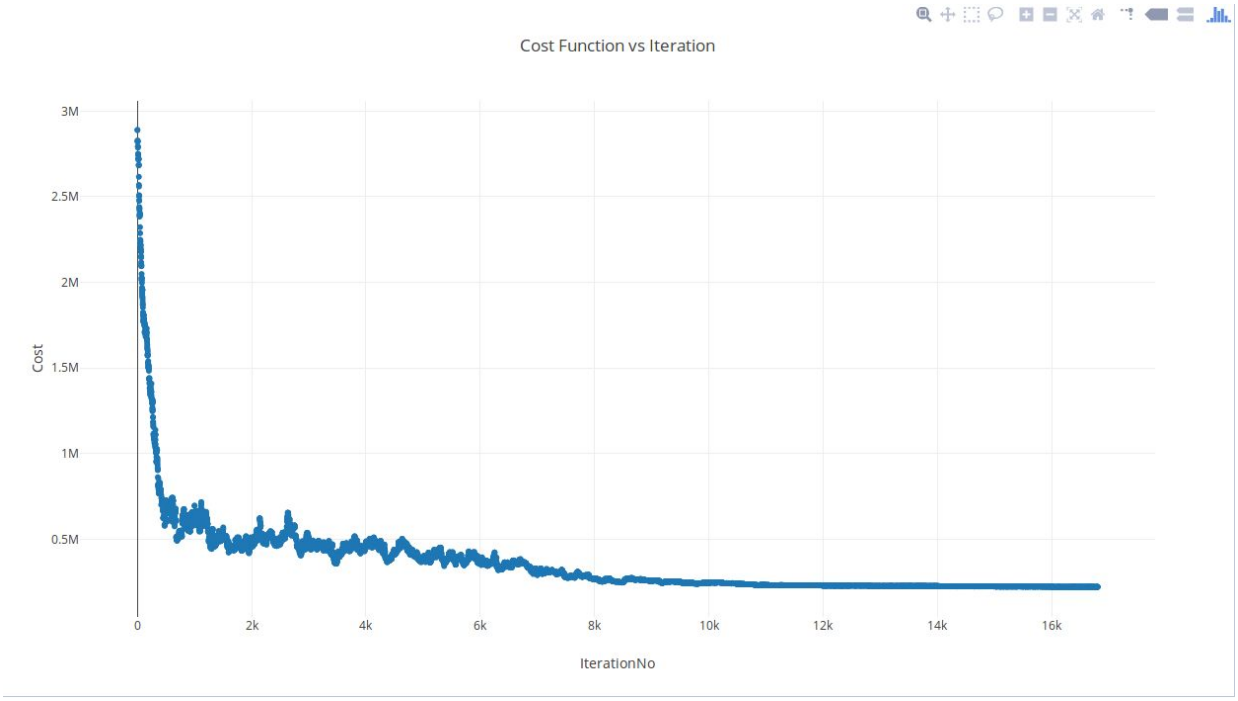
Analysis of Simulated Annealing

The following plots for the input testbench **n100hard.blocks** shows the working of simulated annealing:

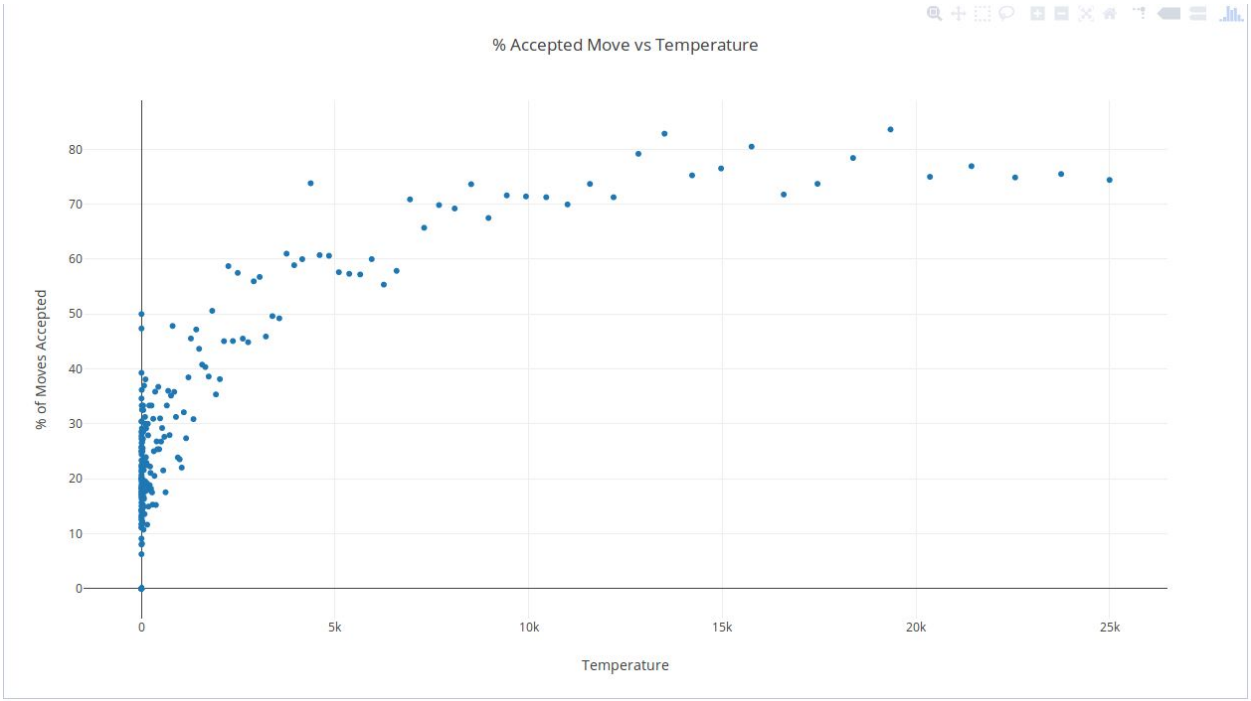
1. **Temperature vs Iteration:**



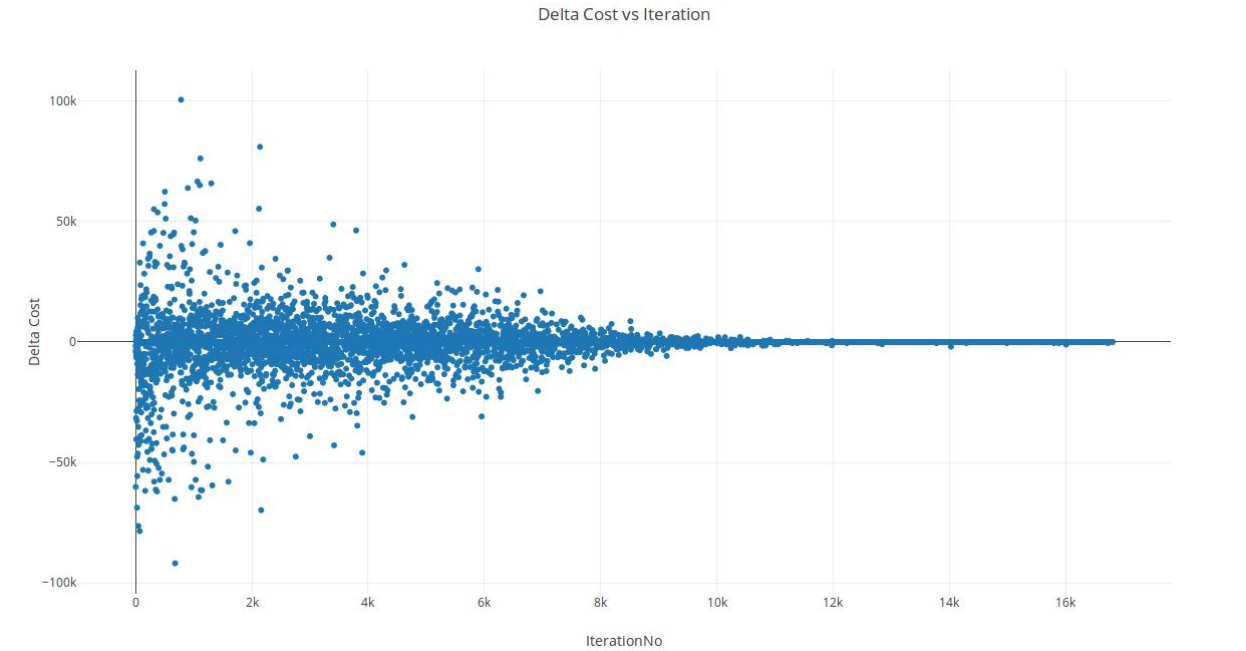
2. Cost Function vs Iteration:



3. Percentage of Accepted Moves vs Temperature (notice how at lower temperatures less number of moves are accepted while at higher temperatures larger number of moves are accepted)



4. Delta Cost vs Iteration (notice how at initial iterations both cost increasing as well as cost decreasing moves are accepted while at later iterations only cost decreasing moves are accepted)



Choice of Parameters:

The program makes use of the following constants with the given use:

1. **CoolDownRate: 0.90**
    - This define the rate at which temperature decreases after a set of moves are performed
    - 90% fall in the temperature after a set of moves is performed is optimum of the given benchmarks. Making this number too small would provide lower runtime but with lower performance as well, as this gives too less opportunity of hill climbing.
2. **StartTemperature: 25000**
    - This defines the temperature with which simulated annealing starts
    - 25000 is optimum of the given test benches, making temperature too low works fine for smaller test cases but fails to provide required performance for larger test cases.
3. **FreezingTemperature: 0.1**
    - This defines the ending point for simulated annealing. It is the temperature at which the simulation stops
    - This should have been as close to zero as possible to enable simulated annealing to find the minima of the trench it's exploring. Keeping it any higher leads to fall in results.
4. **MovesCoolDown : 0.99**
    - This defines the rate at which moves should cool down. The traditional Simulated Annealing doesn't make use of any such configuration. This implementation makes use of this to provide better runtime to the system. It works on the idea that at lower temperature once the minima has been found there is not much need to perform too many moves as any more moves results in zero improvement in cost.
    - This number is chosen to be as close to 1 as possible to implement traditional simulated annealing algorithm with only a very little improvement in moves per set. This play important role when sample size is large in the input testbench.
5. **MovesPerStep: 500**
    - This defines the initial starting point of the moves while performing simulated annealing.
    - This is chosen as 500 as keeping this number too high results in very high runtime with very little improvement in results. While keeping this number is good for smaller inputs it fails for larger inputs.
6. **BoltzmanConstant: 1**
    - This constant is just kept to have 1:1 correspondence with the boltzman function. It can be completely ignored for practical purposes.