VLSI Design Automation Assignment 1

SIMULATED ANNEALING

BY VINAY KUMAR BURUGU

#M08813508

SIMULATED ANNEALING ALGORITHM:

The connectivity matrix is first formed from the benchmark file. Only non-zero entries are stored (connections given in the benchmark file). The matrix is stored in the following way:

The no. of rows in the matrix are the no. of cells (first entry in the benchmark file). Each row's entry will consists of the cell numbers that have the connection with that particular row number. If a number has multiple connections with that row, it is stored twice. So, to calculate the net weight between two numbers, we need to find the no. of occurrences of one of the number in the row of the corresponding other number. To make it efficient, while calculating net weight between two numbers, the row with less no. of entries is considered to traverse.

Simulated Annealing algorithm requires two initial random partitions. This is formed by storing 0s and 1s in a vector (array) of size n (cell count) and randomly shuffled using the standard function available in C++ library. All 0 indexes are considered as partition A and all 1 indexes are considered as partition B.

The next step in the algorithm is to find the cut set of the given two random partitions. We traverse the entire vector and go to the corresponding rows of the connectivity matrix. All entries in the connectivity matrix row are traversed. If the row number index and the corresponding entry number index in the initial vector consists of different values (0 and 1) then we add 1 to the cut set value. After traversing through the entire vector, we get double the value of the actual cut set. Because we have counted each connection twice as a connection (1, 2) is stored in 1^{st} and 2^{nd} rows of the connectivity matrix. Complexity: O(n*m) (m<n) where n is no. of cells and m is the size of maximum row size in the connectivity matrix.

Two random numbers 'a' and' b are choses such that each index stores different values (0 and 1). The index Internal and External connections are calculated by going to the corresponding rows in the connectivity matrix. Using the formulae obtained from KL algorithm, we calculate the gain of the cut set if the two no s chosen are swapped. The formulae is

Gain = Ea-Ia+Eb-Ib-2*Cab; (cab is the connection between a and b).

Complexity : O(m) m is the maximum size of a row in connectivity matrix.

If the gain is positive then the move is automatically accepted. If the gain is negative then based on the random value generated and the boltz value generated, if boltz is greater than random value, then move is accepted, else move is not accepted. In the implementation boltz is chosen such that no negative gain is accepted.

The above process starts with an initial temperature of 400000. It decreases at the rate of 99% of its previous value. And it runs for 400000 iterations of each step in reduction of temperature.

The process is continued till the temperature reaches a low point of 10.

Data Structures used:

1. A vector of vectors was used for the implementation of the connectivity matrix. Each entry in the outer vector corresponds to the cell number and each entry in the inner vector corresponds to the cell number connected to the row number.
2. A vector of size n (no. of elements) is used with equal no. of 0s and 1s to distinguish between the two partitions.

Deviation from the original algorithm:

KL algorithm formulae is used to calculate the gain or cost instead of calculating the cut sets for the newly formed partitions by swapping two random elements chosen which is time consuming .

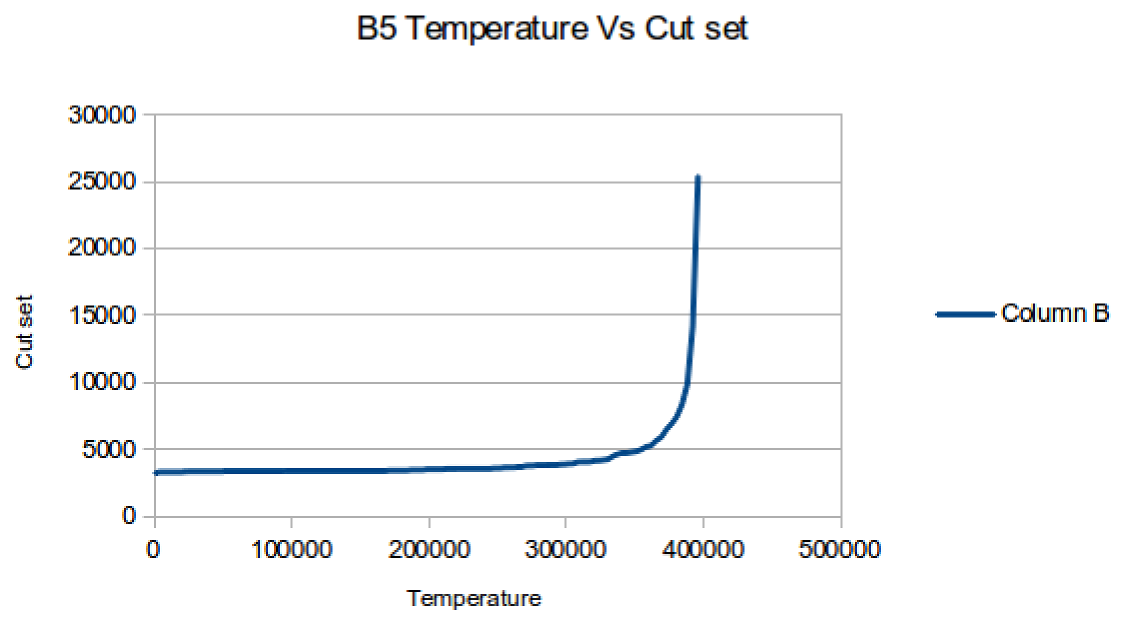
Tuning the algorithm:

For smaller benchmark files the algorithm produces optimal cut set with less initial temperature and less no. of iterations per temperature. But a lot of simulations were performed to obtain the optimal cut set for the final and larger benchmarks. Different temperature and no. of steps per iteration were used to decide on the final temperature and no. of steps per iteration values.

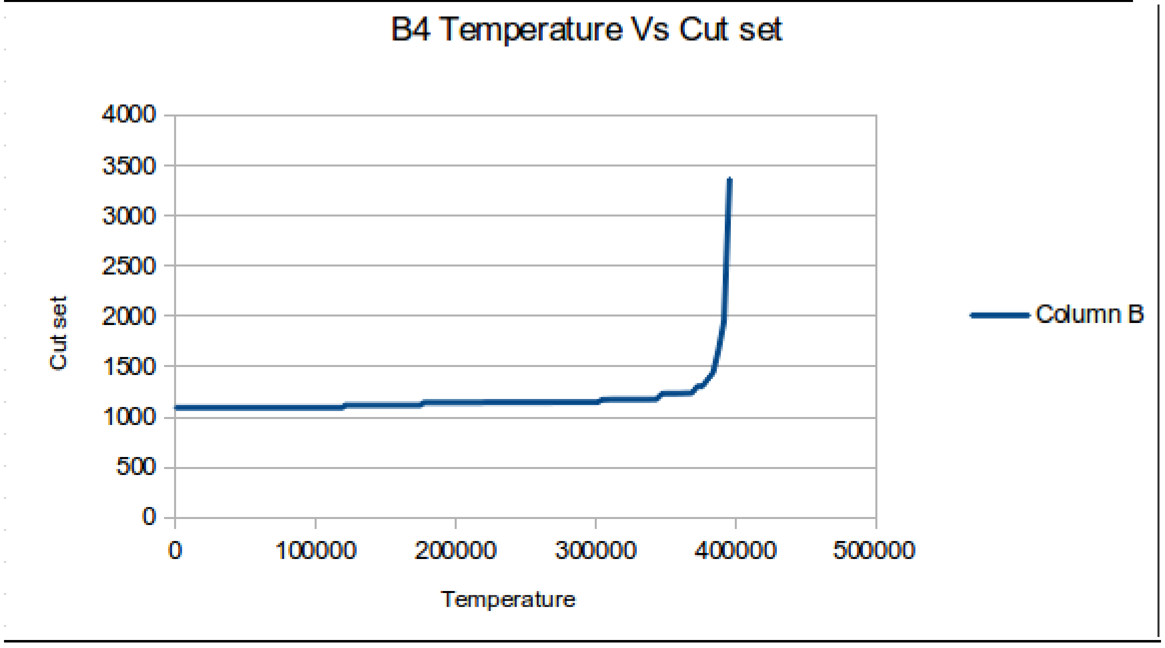
Bench Mark files run information:

Bench Mark	Execution Time	Memory	Optimum cut set	Seed
B1	203.62s	5664kB	11	1456260612
B2	389.77s	5952kB	234	1456261060
B3	338.87s	7872kB	110	1456261741
B4	753.74s	14848kB	1095	1456262406
B5	997.80s	36752kB	3274	1456176983
B6	546s	37232kB	3985	1456191324

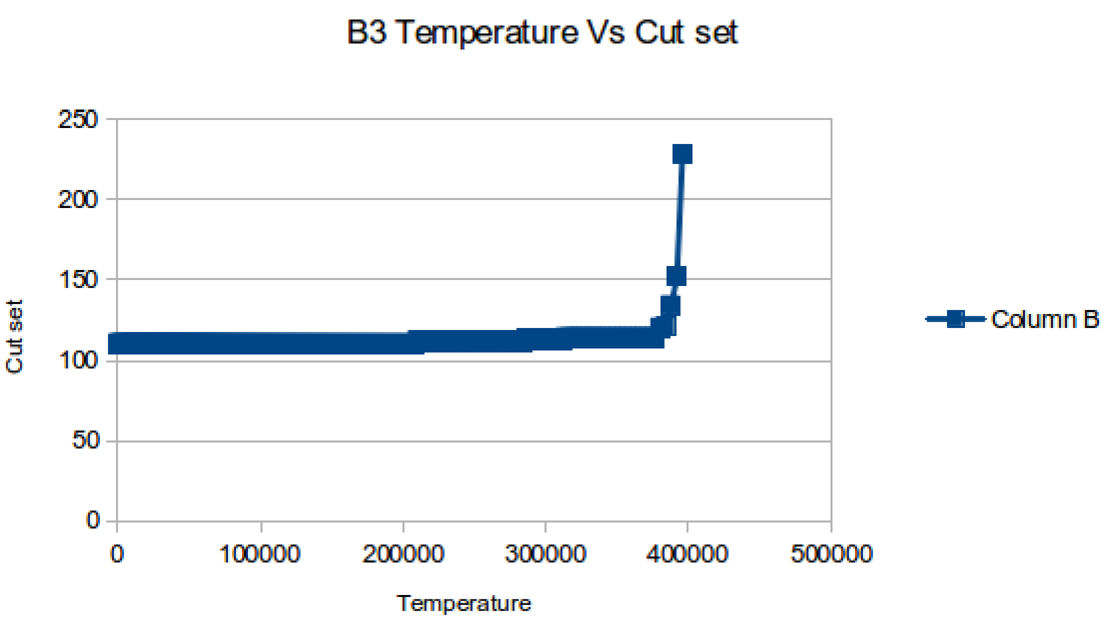
Plots:

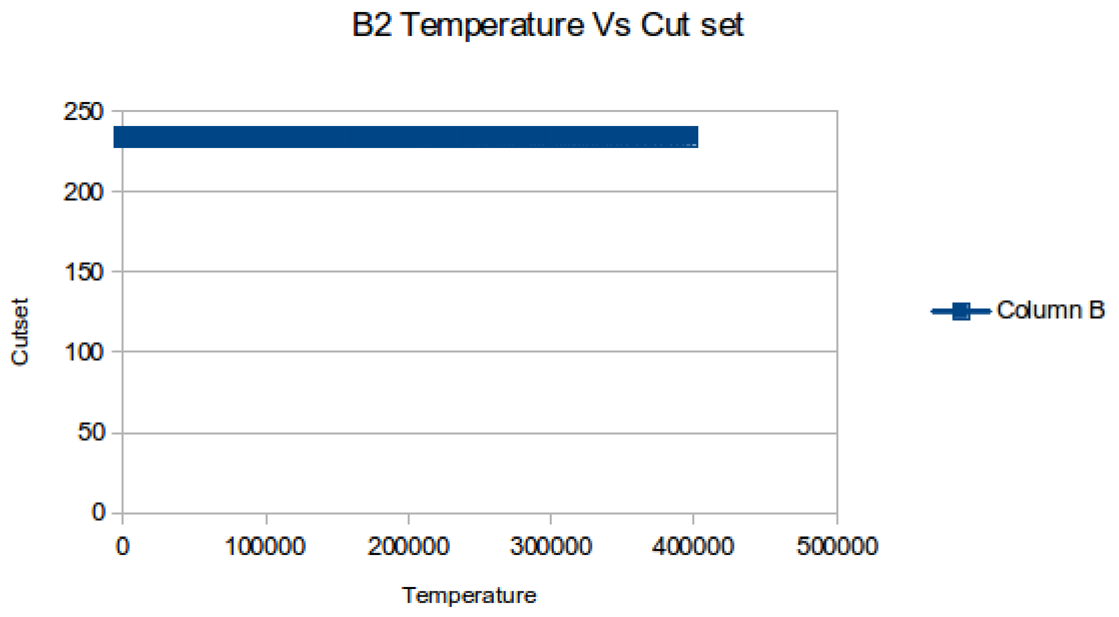


B5 Temperature Vs Cut set
Cut set
Temperature
0
5000
10000
15000
20000
25000
30000
100000
200000
300000
400000
500000
Column B

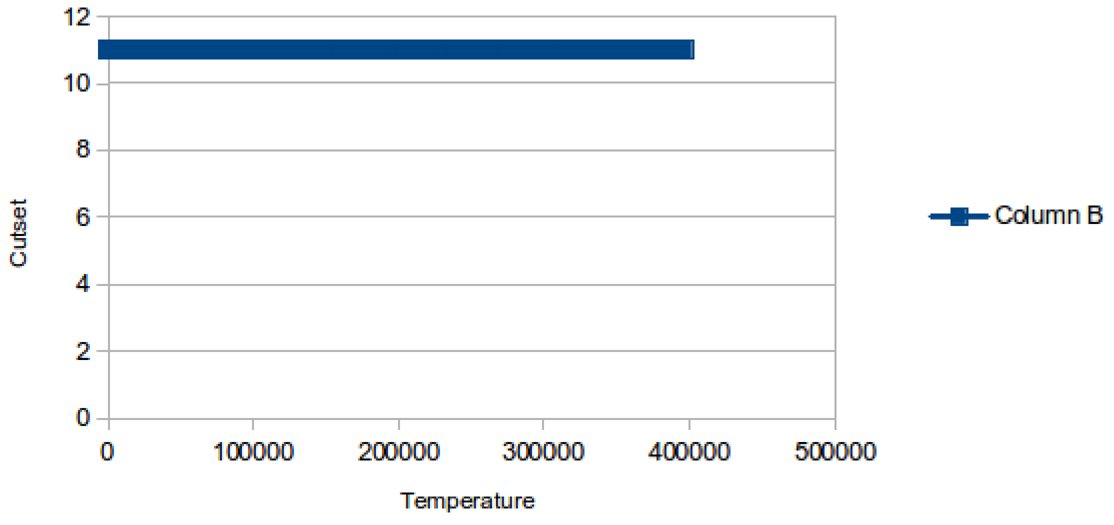


B4 Temperature Vs Cut set
Cut set
Temperature
0
500
1000
1500
2000
2500
3000
3500
4000
100000
200000
300000
400000
500000
Column B

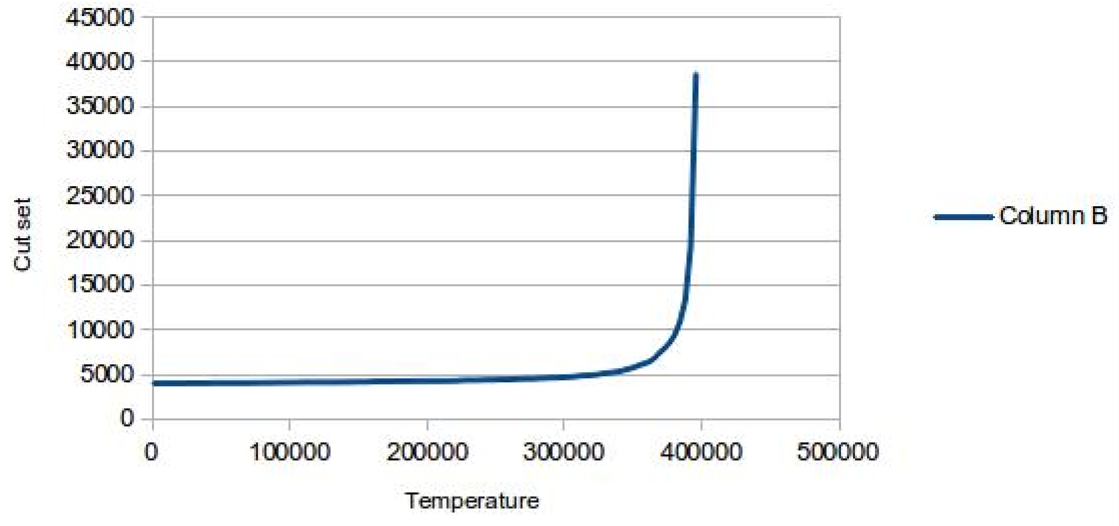

B3 Temperature Vs Cut set
Cut set
Temperature
0
50
100
150
200
250
0
100000
200000
300000
400000
500000
Column B


B2 Temperature Vs Cut set
Cutset
Temperature
0
50
100
150
200
250
0
100000
200000
300000
400000
500000
Column B

B1 Temperature Vs Cutset



B6 Temperature Vs Cut set


Cut set
Temperature
Column B