Digital Design II Project
A Simple Simulated-Annealing Cell Placement Tool

Done by: Amany Shehata, Ahmed ElZoghby, Ziyad Amin

Algorithm:
The algorithm used in this project is based on simulated annealing, it's used for approximating the global optimum of a given function. The objective is to minimize the total wire length in a grid of cells that need to be blacked in a grid to be able to find the wire length.

HPWL:
Half perimeter wire length is used to estimate the wire length of any net. Which is the half perimeter of the smallest bounding box containing the cells connected by the net. The metric is used to calculate by determining the min and max of the x and y coordinates of the cells in the net and then computing the sum of half the perimeter of the bounding box.

Simulated Annealing
The simulated annealing algorithm consists of the process of annealing. It involves heating and then cools the material to remove defects thereby min the systems energy. The energy is the total wire length when computed and the goal is to find the placement of the cells that minimizes this wire length.

1. Initial placement & temperature setting

-Start with random placements of cells in the grid
-Computation the initial wire length
-Set the initial temp proportional to the initial wire length 500 x initial cost

2. Cooling Schedule

-The temp is reduced by multiplying it with the cooling rate of 0.95
-Final temp is set to 5×10−6×(Number of Nets/Initial Cost)

3. Moves per Temperature

-For each temp value the number of moves are performed (20 x the number of cells)

4.
-Swapping Cells: Randomly select two cells and swap their positions.
Wire Length Calculation: We calculated the new wire length if the new wire length is lower, then accept the new placement.
-Calculate the new wire length
-If the new wire length is lower accept the new placement
-If the new wire length is higher then we accept the new placement that decreases exponentially with the increase in wire length and decreases in temp.

-If higher, accept with a probability that decreases exponentially with the increase in wire length and decrease in temperature.

5. Termination
-This process will go on until the temperature drops to the final temperature.
Using the simulated annealing has advantages as it can escape the local minima unlike using greedy algorithms that may get stuck. It requires tuning of parameters such as the initial temperature temperature, cooling rate, and the number of moves per temperature to achieve the best possible results for the wire length.

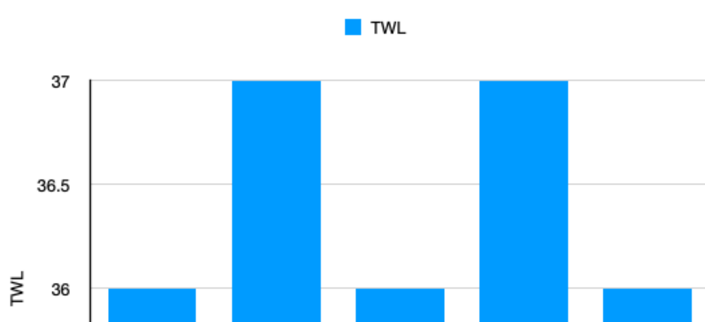
Implementation
The code is constructed into functions to set up the nets and grid positions, parse the input, calculate the wire length, print the grid in binary and normal, and perform the simulated annealing optimization.
Net Struct: Represents a net with the count of connected cells and a vector of connected net IDs.
Position Struct: Represents the coordinates (x, y) of a cell on the grid.
Parsing the Input: The parse function reads the input netlist file and initializes the grid dimensions. It places cells randomly on the grid using a random shuffle, ensuring an initial diverse placement.

Main Functions
wireLength: Calculates the total wire length using HPWL.
printGrid: Prints the current grid configuration.
simulatedAnnealing: Performs the optimization by simulating the annealing process, iteratively improving the cell placement to minimize wire length.
The detailed algorithm ensures a systematic approach to optimize placement and accuracy to achieve minimal wire lengths.

Graphs
Cooling rate vs. TWL graphs
In d0 and d1, the cooling rates (0.75, 0.8, 0.85, 0.9, and 0.95) result in TWLs that are close to each other due to the initially small TWL values. This minimal variation occurs because the TWLs in these graphs are already low, leaving little room for further reduction.To determine the median TWL, I ran the simulated annealing process five times for each cooling rate, sorted the results, and selected the middle value.

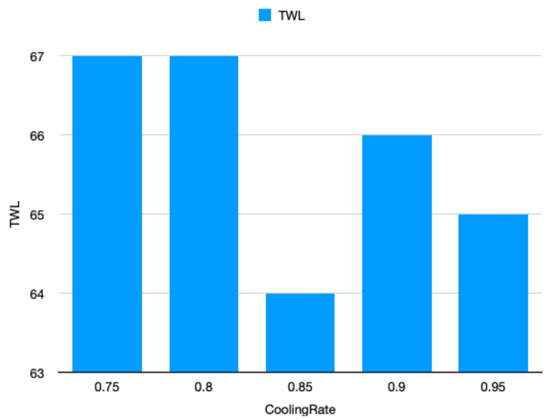
d0.txt_output

CoolingRate	TWL
0.75	36
0.8	37
0.85	36
0.9	37
0.95	36



d1.txt_output

CoolingRate	TWL
0.75	67
0.8	67
0.85	64
0.9	66
0.95	65

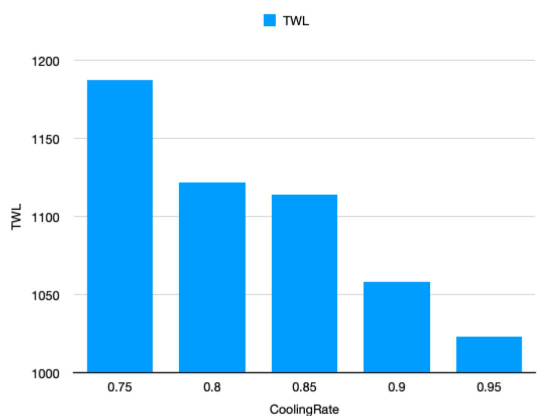


For d2, d3, and t1, the TWL decreases significantly across the same cooling rates. This is due to their initially much higher TWL compared to d0 and d1. The higher starting TWL in these datasets allows for more noticeable improvements during the optimization process.By running the process five times for each cooling rate and sorting the results to find the median, the significant decreases in TWL were clearly observed.

---

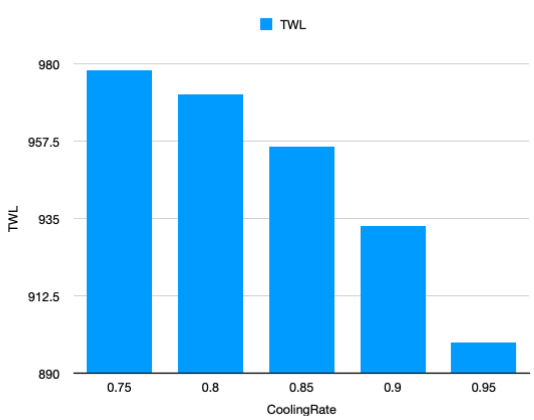
d2.txt_output

CoolingRate	TWL
0.75	1187
0.8	1122
0.85	1114
0.9	1058
0.95	1023



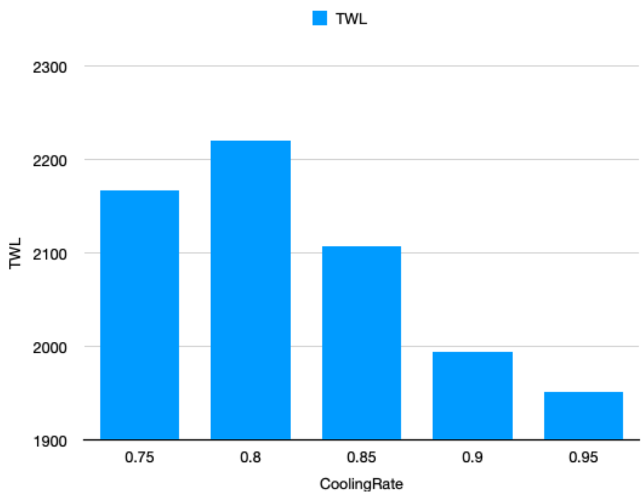
d3.txt_output

CoolingRate	TWL
0.75	978
0.8	971
0.85	956
0.9	933
0.95	899



t1.txt_output

CoolingRate	TWL
0.75	2167
0.8	2220
0.85	2107
0.9	1995
0.95	1952

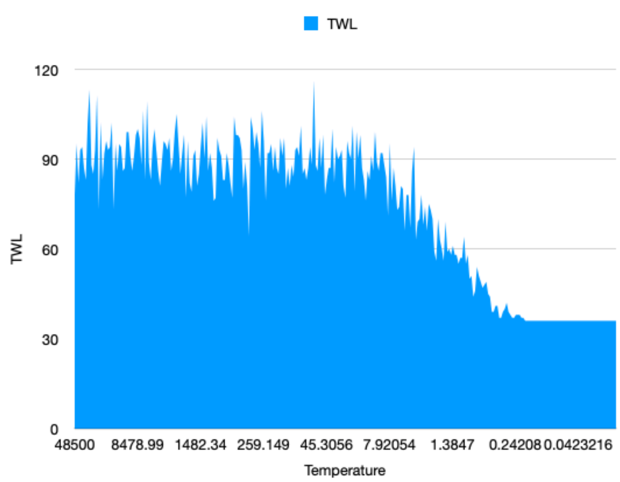


Temperature vs TWL

The temperature vs. TWL graphs show that as temperature decreases the TWL initially drops significantly. At higher temperatures, the simulated annealing algorithm makes larger changes, helping escape local minima. As temperature lowers, changes become smaller. Eventually, the graph levels off, indicating the algorithm has converged to a near-optimal solution where further temperature reductions no longer improve TWL.

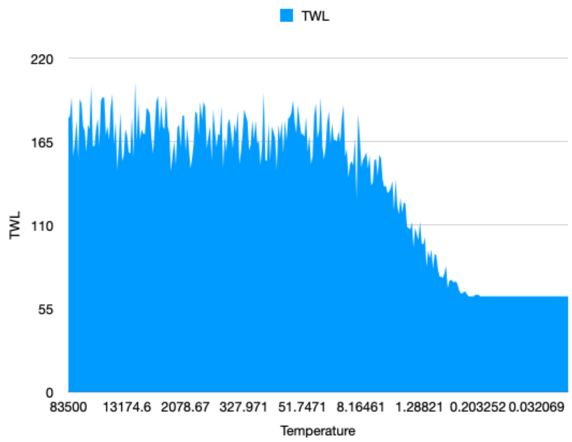
d0.txt_temp_outp

Temperature	TWL
48500	78
46075	95
43771.2	82
41582.7	93
39503.6	94
37528.4	87
35652	83
33869.4	102
32175.9	113
30567.1	88
29038.7	85
27586.8	93
26207.5	111
24897.1	73
23652.2	102
22469.6	83



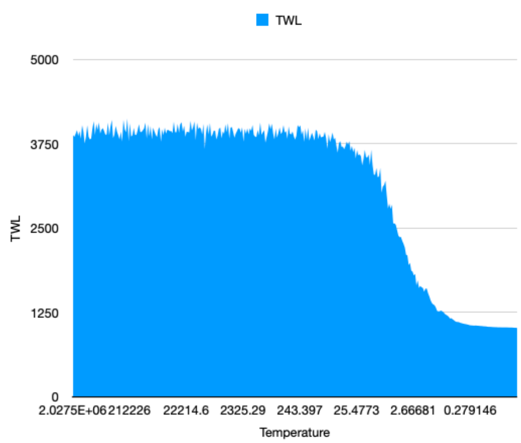
d1.txt_temp_outp

Temperature	TWL
83500	180
79325	182
75358.8	194
71590.8	155
68011.3	165
64610.7	179
61380.2	153
58311.2	193
55395.6	190
52625.8	175
49994.5	172
47494.8	158
45120.1	176
42864.1	173
40720.9	201
38684.8	162
36750.6	162



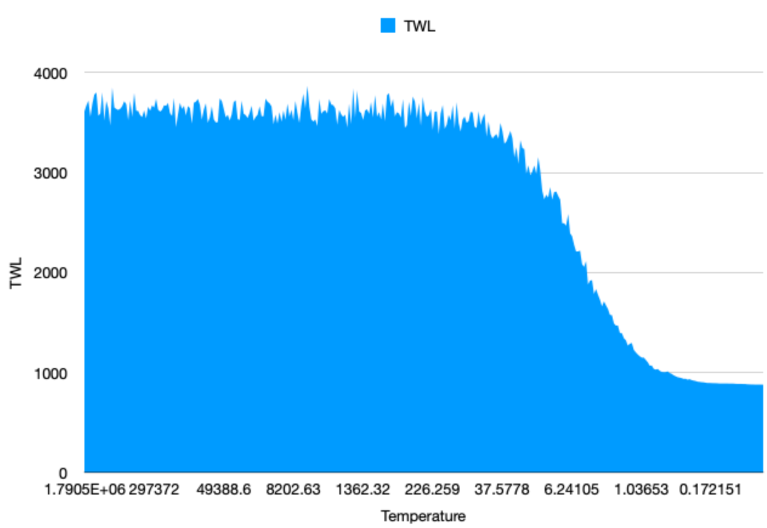
d2.txt_temp_outp

Temperature	TWL
2.0275E+06	3876
1.92612E+06	3848
1.82982E+06	3895
1.73833E+06	3949
1.65141E+06	3860
1.56884E+06	3934
1.4904E+06	3820
1.41588E+06	4020
1.34508E+06	3923
1.27783E+06	3750
1.21394E+06	3855
1.15324E+06	4020
1.09558E+06	3851
1.0408E+06	3811
988761	3824
939323	3972



d3.txt_temp_outp

Temperature	TWL
1.7905E+06	3610
1.70098E+06	3671
1.61593E+06	3718
1.53513E+06	3556
1.45837E+06	3688
1.38545E+06	3779
1.31618E+06	3800
1.25037E+06	3567
1.18785E+06	3589
1.12846E+06	3800
1.07204E+06	3518
1.01844E+06	3715
967515	3639
919139	3468
873182	3847
829523	3650
788047	3634



t1.txt_temp_outpu

Temperature	TWL
3.4085E+06	7002
3.23808E+06	6818
3.07617E+06	7030
2.92236E+06	6940
2.77624E+06	7071
2.63743E+06	7025
2.50556E+06	6968
2.38028E+06	6785
2.26127E+06	6802
2.14821E+06	6944
2.04079E+06	6898
1.93876E+06	6866
1.84182E+06	7012
1.74973E+06	6993
1.66224E+06	7090