EE 5301 - Fall 2018
Project 3: Simulated Annealing Non-Slicing Floorplanning
Phase 2

Initial Chip Width, Height, Area and Wirelength values are shared below the table since the table was getting too crowded for all the values.

Ckt Name	Argument	Chip Width	Chip Height	HPWL	Area	Runtime
n10	-a	465	501	17660.5	232965	0.0878 s
n10	-w	533	571	13633	304343	0.0924 s
n10	-c	489	521	13639	254769	0.0931 s
n30	-a	491	456	46916	223896	0.3722 s
n30	-w	562	514	34169	288868	0.3512 s
n30	-c	516	524	34253	270384	0.3529 s
n50	-a	433	495	104468	214335	0.9388 s
n50	-w	509	564	78243	287076	0.9207 s
n50	-c	468	523	79867.5	244764	0.9172 s
n100	-a	455	455	177832	207025	3.6382 s
n100	-w	481	577	117416	277537	3.5967 s
n100	-c	466	464	133992	216224	3.5802 s
n200	-a	437	480	408525	209760	15.6773 s
n200	-w	545	522	295749	284490	15.5228 s
n200	-c	520	485	301358	252200	15.3969 s
n300	-a	534	560	645910	299040	34.8604 s
n300	-w	725	699	480182	506775	35.0257 s
n300	-c	700	665	498860	465500	35.0407 s

Initial Chip width and heights are mentioned in the files and here as well.

Bench n10.fp:

Initial Wirelength of the Placement is: 32303.5
Initial Width of the Placement is: 1538
Initial Height of the Placement is: 208
Initial Area of the Placement is: 319904

Bench n30.fp:

Initial Wirelength of the Placement is: 133332
Initial Width of the Placement is: 2418
Initial Height of the Placement is: 134
Initial Area of the Placement is: 324012

Bench n50.fp:

Initial Wirelength of the Placement is: 348994
Initial Width of the Placement is: 3080
Initial Height of the Placement is: 94
Initial Area of the Placement is: 289520

Bench n100.fp:

Initial Wirelength of the Placement is: 916466
Initial Width of the Placement is: 4167
Initial Height of the Placement is: 67
Initial Area of the Placement is: 279189

Bench n200.fp:

Initial Wirelength of the Placement is: 2.64666e+06
Initial Width of the Placement is: 5917
Initial Height of the Placement is: 48
Initial Area of the Placement is: 284016

Bench n300.fp:

Initial Wirelength of the Placement is: 5.45597e+06
Initial Width of the Placement is: 9160
Initial Height of the Placement is: 48
Initial Area of the Placement is: 439680

The details of the implementation are discussed in the following section.

Strategy used in the Floorplanning algorithm:

Non Slicing Floorplanning using Simulated Annealing

Parsing the circuit file, using the necessary data structures. In this case, the entire contents of the Bench file were divided into 2 categories. The Floor Planning Blocks were parsed individually and stored in a vector of class block which had various parameters like width, height, x-coordinate and y-coordinate for each corresponding block. The Nets were parsed and each net was put in a vector list of type blocks using pointers, each element in the list representing an individual net containing the connected blocks.

Once the file was parsed, the vectors containing blocks was used to construct the Horizontal Constraint Graph(HCG) to obtain the x-coordinate of each block in the Floorplan and the Vertical Constraint Graph(VG) to obtain their y-coordinates. The construction of HCG and VCG was done using Queues. Another option explored was using Graphs from STL, but this did not seem to improve the runtime by much so was discarded. Queues turned out to be easier to implement and debug. A detailed description of the implementation can be found in the Methodology section below.

Simulated Annealing Engine was then implemented to swap elements from the Sequence Pair and calculate new wirelength and area based on the Boltzman constant defined. Three different moves were implemented, the first one was to swap elements in the Positive Sequence of the Sequence Pair. The second move was to swap two elements in the Negative sequence of the Sequence Pair and then the final move was to swap elements in both the Sequences. Which move was to be executed was selected randomly by generating a random number less than 3.

A Makefile is written to execute the same, and the run commands are shared in the Files section.

Implementation/Methodology

A vector of blocks (aka Class block) was used to store the Block details - the Block Height, the Block Width, x and y coordinates, position in the positive sequence, position in the negative sequence etc. after parsing block details from the bench file. The parsing was done using spaces between block, its width and height details. The first line was parsed as the total number of blocks in the Floorplan and subsequent lines were parsed as first element being the block name followed by its width and then its height. The functions "first_not_of" and "first_of" were used to detect the intermediate spaces.

Next the Nets were parsed in a similar fashion. First the string "Nets" was detected and then the subsequent lines were parsed by detecting the spaces between the connected block names. Each line in the bench represented a Net, so each net contained a variable number of Blocks associated with it. These blocks for each Net were pushed to a vector list of type block (aka class block) pointers. This vector was used to calculate the Wirelength in each case.

The Area was calculated by multiplying the final width and height. The width and height were being updated at each iteration. The width being the maximum of the x-coordinate of the Blocks added to the last Block's width and similarly the final height was maximum y-coordinate of the Blocks added to the height of the last Block in the vertical direction.

Random initial placement was performed by placing the Blocks in a vector of Floor Planning Blocks. Here the chosen initial placement was a linear arrangement of blocks. Initially both the sequence pairs were taken to be the same and after every iteration, the sequence pair was updated based on the most recent placement of Blocks.

For the purpose of Placement and calculation of x and y coordinates of the Blocks, HCG and VCG graphs were implemented using Queues. The Queues were of type block (see class block), were each block was added to the queue once all the preceding blocks were added to the queue i.e, from the Sequence pair, it was considered that the first element had no inputs and all other blocks were its outputs, consequently the second block had one input which was the first element and the blocks following it were the outputs and so on. So, once all the blocks preceding it were added to the queue, the corresponding block was added to the queue. Once all the blocks were processed in the HCG queue, each would have its updated x-coordinate, and when the VCG queue was processed, the y-coordinate was updated.

The Simulated Annealing Engine was implemented for an initial temperature of 40000, decrementing in steps of 0.95 times of the current temperature, till the freezing temperature of 1 was reached. Various temperature ranges like 1000 down to 0.01 and 500 down to 0.001 were experimented with but the one used here was chosen giving an importance to the execution time. Each temperature step also had 100 iterations, which was mostly decided going by trial and error method.

The final Area and Wirelength were reported in 3 different files, along with x and y coordinates of each block - one where area were optimised (a extension), one where wirelength was optimized (w extension) and one where both were optimised (c extension) based on the alpha value which was a parameter used to give different weights to the Wirelength and Area differences. The choosing of Alpha was done based on the plots shown in excel sheet such that the wirelength produced by the best case scenario (w file) was better than c extension file and best case area produced by the best case scenario (a file) was better than c extension file.

Files in the Archive:

- main.cpp - The code that parses the date from the bench file. Uses a class block with members to store various properties of blocks used in the bench file. Calculates and prints the Chip Area, Chip Width and Chip Height along with the Wire Length and Area into files.
- Results.csv - This excel sheet contains details like Temperature, Number of Accepted Moves, Number of Rejected Moves, Wirelength and Area at each temperature.
- benchfile_Sheeparamatti_Arshath.out2<x> - x can be either "a","w" or "c" based on whether Area, Wirelength or Both are being optimized. The file contains details like Initial and Final Area, Wirelength, Width and Height of Chip and the final coordinates of the Blocks.
- Makefile - A Makefile is included to make things easier. Inside the folder containing all the mentioned files, just run "make" command. This should compile the program. To run the file run the following commands -

- Running the files:

```
 make
./parser <filename> <-x>
 (where x can be either “a”, “w” or “c”)
```

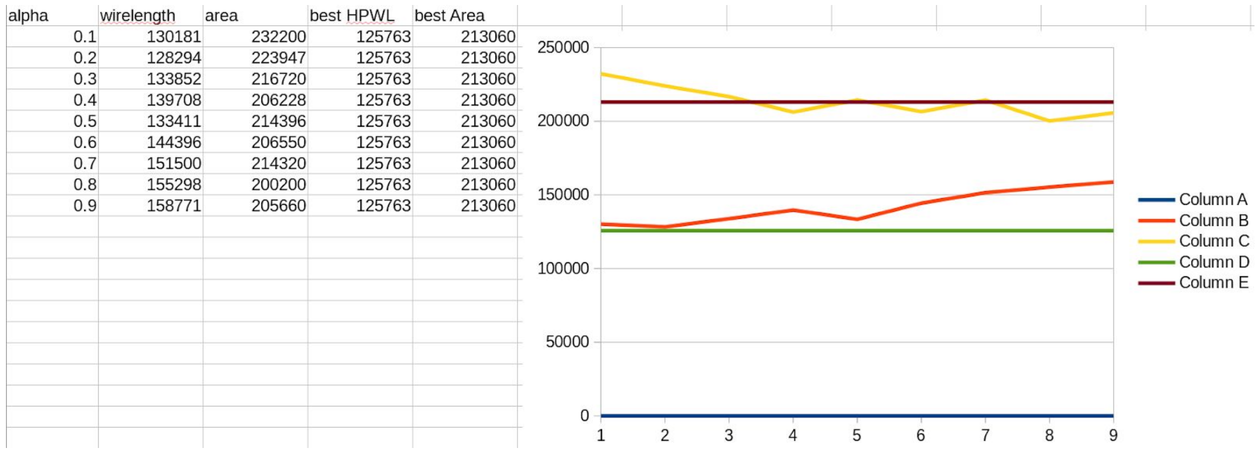
For example: make

```
./parser n100.fp -w
```

Files Generated:
- Results.csv and <benchfile>_Sheeparamatti_Arshath.out2<x>

The Results.csv is regenerated for every file. It is replaced once the simulation is run for other bench files. But the <benchfile>_Sheeparamatti_Arshath.out2<x> is unique to different bench files for different modes.

Selection of Alpha:



alpha	wirelength	area	best HPWL	best Area
0.1	130181	232200	125763	213060
0.2	128294	223947	125763	213060
0.3	133852	216720	125763	213060
0.4	139708	206228	125763	213060
0.5	133411	214396	125763	213060
0.6	144396	206550	125763	213060
0.7	151500	214320	125763	213060
0.8	155298	200200	125763	213060
0.9	158771	205660	125763	213060

Figure 1. Change in Wirelength and Area observed for different values of Alpha

In the above figure, the change in wirelength against different values of Alpha was observed to find the optimal alpha value. Column B is the variation of wirelength value for different alpha values, Column C is the variation of Area for different values of Alpha. Column D is the best case Wirelength and Column E is the worst case wirelength. As can be seen, the simulated annealing run for both area and wirelength with an alpha value between 0 and 1, exclusive of both must produce a wirelength and area that is the closest to both the best wirelength and area produced when alpha is 0 and 1 respectively. As observed from the graph above, the x-axis gives the variation in alpha (read as 0.1 to 0.9), and the orange curve shows the wirelength variation with alpha. So the closest it is to the best wirelength produced (from alpha 0) is from 0.1 to 0.3 alpha values (1 to 3 in graph). Also at some points it is observed that Area goes below the best possible area, which could be due to the fact that when wirelength is highly optimised, it could also result in a better area. But in the alpha value range of 0.1 to 0.3, the value is above the best case Area produced when alpha is 1. So the union of both gives optimal alpha values between 0.1 and 0.3. But if observed carefully, it is observed that at 0.2, the value of Wirelength is closest to the best possible case, and Area is optimally close to the best area (although not particularly the best case). So I have chosen a value of 0.2 for my alpha, which gives a really good wirelength and pretty good Area optimization. The table below might provide more insights where I have shared the Area and Wirelength values for alphas in the range of 0.1 and 0.3.

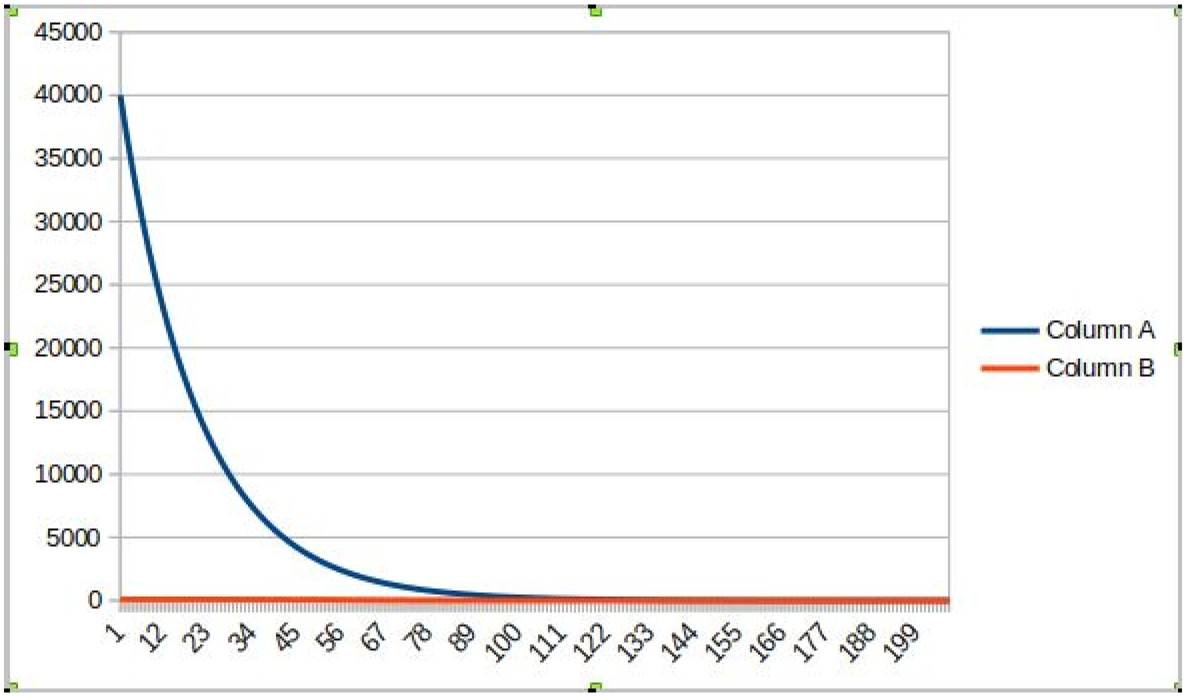
Also experimented with alpha value of 0.5, which meant both the wirelength and area get equal weightage. But this did not result in the best possible solutions for the same. The reason could be because the range of values that both area and wirelength take initially could be very different. For example, in n10, the values of area are approximately 10 times the values taken by the wirelength.

Table 2. Different Wirelength and Area comparison for various alpha values for n300

Alpha value	HPWL(Wirelength)	Area
0.1	480320	414511
0.2	478152	411492
0.25	491367	385795
0.3	498555	398780
0.5	520419	372075

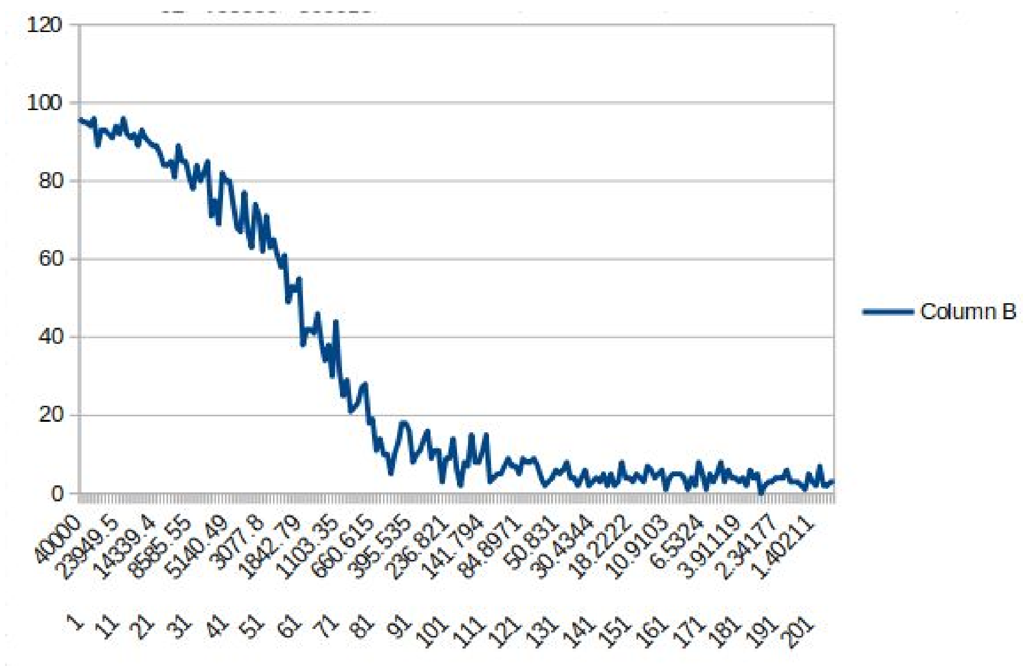
Other things observed:

Variation of number of Accepted moves with respect to temperature:



The graph for Accepted Values against the Temperature steps is shown in the figure above. It can be observed that for lower temperatures the number of accepted moves gradually decreases until it reaches zero. Here Column A represents number of Accepted moves, the Y-axis is the temperature and the X-axis is the Temperature step. The values of accepted moves are not visible but they vary from an initial value of 96 to a least value of 4 as can be seen in the csv file of n100.The number of accepted moves can also be gathered from the next graph.

The number of Accepted moves at different temperatures.



In this graph it is easier to discern the number of accepted moves at each temperature. The Y-axis shows the number of Accepted moves and the X-axis shows the temperature and the step. Initially it can be seen that the number of accepted moves is close to a 100, and in the end, as the temperature decreases to a lower value, the number of accepted moves goes down close to 0 for a total number of iterations run at each step of 100. Column B here represents the Number of Accepted moves.

Selection of k for Simulated Annealing:

The selection of k for Simulated Annealing was mostly trial and error based. For this particular project, the value of 0.95 worked best for k. Although I experimented around with values like 0.05 and 10^(-12), the value 0.95 gave optimal results for both Area and Wirelength. **Temperature** was varied from 40000 to 1,