

Terminal Propagation

- NANDAN BEDEKAR
- PRIYADARSHINI SAVAN ROSHAN

Objective

Brief Description

Overview of implementation

Implementation issues

Results

Brief Description

Algorithm for the placement of cells which targets wirelength reduction

Uses recursive bi-partitioning for performing placement

We have used FM algorithm – move based

The results have been compared for different window sizes.

Overview of implementation

Locked cells were added for each partition which represents external connection to the cells.

These locked cells have a fixed partition during FM algorithm and thus, aren't moved through the passes.

The terminals being propagated are limited by tuning a parameter for each circuit.

Window size is varied from 0 to core width/height. When window size is maximum, no terminals are propagated (mincut).

When window size is zero, all the terminals are propagated.

Overview of implementation

The IO cells are initially placed at the edges as specified. These IOs also act as terminals and pull the cells towards themselves.

When the average number of cells in the partition reaches 50 to 100 we stop partitioning.

Wirelength degradation was observed due to too much terminals being propagated. So the number of terminals propagated was limited by a threshold.

Once the partitions are created, cells are assigned to rows corresponding to that partition randomly. Due to lack of time row balancing techniques were not implemented.

Implementation Issues

KL could not be used because of its very high execution time. The runs of KL would simply not finish for circuits benchmarks with high number of cells.

When the partitions in the circuit become small the number of terminals that are propagated for them become very large leading to very large cut sizes. This led to the terminals dominating cut sizes and bad partitioning.

Significant challenges were faced in tuning the following parameters: window size, max number of terminals per partition, scaling factor for the max number of terminals per partition, number of steps, area constraint for the fm algorithm.

Results

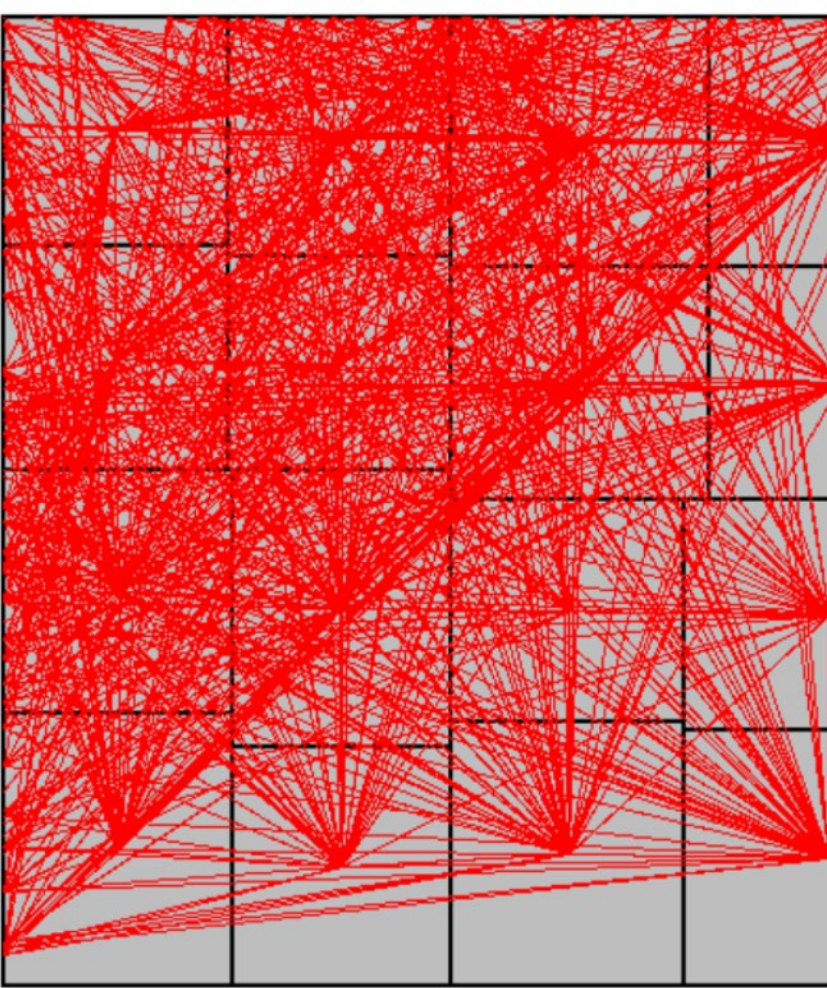
Runtime analysis for the complete algorithm:

Name of trace file	Number of cells	Number of nets	Runtime (in s)
structP	1920	1952	25.90207601
p2	3029	3014	75.49070811
biomedP	5742	6514	2411.632531
industry2	13419	12637	2715.305205
industry3	21923	15406	2722.885562

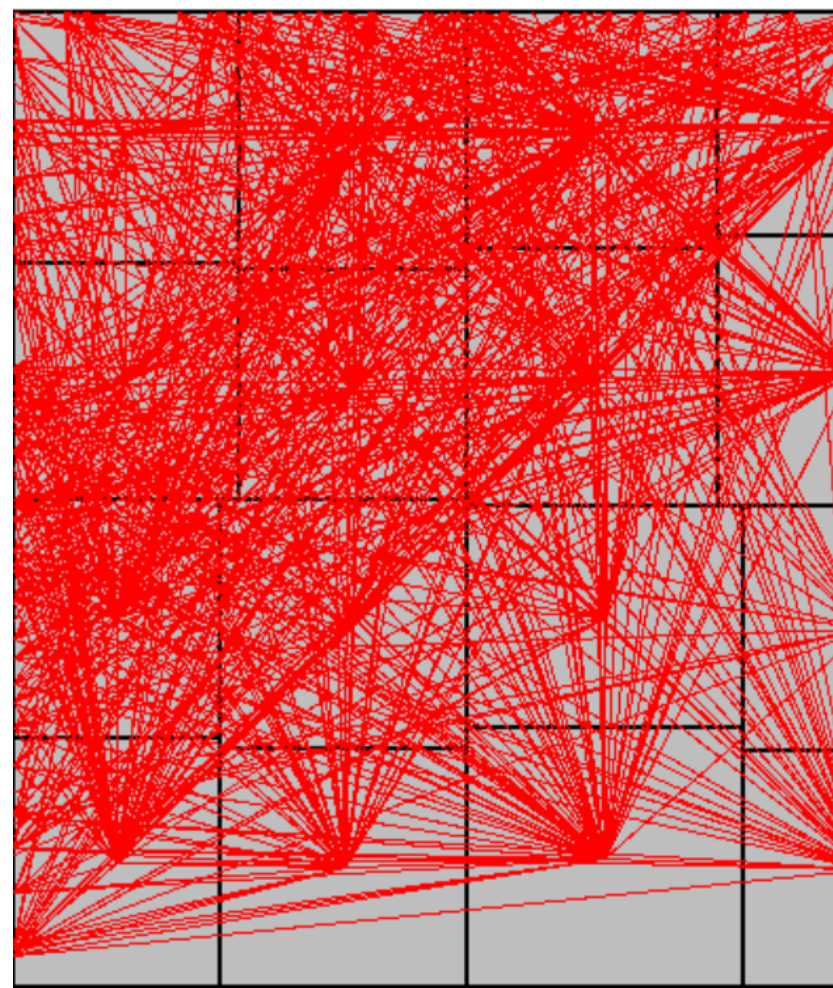
Results

Wirelength analysis for different percentage of window sizes (HPBB):

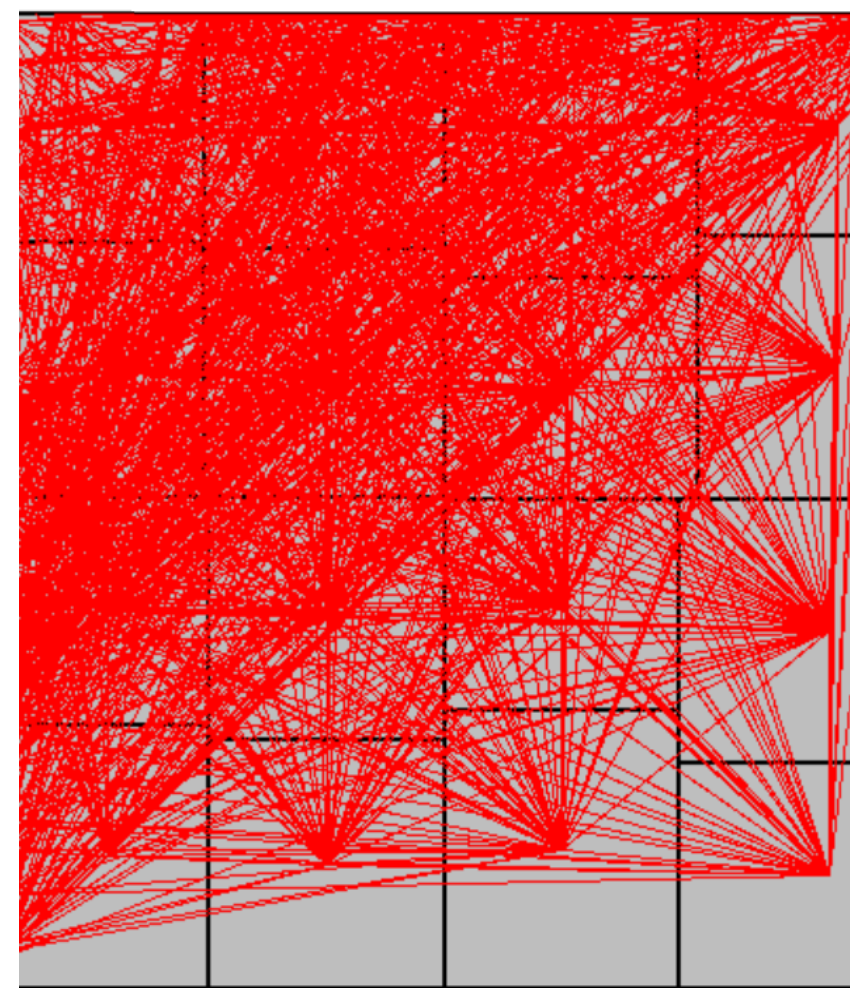
Name of trace file	Wirelength for percent window size		
	0 percent	50 percent	Mincut
structP	75267	73519	77441
p2	232872	229005	241483
biomedP	430287	419735	433179
industry2	1803850	1789820	1826460
industry3	3725490	3647360	3756640



WL = 75267 (WINDOW=0)

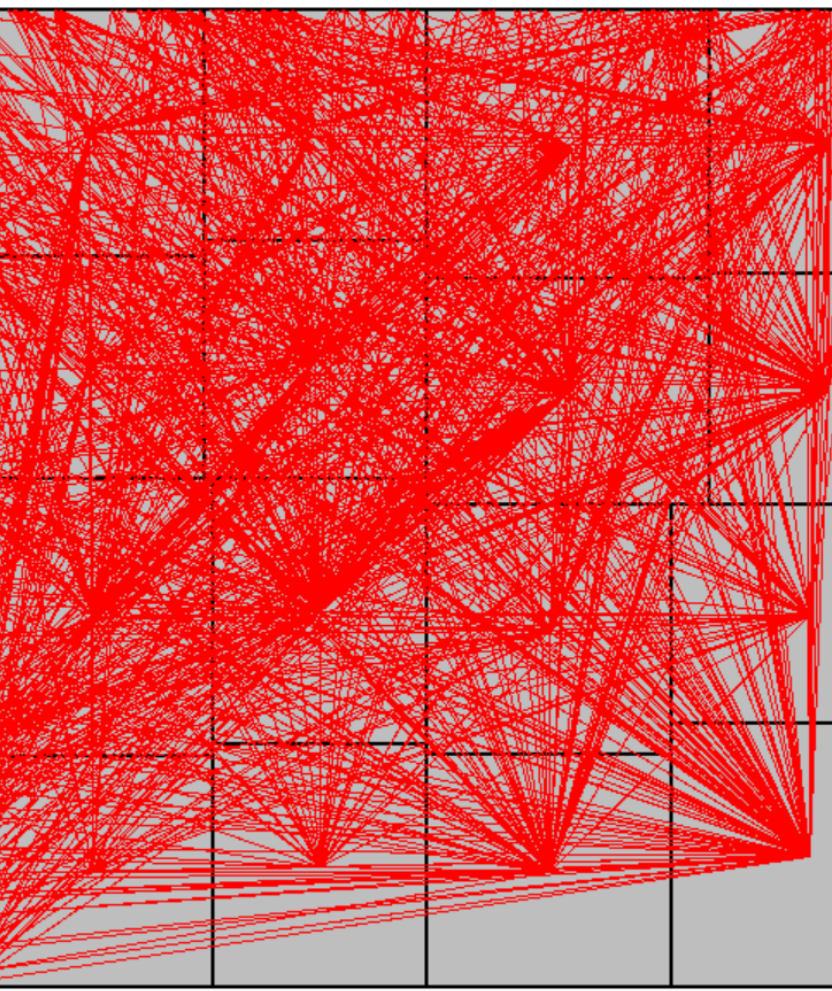


WL = 73519 (WINDOW = 0.5)

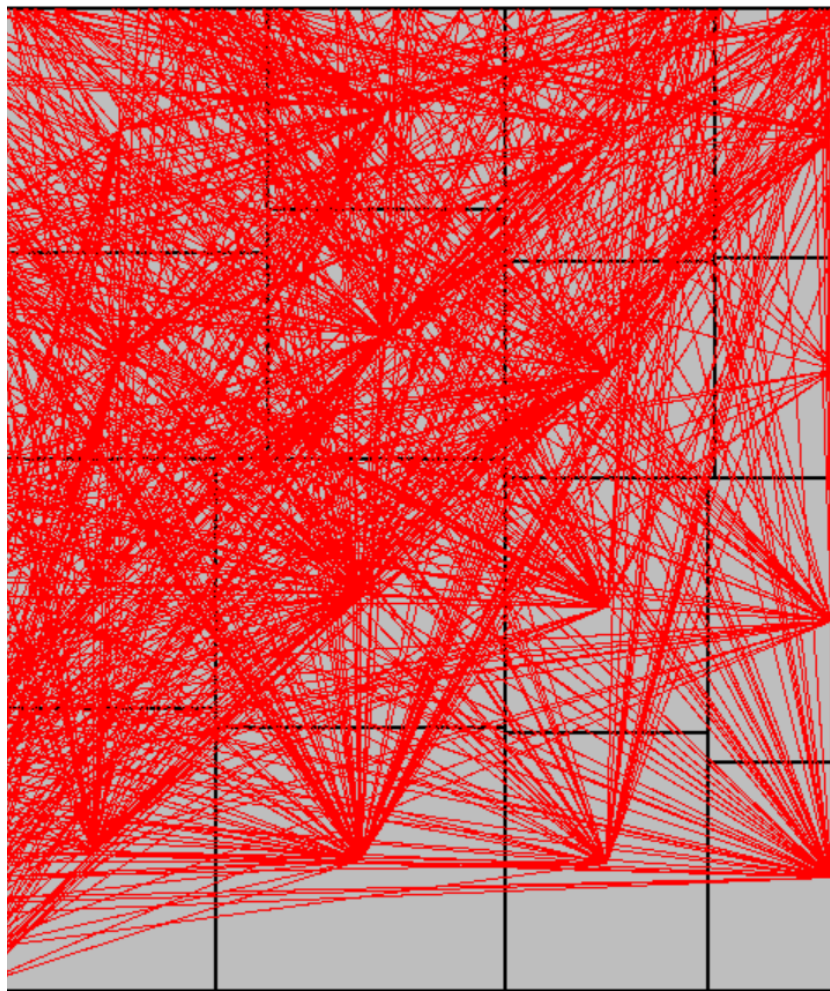


WL = 77441 (WINDOW = 1)

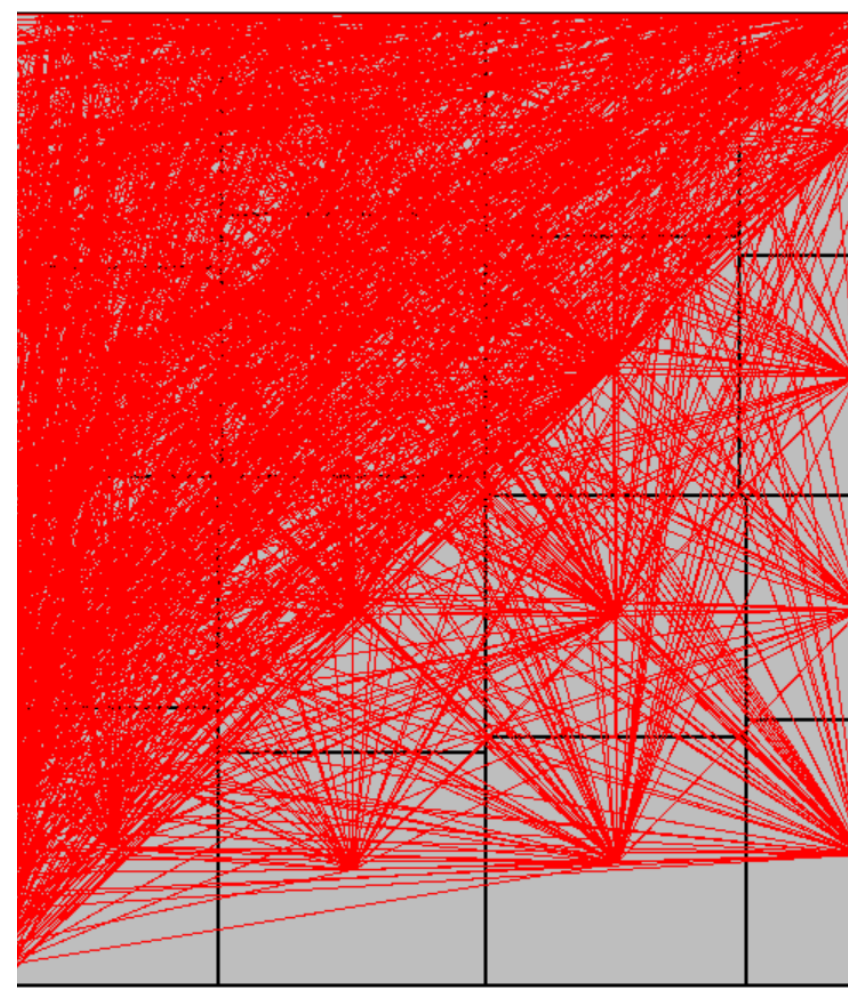
Wirelengths for varying window sizes : structP



WL = 232872 (WIN = 0)



WL = 229005 (WIN = 0.5)



WL = 241483 (WIN = 1)

Wirelengths for varying window sizes : p2



WL = 430287 (WIN = 0)

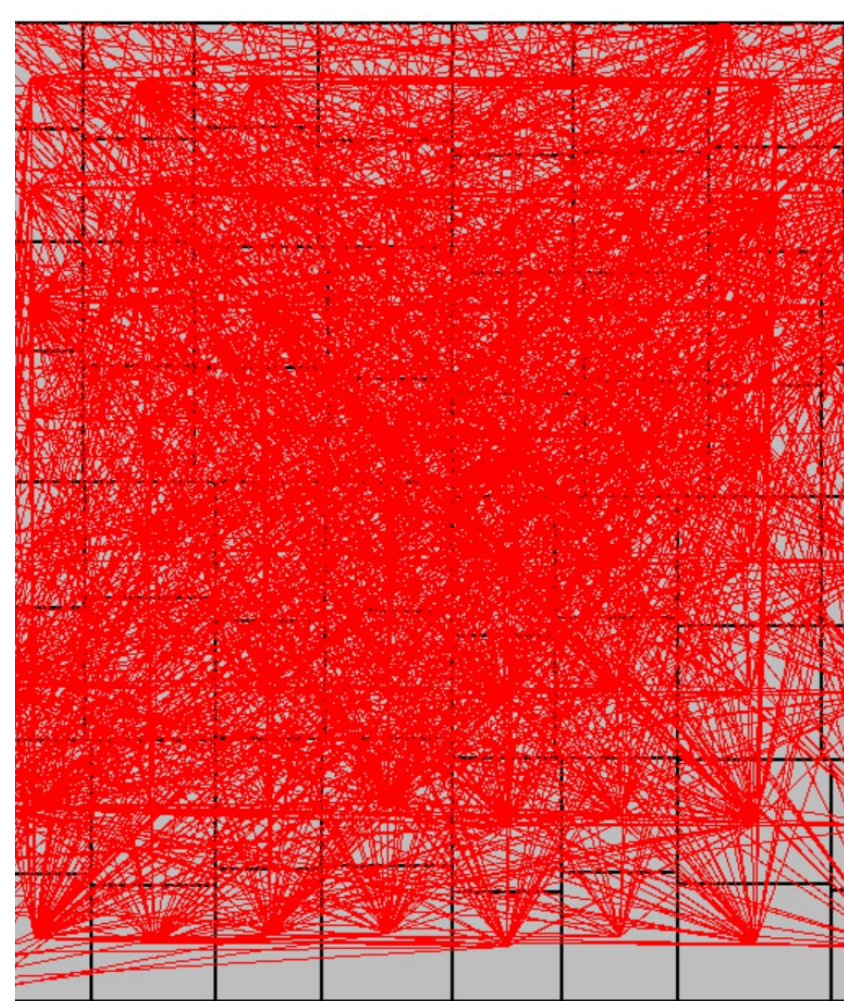


WL = 419735 (WIN = 0.5)

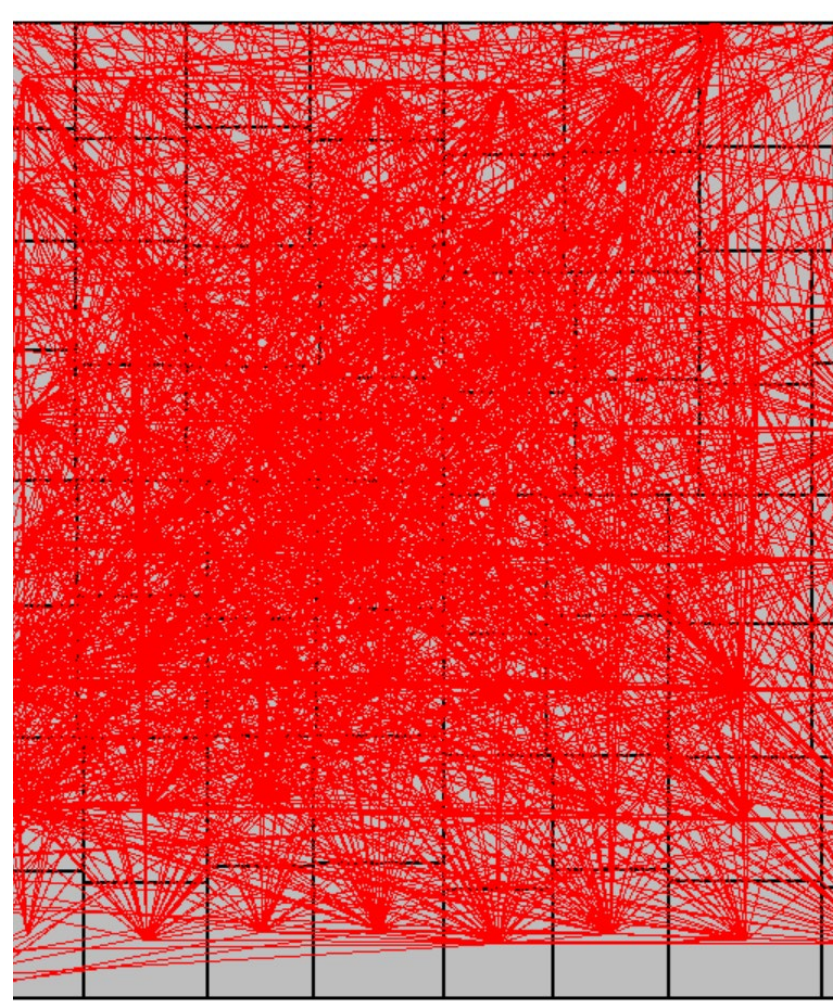


WL = 433179 (WIN = 1)

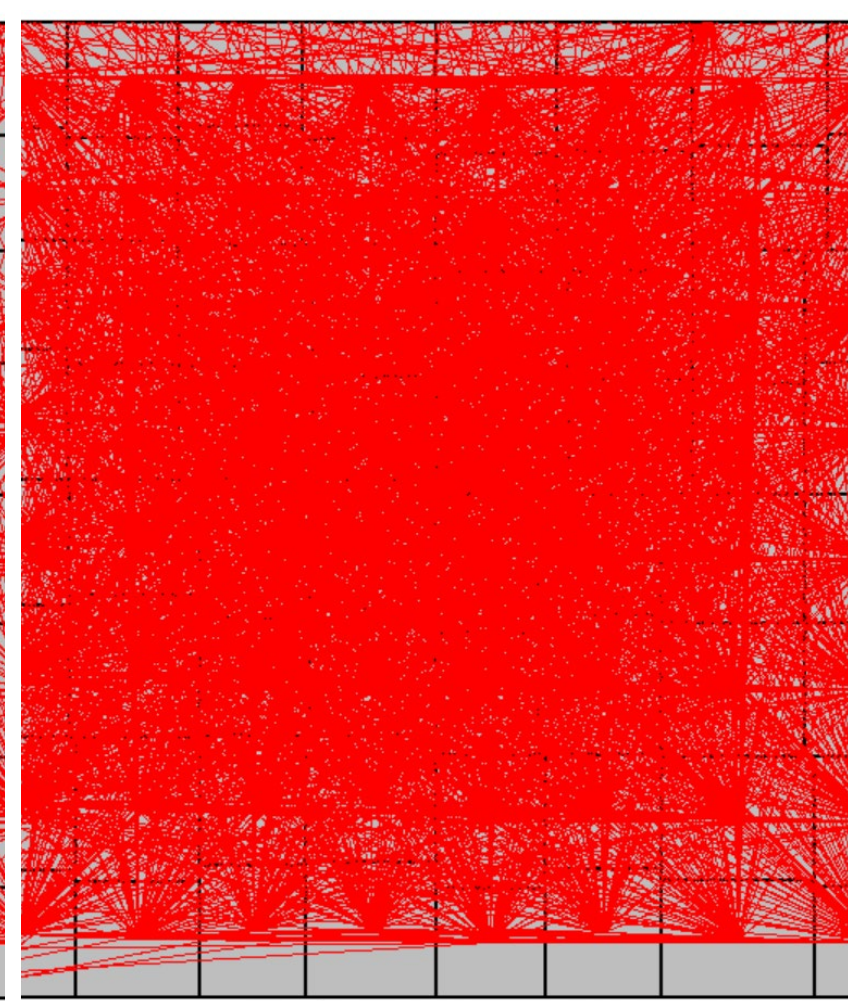
Wirelengths for varying window sizes : biomedP



WL = 1803850 (WIN = 0)

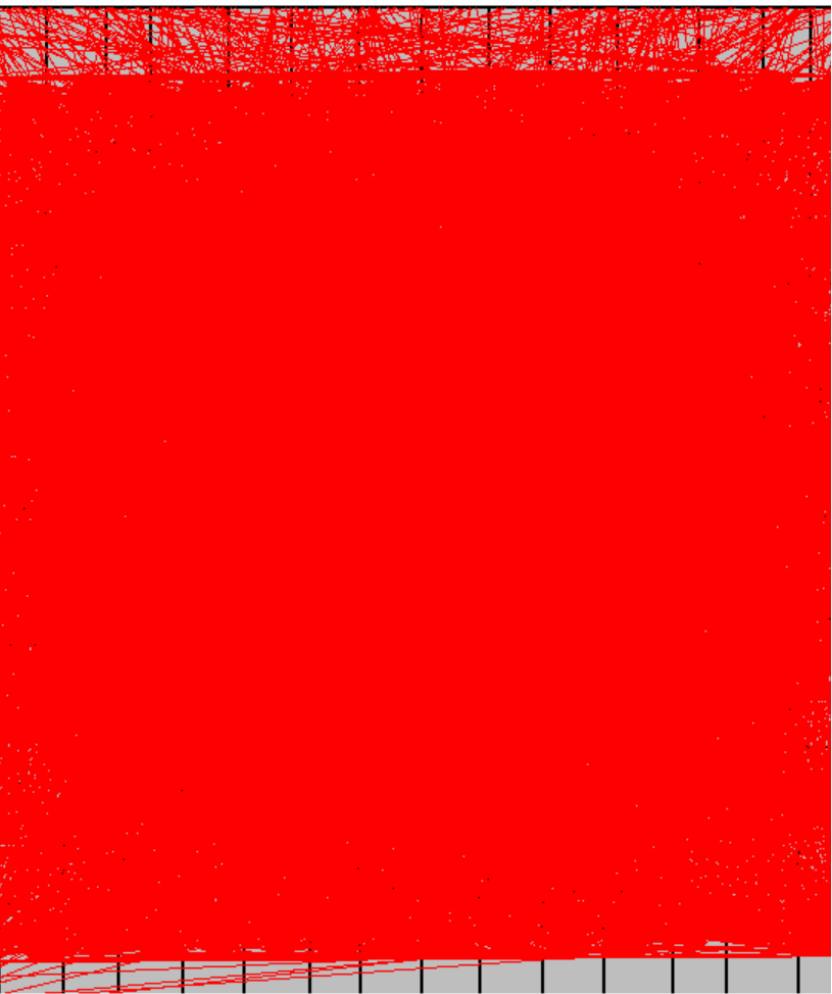


WL = 1789820 (WIN = 0.5)

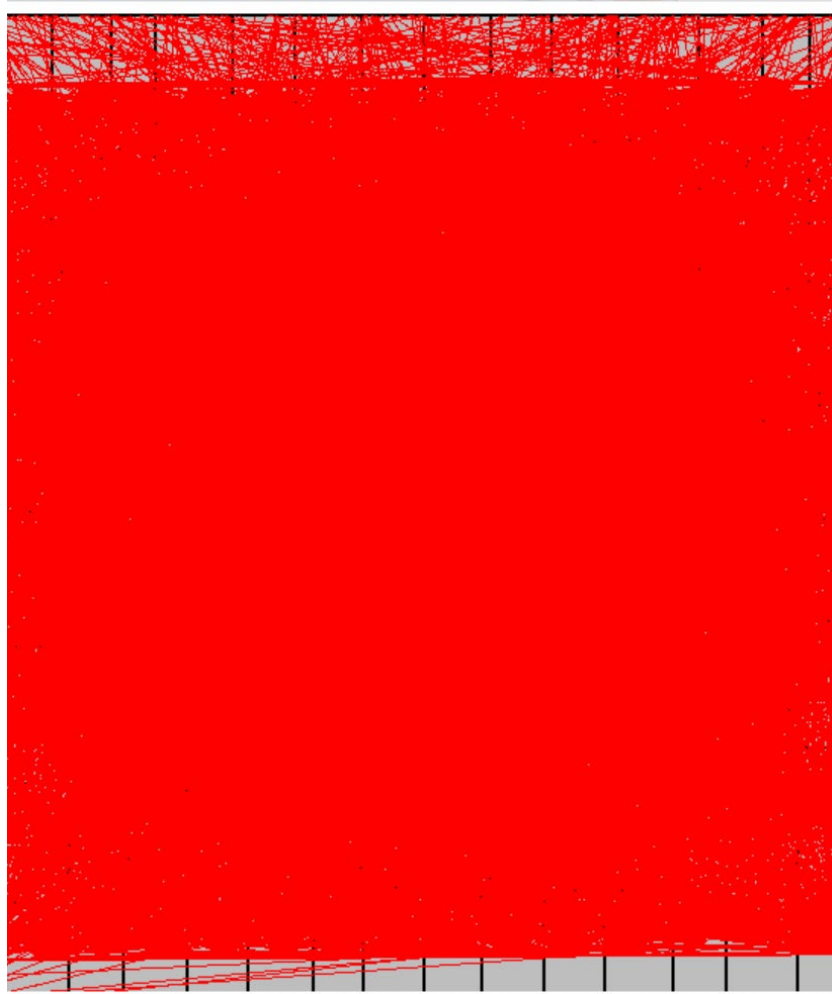


WL = 1826460 (WIN = 1)

Wirelengths for varying window sizes : industry2



WL = 3725490 (WIN = 0)



WL = 3647360 (WIN = 0.5)

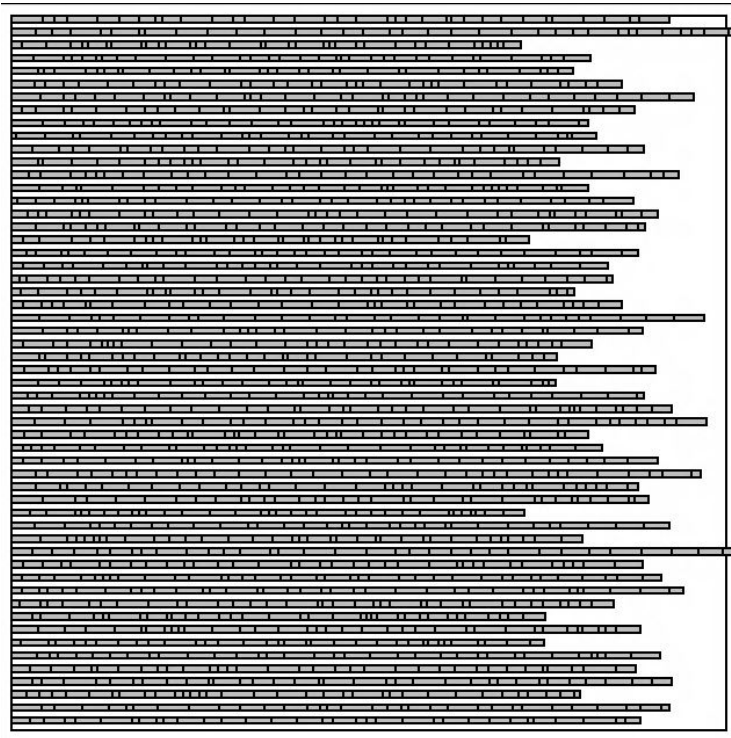


WL = 3756640 (WIN = 1)

Wirelengths for varying window sizes : industry3

Detailed Routing

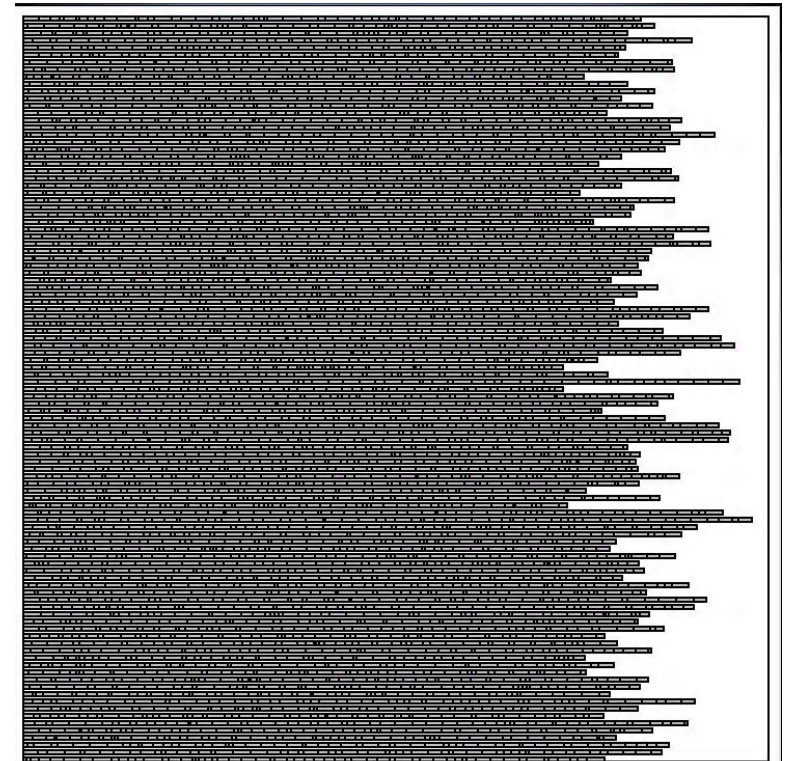
structP



p2

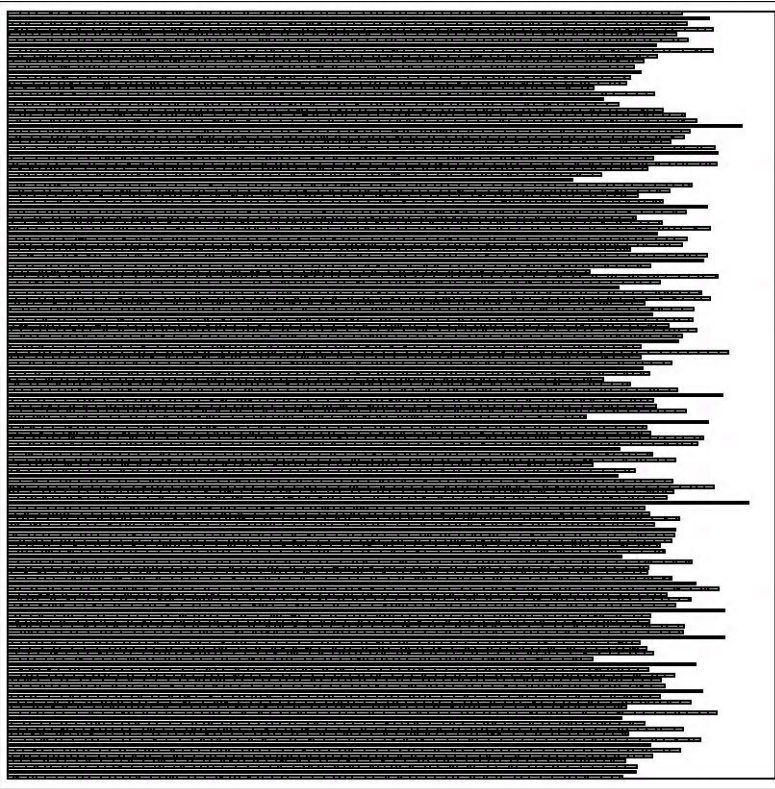


biomedP

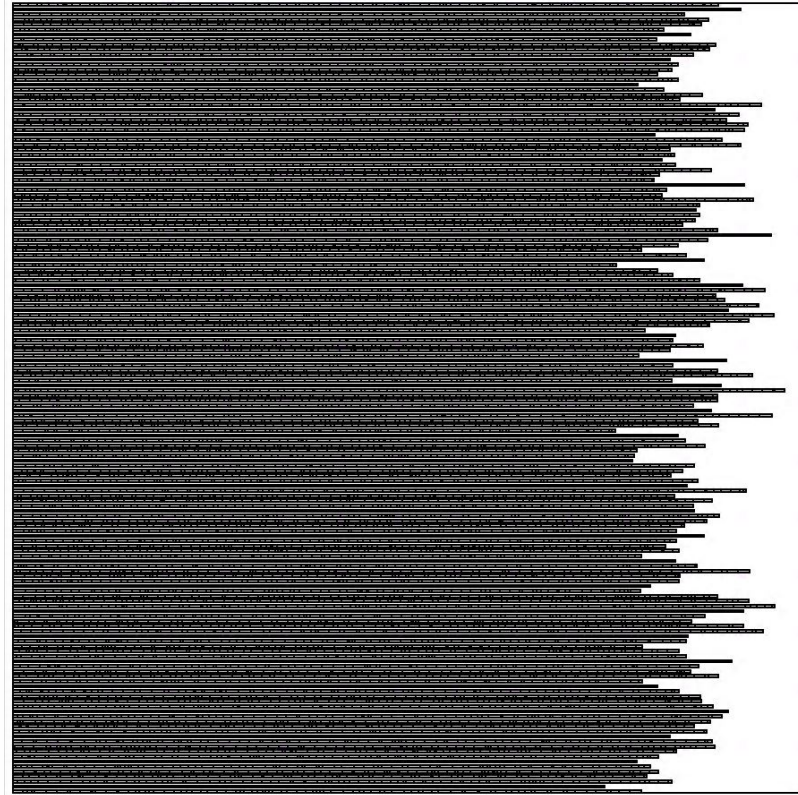


Detailed Routing

industry2



industry3



Conclusion

Terminal Propagation helps in reducing the wirelength, but the number of terminals that are propagated need to be significantly limited to get good results.

Using FM instead of KL gives a significant improvement in runtime. Running big workloads with KL is not a good idea as the runs will not end.

Significant tuning of parameters is required to get even better results for terminal propagation.

Row balancing for detailed placement is required if there are less number of cells per row. If there are a lot of cells per row then random allocation of cells to rows takes care of row balancing. The difference is evident from the detailed placements of structP and industry3.