Simulated Annealing Placement Results

Ckt Name	Argument	Chip Width	Chip Height	Initial Area	Final Area	Initial HPWL	Final HPWL	Runtime (s)
n10	-a	477	557	376480	265608	19585	17093	0.067
n10	-w	575	521	376480	299575	19585	13293	0.064
n10	-c	440	609	376480	267960	19585	23587	0.064
n30	-a	489	580	617895	283620	79987	54301	1.399
n30	-w	669	447	617895	299043	79987	41530	1.434
n30	-c	531	524	617895	52282	79987	278244	1.399
n50	-a	449	603	528528	270747	139548	108831	6.099
n50	-w	607	536	528528	325352	139548	101912	6.063
n50	-c	471	559	528528	263289	139548	112589	6.264
n100	-a	464	532	524160	246848	250119	199033	45.200
n100	-w	592	561	524160	332112	250119	155935	45.363
n100	-c	486	529	524160	257094	250119	194187	45.230
n200	-a	481	548	605241	263588	607356	442366	391.638
n200	-w	558	550	605241	306900	607356	359868	381.468
n200	-c	541	501	605241	271041	607356	414443	375.186
n300	-a	633	642	1113970	406386	1038710	725134	1456.48
n300	-w	708	736	1113970	521088	1038710	594609	1402.3
n300	-c	662	642	1113970	425004	1038710	672572	1374.94

- **Annealing Initialization:**

The annealing engine from MP2 was modified to handle the new input of modules in placement. Specifically, initialization of annealing was done dynamically in MP3 to ensure about 90% of moves will be accepted in the first annealing step. This is done by averaging the delta cost of 50 perturbations and scaling k to meet 90% acceptance. This was different from my approach in MP2 because now we have annealing cost changing depending on user input. This way the annealing generally has the same

acceptance rate at the beginning and converges in similar number of steps for each input option.

- **Initial placement:**

Initial placement was done by randomly creating the positive and negative loci, creating horizontal and vertical constraint graphs from the random loci, and performing longest path traversal of these graphs to obtain x and y locations of each module.

- **Temperature Considerations:**

Initial temp, T_O, was tested to find a balance between runtime and cost reduction. Initially T_O was set large but after plotting cost over time, it was found that the algorithm converges quite quickly and could instead start at a smaller temp like 500. It was noticed with smaller circuits that as annealing converged, there would be several consecutive iterations which would result in no moves made. Here the algorithm automatically terminates after 10 no move steps as the extra time to compute the steps does not provide a return on cost reduction.

- **The cost function**

Realized as:

alpha*area + (1-alpha)*HPWL

the cost function was tested for different values of alpha. It was found that an alpha value of 0.7 finds a balance between the annealing options to optimize for area and HPWL. From my findings, the annealing moves have a harder time to minimize area than HPWL so giving area a higher weight in the combined cost function balanced the two out.