Chapter 4

Experimental Results

Our algorithm described in Chapter 3 is implemented in C language and executed on SUN Sparc Workstations. We use TSMC spice model in *0.18um* technology to perform SPICE simulation for low V_{th} cells and the high V_{th} sleep transistor. The maximum current of one sleep transistor is computed as 432uA under $(\frac{W}{L})_{sleep} \approx 35$ to maintain 5% degradation in circuit performance. Several MCNC benchmark circuits is tested to show our experimental results. The experiment is performed using the gate type in *mcnc.genlib* of MCNC general library. Besides, We use a fast, effective standard cell placement tool called "*Dragon*" [10] to assist us with placement of cells.

Before showing the experimental results of two standard cell placement algorithms proposed in Chapter 3, we first compare the chip area of direct placement with the maximum current computed by functionality-check to

Table 4.1: Chip area of direct placement with and without functionality information

Circuits	Area(T)	Area(T+F)	A_red(%)	Wire(direct)
c880	34720	31640	8.87	8129
c1355	50660	44540	12.08	17222
c1908	69840	60120	13.92	24024
x3	74360	70400	5.33	13704
vda	112320	98400	12.39	50001
dalu	169880	151280	10.95	68090
c3540	115000	106000	7.83	48126
c5315	175360	158720	9.49	66287
c6288	190720	150400	21.14	47358
i10	256040	229400	10.40	125719
Avg.	-	-	11.24	-

that only with topology-check. Table 4.1 shows the results. The column labeled **Area(T)** are the chip area considering only topology information. The column labeled **Area(T+F)** are the chip area considering both topology and functionality information. The column labeled **A_red** are the reduction ratio of chip area taking functionality into consideration and it is computed by:

$$A\_red = \frac{Area(T) - Area(T+F)}{Area(T)} \times 100\ (\%)$$

The column labeled **Wire(direct)** are the total wirelength of the direct placement. The results show that taking both topology and functionality into considering results in about 11.24% reduction of chip area as compared to only considering topology.

Table 4.2 shows the experimental results of two placement algorithms we have proposed. **FP** denotes the results of *functionality directed placement algorithm*, and **DP** denotes those of *direct placement with iterative cell moving algorithm*. The columns labeled **Area** and **Wire** are the chip area and the total wirelength obtained from running our algorithms.

To understand the experimental results of our proposed algorithms more clearly, we compute the area reduction ratio and the wirelength increase ratio of two cell placement algorithms. Table 4.3 shows the results. The columns labeled **A_red** are the reduction ratio of chip area compared to the direct

placement and computed by:

$$A\_red(FP) = \frac{Area(T+F) - Area(FP)}{Area(T+F)} \times 100\ (\%)$$

$$A\_red(DP) = \frac{Area(T+F) - Area(DP)}{Area(T+F)} \times 100\ (\%)$$

The columns labeled **W_inc** are the increase ratio of total wirelength compared to the direct placement and computed by:

$$W\_inc(FP) = \frac{Wire(FP) - Wire(direct)}{Wire(direct)} \times 100\ (\%)$$

$$W\_inc(DP) = \frac{Wire(DP) - Wire(direct)}{Wire(direct)} \times 100\ (\%)$$

On the average, by the *functionality directed placement algorithm*, the chip area is reduced about 14.38% and the total wirelength increased about 32.43% compared to the direct placement. By the *direct placement with iterative cell moving algorithm*, the chip area is reduced about 9.18% and the total wirelength increased about 5.16% compared to the direct placement. The results show that the *functionality directed placement algorithm* makes better chip area. However, the limit of connecting the cells in the same cluster together causes the significant increase of wirelength overhead. On the contrary, the *direct placement with iterative cell moving algorithm* trades total wirelength for chip area, and it makes smaller wirelength overhead and worse chip area than the *functionality directed placement algorithm*.

Table 4.2: Results of two proposed cell placement algorithms

	FP		DP	
Circuits	Area	Wire	Area	Wire
c880	28560	10527	29283	8899
c1355	37740	21869	38422	17901
c1908	51120	34500	53290	25619
x3	58467	15960	136076	72138
vda	81420	67643	91256	52971
dalu	133380	92084	145816	70897
c3540	92665	61726	100253	50413
c5315	137768	88554	143131	69311
c6288	126126	65909	134422	50987
i10	195185	171273	206895	134695

Table 4.3: Area reduction and wirelength increase ratio of two proposed cell placement algorithms

	FP		DP	
Circuits	A_red(%)	W_inc(%)	A_red(%)	W_inc(%)
c880	9.85	29.50	7.45	7.44
c1355	15.78	26.98	13.69	3.94
c1908	14.97	43.61	11.36	6.64
x3	16.95	16.46	10.05	3.91
vda	17.56	35.28	7.26	5.94
dalu	11.83	35.24	6.36	4.12
c3540	12.58	28.26	5.42	4.75
c5315	13.20	33.59	9.82	4.56
c6288	16.14	39.17	10.62	7.66
i10	14.91	36.23	9.81	7.14
Avg.	14.38	32.43	9.18	5.61

To understand how these two placement algorithms trade area and total wirelength, we performed an analysis of the results after each iteration of cell moving procedure. Figure 4.1 and Figure 4.2 show the results.

Figure 4.1 shows an example of the curve of wirelength reduction in the iteration process performed by the algorithm proposed in Section 3.2. It starts from the initial placement in Section 3.2.2. This initial placement has smaller chip area and significant wirelength overhead. In each iteration, we record the reduction of total wirelength. Because, in the *cell moving among clusters algorithm*, the *moving constraint* is defined to limit the increase of one row, and the factor $RD(c_{x,i->j})$ used to reduce the maximum size of all rows in the cost function is given a lower weight, the chip size is almost the same after each iteration. In this figure, X-axis shows the iteration times of the *inner while loop of iterative cell moving* in the *cell moving among clusters algorithm*. Y-axis shows the wirelength reduction ratio compared to the initial placement after each iteration. From this figure, we can see that the total wirelength can be reduced after each iteration of cell moving procedure.

Figure 4.2 shows an example of the curve of area reduction and wirelength increase in the iteration process performed by the algorithm proposed in Section 3.3. It starts from the initial placement from direct placement.

This initial placement has worse chip area and no wirelength overhead. In each iteration, we record the increase of chip area and the reduction of total wirelength. In this figure, X-axis and Y-axis show the ratio of wirelength increase and area reduction compared to the initial placement respectively. From this figure, we can see that this algorithm gradually trades total wirelength for chip area by performing the *cell moving among clusters algorithm*.

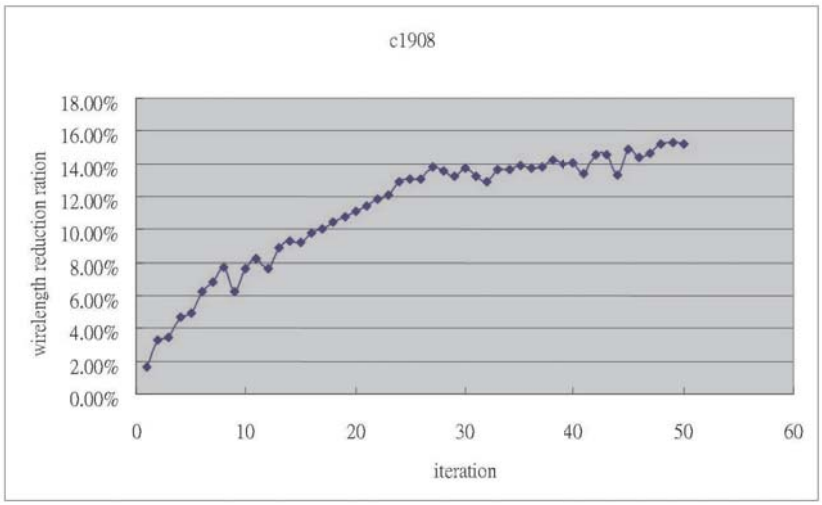

Figure 4.1: An example of the curve of wirelength reduction in the iteration process performed the algorithm proposed in Section 3.2.

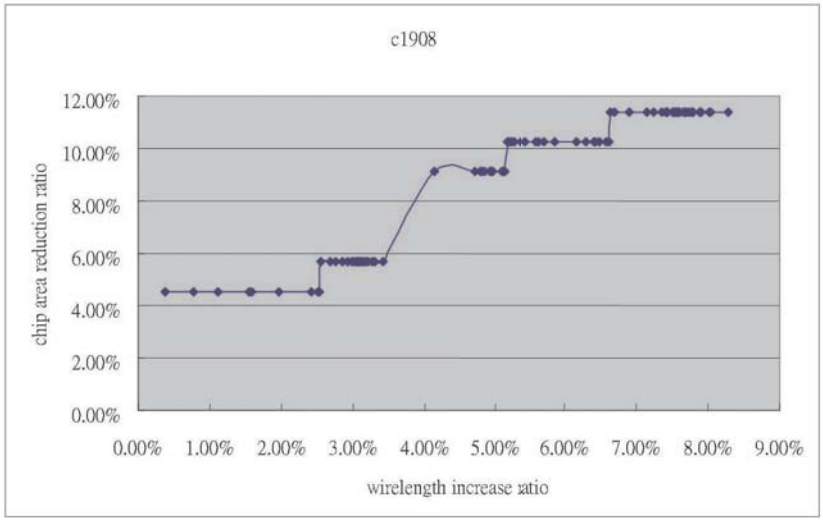

Figure 4.2: An example of the curve of area reduction and wirelength increase in the iteration process performed the algorithm proposed in Section 3.3.