A Snap-On Placement Tool *

Xiaojian Yang, Maogang Wang, Kenneth Eguro and Majid Sarrafzadeh
Department of Electrical and Computer Engineering
Northwestern University
Evanston, IL 60208-3118
xjyang,mgwang,eguro,majid@ece.nwu.edu

ABSTRACT

The standard cell placement problem has been extensively studied in the past twenty years. Many approaches were proposed and proven effective in practice. However, successful placement tools need enormous time in the course of development. In this paper we propose a new *snap-on* placement tool, which is based on multilevel hierarchical placement method. It has great flexibility to combine existing packages and techniques in its top-down framework. In addition, it can be used to build a good placement tool in a short amount of time.

Some important issues in multilevel hierarchical placement are discussed here. We investigate the behavior of net-cut and wirelength objectives in global placement problem, propose a $+\delta$ level clustering technique and design a new top-down placement method based on partitioning, annealing and $+\delta$ level technique. We also work on the trade-off between solution quality and running time during the hierarchical placement. Experimental results show the strength of proposed placement tool, it produces very good results on all benchmarks and the best known result on the largest MCNC benchmark (*avql*).

1. INTRODUCTION

The goal of placement in VLSI physical design is to produce a chip layout with optimized area and routability which also satisfies timing constraint. This classical problem has been extensively studied for over two decades. Traditional placement approaches include min-cut based algorithms [1; 2], annealing based algorithms [3; 4] and quadratic programming algorithms [5; 6]. As the size of VLSI design becomes larger and larger, the hierarchical multilevel methodology becomes the basis of many effective approaches for solving the placement problem. Sun and Sechen [7] proposed a hierarchical based annealing algorithm using a clustering technique. In [8], Sarrafzadeh and Wang showed that solving a hierarchical placement problem helps to reduce the size of the solution space of the original placement problem without sacrificing solution quality. Shin and Kim [9] presented a hierarchical placement algorithm which combines partitioning and simulated annealing techniques. Experiments showed that minimizing net-cut in each placement hierarchical level positively affects the wirelength quality of final solution.

The existing successful VLSI/CAD tools, e.g., TimberWolf [7], have spent huge amount of time in software developing and fine-tuning. With the development of the VLSI industry, the competitive market requires future CAD tools to be more effective and more efficient. However, the next generation CAD tools probably can not afford a long term developing cycle, thus an efficient way to develop a CAD tools is needed. In the placement problem, there are many available techniques (simulated annealing, analytical approach, flow based method, etc.) and software tools (e.g. partitioning package) available. Effectively utilizing existing techniques and tools will lead to a new way in developing new VLSI/CAD approaches. The main challenge here is to understand the placement problem and different tools well enough to know when to use these tools, which specific tool should be used and the reason behind them.

In this paper we design a new "*snap-on*" placement tool framework for the purpose of evaluating the strength of combining different existing techniques. As shown in Figure 1, the snap-on placement tool is composed of placement data box, cost evaluator, control unit and pre-designed software packages. The placement data box contains all the placement circuit information (standard cells, I/O pads, netlist etc.) and current status of the placement. Pre-designed packages are available software packages or functional units using relevant techniques. The cost evaluator calculates the placement cost and sends information to control unit. The control unit initializes and masters the operation of placement workflow. It reads current status and cost of the placement, determines which specific pre-designed package should be used at any specific time. The control unit works according a series of microcodes. These micro codes have to be written by someone who understands the placement problem well and knows many properties of different algorithms. One of the advantages of this snap-on placement tool framework is that it has the potential to produce good placement by using the appropriate algorithms in different stages of the placement workflow. The other advantage is the simplicity of the snap-on tool framework being easy to implement. Comparing with other successful placement tools which need

*This work was supported in part by NSF grant MIP-9527389.

several years to develop and test, this snap-on tool needs several weeks to build.

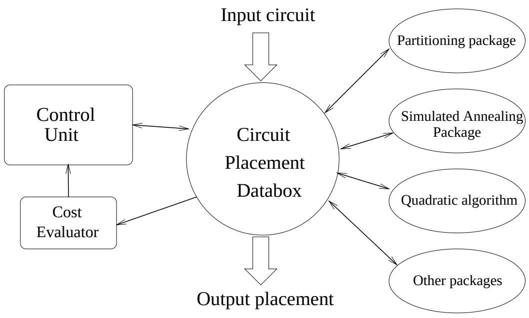


Figure 1: The framework of snap-on placement tool

Since the top-down hierarchical methodology is one of the most effective way to solve the placement problem of large circuits, we use it as the basis for our snap-on placement tool. Other existing software packages and techniques will be added on to this hierarchical framework. In this paper, we will focus on studying the "gluing" mechanism of this framework which can effectively combine other algorithms in one powerful placement tool.

We first analyze the relationship between the net-cut and the wirelength objective. Better understanding this problem enable us to use the leading-edge partitioning tool hMetis ([10; 11]) in our placement tool. We also presented a $+\delta$ level refining technique which can be used in our framework to further improve the placement quality. Our snap-on placement tool includes partitioning, annealing and the $+\delta$ level refining techniques.

The rest of the paper is organized as follows: In Section 2, we formulate the multilevel placement problem. In Section 3 some issues in the top-down placement framework are studied. The placement workflow is presented in Section 4. In Section 5, experimental results are shown and followed by the final conclusions in Section 6.

2. PROBLEM FORMULATION

A typical multilevel placement approach is based on recursive circuit partitioning. It repeatedly divides a given circuit into subcircuits to optimize a given partitioning objective. At each level, the given layout area is partitioned in either the horizontal or the vertical direction or both. Each subcircuit is assigned to a partition. Recursive partitioning is repeated until each subcircuit contains a small number of cells.

We use the concept of global bins to analyze the multilevel placement. At a given hierarchical level, we divide the layout area into N_b rectilinear regions, each of these regions is called *global bin*. Assume we have r rows and c columns of global bins ($N_b = r \times c$). We label the global bin at i_{th} row and j_{th} column as B_{ij}. The center of global bin B_{ij} is denoted by $C_{B_{ij}} = (x_{B_{i,j}}, y_{B_{i,j}})$.

In this paper we assume that we are given a circuit denoted $Ckt(\mathcal{C}, \mathcal{N}, \mathcal{S})$, which consists of a set of cells $\mathcal{C} = \{C_i | i = 1, \ldots, |\mathcal{C}|\}$, a set of nets $\mathcal{N} = \{N_i | i = 1, \ldots, |\mathcal{N}|\}$ and a set of terminals $\mathcal{S} = \{S_i | i = 1, \ldots, |\mathcal{S}|\}$. Each *net* N_k consists of a set of terminals $S(N_k) \subset \mathcal{S}$. Similarly, each cell C_k contains a set of terminals $S(C_k) \subset \mathcal{S}$. The terminal set

$$\mathcal{S} = \bigcup_{k=1}^{|\mathcal{N}|} S(N_k) = \bigcup_{k=1}^{|\mathcal{C}|} S(C_k)$$

Let the location, on the plane, of the cell C_i be denoted by (x_{C_i}, y_{C_i}). Then the location of terminal $S_j \in S(C_i)$ is represented by $(x_{C_i} + x_{S_j}, y_{C_i} + y_{S_j})$, where (x_{S_j}, y_{S_j}) is the offset location of terminal S_j with respect to its parent cell C_i. The wirelength of a net N_i is defined as the half perimeter of the bounding-box for net N_i. The total wirelength of the placed circuit is the summation of the wirelength for all the nets in the circuit.

In the hierarchical placement problem, each global bin $B_{i,j}$ contains a set of cells $P_{i,j} \subset \mathcal{C}$. For any cell in a global bin, the location of the cell is set to the center of that bin. $\forall C_k \in P_{i,j}, (x_{C_j}, y_{C_j}) = (x_{B_{i,j}}, y_{B_{i,j}})$. Cells are placed into global bins to minimize the total wirelength. In order to prevent all the cells from being placed in the same global bin (zero total wirelength), the balancing constraint has to be imposed. The balancing constraint for a certain hierarchical level which has N_b global bins can be described as:

$$(1-u)\frac{A}{N_b} < \sum_{C_k \in P_{i,j}} A_k < (1+u)\frac{A}{N_b}$$

where A is the total cell area, A_k is the area of cell C_k, $0 < u < 1$ is the unbalancing factor.

Net N_i is not cut if and only if all the terminals of N_i are located in the same global bin. The net-cut at a given hierarchical level is defined as the total number of cut nets. This definition is consistent with net-cut definition for a N_b-way partitioning problem. We call all the uncut nets *internal nets* since they are located inside one global bin. We call all the cut nets *external nets* since they span more than one global bin.

3. TOP-DOWN FRAMEWORK

The top-down hierarchical placement methodology is the basis of our snap-on placement tool framework. Generally, a top-down hierarchical placement approach starts from the a top hierarchical level (e.g. 2×2 global bins). In each hierarchical level i, the placement problem is to put cells into $m_i \times n_i$ global bins to minimize a certain placement cost (usually is the total wirelength). We call this problem a hierarchical placement problem. After the hierarchical placement problem in this level is solved, the top-down placement approach goes to the next hierarchical level $i+1$ by splitting each global bin in the current level to four smaller global bins. Thus the $i+1$ level has $m_{i+1} \times n_{i+1} = 2m_i \times 2n_i$ global bins. This procedure will be repeated until the number of cells in each global bin less than a pre-defined value. After the top-down hierarchical placement process is finished, cells are still located at centers of global bins. A final placement stage is needed to perform some local adjustment and produce a legal layout.

We can put different kinds of software packages into this top-down hierarchical placement framework. The net-cut cost which is used in the partitioning problem is globally consistent with the wirelength cost in the placement problem. The partitioning problem is a similar but a much easier problem comparing to the placement problem. A leading-edge partitioner can be faster than a good placement tool by at least an order of magnitude. Thus putting a good partitioning tool into our placement framework may help us get a good initial placement. Simulated annealing is still the best placement algorithm for "small" circuits in terms of quality of the placement result although it may need a lot more running time than other algorithms. So we use simulated annealing in our framework to perform the wirelength optimization. In this section, we will empirically study the gluing mechanism in our framework, i.e., how to combine techniques in our framework to form an effective placement tool. We use larger MCNC benchmark circuits for our experiments in this paper. Table 3 shows the properties of the testing circuits.

ckt	#cells	#pads	#nets	#pins	#rows
primary1	752	81	1266	3303	16
primary2	2907	107	3029	18407	28
biomed	6417	97	5742	26947	46
industry2	12142	495	13419	48404	72
industry3	15059	374	21940	68418	64
avqs	21854	64	30038	84145	80
avql	25114	64	33298	90601	88

Table 1: MCNC benchmark Information

3.1 Net-cut and Wirelength

At coarser hierarchical levels, the net-cut cost is very close to the wirelength cost. For instance, if the number of global bins is two, then minimizing net-cut is equivalent to minimizing wirelength. This fact suggests that the partitioning tools could be added to the placement procedure at coarser hierarchical levels. Intuitively, partitioning the original circuit into several highly connected sub-circuits will always help the placement procedure. Later on, the solution of partitioning gets further away from the solution of placement because partitioning tools do not consider any physical positions of cells. Thus wirelength need be considered at finer levels. We empirically find that the ratio of external nets is a useful metric to decide where is the point to start considering wirelength.

For a cluster in a placement, Rent's rule [12] quantitatively defines the number of external pins can be expressed as $P_m = T_b B_m^r$ where B_m is the average number of cells, T_b is the average number of terminals per block, and $0 \leq r \leq 1$ is called the rent parameter. Rent's rule is experimentally validated for a lot of real circuits and for different partitioning methodologies. For real circuits, the Rent parameter r usually has a value of between 0.3 and 0.8. If a circuit obeys Rent's rule, we can derive a theoretical relationship between the external nets percentage and the number of global bins. Assume we have N_b global bins with all the cells distributed in them evenly, thus we have $\frac{|\mathcal{C}|}{N_b}$ cells in each global bin. According to Rent's rule, the number of external pins on each global bin $P_m = T_b(\frac{|\mathcal{C}|}{N_b})^r$. The total number of external nets will be $\frac{P_m}{p_{avg}} N_b = N_b \frac{T_b}{p_{avg}} (\frac{|\mathcal{C}|}{N_b})^r$ where p_{avg} is the average number of terminals per net. Thus the percentage of the external nets is $p_{ext} = \frac{N_b}{|\mathcal{N}|} \frac{T_b}{p_{avg}} (\frac{|\mathcal{C}|}{N_b})^r$. As defined $T_b = \frac{p_{avg}|\mathcal{N}|}{|\mathcal{C}|}$. Plug this value back into the relation, we get $p_{ext} = (\frac{|\mathcal{C}|}{N_b})^{r-1}$. Since the actual value of the Rent parameter r varies from circuit to circuit, for each circuit, we plot five theoretical curves each one with a different Rent parameter value ranging from 0.3–0.7. Figure 2 shows the curves for four MCNC benchmark circuits (Primary1, Primary2, biomed and avqs).

From Figure 2 we can see that Rent's curve is not exactly obeyed by real circuits. Circuits may have different Rent parameter values in different hierarchical levels. This is consistent with the way a VLSI circuit is designed. The hierarchical design methodology in VLSI tends to combine a number of small sub-circuits into one big circuit. However, the degree of complexity is different according to the size of the circuit. When the size of a circuit is small, it is possible to put very complicated logic in it. Thus it will have a large Rent parameter. When the size of a circuit is large, the logic between sub-circuits will be comparably simple. Thus it will have a small Rent parameter. As shown in Figure 2, most circuits have a smaller Rent parameter in early hierarchical levels (large subcircuits) than in later levels (small subcircuit).

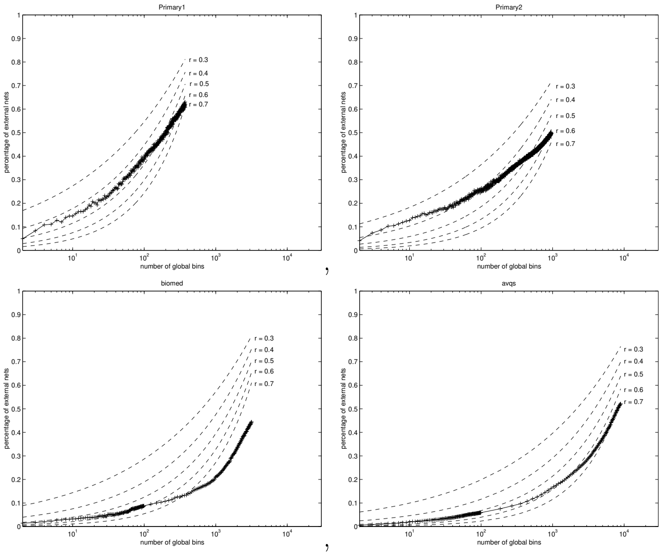


Figure 2: Percentage of external nets vs. number of global bins.

Wang and Sarrafzadeh [13] investigated the relationship between net-cut objective and wirelength objective in the placement problem. The difference between the net-cut and the wirelength objective is the cost of external nets. Thus the more the external nets, the bigger difference there is between these two objectives. Based on the percentage curves of external nets, we empirically found that $20\% - 30\%$ is the percentage where we should start considering wirelength. When less than $20\% - 30\%$ nets are external in a hierarchical level, net-cut is a very good estimation of wirelength. Thus we can use net-cut objective at this hierarchical level. If more than $20\% - 30\%$ nets are external, net-cut is no longer a good objective in order to minimize wirelength. We should start using wirelength as the optimization objective.

This "20%−30% external nets" rule is based on the intuition and the actual experimental results. It is an approximate rule. Constructing a external net ratio curve for a circuit is very time consuming. However, we do not really need the whole curve to determine the place where we need to start considering wirelength. At a given hierarchical level, we can decide whether we should consider wirelength by looking at the external net ratio at this level. Thus it is very easy to make the decision based on the net-cut result.

We conduct an experiment to verify this "$20\% - 30\%$ external nets" rule. We begin our top-down process from the top level which has 2×2 global bins. Successive hierarchical levels are obtained by global bin quadrisections. A switching point is a specified hierarchical level where we start considering wirelength. We perform circuit partitioning without considering wirelength before the switching point. After the switching point, wirelength optimization is performed using simulated annealing.

Table 2 shows the experiment in which we set different global placement levels as switching points for benchmark *avqs*. We compare the total wirelength of the global placement. The first row represents the global wirelength at each hierarchical level produced by the algorithm with a switching point at 2×2 level. The second row corresponds to 4×4 level, etc.

Switch	GP wirelength at hierarchical level					
points	2×2	4×4	8×8	16×16	32×32	64×64
2×2	2.43	3.53	4.39	5.09	5.98	5.91
4×4		3.25	4.15	4.59	5.05	5.73
8×8			4.21	4.55	5.02	**5.63**
16×16				4.81	5.25	5.88
32×32					5.66	6.28
64×64						8.08

Table 2: Global placement(GP) wirelength comparison between algorithms with different switching points(at this level wirelength optimization is started) on benchmark *avqs*, switching at 8×8 level produces the best result as shown in boldface

Note that only comparing the same column data, meaning compare global wirelength of the same bin grids, is valuable in terms of getting good placement. As we know, if the global placement is balanced, i.e., each global bin contains roughly same total cell area, the global wirelength in finer placement level is a good prediction of detail wirelength. We can assume that a better global placement leads to a better detail placement by using a final placer. By observing the global wirelength in 64×64 level in Table 2 one can conclude that a top-down global placement starting from very top level (2×2, 4×4) does not necessarily generate better result comparing with that starting from a lower level. A main reason is that performing wirelength optimization too early means nothing for the quality of finer global placement, but may negatively affect subsequent partitioning results. Experiments on other benchmarks shows similar trends.

3.2 When to stop global placement

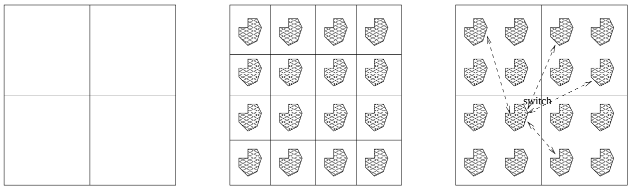
Overlaps between cells exist in global placement since we assume that all the cells within the same global bin are placed at the center of the global bin. After global placement, we will apply final placement procedure to remove these overlaps and make the placement legal. Obviously, the smaller the average number of cells in each global bin, the less effort the final placer will spend. Thus stopping the global placement procedure late will definitely help the final placer.

On the other hand, stopping the global placement procedure too late is not good either. Because cells are placed at the centers of global bins and all the global bins have the same size, wirelength measured at the global placement stage is not the same as the wirelength measured in the final placement stage. Since the main purpose of the global placement is to find an approximate location of each cell, an arbitrarily small global bin size will not help due to the difference between the wirelength measurement in global and final placement stage. Therefore, stopping the global placement procedure too late is unnecessary.

In summary, we should look for the the earliest global placement level which give us the best final placement results. We use experiments to find the appropriate point to stop global placement. In experiments, we stop the top-down global placement procedure at different levels and feed the obtained placement to a final placer which is implemented using simulated annealing. We compare the wirelength of the final placement to see the appropriate point to stop the global placement. Table 3 shows the experimental results for MCNC benchmark circuits[1] The average number of cells in each global bin, N_{avg}, represents the size of global bins. From the result we conclude that an appropriate stopping point is the global placement level where N_{avg} is between 3–6.

3.3 $+\delta$ Level Wirelength Improvement

In the wirelength optimization at a given hierarchical level, we propose a $+1$ level clustering technique to minimize wirelength efficiently: Given a hierarchical level h which has N_b global bins, first we partition each subcircuit in each global bin into g clusters. Then simulated annealing is applied on these gN_b clusters to optimize wirelength at the current level, i.e. cells in the same cluster are located on the global bin center of current level. Figure 3 shows an example of the $+1$ level clustering technique where $g = 4$.



(a) The target level h.

(b) do net-cut optimization at level h+1. Cluster cells based on the net-cut result.

(c) do wirelength optimization optimization by moving clusters back at level h.

Figure 3: The $+1$ *level clustering* technique to improve wirelength.

By splitting the subcircuits into smaller clusters we give clusters more freedom to reach their better locations, while the number of movable objects is still limited. Considering the fact that locations of clusters are fixed at centers of global bins, the annealing process is much faster than moving single cell. Furthermore, if the initial placement before $+1$ level is already a good one, we simply need a low temperature annealing to perform the improvement task well. So the overhead for $+1$ level improvement is relatively low.

The $+1$ level clustering technique can be extended to $+2$, $+3$ or $+\delta$ level clustering technique. As δ increases, the $+\delta$

[1] To save space, from now on we use four of MCNC benchmarks ($primary2$, *biomed*, *avqs*, *avql*) to show the experiment results. Trends are similar in other benchmarks.

	primary2			biomed			avqs			avql		
	W_g	W_f	N_{avg}	W_g	W_f	N_{avg}	W_g	W_f	N_{avg}	W_g	W_f	N_{avg}
2×2	2.38	-	726	1.03	-	1604	2.43	-	5463	2.58	-	6278
4×4	3.38	-	181	1.22	-	401	3.23	-	1366	4.41	-	1569
8×8	3.89	-	45	1.45	-	100	4.21	-	341	4.80	-	392
16×16	3.96	3.89	11	1.86	2.01	25	4.55	7.64	85	5.50	7.35	98
32×32	3.62	**3.70**	2.8	1.85	**1.89**	6.2	5.02	6.71	21	6.53	6.92	24
64×64	3.49	3.93	0.7	1.77	1.94	1.5	5.63	**6.25**	5.3	6.55	**6.30**	6.1
64×128	-	-	-	-	-	-	5.49	6.49	2.7	6.37	6.64	3.1

Table 3: Experiments to determine the appropriate switching point between global placement (GP) and final placement (FP). (W_g: wirelength result after each level of global placement. W_f: wirelength after final placement). Best final placement result is shown in boldface, illustrates the appropriate point to stop global placement. Final placements are produced by NRG detail placer which is based on low temperature annealing.

technique will lead to better solution but the annealing run time will increase accordingly. If there is only one cell at each global bin at level $+\delta$ ($\delta = \frac{\log|\mathcal{C}| - \log N_b}{logg}$), the $+\delta$ clustering technique reduces to the single cell moving strategy. Again, choosing δ is a quality/runtime trade-off for minimizing wirelength in each hierarchical level. Table 4 compares the global placement wirelength on 8x8 level by different $+\delta$ clustering technique. Comparison of running time directs us to adopt +1 level clustering technique for improving wirelength without too much overhead on the running time.

circuit	**different $+\delta$ level**			
	flat	+1 level	+2 level	+3 level
	$WL(rt)$	$WL(rt)$	$WL(rt)$	$WL(rt)$
pri2	3.75(59)	3.67(90)	3.58(128)	3.54(516)
bmd	1.86(189)	1.40(260)	1.34(526)	1.35(1366)
avqs	4.20(533)	4.16(719)	4.15(1415)	4.13(4123)
avql	4.51(613)	4.46(830)	4.42(1679)	4.35(5056)

Table 4: Global placement wirelength(WL) and running time(rt) comparison at 8x8 hierarchical level by using different $+\delta$ level refining techniques. (pri2 is benchmark *primary*2, bmd is *biomed*.)

4. PLACEMENT WORKFLOW

Based on the techniques discussed previously, we design a specific top-down placement workflow for the snap-on placement tool as shown in Figure 4. The workflow utilizes both net-cut and wirelength minimizing steps. At the coarser placement level, net-cut is chosen as the optimization objective to rapidly get an initial placement. We use proven partitioning software hMetis[10; 11] to minimize net-cut. When the top-down flow reaches a certain level in which the number of external nets occupies 20%−30% of the number of total nets, simulated annealing begins to minimize wirelength on two dimensional placement plane. Then the solution is further improved by the +1 level technique comprised of splitting and simulated annealing. Once the global placement in current level is determined, the cell group in each global bin is split into four smaller clusters. The algorithm enters the next global placement level. Based on hierarchical method, partitioning and wirelength minimization are alternatively using to gradually reach a finer placement. This global placement stage ends when the stop criterion is satisfied, i.e., the average number of cells per global bin is less than a certain number. Finally, the achieved global placement is refined by final placer to generate a legal placement as the ultimate solution.

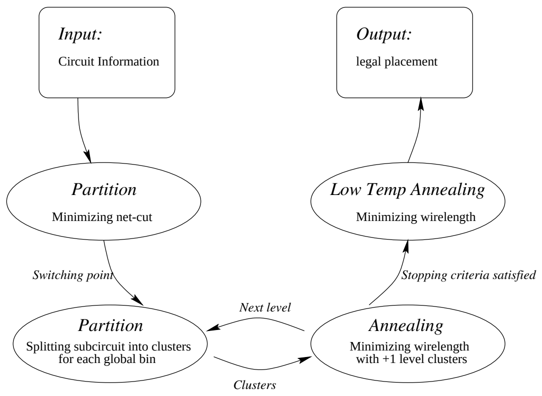


Figure 4: Hierarchical placement and global bins.

At each optimization level, combining both graph partitioning and +1 level annealing techniques helps to overcome disadvantage of using a single method. Partitioning ensures that cells with high connectivity are put into the same cluster while annealing drives clusters to their appropriate locations. By +1 level technique cells or cell clusters are not confined in a given region as they were in the pure min-cut based placement. In addition, +1 level technique also gradually improve the placement quality by breaking and reunioning cell groups in different global placement levels. Figure 5 depicts the trend of wirelength cost in the top-down global placement.

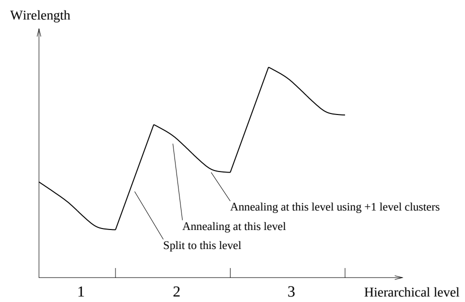


Figure 5: Wirelength change in hierarchical placement

5. RESULTS AND DISCUSSION

The proposed snap-on placement tool has been implemented in C++ language on Sun Ultra10 workstation. We use hMetis [11] partitioning package to minimize net-cut and use simulated annealing to minimize the wirelength in global placement stage. As the final placer, we use NRG[8], a local improvement tool based on low temperature annealing, flipping, switching neighbors and permutation exchange. Total wirelength is measured as the sum of the half perimeter wirelength.

We compared the results of our approach with the state-of-art university placement tools, TimberWolf v7.0 (the latest university version of TimberWolf) and the best known quadratic placement [6]. Table 5 shows the comparison of both final placement wirelength(m) and running time($second$). Our approach produces comparable or better results than TimberWolf on all of MCNC benchmarks, it achieves much better wirelength (23.5%) result than TimberWolf v7.0 on benchmark *avql*. Our results are also comparable with one of the leading-edge quadratic placement tools, Force Quadratic method [6] on most of benchmarks, except *avqs* on which our approach lacks compared to [6]. Given the very short amount of our development time comparing with long development time of other tools, the final results are impressive.

	TW 7.0[7]		Quadratic[6]		Ours	
circuit	WL	$time^2$	WL	$time^3$	WL	$time^4$
primary1	0.99	221	0.87	37	0.95	96
primary2	3.72	1252	3.72	152	3.66	554
biomed	1.88	2164	1.78	284	1.84	734
industry2	14.34	9252	14.6	1283	14.48	3525
industry3	44.67	8766	45.1	1605	44.70	4287
avqs	6.13	13018	4.91	1741	5.15	5094
avql	6.81	15597	5.38	2031	5.21	5632

Table 5: Wirelength and runtime comparison between three placement approaches

The running time of our approach listed in Table 5 is the global placement running time. Most of our contribution in this work focus on global placement so comparing global placement running time is more reasonable. The data structure of the final placer[8] we currently use is not very efficient. We are planning to implement a better version which is expected to give us two or three times speedup.

6. CONCLUSION

We have presented a snap-on placement model which is very easy(once it is fully understood when a certain tool to be used) to develop comparing with existing placement tools, We also proposed a detailed top-down placement work flow which combines net-cut minimization, simulated annealing and +1 level technique. This new method can achieve good placement solution for large circuits in a relatively short amount of time. Experimental results show that the snap-on placement tool is a promising way for solving large placement problems.

[4] runtime on Sun SPARC5 workstation.

[5] runtime on DEC workstation, which is 10 times faster than Sun SPARC5 according the runtime comparison in [6].

[6] runtime for global placement on Sun ULTRA10 workstation, which is 3 times faster than Sun SPARC5.

7. REFERENCES

[1] M.A. Breuer. "A Class of Min-cut Placement Algorithm for the Placement of Standard Cells". In *Design Automation Conference*, pages 284–290. IEEE/ACM 1977.

[2] U. Lauther. "A Min-cut Placement Algorithm for General Cell Assemblies Based on a Graph Representation". In *Design Automation Conference*, pages 1–10. IEEE/ACM 1979.

[3] C. Sechen and K.W. Lee. "An Improved Simulated Annealing Algorithm for Row-Based Placement". In *Design Automation Conference*, pages 180–183. IEEE/ACM, 1988.

[4] S. Mallela and L.K. Grover. "Clustering based Simulated Annealing for Standard Cell Placement". In *Design Automation Conference*, pages 312–317. IEEE/ACM, 1988.

[5] J. M. Kleinhans, G. Sigl, F. M. Johannes, and K. J. Antreich. "GORDIAN: VLSI Placement by Quadratic Programming and Slicing Optimization". *IEEE Transactions on Computer Aided Design*, 10(3):365–365, 1991.

[6] H. Eisenmann and F.M.Johannes. "Generic global Placement and Floorplanning". In *Design Automation Conference*, pages 269–274. IEEE/ACM, June 1998.

[7] W. Sun and C. Sechen. "Efficient and Effective Placement for Very Large Circuits". In *International Conference on Computer-Aided Design*, pages 170–177. IEEE, 1993.

[8] M. Sarrafzadeh and M. Wang. "NRG: Global and Detailed Placement". In *International Conference on Computer-Aided Design*. IEEE, November 1997.

[9] H. Shin, C. Kim, W. Kim and M. O. "A Combined Hierarchical Placement Algorithm". In *International Conference on Computer-Aided Design*, pages 164–169. IEEE, 1993.

[10] G. Karypis, R. Aggarwal, V. Kumar, and S. Shekhar. "Multilevel Hypergraph Partitioning: Application in VLSI Domain". In *Design Automation Conference*, pages 526–529. IEEE/ACM, 1997.

[11] G. Karypis and V. Kumar. "Multilevel k-way Hypergraph Partitioning". In *Design Automation Conference*, pages 343–348, 1999.

[12] B. Landman and R. Russo. "On a pin versus block relationship for partitions of logic graphs". *IEEE Transactions on Computers*, c-20:1469–1479, 1971.

[13] M. Wang and M. Sarrafzadeh "On Wirelength Prediction Using the Net-cut Objective". Submitted to *IEEE Transactions on Very Large Scale Integration (VLSI) Systemss*, 1999