SEQUENCE-PAIR BASED PLACEMENT METHOD FOR HARD/SOFT/PRE-PLACED MODULES

Hiroshi Murata *Ernest S. Kuh*

Department of Electrical Engineering and Computer Science
University of California, Berkeley, USA
murata@EECS.Berkeley.EDU

ABSTRACT

This paper proposes a placement method for a mixed set of hard, soft, and pre-placed modules, based on a placement topology representation called sequence-pair. Under one sequence-pair, a convex optimization problem is efficiently formulated and solved to optimize the aspect ratios of the soft modules. The method is used in two ways: i) directly applied in simulated annealing to present the most exact placement method, ii) applied as a post process in an approximate placement method for faster computation. The performance of these two methods are reported using MCNC benchmark examples.

1. INTRODUCTION

As the circuit size increases, the placement technology is becoming critical especially in the top level of the physical hierarchy in the top-down phase of the design. In such design stage, the modules to be placed have several geometrical diversities in nature. The size varies as well as other geometrical properties: Some modules are not yet designed thus being flexible in shape (soft modules), some are completely designed beforehand (hard modules), even some are completely pre-placed on the target chip (pre-placed modules). This paper studies a placement method to cope with those geometrical diversities.

Once the soft modules are in consideration, previous researches [1, 2] usually limit their scopes themselves to optimize the layout under one specific placement topology given by means of a rectangular dissection. However, known methods to obtain such an input rectangular dissection are limited to slicing structure [3], or conventional rectangular-dual approaches [4] which can not take the geometrical diversities into account.

To handle the size diversity, two methods are presented recently, bounded-sliceline-grid (BSG) [5] and sequence-pair [6, 7]. They provide ways to optimize placement topology beyond the limitation of the slicing structure. The BSG method is extended to handle soft modules [8], but their optimization is not global. The sequence-pair method is extended to handle pre-placed modules without losing the reachability to an area minimum placement [9].

This paper follows the sequence-pair approach to further include soft modules in consideration. The key idea is in an efficient convex formulation for the aspect ratios of the soft modules under one sequence-pair, with respect to two kinds of constraints: Upper and lower limits of the aspect ratios, and the existence of the pre-placed modules. Then, the problem is solved in polynomial time by an inner point method proposed by Vaidya [11]. This method is directly used in simulated annealing to provide the most exact placement method among those studied. An approximate method is also developed for faster computation through experimental comparison with the exact method on MCNC benchmark examples.

The organization of the paper is as follows. Section 2. formally defines the problem. Section 3. formulates and solves the convex problem to optimize the aspect ratios of the soft modules under one sequence-pair. In Section 4., the method is used in simulated annealing to optimize the sequence-pair. An approximate method is also presented in Section 4. with experimental results. Section 5. is for conclusion.

2. PROBLEM DEFINITION

A *module* is a rectangle. The following notation is used to describe a module on the plane.

- w_u, h_u – the width and the height of module u
- $a_u \equiv h_u w_u$ – the area of module u
- $r_u \equiv h_u / w_u$ – the aspect ratio of module u, and
- x_u, y_u – the coordinates of the lower left corner of module u, simply called the coordinates of module u.

We consider three kinds of modules depending on the design freedom on the above parameters. A *hard module* is a module whose width and height are specified but its coordinates are free to design. A *soft module* is a module whose area is specified but the width and height are free to design as far as its aspect ratio is in a given range $[r_{u,\min}, r_{u,\max}]$. The coordinates of soft modules are free to design. A *pre-placed module* is a module whose width, height, and its coordinates are all specified. (no freedom).

Let H be a set of hard modules, F be a set of soft modules, and P be a set of pre-placed modules whose coordinates are specified such that no two pre-placed modules overlap each other and all of them lie in the first quadrant of the plane. Let M denote the union of these three sets of modules. A *packing* of a set of modules is a non-overlapping placement of the modules. A feasible packing of M is a packing of M on the first quadrant of the plane such that all the pre-placed modules are placed at their specified positions, and the width and the height of soft modules are consistent to their area specifications and aspect ratio constraints. We measure the goodness of a feasible packing by the area of the minimum rectangle among rectangles which enclose all the modules other than pre-placed modules and whose lower left corner is at the origin of the plane.

At this moment, let us temporary limit our interest to minimize the area under one placement topology. The placement topology is assumed to be specified by means of "sequence-pair", which is introduced in [6] as follows.

A *sequence-pair* of module set M is a pair of sequences of the names of the modules in M. For example, $S = (uvw, wuv)$ is a sequence-pair of $M = \{u, v, w\}$. A sequence-pair S of M specifies a placement topology through the following rules, called *h/v constraints*.

For every pair of modules u, w in M,

- $S = (\cdots u \cdots w \cdots, \cdots u \cdots w \cdots) \Rightarrow x_u + w_u \leq x_w$, (horizontal constraint)
- $S = (\cdots w \cdots u \cdots, \cdots u \cdots w \cdots) \Rightarrow y_u + h_u \leq y_w$. (vertical constraint)

A sequence-pair is said to be feasible if there is a feasible packing which satisfies the h/v constraint, or said to be infeasible otherwise. A minimum area feasible packing under the h/v constraint of a feasible sequence-pair is said to be an optimal packing under the sequence-pair. Now, we are ready to define our first problem.

> **Primary-Problem:** Given a set H of hard modules, a set F of soft modules, a set P of pre-placed modules, and a sequence-pair S for module set $M = H \cup F \cup P$, determine the feasibility of S, and if S is feasible, find an optimal packing of M under S.

The case of $F = P = \emptyset$ is studied in [6, 7]. They showed that every sequence-pair is feasible and there is a sequence-pair which leads optimal packing. They extended their research to handle the case of $P \neq \emptyset$ in [9], with the following difference comparing to this paper: In [9], the goodness of the feasible packing is defined using the smallest bounding rectangle which encloses all the modules in M, where the pre-placed modules are excluded in our formulation. It is easily understood that this modification removes unnecessary lower bound in the target function. In the next section, we will see that this modification does not require any essential change in their algorithm.

The case of $F \neq \emptyset, P = \emptyset$ is studied in many researches [1, 2] based on a conventional framework of placement topology representation, rectangular dissection. (See [6] and [10] for some known relationships between sequence-pair and rectangular dissection.)

Primary-Problem unifies the problems in those literature. However, it is still a sub problem of a practical placement problem which is modeled in this paper as follows.

> **Placement-Problem:** Given hard/soft/pre-placed modules with nets, optimize sequence-pair for the modules to minimize a linear combination of the packing area and the estimated total wiring length.

Previous algorithm which can handle soft modules [1, 2] limit their scopes themselves to the case of a placement topology being given, and do not present a method to optimize the placement topology, hence Placement-Problem has been left as a manual task. Dissimilarly, we also tackle Placement-Problem in this paper.

3. EXACT ALGORITHM UNDER ONE SEQUENCE-PAIR

3.1. Solving Primary Problem Without Soft Modules

We first consider the following sub problem, where the aspect ratio of each soft module is given in the specified range.

> **Sub-Problem:** Given H, F, P and sequence-pair S of $M = H \cup F \cup P$, and aspect ratios of soft modules in F, determine the coordinates (x_u, y_u) for every module u in M such that the area of the packing is minimized and the following constraints are kept.
> - h/v constraints of S, and
> - $x_p \geq x_p^*$ and $y_p \geq y_p^*$ for p in P
>
> where (x_p^*, y_p^*) is the specified coordinates of pre-placed module p.

Notice that the positions of the pre-placed modules are only constrained from one side, thus the problem always has a solution.

Lemma 1 *The sub-problem can be solved in $O(|M|^2)$ time.*

(Proof) The width and the height of a soft module f are trivially calculated by the following equations.

$$w_f = \sqrt{a_f / r_f}$$
$$h_f = \sqrt{a_f \times r_f}$$

Therefore, the problem is essentially the same as the problem considered in [9] except for the difference in target functions: their target is minimizing the area of the bounding rectangle of all the modules, while our target is to minimize the area of the bounding rectangle of non-pre-placed modules. Despite this difference, their algorithm "propped realization" can solve our problem in $O(|M|^2)$ time, since it independently minimizes X and Y coordinates for every module [9]. □

The procedure "propped realization" [9] is briefly described in the following for the completeness of the discussion.

For each module u, x_u is set to the minimum value to keep the horizontal constraints of S and the horizontal inequality constraints for pre-placed modules. The Y coordinate is similarly calculated using the vertical constraints. This computation can be done in $O(|M|^2)$ time for all the modules by a longest path length calculation on two directed acyclic graphs $G_h(V_h, E_h)$ and $G_v(V_v, E_v)$, called horizontal constraint graph and vertical constraint graph, respectively.

- V_h (V_v): node set consists of a source, a sink, and nodes corresponding to the modules (simply called the modules in the following),
- E_h (E_v): edge set consists of (i) the edges from the source to the modules, (ii) the edges from the modules to the sink, (iii) the edges corresponding to the horizontal (vertical) constraints of S, and (iv) the edges from the source to the pre-placed modules.
- vertex weight for V_h (V_v): zero for source and sink, width (height) of corresponding modules for the other nodes.
- edge weight for E_h (E_v): specified X (Y) coordinate of the corresponding pre-placed modules for the edges of type (iv), zeros for the other edges.

3.2. Convex Formulation of Primary-Problem

Now, we are left with optimizing the aspect ratios of the soft modules. This problem has been extensively studied based on rectangular dissection [2, 1]. Among them, Moh et. al [2] first formulate a convex problem for one rectangular dissection, then solve it by a numerical optimization algorithm. This is the most reasonable approach in the literature, thus followed here. However, in their convex problem formulation, many indirect variables and many constraints on those indirect variables are introduced, and probably because of that, it is reported that the size of the tractable problem is

limited due to the memory requirement. Therefore, we pay more attention to reduce the number of variables and constraint functions. The formulation must be different from the first because we are not using rectangular dissection.

Let us denote the aspect ratios of the soft modules by a vector variable $\vec{r}$, which is the only variable in our formulation. Let us denote the width and the height of the packing obtained by "propped realization" by $W(\vec{r})$ and $H(\vec{r})$, respectively. In the packing, the X (Y) coordinate of a pre-placed module is never smaller than specification, but can be larger. Hence, we define "error function" $E(\vec{r})$ as follows.

$$E(\vec{r}) = \max_{p \in P}(x_p - x_p^*) + \max_{p \in P}(y_p - y_p^*)$$

It is easily understood that $E(\vec{r}) \leq 0$ is a necessary condition for the solution of Sub-Problem being a feasible solution of Primary-Problem. Using the notation $W(\vec{r}), H(\vec{r})$ and $E(\vec{r})$, Primary-Problem is re-written to an aspect ratio optimization problem as follows.

Soft-Module-Problem: (Aspect ratio optimization)

$$\begin{array}{ll} \text{Minimize} & W(\vec{r})H(\vec{r}) \\ \text{subject to:} & E(\vec{r}) \leq 0, \\ & r_{f,\min} \leq r_f \leq r_{f,\max} \text{ for all } f \in F. \end{array}$$

Lemma 2 *Soft-Module-Problem can be translated into a convex optimization problem with convex constraints, by an exponential variable transformation* $r_u \equiv \exp(z_u)$.

(Proof) Since $W(\vec{r})$ is the longest path length in the horizontal constraint graph, it can be written as a function of $\vec{r}$ as follows.

$$\begin{aligned} W(\vec{r}) &= \max_{M_w \in \{M_w\}} \sum_{u \in M_w} \{w_u \text{ or } x_u^*\} \\ &= \max_{M_w \in \{M_w\}} \left[C + \sum_{f \in (M_w \cap F)} C w_f \right] \end{aligned}$$

where, $\{M_w\}$ denotes the set of all the paths from source to sink in G_h, each C represents an individual non-negative constant. Using the exponential variable transformation, $W(\vec{r})$ can be re-written as a function of $\vec{z}$, as follows.

$$W(\vec{z}) = \max_{M_w \in \{M_w\}} \left[C + \sum_{f \in (M_w \cap F)} C \exp(-\frac{z_f}{2}) \right]$$

The term $\exp(-z_f/2)$ is a convex function of z_f. Since sum of convex functions is convex and max of convex functions is also convex, $W(\vec{z})$ is a convex function.

Similarly, $H(\vec{r})$ is translated to $H(\vec{z})$, which is also convex as follows.

$$H(\vec{z}) = \max_{M_h \in \{M_h\}} \left[C + \sum_{g \in (M_h \cap F)} C \exp(\frac{z_g}{2}) \right]$$

where, $\{M_h\}$ is the set of all the paths from source to sink in G_v.

Our objective function is $W(\vec{z})H(\vec{z})$. Although multiplication of two convex functions is not necessarily convex in general, our function is convex as follows.

$$W(\vec{z})H(\vec{z}) = \max_{\substack{M_w \in \{M_w\}, \\ M_h \in \{M_h\}}} \left[C + \sum_{f \in (M_w \cap F)} C \exp(-\frac{z_f}{2}) + \sum_{\substack{f \in (M_w \cap F), \\ g \in (M_h \cap F)}} C \exp(\frac{-z_f + z_g}{2}) + \sum_{g \in (M_h \cap F)} C \exp(\frac{z_g}{2}) \right]$$

The error function $E(\vec{r})$ is also translated to a convex function because it is also defined as sum of path lengths on the constraint graphs. The aspect ratio constraints are trivially translated to a convex constraint.

Hence the claim holds. □

The resultant convex problem is written in the following.

Convex-Problem: (Aspect ratio optimization)

$$\begin{array}{ll} \text{Minimize} & W(\vec{z})H(\vec{z}) \\ \text{subject to} & E(\vec{z}) \leq 0, \\ & z_{f,\min} \leq z_f \leq z_{f,\max} \text{ for all } f \in F \end{array}$$

3.3. Solving Convex Problem

Notice that the variables in Convex-Problem are the aspect ratios of the soft modules only, thus the aspect ratio constraints describe the variable range. An efficient numerical optimization algorithm is proposed for such cases by Vaidya [11]. Vaidya's algorithm utilizes variable range constraint to bound the search space, instead of treating it as a quantity to be controlled. Therefore, the algorithm can handle Convex-Problem virtually as unconstrained problem when $P = \emptyset$. Vaidya's algorithm is also able to take additional convex constraints, which is the pre-placed module constraint ($E(\vec{z}) \leq 0$) in our case. From these reasons, Vaidya's algorithm is selected here to solve our problem.

An implementation of Vaidya's algorithm is described in detail in [12], which deals with a transistor sizing problem. Our implementation follows [12]. However, the outline of the algorithm is described in the following for the completeness of the discussion.

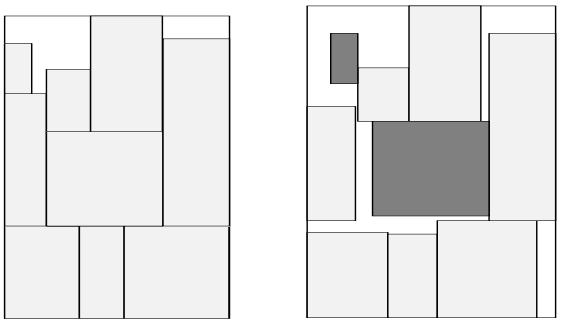
First, a box is constructed in the $|F|$-dimensional space of $\vec{z}$ using the the aspect ratio constraints. Notice that the box is a convex polytope, and the solution, if exists, is guaranteed to be inside the polytope.

In each iteration, an inner point $\vec{z}'$, called the "volume center", of the current polytope is located by maximizing the sum of log-barrier functions using a variation of Newton method, and the point is tested whether it satisfies the pre-placed module constraint ($E(\vec{z}') \leq 0$).

If the pre-placed module constraint is satisfied, a longest path M_w in G_h and a longest path M_h in G_v are identified. The gradient of the objective function $W(\vec{z})H(\vec{z})$ at $\vec{z} = \vec{z}'$ is calculated along these paths. Using the gradient, a hyper plane is constructed so that it passes $\vec{z}'$ and it is tangent to to the gradient vector. Since $\vec{z}'$ is an inner point, the hyper plane divides the current polytope into two parts, one corresponding to the increasing direction of the objective function, and the other corresponding to the decreasing direction. The increasing part is cut off in the current polytope and the remaining part replaces the current polytope.

If the volume center $\vec{z}'$ does not satisfy the pre-placed module constraint, a path in G_v and a path in G_h which together determine $E(\vec{z}')$ are identified, and the gradient of $E(\vec{z})$ at $\vec{z} = \vec{z}'$ is calculated along these paths. Then, the separating hyper plane is constructed and the part corresponding to increase of $E(\vec{z})$ is cut off.

The algorithm repeats cutting off a part of the polytope until it shrinks sufficiently small. After a small polytope is obtained, a point in the resultant small polytope is located, and its feasibility is tested. If feasible, the point is output as the solution. Otherwise, the problem is reported not having a solution.



(a) (b)

Figure 1. Output examples of Primary-Problem. Dark rectangles are pre-placed modules.

Note that only one longest path in a graph is taken into account for calculating gradient of $W(\vec{z})H(\vec{z})$ or $E(\vec{z})$. This is equivalent to take only one maximum term into account in each max function in $W(\vec{z})$, $H(\vec{z})$, or in $E(\vec{z})$. Despite this simple treatment, the solution is guaranteed to be in the remaining side of the hyper plane because of the convexity of the function. This fact helps us from enumerating all the (possibly exponential number of) longest paths.

Convergence is guaranteed by showing the lower bound of the fraction of the cut off part, and the time complexity is polynomially bounded depending on the number of variables, the wideness of the variable range, and the accuracy used in the stopping criterion [11]. Hence the following theorem holds.

Theorem 1 *Primary-Problem can be solved in polynomial time depending on the number of modules, the wideness of the range of the aspect ratio constraint, and the accuracy used in the stopping criterion.*

The above presented approach to Primary-Problem is said to be "exact" in the following, ignoring the inherent error within the accuracy used in the stopping criterion.

Figure 1(a) shows the output of the proposed algorithm for "Example2" in [2]. (X-axis are expanded 2.5 times larger than Y-axis when drawn.) Since this example does not include pre-placed modules, we made such an example by specifying two modules in Figure 1(a) as pre-placed modules, after slightly moving them to upper right direction. The output for this new problem instance is shown in Figure 1(b).

It should be noted that the above discussion does not consider the chip aspect ratio. The chip aspect ratio can be controlled by changing the the target function to the area of the smallest bounding rectangle which has required aspect ratio. All the above discussions remain essentially same for this modification.

3.4. Speed of the Algorithm

The above presented method might be expensive in computation time because of the intensive numerical calculation even though polynomially bounded. Therefore, our first experiment is to know the practical speed of the calculation for the various size of the problem instances.

The problem size in terms of the number of modules was varied from 10 to 500. All modules are soft modules with aspect ratio being constrained in the range of $0.1 \sim 10.0$. The reason why there is no pre-placed module or hard module is that the speed of the algorithm is apparently dominated by the number of soft modules. The areas of the modules are randomly determined in the range of $100^2 \sim 10000^2$. The input sequence-pair is made by simulated annealing described in [6, 7], assuming all the modules are square hard modules.

Table 1. The speed of the soft module optimization

#mod	10	20	50	100	200	500
Area(in)	1.26	1.11	1.09	1.07	1.07	1.09
Area(out)	1.01	1.00	1.01	1.01	1.01	1.02
Time(s)	0.396	4.93	60.5	937	7140	73834

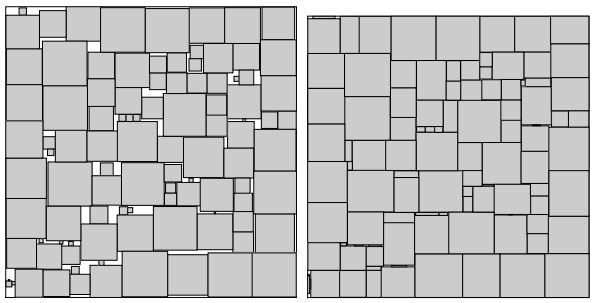


Figure 2. Input (left) and output (right) of a soft module optimization for 100 modules

The reason why such an "optimized" sequence-pair is used for the input is that the method might be used as a post process.

Figure 2 shows the input and output of 100 module case. The chip aspect ratio is constrained to 1 for presentation purpose, for all the experiments. (the reported chip area is the area of the smallest square which encloses the placement.) Table 1 shows the CPU time, the area of the input and the area of the output. Area in the table is normalized using the sum of the areas of all the modules. All the experiments were carried out on a 250 MHz DEC Alpha DEC 21164 CPU, 4Mb cache and 1Gb total memory. The CPU time in the table are approximately on curve $0.0004 \times |F|^{3.0}$.

Moh et. al [2] also report the speed of their method on random data whose size varying in range $|F| = 20 \sim 106$ on a HP9000-735 workstation. In their experiments, they cannot handle the bigger problems because of the memory limitation, where we had no problem in 500 module problem. This is probably because the number of variables and constraints is reduced. However, their speed is faster in many cases than Table 1 by the factor varying from 0.04 to 2.0(smaller number means faster). The reason of this difference is probably because our implementation of Vaidya's algorithm is less optimized comparing to their solver, a package software called MINOS. Especially, it is observed in our experiments that the linear augmentation of hyper planes in the outer loop slows the inner-loop process as the polytope shrinks, since all the hyper planes are kept in our implementation even if they eventually become redundant for the shrunken polytope. The redundant hyper planes may be removed in more advanced implementation as it is suggested in [11].

4. EXACT METHOD AND APPROXIMATE METHOD FOR PLACEMENT-PROBLEM

In this section, two methods for Placement-Problem are developed through experiments using MCNC benchmark examples.

4.1. Exact Placement Method

Previous section shows that Primary-Problem can be solved in polynomial time by Vaidya's algorithm, hence we propose a method to tackle Placement-Problem by iteratively using the algorithm in a simulated annealing. We note that this approach is not literally exact because Primary-Problem is

Table 2. Performance of the "exact" placement method for MCNC examples

data	#mod	#net	area (μm^2)	wlen (μm)	time (sec.)
apte	9	97	46553329	344358	789
xerox	10	203	19509889	401254	1198
hp	11	83	8826841	118819	1346
ami33	33	123	1159929	53393	75684
ami49	49	408	35581225	775104	612103

solved solely for area. For instance, if the output of Primary-Problem includes a small unused area which allows a module to be slightly shifted, then there is a chance to further minimize the total wire length by utilizing that freedom. However, this method is the extreme case of pursuing the exactness of the solution, among those studied, thus called "exact method" here.

In some iteration, the input sequence-pair might be found infeasible since Primary-Problem may have no solution when there is a pre-placed module. To cope with this difficulty, we extend "adaptation" procedure in [9] which changes a sequence-pair so that it becomes consistent to the pre-placed modules. In our case, the adaptation procedure must consider soft modules also, since the feasibility of a sequence-pair relies on the flexibility of the soft modules. Therefore, in our adaptation, we first perform the numerical algorithm to determine the feasibility of the sequence-pair. If it is found infeasible, then we legalize the sequence-pair with respect to the pre-placed modules by the algorithm in [9].

The experiments are carried out for the MCNC building block examples. All the modules are originally defined as hard modules, but they are interpreted as soft modules with aspect ratio constraint $0.1 \sim 10$. The evaluating function is a weighted sum of the area of the packing and the total wire length based on the half perimeter estimate for each net, where the terminals are assumed to be at the center of the modules. The weight is decided so that the area term and the wire length term are approximately balanced. The initial temperature is decided such that an accept ratio is between 95 % and 100 %. The temperature is exponentially lowered in 4 decades by 20 steps. The number of iterations for one temperature step is set to ten times the number of modules. The initial sequence-pair is created at random, and changed by the move operations called "half-exchange" and "full-exchange" [6, 7].

The result is shown in Table 2. Area, wire length, and cpu-time are listed. Figure 3 shows the layout for each example. There is almost no unused area in the results, which implies that the additional freedom for wire length minimization is most likely small, and the quality of the result must be almost exact. However, the CPU time is formidable for the largest two problems, ami33 and ami49. Therefore, we propose an approximate algorithm next.

4.2. Approximate Placement Method

To design an approximate algorithm for Placement-Problem, experiments are carried out to test four candidate methods, namely, "square", "discrete", "sq+post", and "dis+post". All of them are based on simulated annealing, similar to "exact", with the following differences.

"square" All modules are first reshaped into square hard modules and never reshaped.

"discrete" Starting with square modules, aspect ratio of an arbitrary module is increased or decreased by 10 percent by a newly introduced move operation.

"sq+post" "square" followed by the numerical optimization as a post process.

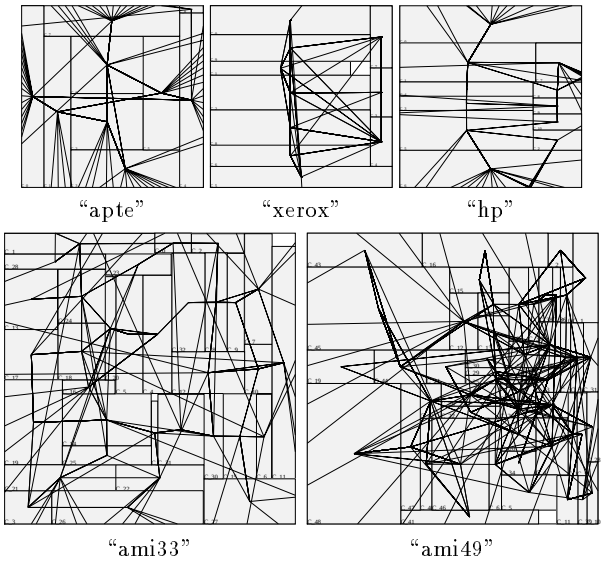


Figure 3. Output of "exact" placement method for MCNC examples

"dis+post" "discrete" followed by the numerical optimization as a post process.

Table 3 shows the result of these experiments for MCNC benchmark examples. Figure 4 shows the output of these methods for one problem instance, "hp". Both of "sq+post" and "dis+post" show acceptable CPU time and good approximation in terms of area comparing to "exact". However, both of the methods give poor approximation in terms of wire length. For example, the wire length of "hp" obtained by "sq+post" and "dis+post" are 56, 49 % worse than that of "exact", respectively. This tendency is worse for larger examples. The reason can be observed by comparing the columns "square" and "discrete" as: the freedom of changing the aspect ratios in "discrete" was mainly used for area minimization and wire length was even degraded sometimes.

From the above observation, a better approximation method would be obtained based on "dis+post" by replacing the weight in the evaluating function with heavier value for wire term.

"dis2+post" "discrete" with heavier weight for wires than for area, followed by the numerical optimization as a post process.

Some sacrifice on area may happen in the "discrete" phase, but the area will be sufficiently reduced in the post process. The performance is listed in Table 4. The weight is set so that it is approximately 100 times heavier for wire length than for area. It is observed from Table 4 that the intent is realized especially significant for larger examples. From the above study, "dis2+post" is concluded here as a good approximate method for faster computation.

5. CONCLUSION

A convex optimization problem is efficiently formulated under one sequence-pair to optimize the aspect ratios of the soft modules in a mixed set of hard/soft/pre-placed modules, and a numerical optimization algorithm is carried out to solve the problem exactly in polynomial time. Then, the method is iterated using simulated annealing to optimize sequence-pair to achieve the best placement. Along with

Table 3. Performance of the basic approximate methods

data	square	discrete	sq+post	dis+post
apte	52490025	52215076	46840336	46717225
(9,97)	413354	415090	408635	400514
	0.663	0.673	1.137	1.342
xerox	24661156	20839225	19465744	1936000
(10,203)	551971	551518	545978	551248
	1.408	1.423	2.671	2.658
hp	11840481	9610000	8928144	8832784
(11,83)	191483	185759	185481	177592
	0.867	0.851	1.497	2.148
ami33	1404225	1256641	1210000	1168561
(33,123)	59243	64593	56114	63042
	4.002	4.089	7.983	25.663
ami49	48790225	41293476	35916049	35844169
(49,408)	949571	1118411	905818	1130677
	11.045	11.712	100.826	98.596
(#mod, #net)	area(μm^2) wlen(μm) time(sec.)			

Table 4. Performance of the approximate method "dis2+post" for MCNC examples

data	#mod	#net	area (μm^2)	wlen (μm)	time (sec.)
apte	9	97	46635241	421174	1.316
xerox	10	203	19474569	533990	2.062
hp	11	83	8868484	157166	2.051
ami33	33	123	1162084	51346	28.088
ami49	49	408	36048016	850305	50.399

this exact but computationally expensive method, an approximate method is also developed for faster computation through experimental comparison on MCNC benchmark examples.

Wires are considered in the total wire length measure only. Further research is necessary for ensuring routability and satisfying timing constraint.

ACKNOWLEDGMENT

The authors would like to thank Mr. Atsushi Sakurai, Applied System LSI Division, Fujitsu, for the helpful discussion. This work has been supported in part by Fujitsu and the Semiconductor Research Corporation (SRC) 324-018.

REFERENCES

[1] K. Wang and W.-K. Chen. Floorplan area optimization using network analogous approach. In *Proc. IEEE International Symposium on Circuits and Systems*, pages 167–170, 1995.

[2] T.-S. Moh, T.-S. Chang, and S. L. Hakimi. Globally optimal floorplanning for a layout problem. *IEEE Trans. on Circuit and Systems – I: Fundamental Theory and Applications*, 43(9):713–720, Sep 1996.

[3] D. F. Wong and C. L. Liu. A new algorithm for floorplan designs. In *Proc. 23rd ACM/IEEE Design Automation Conference*, 1986.

[4] Y. Lai and S. Leiwand. Algorithms for floorplan design via rectangular dualization. *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, 7(12):1278–1289, Dec 1988.

[5] S. Nakatake, K. Fujiyoshi, H. Murata, and Y. Kajitani. Module placement on BSG-structure and IC layout applications. In *Proc. IEEE Intl. Conf. on Computer Aided Design*, pages 484–491, 1996.

[6] H. Murata, K. Fujiyoshi, S. Nakatake, and Y. Kajitani. Rectangle-packing-based module placement. In *Proc. IEEE Intl. Conf. on Computer Aided Design*, pages 472–479, 1995.

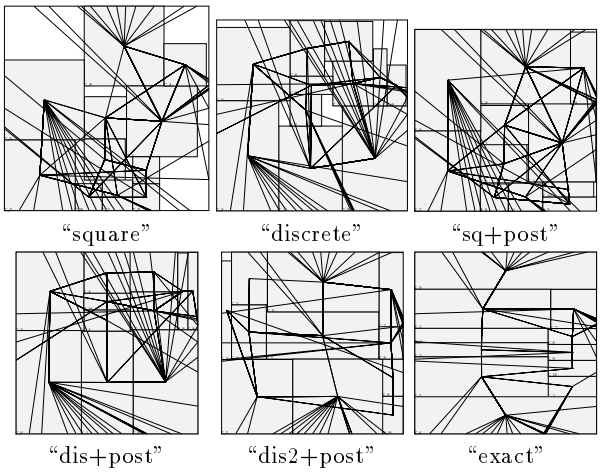


Figure 4. Output examples of the six methods for problem "hp"

[7] H. Murata, K. Fujiyoshi, S. Nakatake, and Y. Kajitani. VLSI module placement based on rectangle-packing by the sequence-pair. *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, 15(12):1518–1524, Dec 1996.

[8] M. Kang and W. Dai. General floorplanning with L-shaped, T-shaped and soft blocks based on bounded slicing grid structure. In *Proc. Asia and South Pacific Design Automation Conf. 1997*, pages 265–270, 1997.

[9] H. Murata, K. Fujiyoshi, and M. Kaneko. VLSI/PCB placement with obstacles based on sequence-pair. In *Intl. Symp. on Physical Design*, pages 26–31, 1997.

[10] H. Murata, K. Fujiyoshi, T. Watanabe, and Y. Kajitani. A mapping from sequence-pair to rectangular dissection. In *Proc. Asia and South Pacific Design Automation Conf. 1997*, pages 625–633, 1997.

[11] P. M. Vaidya. A new algorithm for minimizing convex functions over convex sets. *Mathematical Programming*, 73:291–341, 1996.

[12] S. S. Sapatnekar, V. B. Rao, P. M. Vaidya, and S. M. Kang. An exact solution to the transistor sizing problem for CMOS circuits using convex optimization. *IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems*, 12(11):1621–1634, Nov 1993.