# Flipping Modules To Improve Circuit Performance And Routability

*Keumog Ahn*
University of Minnesota

and

*Sartaj Sahni**
University of Florida

**University of Florida TR 92-23**

## Abstract

Preplaced modules may be flipped to improve performance and/or routability. We show that several versions of the module flipping problem are NP-hard. We propose algorithms to obtain module orientations that minimize the total wire length for module matrix and standard cell models. Additionally, we propose a heuristic to flip modules so as to minimize the length of the longest signal path. The results of experiments performed with this heuristic are also presented.

## Keywords and Phrases

Module flipping, circuit performance, routability, heuristics

* Research supported, in part, by the National Science Foundation under grant MIP-9103379 and in part by an award from AT&T Bell Laboratories.

## 1 Introduction

The performance and routability of a circuit are affected by flipping and/or rotating the circuit modules while keeping the placement fixed. One may formulate several performance and/or routability measures that are based on estimates of wire length. In these estimates, the length of a wire is estimated either by the Euclidean distance between the wire end points or by the Manhattan distance. The developments of this chapter use the Manhattan Metric. Let ETL denote the estimated total wire length and let MEL denote the maximum estimated wire length (i.e., length of the longest wire).

The problems of minimizing ETL and MEL by flipping modules have been studied by Yamada and Lin [YAMA88], Libeskind-Hadas and Liu [LIBE89], and Chong and Sahni [CHON91 and 92]. All of these start with a preplaced circuit. It is assumed that modules may be flipped about their horizontal and/or vertical axes. Hence, there are four possible orientations of a module under flipping (Figure 1). The initial or reference orientation is denoted by $\phi$, *h*, *v*, and *b*, respectively, denote the orientation that results from flipping about the horizontal axis, vertical axis, and both axes. For rotation, rotations of 0, 90, 180, and 270 degrees are permitted and it is assumed that these rotations do not result in module overlaps.

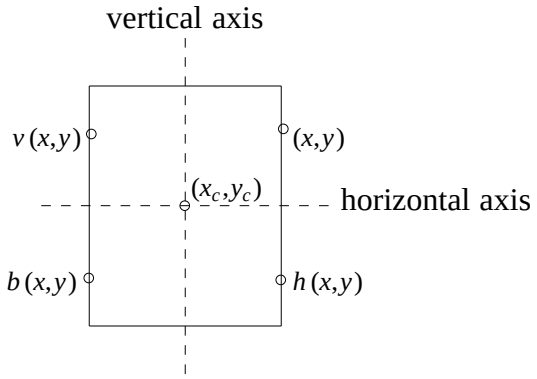



**Figure 1** $\phi$, *h*, *v*, *b*

Let ETLF (ETLR) denote the problem of flipping (rotating) modules so as to minimize the estimated total wire length. In both [YAMA88] and [LIBE89], the Euclidean distance between wire end points is used to estimate wire length. Both ETLF and ETLR are shown to be NP-hard in [LIBE89]. The proofs of [LIBE89] are easily modified for the case when wire length is estimated using the Manhattan distance between the wire end points. Further, the proof for ETLF holds even if flips are restricted to be made along only the vertical (horizontal) axis. Yamada and Liu [YAMA88] propose an analytical method to obtain suboptimal solutions for ETLF. This algorithm is shown (experimentally) to be competitive with hill-climbing and simulated annealing algorithms for ETLF. In [LIBE89], Libeskind-Hadas and Liu propose neural network formulations for ETLF and ETLR.

Chong and Sahni [CHON92] show that ETLF is linearly solvable for the case when the modules are arranged as a matrix in which wires connect only pairs of modules that are in adjacent columns. They also show that ETLF is polynomially solvable for standard cell designs in which wires connect modules in adjacent columns and either the number of module columns is two or the number of modules dependent on any other module is bounded by some constant. Chong and Sahni [CHON92] also evaluate a simple greedy heuristic that attempts to minimize ETL by flipping modules. Experimental results reported by them indicate this heuristic is superior to the neural network approach of [LIBE89].

The problem of minimizing the maximum estimated wire length was studied by Chong and Sahni [CHON91]. They considered two versions of the problem MEL4 and MEL2. In MEL4 each module is permitted four orientations as in Figure 1. In MEL2, only two these four orientations are permitted. Chong and Sahni showed that MEL4 is NP-hard even for a single row or column of modules. They developed polynomial time algorithms for restricted versions of MEL4 and obtained an $O(n\log n)$ algorithm for MEL2.

In [AHN92], we consider the problem of rotating modules to improve circuit performance and routability. Let *MaxDelay* denote the problem of reorienting modules so that the length of the longest signal flow path is minimized. In [AHN92], we showed that the *MaxDelay* problem is NP-hard for single column and single row instances with equal size modules when only rotations are permitted. In this paper, we also showed that the ETLR problem is NP-hard for a single column or row of equal size modules even when only 0° and 90° rotations are permitted. Note

that the ETLR proof of [LIBE89] comes close to having three rows of modules. This proof uses modules of different size. Our proof applies to the simpler case of equal size modules in a single row or single column.

In this paper, we restrict ourselves to module rotation problems. In Section 2, we obtain the following NP-hard results:

1. MaxDelay is NP-hard when only horizontal flips, only vertical flips, and both horizontal and vertical flips, are permitted. This is so even for single column or single row instances with equal size modules.

2. The ETLF problem is NP-hard for a single column of equal size modules even when only vertical flips are permitted.

The ETLF proof of [LIBE89] uses modules of two different sizes. While most of these are placed in a single row arrangement, two modules are stacked into a column. So, their construction does not apply to the case of single row instances with equal size modules and flips restricted to horizontal ones (this would be symmetric to single column, equal size modules, vertical flips only). So, our ETLF proof applies to even simpler module layouts than does the proof of [LIBE89].

In Section 2, we also show that single column ETLF can be solved in linear time when only horizontal flips are permitted. In Section 3, we consider the ETLF problem for layouts that follow a module matrix model as well as for standard cell layouts. Algorithms to obtain optimal solutions are proposed here. In Section 4, we propose a heuristic for the MaxDelay problem and evaluate it experimentally.

## 2 NP-hard Results

To prove our NP-hard results, we use the following problems that are known to be NP-hard.

*3SAT problem* [GARE79]

*Input:* A Boolean function $I = C_1, C_2, ..., C_m$ in $n$ variables $x_1, x_2, ..., x_n$. Each clause $C_i$ is the disjunction of exactly three literals.

*Output:* "*Yes*" iff there is a truth assignment for the $n$ variables such that $I$ = true. "*No*" otherwise.

*NE3SAT problem* [GARE79]

*Input:* Same as for 3SAT.

*Output:* "*Yes*" iff there is a truth assignment for the $n$ variables such that in each clause $C_i$ at least one literal evaluates to true and at least one to false (i.e., all three literals do not have the same truth value). "*No*" otherwise.

**Theorem 1:** MaxDelay is NP-hard for a single column of equal size modules even when the only flips permitted are vertical flips.

**Proof:** We shall prove this using the NE3SAT problem. Let $I$ be any instance of this problem. Let $n$ and $m$, respectively, be the number of variables and clauses in $I$. Construct an $n$ module single column instance of MaxDelay as follows. Each module has unit delay, unit height, and width equal to $7(m+1)$. Each module is logically divided into three parts left, middle, and right (Figure 2(a)). These parts have width $m+1$, $5m+5$, and $m+1$, respectively. The left and right parts each have $m$ pin positions with unit seperation on their top and bottom boundaries. The $n$ modules are stacked one on top of the other as in Figure 2(b). The top of one module is at a unit distance from the bottom of the module above it. The $i$´th module from the top represents variable $x_i$ of $I$.

Each clause of $I$ will be represented by an independent signal path that begins at the top boundary of Figure 2(b) and ends at the bottom boundary. We place $m$ pins in the center of the middle part of the top and bottom boundaries. We may view the pins on the top boundary as the circuit input pins and those on the bottom boundary as output pins. The signal path for clause $C_j$ of $I$ begins at the $j$´th pin (from the left) of the top boundary, goes through the three modules that represent the three variables in $C_j$ and ends at the $j$´th pin of the bottom boundary. When the path goes through a module it enters from the module top using the $j$´th pin from the left or right end of the top boundary and exits from the corresponding pin on the bottom side. Suppose that $C_j$ includes variables $x_a$, $x_b$, and $x_c$ and that $a < b < c$. The signal path that we construct depends on

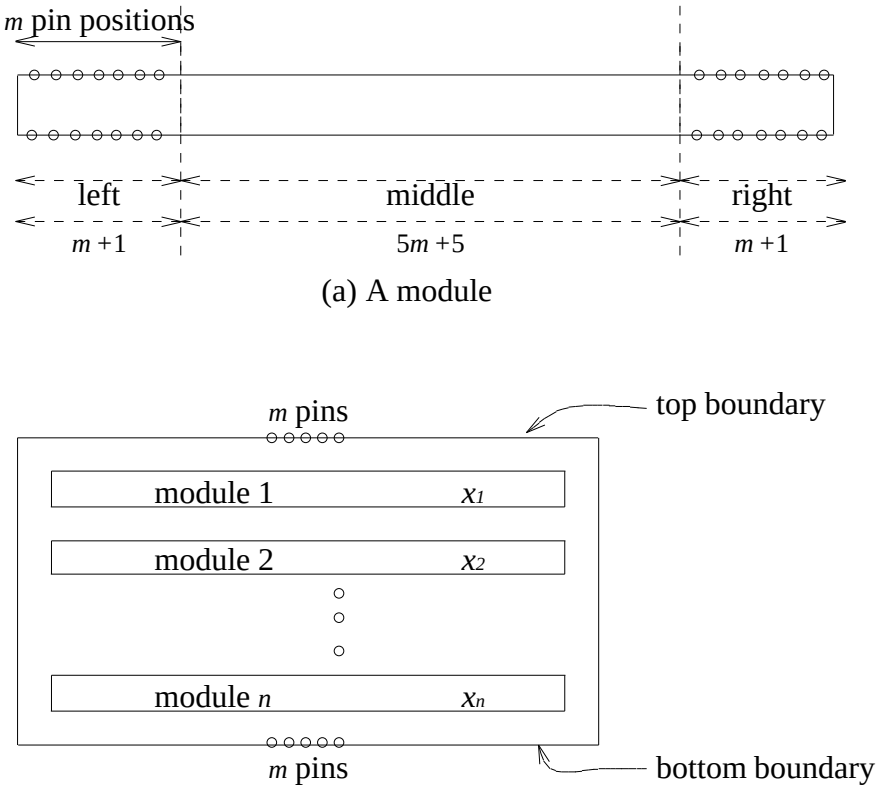
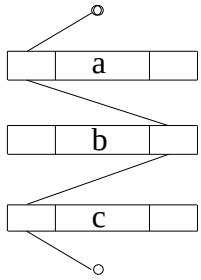


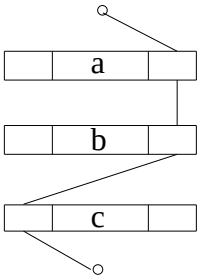

**Figure 2** A module and module column

which of the variables are complemented in $C_j$. The signal path for each of the eight possibilities for $C_i$ is given in Figure 3.

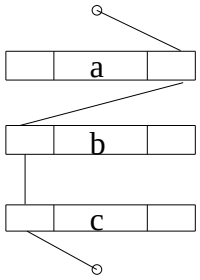
Figure 4 shows the complete construction for the case $I = (x_1+\overline{x_2}+x_3)(x_2+x_4+x_5)(x_3+\overline{x_4}+\overline{x_5})(\overline{x_1}+x_2+x_5)$. The current orientation of each module corresponds to each variable being true. When module $i$ is flipped along its vertical axis, the module orientation corresponds to $x_i$ being false. Consider the case $C_i=\overline{x_a}+x_b+x_c$. There are eight possible orientations for the module set $\{a,b,c\}$. These correspond, respectively, to $(x_{a,}x_{b,}x_c)$ having the truth assignment $(t,t,t)$, ..., $(f,f,f)$. The signal path corresponding to $C_i$ for each of these



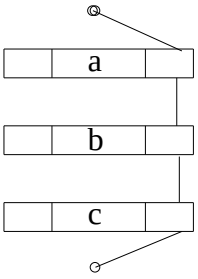

(a) $C_i=x_a+x_b+x_c$



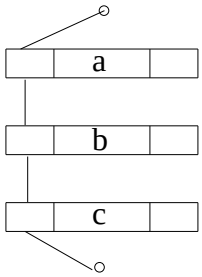

(b) $C_i=\bar{x}_a+x_b+x_c$



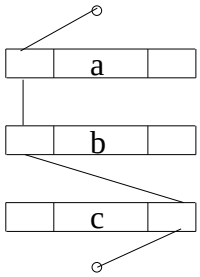

(c) $C_i=\bar{x}_a+\bar{x}_b+x_c$



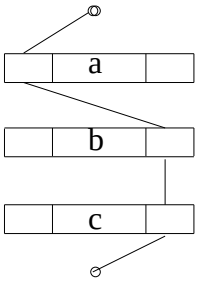

(d) $C_i=\bar{x}_a+x_b+\bar{x}_c$



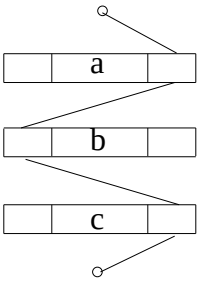

(e) $C_i=x_a+\bar{x}_b+x_c$




(f) $C_i=x_a+\bar{x}_b+\bar{x}_c$




(g) $C_i=x_a+x_b+\bar{x}_c$




(h) $C_i=\bar{x}_a+\bar{x}_b+\bar{x}_c$

**Figure 3** Signal path for different types of clauses

orientation sets is given in Figure 5. We make the following observations :

(1) When all three literals of $C_i$ are true or all are false, the signal path crosses from the left part of one module to the right of the next (or the reverse) twice. Under the assumption that the propagation delay is proportional to the wire length plus the module delays (which are unit) and using the Manhattan metric, the signal delay ($sd$) is 3 (module delays) + $(2n-2)$ (vertical components of wire lengths) + $2(3m+3)$ (horizontal segments of first and last wires) + $2(6m+6)$ (horizontal segments of inter module wires) = $18m+2n+19$.

(2) When the literal truth values are $(f,t,t)$, $(t,f,f)$, $(t,t,f)$, or $(f,f,t)$, only one of the intermodule wires crosses from the left part of one module to the right of the next (or the reverse). The signal delay is $3+(2n-2)+2(3m+3)+(6m+6)=12m+2n+13$.

(3) In the remaining two cases $(t,f,t)$ and $(f,t,f)$ the signal delay is $3+(2n-2)+2(3m+3)=6m+2n+7$.

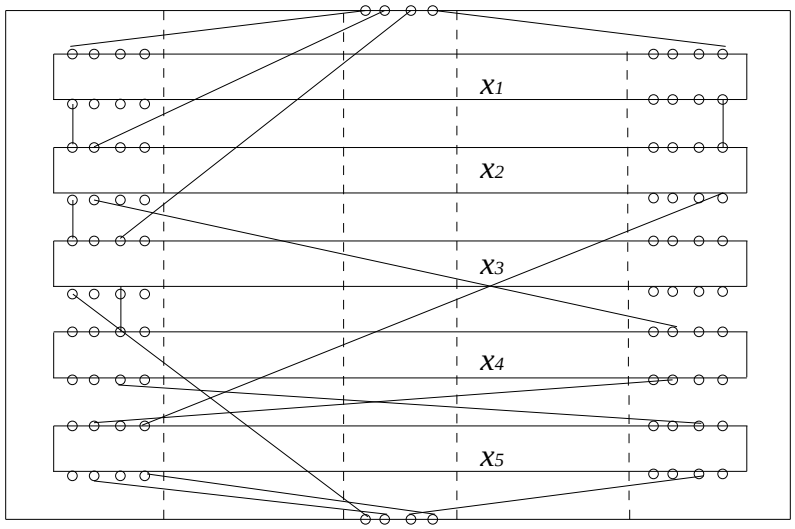
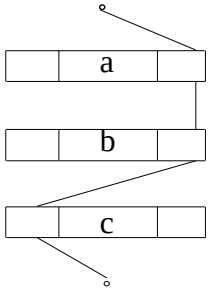
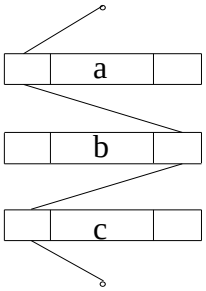



**Figure 4** Construction for I = $(x_1+\overline{x_2}+x_3)(x_2+x_4+x_5)(x_3+\overline{x_4}+\overline{x_5})(\overline{x_1}+x_2+x_5)$

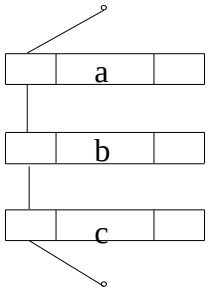
Hence the maximum of the signal delays for the $m$ clauses $C_1, C_2, \cdots C_m$ is less than $18m+2n+19$ iff there is a truth assignment for which every clause has at least one true and one false literal.



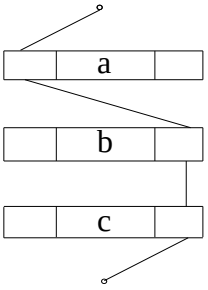

(a) $(x_a,x_b,x_c)=(t,t,t)$
$(\bar{x}_a,x_b,x_c)=(f,t,t)$



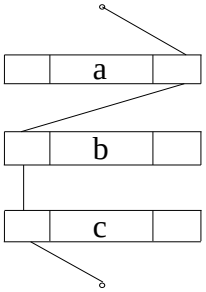

(b) $(x_a,x_b,x_c)=(f,t,t)$
$(\bar{x}_a,x_b,x_c)=(t,t,t)$



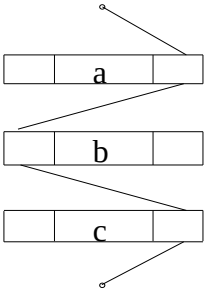

(c) $(x_a,x_b,x_c)=(f,f,t)$
$(\bar{x}_a,x_b,x_c)=(t,f,t)$



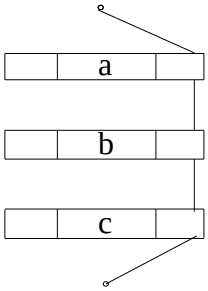

(d) $(x_a,x_b,x_c)=(f,t,f)$
$(\bar{x}_a,x_b,x_c)=(t,t,f)$



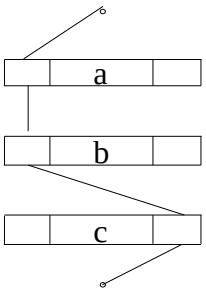

(e) $(x_a,x_b,x_c)=(t,f,t)$
$(\bar{x}_a,x_b,x_c)=(f,f,t)$




(f)$(x_a,x_b,x_c)=(t,f,f)$
$(\bar{x}_a,x_b,x_c)=(f,f,f)$




(g) $(x_a,x_b,x_c)=(t,t,f)$
$(\bar{x}_a,x_b,x_c)=(f,t,f)$




(h) $(x_a,x_b,x_c)=(f,f,f)$
$(\bar{x}_a,x_b,x_c)=(t,f,f)$

**Figure 5** Different truth assignments to $x_a$, $x_b$, $x_c$

I.e., the constructed MaxDelay instance has a solution with delay $< 18m + 2n + 19$ iff the NE3SAT instance $I$ has answer "yes". Since the construction can be done in polynomial time, MaxDelay is NP-hard. □

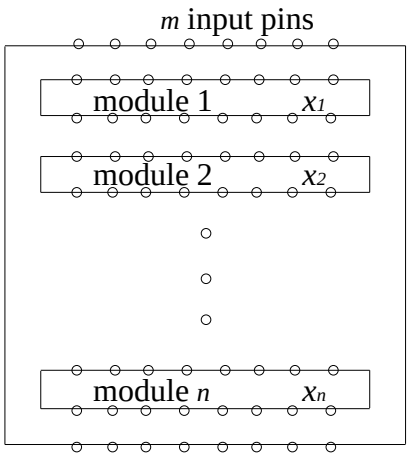
**Theorem 2:** MaxDelay is NP-hard for a single column of equal size modules when both horizontal and vertical flips are permitted.

**Proof:** The proof is the same as that for Theorem 1. In that construction, horizontal flips do not reduce the signal delay. Hence permitting these flips is of no benefit. □

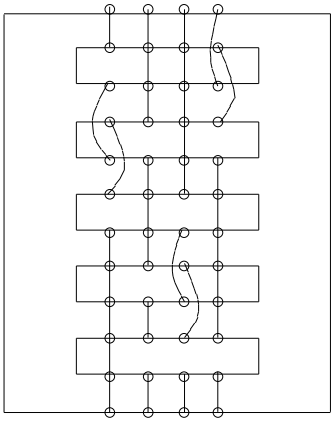
**Theorem 3:** MaxDelay is NP-hard for single column of equal size modules even when the only flips permitted are horizontal flips.

**Proof:** Let $I$ be an $m$ clause $n$ variable instance of 3SAT. Corresponding to $I$, we construct an $n$ module single column instance of MaxDelay. Each module is of unit height, has width $m+1$, and has $m$ uniformly spaced terminals on both its top and bottom sides (Figure 6(a)). The bottom of each module is one unit away from the top of the one below it. The $i$´th module from the top represents variable $x_i$. The module orientations of Figure 6(a) correspond to all variables being true. If module $i$ is flipped about its horizontal axis, its orientation will correspond to $x_i$ being false. For each clause $C_j$ of $I$ we establish a signal path that begins at the $j$´th input pin on the top boundary and ends at the $j$´th output pin at the bottom boundary. If the variables in $C_j$ are $x_a$, $x_b$, and $x_c$, $a<b<c$, then the signal path goes through the $j$´th pins of modules $a$, $b$, and $c$. In case $x_y$, $y \in \{a,b,c\}$ is a literal of $C_j$, then the signal path enters at the $j$´th pin on the top boundary of module $y$ and exits at the $j$´th pin on the bottom boundary. If $x_y$ is not a literal of $C_j$ then $\overline{x_y}$ is and the signal path enters at the bottom and exits from the top. The construction for the case $I = (x_1 + \overline{x_2} + x_3)(x_2 + x_4 + x_5)(x_3 + \overline{x_4} + x_5)(\overline{x_1} + x_2 + x_5)$ is given in Figure 6(b).

When a signal enters the $j$'th top pin and leaves from the $j$'th bottom pin of module $a$, we shall say that the literal corresponding to variable $x_a$ in $C_j$ is true. Otherwise the literal is false. A false literal increases the wire length for the clause by two units over that for a true literal. By horizontally flipping a module, the corresponding variables truth assignment changes and previously false literals become true and true ones become false. This is reflected in our model as a reduction or increase in wire length.




(a) Module column



(b) $I = (x_1 + \overline{x_2} + x_3)(x_2 + x_4 + x_5)(x_3 + \overline{x_4} + x_5)(\overline{x_1} + x_2 + x_5)$

**Figure 6** Construction of Theorem 3

Assume that the module delays are unit and the interconnect delay is given by the Manhattan distance between the wire end points. The signal delay corresponding to clause $C_j$ is 3 (module delays) $+(2n-2)+2$(*number of false literals in* $C_j$). When all three literals are false the signal delay is $2n+7$. When at least one literal is true (i.e., $C_j$ is true) the delay for $C_j$ is at most $2n+5$. The maximum of the signal delays for the $m$ clauses is $2n+7$ iff at least one clause has all three literals false. If at least one literal of each clause is true (and hence the formula $I$ is true), the maximum signal delay is $\leq 2n+5$.

It is easy to see that $I$ is satisfiable iff by horizontally flipping some of the modules in the constructed MaxDelay instance, the max delay is no more than $2n+5$. Since the construction takes polynomial time, the MaxDelay problem for a single module column and restricted to horizontal flipping only is NP-hard. □

**Theorem 4:** ETLF is NP-hard for a single column of equal size modules even when the only flips permitted are vertical flips.

**Proof:** Let $I$ be any $n$ variable $m$ clause instance of NE3SAT. We shall show how to construct, in polynomial time, an instance $T$ of ETLF such from the solution to $T$ one can determine in $O(1)$ time whether $I$ has answer "yes" or "no". For each clause $C_i=(l_{i_1}+l_{i_2}+l_{i_3})$ of $I$, construct a *clause block* as in Figure 7(a). Such a block consists of four modules of width 5. The modules are labeled $L_{i1,}$ $L_{i2,}$ $L_{i3,}$ and $L'_{i1}$ and stacked one on top of the other (in that order) to form a column of modules. Modules $L_{i1}$, $L_{i2}$, and $L_{i3}$, respectively, represent literals $l_{i_1}$, $l_{i_2}$, and $l_{i_3}$. Module $L'_{i1}$ represents no literal. Each module has six uniformly spaced pins on its top and bottom boundaries. Thus two adjacent pins are one unit apart. The distance between two adjacent modules (bottom of one to top of lower one) is also one unit. The four modules are connected by 8 two pin nets as shown in Figure 7(a). The module orientations of Figure 7(a) represent the situation when all three literals are true. Flipping a module about its vertical axis corresponds to changing the truth value of the corresponding literal. Flipping also changes the total wire length of the 8 nets in the clause block. However, only the horizontal component of the Manhattan lengths changes. For the configuration of Figure 7(a), this is 6. If one (but not both) of $L_{i1}$ and $L'_{i1}$ is flipped, the total horizontal length is at least 6 (this is because of the two nets that connect $L_{i1}$ and $L'_{i1}$). Flipping just one these modules corresponds to an inconsistent truth assignment which requires $l_{i1}$ to be both true and false. When all four modules are flipped the horizontal length is again 6. This situation corresponds to all three literals being false. One may verify that when exactly one or two the block subsets $\{L_{i1,}\ L'_{i1}\}$, $\{L_{i2}\}$, and $\{L_{i3}\}$ are flipped, the horizontal wire length is 2. These situations correspond to exactly one or two literals being true. Figure 7(b) shows the case when only $L_{i2}$ is flipped and corresponds to $l_{i1}$ and $l_{i3}$ true and $l_{i2}$ false.

Let $v$ be the total vertical wire length in a clause block. $v=16$ when the blocks are of unit height. The total wire length in a clause block is $v+2$ when either one or two literals are true and $v+6$ when either all three literals are true or all three are false.

The $m$ clause blocks are arranged into a column as in Figure 8(a) with the block for $C_1$ at the top followed by those for $C_{2,}C_{3,}...,C_m$. Additional two pin nets are defined between pairs of modules that are in different clause blocks and which represent literals that are of the same variable. Figure 8(b) shows the case when the modules $L_{i2}$ and $L_{j1}$ represent the same literals (i.e., $l_{i2}=l_{j1}$) and none of the modules in between represent either $l_{i2}$ or $\overline{l_{i2}}$. The 2 two pin nets defined use straight connections. Figure 8(c) shows the case when the modules $L_{i3}$ and $L_{j2}$ represent literals of the same variable but of different polarity (i.e., $l_{i3}=\overline{l_{j2}}$) and no modules in between

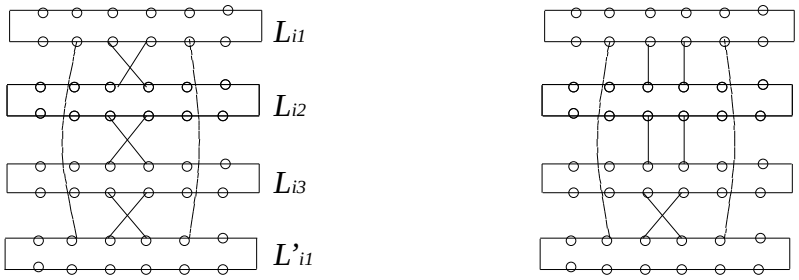



**Figure 7** Clause block for $C_i$

represent either $l_{i3}$ or $\overline{l_{i3}}$. In this case 2 two pin cross connected nets are defined. Note that pairs of two pin nets are defined between two modules $L_{ia}$ and $L_{jb}$ iff both of the following are true:

(1) $l_{ia} = l_{jb}$ or $l_{ia} = \overline{l_{jb}}$

(2) no module between $L_{ia}$ and $L_{jb}$ represents either of the literals $l_{ia}$ or $\overline{l_{ia}}$.

When $l_{ia}=l_{jb}$, straight nets are defined and when $l_{ia}=\overline{l_{jb}}$, cross nets are defined.

By flipping modules about the vertical axis, only the horizontal length of the inter clause block wires is changed. Let $V$ be the total vertical length of these inter block wires. When all modules correspond to consistent truth values for the literals (i.e. each literal has the same truth value in all clause blocks and its complement has the negated value in all clause blocks), then the horizontal length of the inter block wires is zero.

The total length of the inter block wires is therefore $V$ when the module orientations correspond to a truth assignment to the $n$ variables and greater than $V$ when they don't. Hence there is a set of module orientations that results in a wire length $V+m(v+2)$ iff there is a truth assignment which results in each clause of $I$ having at least one true and one false literal. Otherwise, all orientation sets result in the total wire length being greater than $V+m(v+2)$. This completes the proof. □

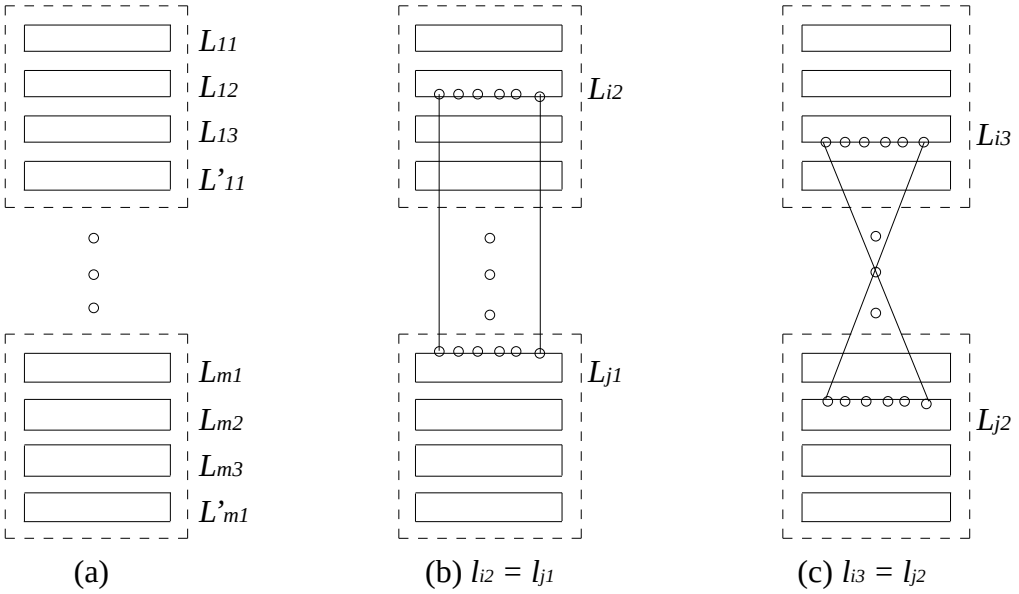



**Figure 8** Combining clause blocks

**Theorem 5:** ETLF is NP-hard for a single column of equal size modules when both horizontal and vertical flips are permitted.

**Proof:** In the construction of Theorem 4, horizontal flips do not reduce wire length. Hence there is no benefit to performing these. □

Theorems 4 and 5 leave unanswered the question: "What is the complexity of single column ETLF when only horizontal flips are permitted?" Horizontal flips affect only the vertical component of a wire's length. One may verify that the simple algorithm of Figure 9 obtains an optimal solution. The correctness of the algorithm also follows from some more elaborate results developed in Section 3.

---

**for** each module $M_i$ **do**
**begin**

- Arbitrarily fix the orientation of all modules other than $M_i$ ;
- Determine the sum of the length of all wires connecting to module $M_i$ using the current and horizontally flipped orientations of $M_i$ ;
- Select the orientation that has a smaller sum. Ties are broken arbitrarily

**end** ;

---

**Figure 9** Algorithm for single column ETLF when only horizontal flips are permitted

## 3 Optimal Orientations

In this section, we assume that module flips (both horizontal and vertical) are permitted but rotations are not. We are interested in developing algorithms that minimize the estimated total wire length (i.e., we study ETLF).

### 3.1 Independence Theorem

For ETLF, we first observe the independence of horizontal and vertical flips when the Manhattan distance metric is used. Suppose a wire connects the two end points $(x_1, y_1)$ and $(x_2, y_2)$. The *horizontal length* of this wire is defined to be $|x_1-x_2|$ and its *vertical length* is defined to be $|y_1-y_2|$. The *wire length* is the sum of its horizontal and vertical lengths, i.e., $|x_1-x_2|+|y_1-y_2|$. We make the following observations:

(O1) The horizontal length of a wire is not affected by a horizontal flip.

(O2) The vertical length of a wire is not affected by a vertical flip.

For any set of module orientations, we may define the *total horizontal (vertical) wire length* as the sum of the *horizontal (vertical) lengths* of all wires. From observations O1 and O2, it follows that the total horizontal (vertical) length is unaffected by horizontal (vertical) flips. This implies the following theorem.

**Theorem 7:** [Independence Theorem] The ETLF problem may be solved by (1) making horizontal flips so as to minimize the total vertical wire length and (2) making vertical flips so as to minimize the total horizontal wire length. (1) and (2) are independent and may be done in either order (i.e., do (1) first then (2), or do (2) first then (1)). □

The independence theorem immediately reduces the complexity of an exhaustive search solution as for an $n$ module circuit at most $2^n$ horizontal orientations and $2^n$ vertical orientations need to be considered. Without the independence theorem, one needs to consider up to $4^n$ possible module orientation sets.

### 3.2 Module Matrix

Chong and Sahni [CHON92] have developed an algorithm that solves ETLF for an $m \times n$ matrix of modules with the property that wires connect pairs of modules in adjacent columns only (Figure 10). Their algorithm has complexity $O(mn+w)$ where $w$ is the number of wires.

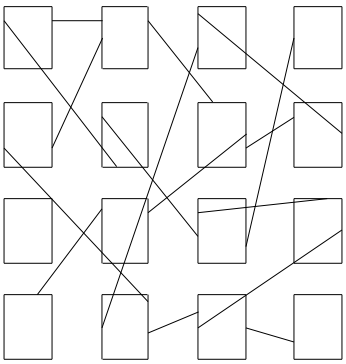
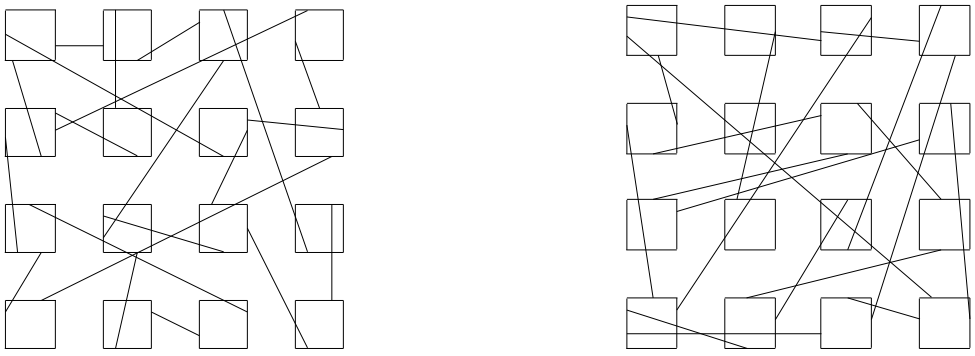


**Figure 10** $4 \times 4$ module matrix with adjacent column connections only

In this section, we generalize the module matrix model of [CHON92]. We permit wires to connect any pair of modules. However if a wire connects two modules in the same row or column, then we limit the number of intermediate modules to $k-1$. Thus, when $k=1$ wires between modules in the same row or column are confined to be between adjacent modules. Figure 11 shows possible matrix modules for the cases $k=1$ and 2.



(a) $k=1$ (b) $k=2$

**Figure 11** Example module matrices

Notice that we permit intracolumn wires while the model of [CHON92] does not. Also, note that when wires may connect modules of the same row or column that are arbitrarily far apart, the ETLF problem is NP-hard even when $n=1$ or $m=1$ (Theorems 4 and 5). Consequently, we do not expect to find a polynomial time algorithm for our model (i.e., polynomial in, $n$, $m$, $w$, and $k$). The algorithm we develop has complexity $O(mn2^k+w)$.

Chong and Sahni [CHON92] have shown that for their matrix model ETLF could be solved by obtaining optimal orientations for each row of modules while arbitrarily selecting the orientations of modules in other rows. That is, they proved that the procedure of Figure 12 minimizes the estimated total wire length. It is easy to see that moving line 3 out of the **for** loop doesn't affect the outcome.

```
1.    for r = 1  to m do
2.    begin
3.          arbitrarily set the orientations of the modules in rows other than r ;
4.          determine the orientation of the modules in row r that minimizes ETLF under the
             constraint that the orientations of the modules in the remaining rows cannot be
             changed ;
5.    end
6.    For each module, use the orientation determined in line 4.
```

**Figure 12** Algorithm of [CHON92]

Using the techniques of Chong and Sahni [CHON92], one may show that for our module matrix model, the total vertical wire length is minimized by the algorithm of Figure 13 and the total horizontal wire length is minimized by the algorithm of Figure 14.

```
1.    arbitrarily select an orientation for each module ;
2.    for r := 1 to m do
3.          perform horizontal flips on the modules in row r so as to minimize the total vertical
             wire length (do not flip modules in other rows) ;
```

**Figure 13** Algorithm to minimize total vertical wire length

```
1.    arbitrarily select an orientation for each module ;
2.    for  c := 1 to n do
3.          perform vertical flips on the modules in column c so as to minimize the total horizon-
             tal wire length (do not flip modules in other columns) ;
```

**Figure 14** Algorithm to minimize total horizontal wire length

Using the independence theorem of the preceeding section, we see that if we eliminate step 1 of the horizontal wire length algorithm (Figure 14) and just perform the vertical wire length algorithm of Figure 13 followed by lines 2 and 3 of Figure 14 the resulting module orientations will have minimum total wire length. An exhaustive search implementation of line 3 of Figure 13 (Figure 14) would examine $O(2^n)$ ($O(2^m)$) possible orientations for the modules in row $r$ (column $c$). However, since wires span at most $k$ rows and/or columns, the worst case complexity can be reduced.

We now consider how to obtain the optimal orientations for a single row or column of modules under the assumption that a wire can span at most $k$ modules in that row or column. Since the algorithm for a single row is the same as that for a single column, we explicitly develop only the former. Let the modules in the row be $M_1, ..., M_n$. Let $tvwl(i, O_i, O_{i+1}, ..., O_{i+k-1})$ be the minimum total wire length for all wires with one end point in modules $M_1, ..., M_{i-1}$ when modules $M_i, ..., M_{i+k-1}$ have the orientations $O_i, O_{i+1}, ..., O_{i+k-1}$, respectively. This sum includes the vertical length of wires to other rows as well as those confined to this row, provided that at least one of the end points is in module $M_1, M_2, ...,$ or $M_{i-1}$. $tvwl(i, \cdots)$ is defined for $1 \le i \le n-k+1$. We note that $tvwl(1, O_1, O_2, ..., O_k) = 0$ for all orientations $O_1, O_2, ..., O_k$ and the $O_j$ have only two possible values each as only horizontal flips are permitted. Let $sv(j, O_j, ..., O_{j+k-1})$ be the sum of the vertical wire lengths of all wires with one end point in module $M_j$ and the other not in $M_1, ..., M_{j-1}$ when $M_j, ..., M_{j+k-1}$ have orientations $O_j, O_{j+1,} ..., O_{j+k-1}$, respectively. Note that $sv$ is easily computed as all wires that contribute to $sv$ have both end points in $T=\{M_j, ..., M_{j+k-1}\} \cup \{$*modules in other rows*$\}$ and the orientations of all modules in $T$ is known. The following recurrence is easily verified:

$$tvwl(i, O_i, O_{i+1}, ..., O_{i+k-1}) = \min\{ tvwl(i-1, O'_{i-1}, O_i, O_{i+1}, ..., O_{i+k-2}) + sv(i-1, O_i, O_{i+1}, ..., O_{i+k-1}) \mid O'_{i-1} \text{ is a possible orientation for } M_{i-1} \} \quad (1)$$

Note that since only horizontal flips are permitted, we need to take the minimum of only two quantities in Equation (1). Equation (1) can be used to compute $tvwl(i, ...)$ for $i = 2,3,...,n-k+1$ in that order. However, when computing for $i = n-k+1$ we modify the definition of $sv$ to:

$sv(n-k+1, O_{n-k+1}, \ldots, O_n)$ = sum of the vertical wire lengths of all wires with one end point in $M_{n-k+1}, M_{n-k+2}, \cdots,$ or $M_n$ and the other not in $M_1, M_2, \cdots,$ or $M_{n-k}$ when modules $M_{n-k+1}, ..., M_n$ have the orientation $O_{n-k+1}, ..., O_n$, respectively.

It is easy to see that $tvwl(n-k+1, O_{n-k+1}, ..., O_n)$ gives the minimum total wire length possible when $M_{n-k+1}, ..., M_n$ are required to have the orientation $O_{n-k+1}, ..., O_n$, respectively and the orientation of the modules in the remaining rows is fixed. The total minimum vertical wire length when the orientation of modules in the remaining rows is fixed is

$$\min\{\ tvwl(n-k+1, O_{n-k+1}, ..., O_n) \mid \text{all possible orientations } O_{n-k+1}, ..., O_n\}$$

All the *tvwl* values may be computed in $O(n2^k$ + the number of wires with at least one end point in $M_1, ..., M_n)$ time. The trace back [HORO76] to find the optimal orientations for $M_1, ..., M_n$ takes less time than this. To compute the *tvwl*'s for all $m$ rows takes $O(mn2^k+w)$ time where $w$ is the number of wires. To solve the $n$ column optimization problems also takes $O(mn2^k+w)$ time. So, the overall time needed to obtain the optimal orientations for our module matrix model is $O(mn2^k+w)$. The case considered in [CHON92] is a special case of our model with $k=1$. The asymptotic complexity of our algorithm for the case $k=1$ is the same as that of the algorithm of [CHON92].

### 3.3 Standard Cells

Chong and Sahni [CHON92] considered the ETLF problem for standard cell designs in which modules are arranged in columns and wires connect pairs of modules in adjacent columns. Only horizontal flips are permitted. They define module dependence in the following way: "Two modules are dependent iff they are in adjacent columns and if the $y$ coordinate of the top of one is contained in the $y$ coordinate range of the other." Chong and Sahni have developed an algorithm of complexity $O(n2^{2km})$ where $n$ is the number of modules, $m$ the number of columns, and $k$ is the maximum number of modules in an adjacent column that any module can be dependent with. For the standard cell problem, we develop an algorithm whose complexity is $O(m2^{2p})$ where $p$ is the maximum number of modules in a column. When modules are uniformly distributed, $p=n/m$ and our algorithm has complexity $O(m2^{2n/m})$. Our algorithm is practical for circuit designs with a limited number of modules per column irrespective of the number of columns in the design. This is not so for the algorithm of [CHON92]. However, their algorithm is superior for designs in which the number of columns is small (small $m$) and $k$ is also small. Note also that the complexity of our algorithm is independent of $k$.

Our standard cell algorithm employs dynamic programming. Let $n_i$ be the number of modules in column $i$. Note that $n = \sum_{1}^{m} n_i$. We use $twl(i, O_1, O_2, ..., O_{n_i})$ to denote the minimum total wire length of wires connecting module pairs from columns 1, ..., $i$ under the assumption that the $j$'th module of column $i$ has orientation $O_j$. Since each module has only two possible orientations (only horizontal flips are permitted), there are $2^{n_i}$ different $(O_1, O_2, ..., O_{n_i})$ tuples. It is easy to see that $twl(1, O_1, O_2, ..., O_{n_1}) = 0$ for each of the $2^{n_1}$ different tuples of the form $(O_1, O_2, ..., O_{n_1})$. The following dynamic programming recurrence is easily verified:

$twl(i, O_1, O_2, ..., O_{n_i})$ = min{ $twl(i-1, O'_1, O'_2, ..., O'_{n_{i-1}})$ + sum of lengths of wires between columns $i-1$ and $i$ under the assumption that the column $i-1$ modules have orientation $(O'_1, O'_2, ..., O'_{n_{i-1}})$ and the column $i$ modules have orientation $(O_1, O_2, ..., O_{n_i})$ | $(O'_1, O'_2, ..., O'_{n_{i-1}})$ is one of the $2^{n_{i-1}}$ orientation sets for column $i-1$ } (2)

Equation (2) may be solved in the order $i$=2,3,...,$n$. The time taken to obtain a single $twl(i, O_1, O_2, ..., O_{n_i})$ is $O(2^{n_{i-1}})$. The time required to obtain all $2^{n_i}$ $twl(i, O_1, O_2, ..., O_{n_i})$ values is $O(2^{n_{i-1}+n_i})$. To do this for all $i$, the time is $O(\sum_{i=2}^{n} 2^{n_{i-1}+n_i}) = O(m2^{2p})$ where $p = \max\{ n_i \mid 1 \le i \le m \}$.

Using standard dynamic programming trace back techniques [HORO76] one can obtain the optimal orientations in additional $O(n)$ time beginning with the $f(m, O_1, O_2, ..., O_{n_m})$ that is minimum.

## 4 Heuristic For MaxDelay

Our heuristic for the MaxDelay problem works in two phases:

Phase1: [Initial Assignment] Obtain an initial orientation for each module.

Phase2: [Refinement] Change some of the initial orientations so as to reduce the maximum delay.

### 4.1 Phase1: Initial Assignment

We assume that the circuit graph is acyclic as otherwise there are signal paths of infinite delay. There is delay associated with each edge and vertex of the graph. Vertex delays correspond to module delays while edge delays correspond to signal propagation delays. The graph model is easily generalized to the case when the module delay is a function of the module input-output pin pair along which the signal flows. False paths, however, are not accounted for.

Assume that the modules $M_1, M_2, ..., M_n$ have been numbered in topological order [HORO90]. So, if there is a path from $M_i$ to $M_j$ in the circuit graph, then $i < j$. For each module $M_i$ and module orientation $r$, we compute two numbers InputDelay($i,r$) and OutputDelay($i,r$). InputDelay($i,r$) is the maximum signal delay from any circuit input pin to any input pin of module $M_i$ under the assumption that $M_i$ has orientation $r$. OutputDelay($i,r$) is the maximum signal delay from any input pin of $M_i$ to any circuit output pin under the assumption that module $M_i$ has orientation $r$. Let $<j,i>$ denote a wire from a circuit input pin $j$ or from a module $M_j$ (we assume circuit input pin numbers are distinguishable from module numbers) to an input pin of module $M_i$. The direction of signal flow is from $j$ to $i$. Similarly, let $<i,j>$ denote a wire from an output pin of $M_i$ to a circuit output pin $j$ or to an input pin of $M_j$. Let WireDelay($i,j,r,s$) denote the signal transmission delay along the wire $<i,j>$ when module/pin $i$ has orientation $r$ and $j$ has orientation $s$. Let InputDelay($i,r$) be zero in case $i$ denotes a circuit input pin; and let OutputDelay($i,r$) be zero in case $i$ denotes a circuit output pin. Also, Delay($i$) is zero for $i$, a circuit input or output pin. The following recurrences are easily verified for the case when there are no false paths or when these are ignored:

InputDelay($i,r$) = max { InputDelay($j,s$) + Delay($j$) + WireDelay($j,i,s,r$) | $<j,i>$ is a wire and $s$ is an allowable orientation }

OutputDelay($i,r$) = max { Delay($i$) + WireDelay($i,j,r,s$) +OutputDelay($j,s$) | $<i,j>$ is a wire and $s$ is an allowable orientation }

The recurrence for InputDelay is easily solved in topological order, i.e., $i=1,2,...,n$ and that for OutputDelay is easily solved in reverse topological order, i.e., $i=n,n-1,...,1$.

For each module $M_i$, we pick as the initial orientation an orientation $r$ such that

InputDelay($i,r$) + OutputDelay($i,r$) = min { InputDelay($i,s$) + OutputDelay($i,s$) | $s$ is an allowable orientation }.

The time required to obtain the initial module orientation is O($n+w$) where $n$ is the number of modules and $w$ the number of wires (see [HORO90] for a linear time topological ordering algorithm).

### 4.2 Phase2: Refinement

To reduce the maximum delay further, we follow a two step process. In step 1, we begin with a path, $P_L$, of maximum delay. Let the modules on this path be $M_{i_1}$, $M_{i_2}$, ..., $M_{i_k}$ and let $O_j$ be the current orientation of $M_{i_j}$. Let $O'_j$, $1 \le j \le k$ define orientations for these modules that result in the least possible delay along path $P_L$. To determine these orientations, let $D(a,s)$ represent the least delay possible on the path segment of $P_L$ from $M_{i_a}$ to the output pin under the assumption that $M_{i_a}$ has orientation $s$. Clearly,

$$D(k,s) = \text{Delay}(k) + \text{delay on wire from } M_{i_k} \text{ to the output pin of } P_L$$

For all other modules, we have

$$D(a,s) = \min \{ \text{Delay}(a) + D(a+1,r) + \text{WireDelay}(i_a, i_{a+1}, s, r) \mid r \text{ is an allowable orientation of } M_{i_{a+1}} \}.$$

Let $b$ denote the input pin of $P_L$. The minimum possible delay, $D_{\min}(P_L)$, on $P_L$ is min { $D(1,s)$ + WireDelay($b, i_1, r, s$) | $r$ is a pin orientation and $s$ is an allowable orientation of $M_{i_1}$ }.

By keeping track of the orientations $r$ that minimize the right hand side of the equation for $D(a,s)$, one can obtain the orientation $O'_i$, $1 \le i \le k$ of the modules on $P_L$ that results in a delay of $D_{\min}(P_L)$. Having determined these orientations, we note that if $D_{\min}(P_L)$ = delay of $P_L$ using orientation $O_i$ (denoted delay($P_L$, $O$)), $1 \le i \le k$, then we have an optimal solution and this cannot be improved further. If not, the orientations of the modules on $P_L$ are changed to $O'_i$, $1 \le i \le k$ and we see if any of the modules on $P_L$ are on paths of length greater than delay ($P_L$, $O$). If so, modules, $M_{i_j}$, on such paths have their orientations reset to $O_j$. If this results in the delay of $P_L$ being less than delay($P_L$, $O$) then we find a new longest path and attempt to reduce its length in this way. If the length of $P_L$ exceeds delay($P_L$, $O$), we fix the orientations of the modules whose orientation was reset to the $O_j$ orientations and find a new set of orientations for the remaining modules on $P_L$ such that the length of $P_L$ is minimized under the restriction that the orientation of the fixed modules is not changed. These orientations are determined in a manner similar to that used to obtain the $O'$ orientations. This process continues until all modules on the current longest path

have their orientation fixed. Then we move to step 2.

In step 2, we try to obtain a further reduction in the maximum delay by considering modules on a path of maximum delay one at a time. We attempt to change the orientation of a single module on $P_L$ so as to reduce the maximum delay of the whole circuit. This continues until there is no single module whose orientation can be changed so as to reduce the maximum delay. If the maximum delay is reduced in step 2, we go back to step 1. Otherwise, we terminate.

A more formal description of the refinement steps is given in Figure 15.

### 4.3 Experimental Results

Our two phase heuristic was programmed in Pascal and experiments were conducted on an Apollo 3500 workstation. The test data consisted of ten circuits having a number of modules that ranged from 12 to 96. Each of the modules was permitted to be flipped horizontally and/or vertically. Rotations were prohibited as these would result in module overlap for our test circuits. The results of our experiments are given in Table 1. The second column gives the maximum delay using the input module orientations. The fourth column gives the delay following Phase 1 (i.e., after initial assignment). The time required for this phase is given in column three. As can be seen, this phase is quite fast and reduces the maximum delay 10% to 20%. The results following the second phase (refinement) are given in columns five and six. The time column gives the time for both phases 1 and 2. Phase 2 takes more time than phase 1 but does provide some further reduction in the maximum delay.

We conducted another experiment to measure the relative effectiveness of steps 1 and 2. In this, Phase 1 (initial assignment) was followed directly by step 2 of Phase 2 (i.e., step 1 of Phase 2 was eliminated). The results of this experiment are given in columns seven and eight of Table 1. In seven of the ten test cases this was enough to obtain a solution as good as that obtained by running both steps of Phase 2. Thus, if computer time is a concern, one could eliminate step 1 of Phase2.

```
repeat
   find the path P_L with maximum delay ;
   find the optimal orientations O´ for modules on P_L ;
   if delay(P_L, O) = delay(P_L, O´) then OptimalStop:=true ;   (* optimal STOP *)
   else L := delay(P_L, O) ;
        improved := false ;
    (* step 1 *)
        repeat
             newlyfixed := false ;
             change orientations to O´ temporarily ;
          (* find a longest path containing an unfixed module on P_L with delay more than L *)
             bigval := L;   bigind := 0;
             For each module md on P_L which is not fixed do
                  find a longest path L_md passing md
                  if delay(L_md, O´) > bigval
                  then      bigind := md;  bigval := delay(L_md, O´)
             endfor
          (* end *)
             if  bigind > 0  (* if there is a path passing an unfixed module on P_L & longer than L *)
             then  [ fix the orientation of module bigind as original status;
                      (* no change for that module within current iteration *)
                     newlyfixed := true;
                     find the new orientations O´ minimizing P_L without changing the fixed modules'
                      orientations; ]
        until (not newlyfixed);
        Minimize the length of P_L by changing orientations to O´
    (* step 2 *)
        if delay(P_L, O) = delay (P_L, O´)  (* if no improvement *)
        then   premin := delay(P_L, O) ;
             while there is a module Md on P_L which is not processed do
                  find an orientation s of Md which minimizes the maximum delay in the circuit;
                  Let this new delay be D;
                  if D < premin
                  then [change the orientation of Md to s;
                        premin := D;
                        improved := true;
                        mark Md as processed;]
             end of while
        else   improved := true
until (not improved) or (OptimalStop)
```

**Figure 15** Refinement Step

| | at initial time | after initial assignment | | after final assignment | | step 2 only | |
|---|---|---|---|---|---|---|---|
| # of modules | Max Delay | time | Max Delay | time | Max Delay | time | Max Delay |
| 12 | 199 | 0.66 | 176 | 1.49 | 175 | 1.00 | 175 |
| 16 | 237.22 | 1.20 | 196.00 | 4.96 | 184.56 | 2.09 | 184.56 |
| 24 | 335.66 | 1.19 | 286.90 | 4.28 | 266.65 | 1.73 | 281.11 |
| 32 | 396.00 | 1.38 | 337.00 | 3.54 | 327.00 | 1.98 | 329.00 |
| 40 | 980.00 | 1.39 | 887.53 | 3.70 | 883.09 | 1.93 | 883.09 |
| 48 | 690.93 | 2.67 | 575.53 | 7.38 | 574.82 | 3.76 | 574.82 |
| 56 | 945.80 | 2.46 | 826.50 | 6.40 | 815.55 | 3.56 | 815.55 |
| 64 | 455.82 | 2.92 | 375.37 | 17.62 | 345.95 | 4.42 | 347.67 |
| 72 | 1214.14 | 4.81 | 979.05 | 64.09 | 949.46 | 10.88 | 949.46 |
| 96 | 2607.63 | 5.55 | 2099.44 | 84.73 | 2002.38 | 12.61 | 2002.38 |

**Table 1** Experimental results (time is in seconds)

## 5 Conclusions

We have shown that several simple versions of the module flipping problem are NP-hard. For the ETLF problem, we developed algorithms to obtain optimal solutions for module matrix and standard cell designs. A heuristic for MaxDelay was also developed and evaluated experimentally.

## 6 References

[AHN92] K. Ahn and S. Sahni, 'NP-hard module rotation problems', University of Florida, Technical Report, August, 1992.

[CHON91] K.R.Chong and S.Sahni, 'Flipping modules to minimize maximum wire length', Proc. IEEE Intl' Conf. On Computer Design, ICCD'91, pp 528-531.

[CHON92] K.R.Chong and S.Sahni, 'Minimizing total wire length by flipping modules', Proceedings VLSI Design'92, IEEE Press, pp 25-30.

[GARE79] M.R.Garey and D.S.Johnson, 'Computers and Intractability: A Guide to the theory of NP-completeness', W.H.Freeman and Co., New York, 1979.

[HORO76}]E. Horowitz and S.Sahni, Fundamentals of computer algorithms, Computer Science Press, Maryland, 1976.

[HORO90] E. Horowitz and S. Sahni, Fundamentals of data structures in Pascal, 3rd Edition, Computer Science Press, Maryland, 1990.

[LIBE89] R.Libeskind-Hadas and C.L.Liu, 'Solutions to the modules orientation and rotation problems by neural computation network', Proc. 26th Design Automation Conference, pp 400-405, 1989.

[YAMA88] M.Yamada and C.L.Liu, 'An analytical method for optimal module orientation', Proc. 1988 International Symp. on Circuits and Systems, pp 1679-1682.