DIMACS Technical Report 98-03
December 1997

Optimal cell flipping to minimize channel density in VLSI design and pseudo-Boolean optimization[1]

by

Endre Boros[2] Peter L. Hammer[2] Michel Minoux[3]
David J. Rader, Jr.[4]

[1]Research supported in part by the Office of Naval Research (Grants N0001492F1375, N0001492F4083, and N000149310890), by the Air Force Office of Scientific Research (Grant F49620-93-1-0041), and by NATO (Grant CRG 931531).
[2]Permanent member of DIMACS and RUTCOR, Rutgers, The State University of New Jersey, 640 Bartholomew Road, Piscataway, NJ 08854-8003, USA; Email: {*boros*, *hammer*}@*rutcor.rutgers.edu*
[3]Laboratoire Masi, Université Paris 6, 4 place Jussieu, 75005 Paris, France; Email: *Michel.Minoux@lip6.fr*
[4]Department of Mathematics, Rose-Hulman Institute of Technology, Terre Haute, IN 47803, USA; Email: *David.Rader@rose-hulman.edu*.

ABSTRACT

Cell flipping in VLSI design is an operation in which some of the cells are replaced with their "mirror images" with respect to a vertical axis, while keeping them in the same slot. After the placement of all the cells, one can apply cell flipping in order to further decrease the total area, approximating this objective by minimizing total wire length, channel width, etc. However, finding an optimal set of cells to be flipped is usually a difficult problem. In this paper we show that cell flipping can be efficiently applied to minimize channel density in the standard cell technology. We show that an optimal flipping pattern can be found in $O(p(\frac{n}{c})^c)$ time, where n, p and c denote the number of nets, pins and channels, respectively. Moreover, in the one channel case (i.e. when $c = 1$) the cell flipping problem can be solved in $O(p \log n)$ time. For the multi-channel case we present both an exact enumeration scheme and a mixed-integer program that generates an approximate solution very quickly. We present computational results on examples up to 139 channels and 65 000 cells.

1 Introduction

This paper investigates a combinatorial problem arising from the computer aided design of VLSI circuits in the so-called "*standard cell*" technology. It is assumed that cell placement has already been performed, i.e. all the cells have been given a physical location (slot) in a set of horizontal rows. The only remaining degree of freedom at this stage, is that of performing *cell flipping*, i.e. the operation of replacing some of the cells with their "mirror image" with respect to a vertical axis, while keeping them in the same slot of the same row.

Examples where flipping a few cells improves the design by reducing *channel width* can be easily found, raising the combinatorial problem of determining an optimal subset of cells to be flipped, so that channel width is minimized. Note that in standard cell design, minimizing channel width is a particularly relevant criterion since, when the number of channels is fixed, the total surface of the chip is simply proportional to the sum of channel widths.

Unfortunately, even without cell flipping, it is known that finding a minimum width channel routing is a difficult problem (see e.g. LA PAUGH 1980, who showed that channel width minimization in the jog-free Manhattan Model is NP-hard). However, numerous computational studies have shown that, in most practical cases, channel width may be very closely approximated by the so-called *channel density*, an easily computable quantity.

Cell flipping was considered in [2] for minimizing total wire length in the more general custom cell technology, and was shown to lead to an NP-hard formulation. In [4] a cell flipping formulation was considered to satisfy channel density and net span constraints. In case of one channel an efficient algorithm, based on 2-satisfiability, was proposed.

Our purpose in this paper is to show a different pseudo-Boolean formulation for channel density minimization. Using this formulation we are able to show that channel density minimization in the standard cell technology can be solved in polynomial time for any fixed number of channels. Furthermore, this pseudo-Boolean formulation allows us to derive an mixed-integer programming approximation model for the multi-channel case. A large number of the variables and constraints could be eliminated by applying a linearization technique, and thus even very large problems could be solved approximately by standard packages.

We give first, in Section 2, a detailed description of the one channel case, and show in Section 3 that the cell flipping problem for one channel can be reformulated as a min-max problem involving pseudo-Boolean functions, i.e. real valued functions in $0-1$ variables. More precisely, the problem becomes to minimize the pointwise maximum of a collection of quadratic pseudo-Boolean functions, each one of which depends on at most two of the variables. In Section 4, this problem is shown to be polynomially solvable via a binary search. At each step of this search, the subproblem of deciding whether the optimum value is less than or equal to the current threshold value, is reduced to a 2-satisfiability problem, and from this, a polynomial complexity bound is easily derived. In Section 5, we show how the approach may be extended to the case of several channels, leading to a polynomial time algorithm for any *fixed* number C of channels. In Section 6 we present a technique for efficient computation of the lower bounds of a given channel's density as well as a mixed-integer programming approximation model for the multi channel case, Finally, in Section 7, we present computational results for actual VLSI chips with up to 139 channels and over 65 000 cells. We would like to thank Frédéric Pétrot, University of Paris 6, CAD Research Team, for providing us with the data for these examples.

2 Optimal cell flipping: the one channel case

Standard cell technology has been widely used as an alternative to custom design of VLSI circuits because of its cheaper development and production costs. In standard cell design (see e.g. LENGAUER 1990) a *channel* is the area between two consecutive rows of *cells*.

The cells contained in the first and second rows will be denoted by C_i, with i in the index sets I_1 and I_2, respectively. Each cell has electrical inputs/outputs called *pins* (which are located on the side of the channel) by which it is connected to other cells either contained in the same row, or in the other row (see figure 1).

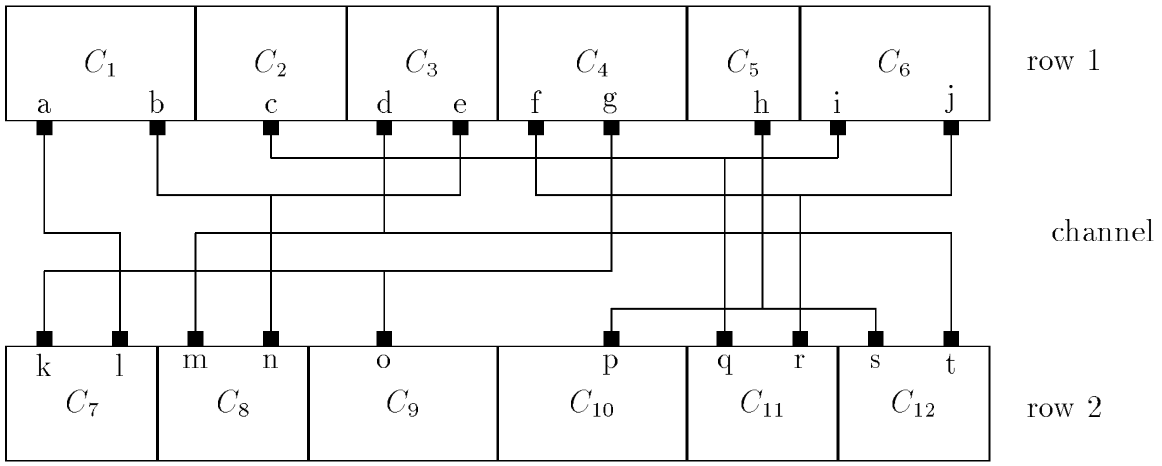

Figure 1: A channel with two rows of 6 cells each. Dots indicate *vias* i.e. electrical connections between a horizontal and a vertical wire. Dark squares indicate pins.

The pins of the cells are linked through electrical connections called equipotentials or *nets*. For instance, on figure 1, there is a net connecting pin b of C_1, pin e of C_3 and pin n of C_8.

The list of nets (so called *net list*) corresponding to figure 1 is:

$$\{a,l\}\{b,e,n\}\{c,i,q\}\{d,m,t\}\{f,r,j,\}\{g,k,o\}\{h,p,s\}.$$

In the fabrication process, nets are physically routed by using two *metal layers*, one horizontal and one vertical[1]. This requires space between the two rows of active compotents (cells). Of course, one of the objectives of all layout systems is to reduce the surface of the chip implementing the design. In the case of a single channel, this just amounts to minimizing *channel width*, i.e. the total number of distinct horizontal tracks necessary to route all the nets.

In the routing of the 7 nets shown on figure 1, it can be seen that channel width is equal to 5 (because 5 distinct horizontal tracks are needed to perform the routing). Optimizing cell placement to miminize channel width is a difficult problem which has been extensively addressed in the literature (see e.g. LENGAUER 1990).

In many cases, the minimum achievable channel width has been shown to be closely related to the so-called *channel density*, which is defined as follows. Taking account of the physical locations of the pins, each net may be represented as a closed interval of the real line: the left end of the interval corresponds to the leftmost pin involved in the set, and the right end of the interval, to the rightmost pin in the net.

For any $\alpha \in$, let $\nu(\alpha)$ denote the number of intervals in the family containing α. $\nu(\alpha)$ will also be called the *local density* of the channel at α. Then the *channel density* is

$$d = \max_{\alpha \in} \{\nu(\alpha)\}.$$

Clearly the channel width can never be less than the channel density. On the other hand, numerous computational studies have shown that, in a high proportion of cases, minimum channel width and channel density are equal and, when this does not occur, the relative difference is always rather small. This explains why, in most CAD tools for VLSI circuit layout, minimum channel density is looked for as a good approximation to minimum channel width.

In the present paper, we assume that the cells have already been placed on the two sides of the channel and that the only freedom we have to further improve the design, is to perform *cell flipping*.

Cell flipping is the operation by which a given cell is replaced by its symmetric with respect to a vertical axis Δ through its center. Through this operation, we observe that the location of the pins of the cell are changed accordingly (see figure 2).

The fact that a given design may be improved with respect to channel density by using cell flipping can be illustrated on the example of *figure 1*. Indeed, flipping cell C_4 leads to the new configuration shown on *figure 4* where it is seen that channel density has been reduced by 1 and routing with channel width = channel density = 4 is now possible.

[1] Some new emerging technologies, feature more flexible routing capabilities by using more than two layers. Here we limit ourselves to the standard two-layer technology.

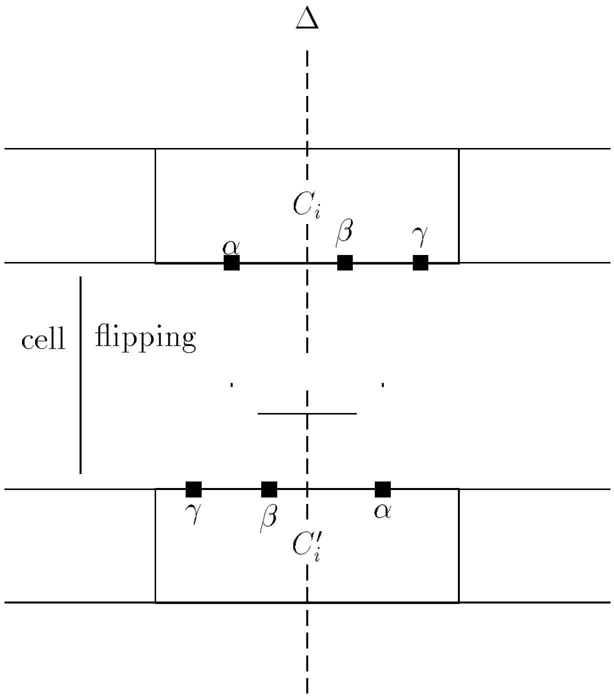

Figure 2: Flipping cell C_i.

3 Formulation of the one channel flipping problem as a pseudo-Boolean optimization problem

With each cell C_i of the design under consideration, let us associate a boolean variable x_i where $x_i = 1$ if and only if cell C_i is flipped. We therefore have $n = |I_1| + |I_2|$ boolean variables x_i $(i = 1, \ldots, n)$.

For any possible choice of $x = (x_i) \in \{0,1\}^n$, let $d(x_1, \ldots, x_n)$ denote the corresponding channel density. Thus, the problem is to find $x^* \in \{0,1\}^n$ such that

$$d(x^*) = \min_{x \in \{0,1\}^n} \{d(x)\}.$$

We now show how to obtain an algebraic expression of the pseudo-Boolean function d. The problem data essentially consist of:

- the location of each cell and its pins;
- the net list.

Thus, in order to specify an arbitrary instance of the problem, we are given the following data:

- for each cell C_i $(i \in I_1 \cup I_2)$ the abscissa μ_i of the center of the cell;

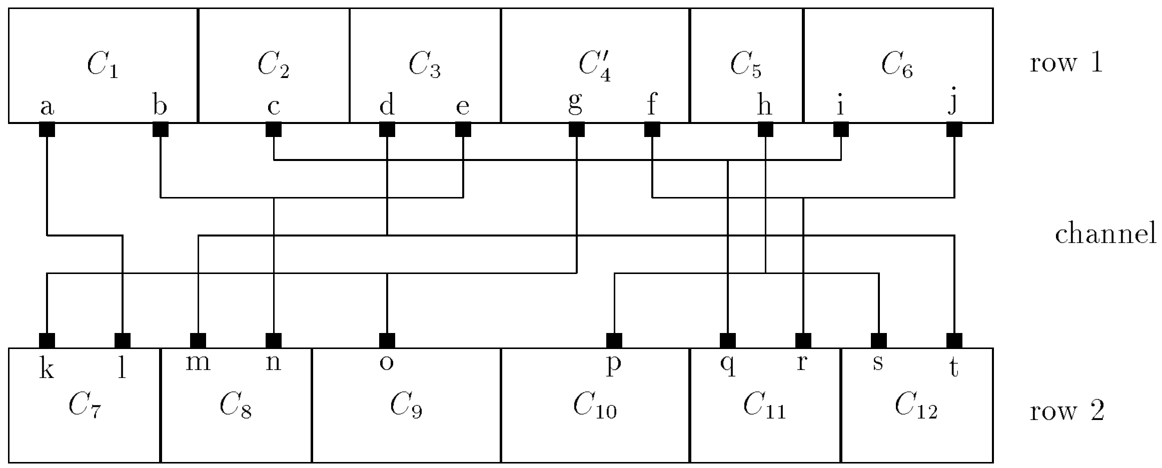

Figure 3: The new channel configuration obtained from Figure 1 after flipping cell C_4 (C_4' denotes the result of flipping C_4).

- for each cell C_i ($i \in I_1 \cup I_2$) the set $P(i)$ giving the list of the names (indices) of the various pins of C_i;
- for each cell C_i and each pin $p \in P(i)$, the abscissa α_p of pin p;
- the number L of nets in the net list and, for each $l \in [1, L]$, the list S_l of pins to be electrically connected though net l;
- for each pin p, the index number $\varphi(p)$ of the cell to which it belongs.

The above data is sufficient to derive all the necessary information about the cells C_i' deduced from C_i by flipping. Clearly, cell C_i' is still centered at μ_i, and has the same set of pins $P(i)$. Also we note that the abscissae α_p of the pins $p \in P(i)$ in C_i are transformed to α_p' in C_i' where:

$$\forall p \in P(i) \ : \ \alpha_p' = 2\mu_i - \alpha_p.$$

The *range* of a cell C_i, $i \in I_1 \cup I_2$, is the closed real interval $R_i = [\gamma_i, \theta_i]$ where:

$$\gamma_i = \min_{p \in P(i)} \{\alpha_p\}, \quad \text{and} \quad \theta_i = \max_{p \in P(i)} \{\alpha_p\}.$$

and the range of C_i', i.e. cell i after flipping, is $R_i' = [\gamma_i', \theta_i']$ where

$$\gamma_i' = \min_{p \in P(i)} \{\alpha_p'\}, \quad \text{and} \quad \theta_i' = \max_{p \in P(i)} \{\alpha_p'\}.$$

Now consider any configuration of the cells defined by a given boolean vector $\hat{x} \in \{0,1\}^n$ and let $l \in [1, L]$ be a net.

The range of this net in the configuration specified by $\hat{x}$ is defined as the closed interval

$$T_l(\hat{x}) = [\sigma_l(\hat{x}), \rho_l(\hat{x})]$$

where

$$\sigma_l(\hat{x}) \quad = \quad \min \quad \min_{\substack{p \in S_l \\ \hat{x}_{\varphi(p)}=0}} \{\alpha_p\} \ ; \ \min_{\substack{p \in S_l \\ \hat{x}_{\varphi(p)}=1}} \{\alpha'_p\}$$

$$\rho_l(\hat{x}) \quad = \quad \max \quad \max_{\substack{p \in S_l \\ \hat{x}_{\varphi(p)}=0}} \{\alpha_p\} \ ; \ \max_{\substack{p \in S_l \\ \hat{x}_{\varphi(p)}=1}} \{\alpha'_p\}$$

Now let $\delta(\alpha, \hat{x})$ denote the local channel density obtained at the abscissa $\alpha \in$ when the variables $x_1, \ldots, x_n$ are assigned the values $\hat{x}_1, \hat{x}_2, \ldots, \hat{x}_n$ respectively; we have:

$$\delta(\alpha, \hat{x}) = |\{l \in [1, L] : \alpha \in T_l(\hat{x})\}|, \tag{1}$$

and from the definition of channel density:

$$d(\hat{x}_1, \hat{x}_2, \ldots, \hat{x}_n) = \max_{\alpha \in} \{\delta(\alpha, \hat{x})\}. \tag{2}$$

We observe that $\forall l \in [1, L]$, and $\forall \hat{x} \in \{0,1\}^n$, all the possible intervals $T_l(\hat{x})$ have endpoints in the set of real numbers:

$$A = \mathop{}_{i \in I} (\{\alpha_p | p \in P(i)\} \cup \{\alpha'_p | p \in P(i)\}).$$

Note that $|A| \leq 2N_P = 2 \mathop{}_{i \in I} |P(i)|$.

Therefore it is easily seen that the maximum defined by (2) remains the same by considering only those values of α which are in A. In other words, for any $\hat{x} \in \{0,1\}^n$:

$$d(\hat{x}_1, \hat{x}_2, \ldots, \hat{x}_n) = \max_{\alpha \in A} \{\delta(\alpha, \hat{x})\}. \tag{3}$$

From this, it follows that the pseudo-Boolean function d is defined as the pointwise maximum of at most $2 \times N_P$ pseudo-Boolean functions $\delta(\alpha, x_1, \ldots, x_n)$, where N_P is the total number of pins in the problem.

We shall show now that, for each $\alpha \in A$, $\delta(\alpha, x_1, x_2, \ldots, x_n)$ takes on a particularly simple form.

Consider an arbitrary $\alpha \in A$. There exists at most one cell $i \in I_1$ such that $\alpha \in R_i \cup R'_i$. Similarly, there exists at most one cell $j \in I_2$ such that $\alpha \in R_j \cup R'_j$.

Clearly, the local channel density $\delta(\alpha, x)$ remains constant for fixed x_i and x_j, for any flipping of the other cells $k \neq i$ $k \neq j$. In other words, $\delta(\alpha, x)$ only depends on the two boolean variables x_i and x_j.

Therefore, for each given α, we only have to compute four values, namely:

$$
\begin{aligned}
\beta_{ij}(0,0) &= \delta(\alpha, 0, \ldots, \overset{\overset{i}{\downarrow}}{0}, 0, \ldots, \overset{\overset{j}{\downarrow}}{0}, \ldots, 0) \\
\beta_{ij}(0,1) &= \delta(\alpha, 0, \ldots, 0, 0, \ldots, 1, \ldots, 0) \\
\beta_{ij}(1,0) &= \delta(\alpha, 0, \ldots, 1, 0, \ldots, 0, \ldots, 0) \\
\beta_{ij}(1,1) &= \delta(\alpha, 0, \ldots, 1, 0, \ldots, 1, \ldots, 0).
\end{aligned}
$$

This computation may easily be carried out as follows:

We assume that, for each net l, the index number $\min(l)$ (resp. $\max(l)$) of the pin in S_l having minimum abscissa (resp. maximum abscissa) in the initial configuration of the cells has been precomputed (it is easy to see that this precomputation has to be carried out only once, and can be done in $\mathcal{O}(N_P)$ time, where $N_P = \underset{i \in I_1 \cup I_2}{\quad} |P(i)|$ is the total number of pins).

Now, to compute the values β_{ij}, for each value of the pair (x_i, x_j) (i.e. $(0,1)$, $(1,0)$ and $(1,1)$), we do the following:

Set $\beta_{ij}(0,0) \leftarrow 0$, $\beta_{ij}(0,1) \leftarrow 0$, $\beta_{ij}(1,0) \leftarrow 0$, $\beta_{ij}(1,1) \leftarrow 0$, and set $\mathcal{L} \leftarrow \emptyset$.
For all $p \in P(i)$ do: $\mathcal{L} \leftarrow \mathcal{L} \cup \{\varphi(p)\}$.
For all $p \in P(j)$ do: $\mathcal{L} \leftarrow \mathcal{L} \cup \{\varphi(p)\}$.
For all $(x_i, x_j) \in \{0,1\}^2$ do:
For all $l \in \mathcal{L}$ do:
    $y_l \leftarrow +\infty$
    $z_l \leftarrow -\infty$
    for all $p \in S_l$ do:
        if $p \notin P(i) \cup P(j)$ then
            if $\alpha_p < y_l$ set $y_l \leftarrow \alpha_p$
            if $\alpha_p z_l$ set $z_l \leftarrow \alpha_p$
        else
        if ($p \in P(i)$ and $x_i = 0$) or ($p \in P(j)$ and $x_j = 0$) then:
            if $\alpha_p < y_l$ set $y_l \leftarrow \alpha_p$
            if $\alpha_p z_l$ set $z_l \leftarrow \alpha_p$
        if ($p \in P(i)$ and $x_i = 1$) or ($p \in P(j)$ and $x_j = 1$) then:
            if $\alpha'_p < y_l$ set $y_l \leftarrow \alpha'_p$
            if $\alpha'_p z_l$ set $z_l \leftarrow \alpha'_p$
    if $\alpha \in [y_l, z_l]$ then
            $\beta_{ij}(x_i, x_j) \leftarrow \beta_{ij}(x_i, x_j) + 1$

Compute $\Delta = \delta(\alpha, 0, 0, ..., 0) - \beta_{ij}(0,0)$ and set
$\beta_{ij}(0,0) \leftarrow \beta_{ij}(0,0) + \Delta$,
$\beta_{ij}(1,0) \leftarrow \beta_{ij}(1,0) + \Delta$,
$\beta_{ij}(0,1) \leftarrow \beta_{ij}(0,1) + \Delta$,
$\beta_{ij}(1,1) \leftarrow \beta_{ij}(1,1) + \Delta$.

Let us compute now the complexity of the above procedure for obtaining $\beta_{ij}(x_i, x_j)$ for each value of the pair (x_i, x_j) . The computation of y_l and z_l has only to be carried out for at most $|P(i)| + |P(j)|$ distinct nets. So, if $S_{\max} = \max_{l \in [1,L]} \{|S_l|\}$ denotes the maximum cardinality of a net and $P_{\max} = \max_{i \in I_1 \cup I_2} \{|P(i)|\}$ denotes the maximum number of pins in any cell, then the complexity is $\mathcal{O}(P_{\max} \times S_{\max})$. This is also the complexity for obtaining all three values $\beta_{ij}(0,1)$, $\beta_{ij}(1,0)$ and $\beta_{ij}(1,1)$ for each value of α. This computation will have to be repeated for all possible values of α, i.e. at most $2 \times N_P = 2 \sum_{i \in I_1 \cup I_2} |P(i)|$ times, leading to a complexity $\mathcal{O}(N_P \times P_{\max} \times S_{\max})$.

Once the β_{ij} values have been determined for some given $\alpha \in A$, we can readily obtain the formal expression of $\delta(\alpha, x, \dots, x_n)$ as:

$$
\begin{aligned}
\delta(\alpha, x, \dots, x_n) = {} & \delta(\alpha, 0, 0, \dots, 0) \\
& +x_i \times [\beta_{ij}(1,0) - \delta(\alpha, 0, 0, \dots, 0)] \\
& +x_j \times [\beta_{ij}(0,1) - \delta(\alpha, 0, \dots, 0)] \\
& +x_i x_j [\beta_{ij}(1,1) - \beta_{ij}(1,0) - \beta_{ij}(0,1) + \delta(\alpha, 0, 0, \dots, 0)].
\end{aligned} \tag{4}
$$

Therefore, the channel density defined in (2) as a pseudo-Boolean function of the n boolean variables $(x_1, \dots, x_n)$ is the pointwise maximum of $|A|$ quadratic pseudo-Boolean functions in at most 2 variables given by (4).

To make notation simpler, each value of α in A will be given an index number k, k ranging from 1 to $K = |A|$. For each k corresponding to some $\alpha \in A$, we denote by i_k, j_k the indices of the two boolean variables involved in the formal expressionof $\delta(\alpha, x_1, \dots, x_n)$ and let

- u^k be the constant term;
- $v^k_{i_k}$ and $v^k_{j_k}$ be the coefficients of the linear terms;
- $w^k_{i_k j_k}$ be the coefficient of the quadratic term.

With this notation, (4) may be rewritten as:

$$
\begin{aligned}
\delta(\alpha, x_1, x_2, \dots, x_n) &= g^k(x_{i_k} x_{j_k}) \\
&= u^k + v^k_{i_k} x_{i_k} + v^k_{j_k} x_{j_k} + w^k_{i_k j_k} x_{i_k} x_{j_k}
\end{aligned} \tag{5}
$$

and the channel density $d(x_1, \dots, x_n)$ takes the form:

$$
d(x_1, x_2, \dots, x_n) = \max_{k=1,\dots,K} \{g^k(x_{i_k}, x_{i_k})\}. \tag{6}
$$

Let us note that some of the coefficients in (5) may be equal to 0.

Also, remember that the formal expressions (5) and (6) can be obtained from the problem data in $\mathcal{O}(N_P \times P_{\max} \times S_{\max})$ time. In practical applications $P_{\max}$ and $S_{\max}$ are usually small integers ($P_{\max}$ being typically less than 6-8, and $S_{\max}$ very rarely exceeding 10).

Let us conclude this section with an example (shown in Figure 4). This example consists of 2 rows with 5 and 4 cells in each, respectively, and 11 nets, with two pins in each. The original placement cannot be drawn on less than 4 tracks, since e.g. in the horizontal overlap of cells C_2 and C_6, or in the overlap of cells C_3 and C_7 the maximal channel density is 4.

Let us compute now the local density functions $\delta(\alpha, x_1, ..., x_n)$ defined as above. Let L_{ij} denote the interval of the horizontal overlap of cells C_i and C_j, and let us introduce

$$
g^{ij}(x_1, ..., x_n) = \max_{\alpha \in L_{ij}} \delta(\alpha, x_1, ..., x_n).
$$

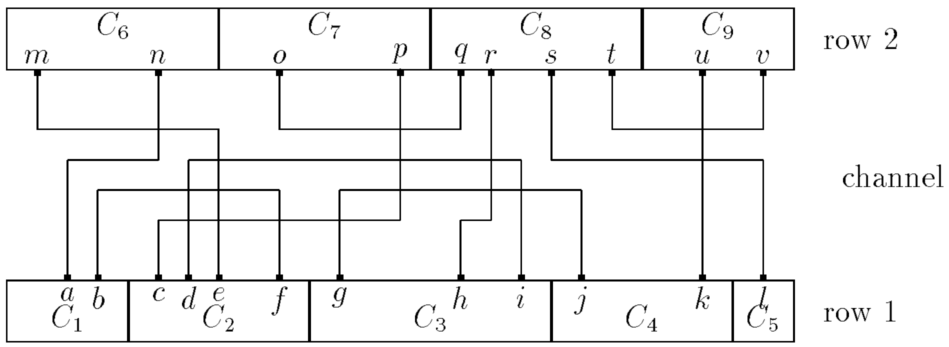

Figure 4: Example for a single channel problem.

Obviously, g^{ij} depends only on x_i and x_j, since $\delta(\alpha, x_1, ..., x_n)$ depends only on x_i and x_j, whenever $\alpha \in L_{ij}$, as we observed earlier, and g^{ij} will only have to be computed for those pairs (i, j) for which the overlap L_{ij} is not empty. The following table lists the resulting functions.

$$\begin{aligned}
&g^{1,6}(x_1, x_6) = 3 - 2x_6 + x_1x_6\\
&g^{2,6}(x_2, x_6) = 4 - x_2 - 2x_2x_6\\
&g^{2,7}(x_2, x_7) = 4 - x_2 - x_2x_7\\
&g^{3,7}(x_3, x_7) = 4 - x_3 - x_7\\
&g^{3,8}(x_3, x_8) = 4 - 2x_3 + 2x_8 + x_3x_8\\
&g^{4,8}(x_4, x_8) = 2 + 2x_4 + 3x_8 - x_4x_8\\
&g^{4,9}(x_4, x_9) = 3 + x_4 - x_9 + x_4x_9\\
&g^{5,9}(x_5, x_9) = 2
\end{aligned}$$

$$\begin{aligned}
d(x_1, x_2, ..., x_9) = \max\{ \ & 2;\ 3 - 2x_6 + x_1x_6;\ 4 - x_2 - 2x_2x_6;\ 4 - x_2 - x_2x_7;\ 4 - x_3 - x_7;\\
& 4 - 2x_3 + 2x_8 + x_3x_8;\ 2 + 2x_4 + 3x_8 - x_4x_8;\ 3 + x_4 - x_9 + x_4x_9\}.
\end{aligned} \tag{7}$$

4 A polynomial algorithm for the one channel case

In this section we show how to exploit the special structure of the pseudo-Boolean function $d(x_1, x_2, \ldots, x_n)$ to be minimized, in order to derive a polynomial algorithm based on a *binary search.*

Let $d(x_1, x_2, \ldots, x_n)$ be given by (6) with $\forall k = 1, \ldots, K$:

$$g^k(x_{i_k}, x_{j_k}) = u^k + v^k_{i_k} x_{i_k} + v^k_{j_k} x_{j_k} + w^k_{i_k j_k} x_{i_k} x_{j_k}.$$

Let

$$z_0 = d(0, 0, \ldots, 0) = \max_{k=1,\ldots,K} \{u^k\}$$

and consider any integer $z \in [0, z_0]$. Since $z_0 \leq L$ (the total number of nets in the problem), it is clear that a polynomial time algorithm for minimizing $d(x_1, \ldots, x_n)$ will be obtained if we are able to solve in polynomial time the problem:

$$\pi(z): \quad \begin{array}{c} \text{does there exist } (x_1, \ldots, x_n) \in \{0,1\}^n \\ \text{such that } d(x_1, \ldots, x_n) \leq z \text{ ?} \end{array}$$

We shall show now that, for any $z \in$, problem $\pi(z)$ can be reformulated as a quadratic Boolean equation (i.e. a 2-SAT problem). To see this, let us consider the following four cases for each of the quadratic functions $g^k(x_{i_k}, x_{j_k})$, $k = 1, ..., K$:

Case 1 If $g^k(0,0)z$, then binary vectors for which $x_{i_k} = 0$ and $x_{j_k} = 0$ must be excluded. This may be achieved by requiring that

$$\overline{x}_{i_k}\overline{x}_{j_k} = 0.$$

Case 2 If $g^k(1,0)z$, then the combination $x_{i_k} = 1$ and $x_{j_k} = 0$ must be excluded, which is equivalent to the equation

$$x_{i_k}\overline{x}_{j_k} = 0.$$

Case 3 If $g^k(0,1)z$, then $x_{i_k} = 0$ and $x_{j_k} = 1$ are not allowed, hence

$$\overline{x}_{i_k}x_{j_k} = 0.$$

Case 4 If $g^k(1,1)z$, then $x_{i_k} = x_{j_k} = 1$ should be excluded, which is equivalent with requiring

$$x_{i_k}x_{j_k} = 0.$$

Of course, for a given $k \in [1, K]$, any combination of the above four cases may arise. For instance, if we have

$$g^k(0,0) \leq z,\ g^k(1,0)z,\ g^k(0,1) \leq z \text{ and } g^k(1,1)z$$

then we will have to satisfy the Boolean equations

$$\overline{x}_{i_k}x_{j_k} = \overline{x}_{i_k}\overline{x}_{j_k} = 0$$

formed by the two equations arising from cases 2 and 4.

Observe that, if all four values of g^k are greater than z, then we get the inconsistent system of equations

$$x_{i_k} x_{j_k} = \overline{x}_{i_k} x_{j_k} = x_{i_k} \overline{x}_{j_k} = \overline{x}_{i_k} \overline{x}_{j_k} = 0.$$

In such a situation, it will immediately be clear that the current value of z is not feasible for d, but it is a lower bound on the optimal value, indicating thus that a larger value for z has to be tested for.

We will denote by F_z the system of Boolean equations obtained as above for $k = 1, 2, \dots, K$. It is readily seen that

Proposition 1 *For any integral value $z \in [0, z_0]$ the answer to problem $\pi(z)$ is YES if and only if F_z has a solution $x_1^*, \dots, x_n^*$.*

Since F_z is a system of quadratic Boolean equations, and since $2 - SAT$ is solvable in linear time (see e.g. [1]), we have

Proposition 2 *The size of F_z (i.e. the total number of literals appearing in the equations) is $\mathcal{O}(n)$. Therefore F_z can be solved in linear time $\mathcal{O}(n)$.*

Proof: If we represent the cells in the upper row (resp. in the lower row) as a set of disjoint consecutive open intervals of the real line, it is easy to see that the only pairs of variables involved in clauses of F_z correspond to the edges of the *intersection graph* (see e.g. [5]) of this family of intervals. The latter is bipartite without cycles (note that it may not be connected). Hence the intersection graph has at most $n-1$ edges. Therefore, there are at most $n-1$ pairs of variables involved in F_z. Since for each pair of variables, there may be at most 4 terms, we can conclude that the size of F_z is $\mathcal{O}(n)$.

By using the well-known linear time algorithm described in ASPVALL, PLASS & TARJAN (1979), the solution to F_z can be obtained in time $\mathcal{O}(n)$ (linear in the number of cells).

From this we can easily deduce an efficient polynomial algorithm to solve the minimum channel density problem. Indeed, the optimal value z^* (and a corresponding solution) can be obtained by applying binary search, as follows:

BINARY SEARCH TO FIND AN OPTIMAL CHANNEL DENSITY SOLUTION

(a) Set $z_{\min} \leftarrow 0 \quad z_{\max} \leftarrow z^0 \quad x^* \leftarrow (0, 0, \dots, 0)$

(b) If $z_{\min} - z_{\max} \leq 1$, STOP. Output $z^* = z_{\max}$ and $x^* = x$.

(c) If $z_{\max} 1 + z_{\min}$, then let $z = \lfloor \frac{z_{\min} + z_{\max}}{2} \rfloor$.

(d) Construct the quadratic Boolean equations F_z and find a solution or conclude that there is no such solution (in linear time, see [1]).

(e) If F_z has no solution, then set $z_{\min} \leftarrow z$ and return to **(b)**.

(f) If $F_z$has a solution, $x = (x_1, \dots, x_n)$ then set $z_{\max} \leftarrow z$ and $x^* \leftarrow x$, and return to **(b)**.

Since $z_0 \leq L$ (the total number of nets in the problem) it is easily seen that the number of iterations necessary to obtain z^* is $\mathcal{O}(\log_2 L)$. Thus we obtain:

Theorem 1 *Minimizing channel density in the one channel case can be carried out in polynomial time $\mathcal{O}(N_P \log_2 L)$, where L is the total number of nets ("equipotentials") and $N_P = \sum_{i \in I_1 \cup I_2} |P(i)|$ is the total number of pins.*

Proof: For each value of z, constructing the quadratic Boolean equations F_z requires $\mathcal{O}(K)$ elementary operations, and since $K \leq 2\ N_P$ the complexity is $\mathcal{O}(N_P)$. Solving then F_z requires $\mathcal{O}(n)$ time. Therefore, for each value of z, the complexity is $\mathcal{O}(N_P + n) = \mathcal{O}(N_P)$, since $n \leq N_P$ (each cell has at least one pin). The result follows, since there are at most $\log_2 L$ values to be tested for.

Let us return now to the example given in Figure 4. It is easy to see that the density function (7) of this example can also be written as

$$\begin{aligned} d(x_1, x_2, ..., x_9) &= \max\{ \ 6\overline{x}_3 x_8, 6x_4 x_8, 5x_3 x_8, 5\overline{x}_4 x_8, 4\overline{x}_2, 4\overline{x}_3\overline{x}_7, 4\overline{x}_3\overline{x}_8, 4x_4, \\ &\qquad 3\overline{x}_6, 3x_2\overline{x}_7, 3\overline{x}_3 x_7, 3x_3\overline{x}_7, 3\overline{x}_4\overline{x}_9, 2\}. \end{aligned} \tag{8}$$

Using expression (8) is advantegeous in our problem, since, e.g., if we want to test whether $d(x_1, ..., x_n)$ can take a value smaller than 5, all terms in (8) having coefficients ≥ 5 must be cancelled. This provides us directly with the desired 2-SAT formulation.

In order to find the optimal cell flipping, let us carry out now the dichotomic search of this section.

We can see from (8) that $d(x_1, ..., x_9) \leq z_{max} = 6$, and also that $d(x_1, ..., x_9) z_{min} = 1$.

Therefore, let us start with $z = \lfloor \frac{1+6}{2} \rfloor = 3$. To decide the feasibility of $d(x_1, ..., x_n) \leq z$ we have to find a binary assignment which cancells all terms of $d(x_1, ..., x_9)$ with coefficients greater than $z = 3$. The corresponding Boolean equation F_z is as follows:

$$\overline{x}_3 x_8 \vee x_4 x_8, \vee x_3 x_8 \vee \overline{x}_4 x_8 \vee \overline{x}_2 \vee \overline{x}_3 \overline{x}_7 \vee \overline{x}_3 \overline{x}_8 \vee x_4 = 0.$$

This equation is consistent ($x_2 = x_3 = 1$, $x_4 = x_8 = 0$ satisfies all terms), thus we can set $z_{max} = 3$, and hence $z = \lfloor \frac{3+1}{2} \rfloor = 2$ in the next iteration. The corresponding Boolean equation F_z

$$\overline{x}_3 x_8 \vee x_4 x_8, \vee x_3 x_8 \vee \overline{x}_4 x_8 \vee \overline{x}_2 \vee \overline{x}_3 \overline{x}_7 \vee \overline{x}_3 \overline{x}_8 \vee x_4 \vee \overline{x}_6 \vee x_2 \overline{x}_7 \vee \overline{x}_3 x_7 \vee x_3 \overline{x}_7 \vee \overline{x}_4 \overline{x}_9 = 0.$$

is again consistent, e.g. $x^* = (0, 1, 1, 0, 0, 1, 1, 0, 1)$ is a solution. Hence we get $z_{max} = 2$. Since $z_{max} - z_{min} \leq 1$, we can stop, and conclude that $z^* = 2$ is an optimal solution, and that x^* is an optimal flipping assignment. Figure 5 below depicts the same channel after the optimal flipping, i.e. after flipping cells C_2, C_3, C_6, C_7 and C_9. The optimal channel density is 2.

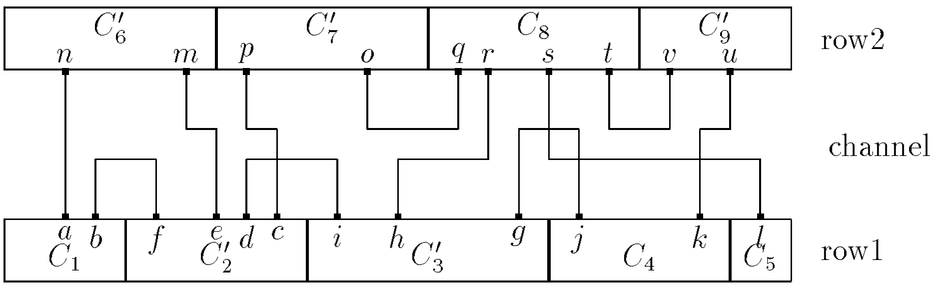

Figure 5: Optimal flipping of example of Figure 5.

5 The multichannel case

We now extend our analysis to the case where there are Q channels formed by $Q + 1$ distinct rows of cells.

In this case, nets may connect terminals which are not located on the sides of a single channel. Figure 6 shows an example where there are 4 rows with 3 channels and a net connecting terminals a and b in row 1, terminal c in row 2 and terminal d in row 4. This net traverses the intermediate rows 2 and 3 at places (called *feedthroughs*) which have already been determined in the placement and the global routing phases.

Since the locations of the feedthroughs are known, as far as the cell flipping problem is concerned, they can be represented as new terminals attached to the cells which are traversed.

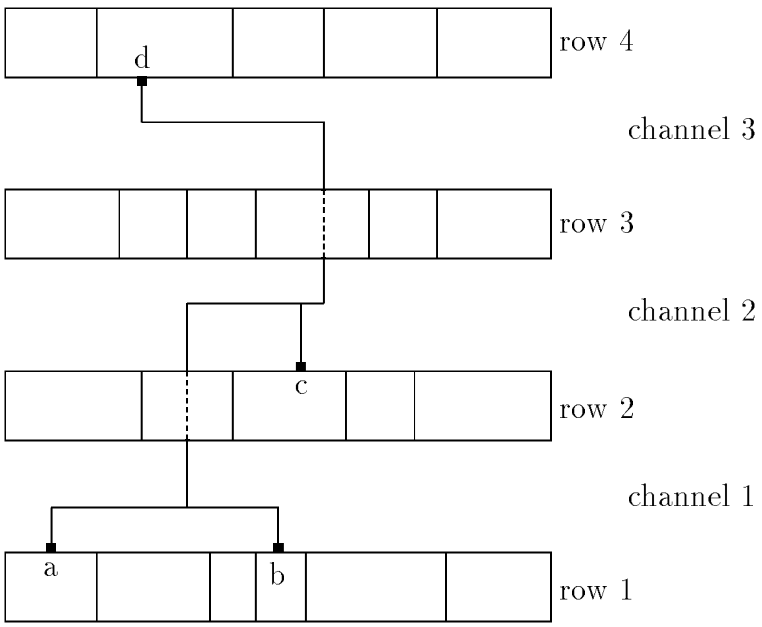

Figure 6: An example illustrating the multichannel case and a net connecting terminals on three different rows. Feedthroughs are indicated in dashed lines.

More precisely, each feedthough used by a net to traverse a row r is replaced by two new terminals, one on each side of the cell. The original net is replaced by two distinct nets, one on each side of row r. In that way, it is seen that it is not restrictive to assume that each of the nets in the given cell flipping problem is located in one single channel and connects only terminals located on one or both sides of that channel (see figure 7).

Now, for each channel $q = 1, \ldots, Q$ let us denote by $d_q(x_1, \ldots, x_n)$ the pseudo-Boolean function representing the density of channel q in terms of all the variables $x_1, \ldots, x_n$ of the problem (here n is equal to the total number of cells appearing in all the rows). An expression for d_q is:

$$d_q(x_1, \ldots, x_n) = \max_{k \in J_q} \{ g_q^k(x_{i_k} x_{j_k}) \}$$

where J_q is a set of indices corresponding to channel q and $g_q^k(x_{i_k} x_{j_k})$ is a quadratic pseudo-Boolean function in the two variables $x_{i_k} x_{j_k}$.

Since the total surface of the design is now proportional to the sum of the widths of the channels, the objective of the cell flipping problem is to minimize the *sum of the channel densities*, in other words:

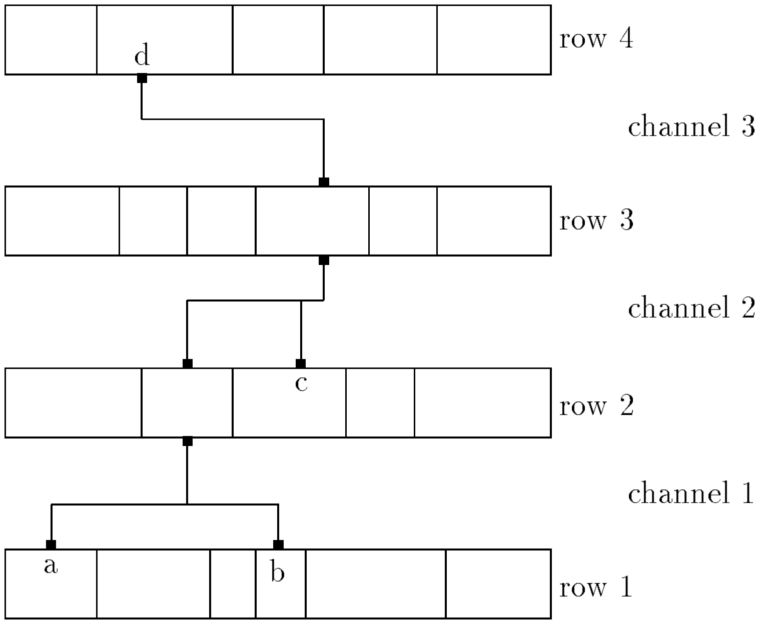

Figure 7: The cell flipping problem equivalent to the one on Figure 5 obtained by replacing each feedthrough by two new terminals and splitting the net into three (one for each channel where the original net had terminals).

$$\text{Minimize} \sum_{q=1}^{Q} d_q(x_1, \ldots, x_n). \tag{9}$$

We note that the problems in the various channels cannot be solved separately since the variables x_i corresponding to cells in one intermediate row r appear in both expressions of $d_r(x_1, \ldots, x_n)$ and $d_{r-1}(x_1, \ldots, x_n)$.

Let $z_1^0 z_2^0, \ldots, z_Q^0$ denote the channel densities of the various channels in the initial configuration for which $x_1 = x_2 = \cdots = x_n = 0$. If we denote by $L_1, L_2, \ldots, L_Q$ the number of nets in channels $1, 2, \ldots, Q$, and

$$L_{\max} = \max_{q=1,\ldots,Q} \{L_q\},$$

we have

$$z_1^0 \leq L_1, \; z_2^0 \leq L_2, \; \ldots, \; z_Q^0 \leq L_Q. \tag{10}$$

For $q = 1, \ldots, Q$, let us denote by $\hat{z}_q$ the minimum width of channel q determined by the algorithm of section 4. It is the minimum value for z_q such that the quadratic Boolean equation $F^q_{\hat{z}_q}$ is consistent. Here F^q_z denotes the system of quadratic Boolean equations expressing the condition that the density of channel q is not more than z.

Then problem (9) reduces to the search for Q integer values $z_1, z_2, \ldots, z_Q$ such that:

(i) $\forall q = 1, \ldots, Q$: $\hat{z}_q \leq z_q \leq L_q$.

(ii) There exists a binary vector $\hat{x}$ which solves simultaneously the equations $F^q_{z_q}$, $q = 1, ..., Q$.

(iii) $\sum_{q=1}^{Q} z_q$ is minimized.

If we enumerate all possible values for $(z_1, z_2, \ldots, z_Q)$ satisfying (i), each time checking the feasibility of (ii) by solving the corresponding 2-SAT equation, then we get the following complexity result.

Theorem 2 *The complexity of solving the optimal cell flipping problem for a total number of terminals N_P, Q channels and L nets is*

$$\mathcal{O}\left(N_P \left(\frac{L}{Q}\right)^Q\right).$$

Hence the problem is polynomially solvable for fixed Q.

Proof: For each value of $z_1, z_2, \ldots, z_Q$, constructing all the quadratic Boolean equations $F^q_{z_q}$ takes $\mathcal{O}(N_P)$ time. Also, since the total number of terms in these equations for $q = 1, ..., Q$ is $\mathcal{O}(n)$, the complexity of solving (ii) is $\mathcal{O}(n)$. Therefore, for each combination $(z_1, z_2, \ldots, z_Q)$ to be checked, the computing time is $\mathcal{O}(N_P)$. In view of (10), the total number of combinations to examine is bounded by

$$L_1 \times L_2 \times \cdots \times L_Q, \tag{11}$$

and since $L_1 + L_2 + \cdots + L_Q = L$, the quantity in (11) cannot be greater than $\left(\frac{L}{Q}\right)^Q$. The result then follows.

6 Improved Lower Bounds and a Heuristic Algorithm

We now describe methods that both improve efficiency in computing the initial lower bounds of each channel's density and provide an initial set of channel densities that closely approximate the optimal densities. The lower bounds are derived without extra computational costs, while the heuristic densities are obtained through the solution of a mixed-integer program.

Recall that, for each channel $q = 1, \ldots, Q$, $d_q(x_1, \ldots, x_n)$ is the pseudo-Boolean function representing the density of channel q, and can be written as

$$d_q(x_1, \ldots, x_n) = \max_{k \in J_q} g_q^k(x_{i_k}, x_{j_k}),$$

where J_q is the set of indices of variables corresponding to channel Q and $g_q^k(x_{i_k}, x_{j_k})$ is a quadratic pseudo-Boolean function in the two variables x_{i_k} and x_{j_k}. Hence, the problem we wish to solve is

$$\text{Minimize} \sum_{q=1}^{Q} d_q(x_1, \ldots, x_n).$$

There are many different methods for obtaining lower bounds on $d_q(x_1, \ldots, x_n)$. For example, one can solve the single channel problem for each q, thus providing the lower bounds $\hat{z}_q$ (see Section 5); however, this yields an $O(nQ)$ algorithm. A much more efficient way of deriving good lower bounds is as follows: for each $k \in J_q$, let

$$LB_q^k = \min_{x_{i_k}, x_{j_k} \in \{0,1\}} g_q^k(x_{i_k}, x_{j_k})$$

and

$$LB_q = \max_{k \in J_q} LB_q^k.$$

This lower bound can be obtained in-line while each $g_q^k(x_{i_k}, x_{j_k})$ is being computed. This implies that no extra time is needed to compute these bounds.

In order to obtain a good approximation of the density of each channel, more work is required. To this end, we shall construct linear upper bounds for each $g_q^k(x_{i_k}, x_{j_k})$ and then find a series of cell flippings that minimize the total sum of the upper bounds. In this way, we generate feasible channel densities; however, note that the densities obtained are not guaranteed to be upper bounds of the optimal individual channel densities.

Recall that, given a quadratic pseudo-Boolean function $f(x_1, \ldots, x_n)$, a linear function $l(x_1, \ldots, x_n)$ can be constructed so that $l(x_1, \ldots, x_n) \geq f(x_1, \ldots, x_n)$ for all $0 - 1$ vectors $(x_1, \ldots, x_n)$. Linear functions constructed using the methodology of [3] are known as *roofs*. A simple example of a roof $l(\mathbf{x})$ can be obtained by replacing each $c_{ij}x_ix_j$ term, with $c_{ij} > 0$, by $(c_{ij}/2)(x_i + x_j)$ and each $c_{ij}x_ix_j$ term, with $c_{ij} < 0$, by $(c_{ij}/2)(x_i + x_j - 1)$. We shall call such linear functions *half-roofs*.

For each $k \in J_q$, let $l_q^k(x_{i_k}, x_{j_k})$ be the half-roof of $g_q^k(x_{i_k}, x_{j_k})$. Notice that, since each $g_q^k(x_{i_k}, x_{j_k})$ has at most one quadratic term, $l_q^k(x_{i_k}, x_{j_k})$ can be easily obtained. Now, we wish to find a λ_q such that

$$\lambda_q \geq l_q^k(x_{i_k}, x_{j_k})$$

for each $k \in J_q$ and every $(x_{i_k}, x_{j_k}) \in \{0,1\}^2$. This leads to the following mixed-integer program:

$$\text{Minimize} \quad \sum_{q=1}^{Q} \lambda_q \tag{12}$$

subject to

$$\lambda_q \geq l_q^k(x_{i_k}, x_{j_k}) \quad k \in J_q, \quad q = 1, \ldots, Q \tag{13}$$

$$x_i \in \{0,1\} \qquad i = 1, \ldots, n. \tag{14}$$

Now, let $UB_q = \lfloor \lambda_q \rfloor$. It is easy to see that $\sum_{q=1}^{Q} UB_q$ is an upper bound on the optimal channel density. Let us remark that, if we would try to relax the integrality constraints (14), and solve the linear relaxation of (12) - (14), the corresponding λ_q values may not be valid upper bounds on $d_q(x_1, \ldots, x_n)$, for each q.

Since the integer program (12) - (14) may have a large number of variables and constraints, it may not be possible to solve it in a reasonable amount of time. One way to improve upon the formulation is to restrict the number of rows and variables as follows: if LB_q is a lower bound for $d_q(x_1, \ldots, x_n)$, let

$$\hat{J}_q = \{k \in J_q : \exists (x_{i_k}, x_{j_k}) \in \{0,1\}^2 \text{ such that } g_q^k(x_{i_k}, x_{j_k}) > LB_q\}.$$

In other words, $\hat{J}_q$ represents those quadratic functions that can actually affect the density of channel q. All other functions have values less than or equal to the known lower bound. Let us thus consider the following mixed-integer program:

$$\text{Minimize} \quad \sum_{q=1}^{Q} \lambda_q \tag{15}$$

subject to

$$\lambda_q \geq l_q^k(x_{i_k}, x_{j_k}) \quad k \in \hat{J}_q, \quad q = 1, \ldots, Q \tag{16}$$

$$\lambda_q \geq LB_q \tag{17}$$

$$x_i \in \{0,1\} \qquad i = 1, \ldots, n. \tag{18}$$

Clearly problem (15) - (18) yields the same optimal values of λ_q as problem (12)-(14), but with much fewer variables and constraints, as shown from the computational results below. It follows that, in our heuristic solution, we may give those variables not appearing in (15) - (18) a value of 0, since they have no affect on the solution.

7 Computational Results

In this section we describe some computational results obtained by implementing our algorithm on some sample data. All data was obtained from the Architecture team, MASI laboratory, University Pierre et Marie Curie (Paris VI), Paris, France, and comes from the placement of cells from acutal VLSI chips. Table 1 gives the sizes of the problems we solved as well as the time, in CPU seconds, it took our algorithm to read in the data.

Problem	Channels	Cells	Pins	Nets	Input Times (seconds)
scman	4	188	591	187	0.23
amd	9	528	2056	807	0.98
scseq	11	1855	5006	2136	5.07
scexec	14	2540	7088	3186	8.02
scage	111	56258	120358	60167	446.87
scmul	139	65693	142039	70713	495.43
ptx	24	5866	16920	7630	25.99

Table 1: Sizes of Problems

To solve these problems, we implemented in C++ a revised version of the enumeration algorithm described in Section 5. We enumerate over all channel densities z_q in the interval $[LB_q, L_q]$, where L_q was defined in Section 5 and LB_q was defined in Section 6. For each set of channel densities z_q, we check its feasibility by solving an appropriate 2-SAT problem. If a feasible set of channel densities has total density lower than our current best solution, we save it.

Problem	Number of Variables	Number of Constraints	Solution Times (CPU seconds)
scman	16	15	< 0.01
amd	42	44	< 0.01
scseq	28	36	< 0.01
scexec	53	44	0.01
scage	23	30	0.01
scmul	47	61	0.01
ptx	94	118	0.01

Table 2: Sizes of Mixed Integer Programs

We incorporate specific improvements to this total enumeration scheme in order to decrease the total number of channel densities we need to consider. For example, when a full enumeration is done, the densities of all channels but one are fixed, and the remaining channel's density is enumerated. Once we obtain a density where the 2-SAT problem is no

Problem	Solutions		
	Default (Without Cell Flipping)	Heuristic	Optimal
scman	51	48	48
amd	143	138	138
scseq	421	418	418
scexec	628	622	622
scage	7772	7768	7768
scmul	9417	9407	9407
ptx	1168	1158	1157

Table 3: Solutions of Cell Flipping Problems

longer consistent, we need not examine any of the remaining densities for that channel. This eliminates a portion of the combinations we need to consider. Another change we made to improve the running time of the algorithm is to examine, via the solution of a 2-SAT equation, only those values $(z_1, z_2, \ldots, z_Q)$ whose sum $\sum_{q=1}^{Q} z_q$ is smaller than the current best solution. This can greatly improve the running time of the algorithm since we only do the "bounding" routine at those nodes whose individual channel densities can improve the current solution. We also included our heuristic method, described in Section 6, where we solve the mixed-integer program (12) - (14).

We then ran our enumeration algorithm on a Silicon Graphics MIPs 10000 with 192M memory and 175 Mhz speed. The mixed-integer program was solved using CPLEX 4.0. Sizes of the reduced mixed-integer programs, as well as their computational times (in CPU seconds) are given in Table 2. In Table 3 we show the initial (default) solution, the heuristic solution derived from the mixed-integer program (15) - (18), and the optimal solution for each problem we considered. We see that, in every case, our heuristic solution is either equal to or nearly equal to the optimal channel densities. In addition, our heuristic solution improves upon the default solution (without any cell flippings). However, this solution would not be of importance if it were difficult to obtain; fortunately, this has not been the case. In Table 4 we show the computational results obtained by solving the problems with the exact algorithm using our heuristic solution as an initial starting point. We can see that our exact method works very well; in fact, we use relatively few 2-SAT calculations, even for problem *scmul*. Unfortunately, not all problems can be efficiently solved to optimality. Problem *ptx*, when starting with our heuristic solution, took over 60 CPU hours to solve, requiring over 4.2 billion nodes visited with over 38 million 2-SAT formulations. However, the mixed-integer program yielding our heuristic solution was solved in about 0.01 second, thus yielding an improvement over the default solution very quickly. In addition, from Tables 1, 2, and 4 we can see that the majority of the total time needed to obtain a solution comes from reading in and initializing the problem data.

Let us remark finally that, as Table 3 shows, in all examples we found the exact optimum improving slightly on the implemented chip design. Even though these improvements are

Problem	Time (CPU seconds)	Nodes	2-SAT Formulations
scman	0.02	36	0
amd	0.08	2583	32
scseq	0.18	520	48
scexec	0.08	13823	12
scage	0.45	759	30
scmul	10.63	883833	2882

Table 4: Optimization times and number of nodes in the exact enumeration procedure

not substantial, the computational cost of obtaining them is negligible.

References

[1] Aspvall B., Plass M.F., Tarjan R.E. (1979) "A linear time algorithm for testing the truth of certain quantified boolean formulas". Information Processing Letters 8, pp. 121-123.

[2] Cheng, C.K., Yao, S.Z., and Hu, T.C., "The orientation of modules based on graph decomposition", *IEEE Transactions on Computers* **40** (1991), 774-780.

[3] Hammer, P.L., Hansen, P., and Simeone, B., "Roof duality, complementation and persistency in quadratic $0-1$ optimization", *Math. Programming*, 28 (1984), pp. 121-155.

[4] Her, T.W., Wang, Ting-Chi, and Wong, D.F., Preformance driven channel pin assignment algorithms, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, **14** (1995), pp. 849-857.

[5] Lekkerkerker, C.G. and Boland, J. Ch., "Representation of a finite graph by a set of intervals on the real line", *Fundam. Math.* **51** (1962), 45-64.

[6] Lengauer T. (1990) "Combinatorial algorithms for integrated circuit layout". Wiley-Teubner Series in Computer Science, 697.

[7] La Paugh A.S. (1980) "Algorithms for integrated circuit layout: an analytic approach". Ph.D. Thesis, Dept. Electrical Engineering and Computer Sc., M.I.T., Cambridge, MA, USA.