Simultaneous Transistor Pairing and Placement for CMOS Standard Cells

Ang Lu, Hsueh-Ju Lu, En-Jang Jang, Yu-Po Lin, Chun-Hsiang Hung, Chun-Chih Chuang, Rung-Bin Lin
Computer Science and Engineering
Yuan Ze University
Chung-Li, Taiwan

***Abstract*—This paper presents an integer linear programming approach to transistor placement problem for CMOS standard cells with objectives of minimizing cell width, wiring density, wiring length, diffusion contour roughness, and misalignments of common ploy gates. Our approach considers transistor pairing and transistor placement simultaneously. It can achieve a smaller number of transistor chains than the well-known bipartite approach. About 31% of the 185 cells created by it have smaller widths and no cells whose widths are larger than their handcrafted counterparts.**

Keywords—Transistor placement; transistor pairing; transistor folding; standard cell

I. INTRODUCTION

Regular layouts are essential to advance semiconductor technology beyond 32nm node [1] employing multiple patterning lithography. Layout regularity makes the concept of automatic layout generation more appealing, for it to be used as a way of designing standard cells for production, a way of benchmarking manually designed standard cells, or even a way of exploring design space. Standard cell layout generation consists of a serious of tasks: transistor folding, pairing, placement, and routing. It is difficult to solve all these tasks at once. Transistor folding is a task [2] normally done first, sometimes also taking transistor placement into account. Recently, J. Cortadelia [3] proposed an area-optimal transistor folding approach considering different layout parameters such as parameterized diffusion break, flexible transistor size, etc. Traditional transistor pairing approach was somewhat intuitive. It normally paired a P- and an N-type transistor with a common gate signal together. Transistor placement, also called transistor chaining or transistor ordering, were investigated extensively in the past. Stochastic algorithms such as Simulated Annealing [4] or genetic algorithms [5] were widely used. A stochastic approach could flexibly consider more factors, but it also took more runtime. A deterministic approach could be a graph algorithm [6, 7, 8], an integer linear programming model [3, 9, 10], or a Boolean satisfiability formulation [11, 12]. C. Y. Hwang et al. [7] proposed a fast transistor chaining algorithm by modeling possible diffusion sharing between the transistor pairs as a bipartite graph. Although a solution with a least number of transistors could be found in most of the cases, transistor pairing and ordering were independently solved. The work in [13] extended the bipartite graph approach [7] to also considering transistor pairing. Unfortunately, its algorithmic details were missing. Maziasz and Hayes [6] focused on minimizing cell height and width, but their method could not be applied to a cell with non-dual P- and N-type transistors. Wu et al. [14] presented a 1-D cell generation algorithm that simultaneously minimized 1-D cell area and enhanced the printability. Cell area was minimized by reducing the diffusion gaps, each of which is created between two adjacent transistors of different widths or between two chains.

Integer linear programing (ILP) and Boolean satisfiability (SAT) formulations are suitable for a decision problem like transistor placement. Due to their long runtime, the problem size for ILP or SAT cannot be too large. However, they are still viable for a placement instance of a few dozen of transistors. Gupta and Hayes [9] proposed an ILP model to minimize the width of multi-row transistor placement with dual P- and N-type transistors. In [10], Gupta and Hayes integrated transistor folding into the generation of optimal cell layout based on ILP, but transistor folding and placement were solved independently. The work in [11] proposed a minimum-width transistor placement method for CMOS cells with non-dual P- and N-type transistor using SAT. The work in [12] extended the concept of [11] to multi-row transistor placement. All the above methods have advanced standard cell layout generation in many aspects. Nevertheless, a true simultaneous transistor pairing and placement approach that minimizes cell width, wiring density, wire length, misalignments of common ploy gates, and diffusion contour roughness is yet to be developed.

In this work we formulated the transistor pairing and placement problem as a whole into an ILP model optimizing cell width, poly gate sharing, wire length, wiring density, and diffusion contour roughness. Our approach could obtain a smaller number of transistor chains than the bipartite approach [7]. About 31% of the 185 cells created by it had smaller widths and no cells whose widths were larger than their handcrafted counterparts. We also proposed an approach to reducing ILP runtime without compromising cell width.

The rest of this paper is organized as follows. Section II presents our problem formulation. Section III details our transistor folding approach and ILP formulation. Section IV presents experimental results. Section V draws conclusions.

II. PROBLEM FORMULATION

Below are assumptions made in this work.

- Standard cell layout style employs only two diffusion strips, one for forming P-type transistors on the top and

the other for N-type transistors at the bottom as shown in Fig. 1. Moreover, a poly wire cannot be bended.

- There are no specific restrictions on the structure of a circuit network specified by a SPICE netlist. A circuit topology should not be changed [15]..
- A P- and an N-type transistor on the same column form a transistor pair. Their gates can be connected using a straight poly wire if they have the same gate signal.
- If the numbers of P- and N-type transistors after transistor folding are different, dummy transistors are employed to make them equal.
- An isolation transistor (gate) separates two neighboring transistors where diffusion sharing is not possible.
- The top edges of P-type transistors' diffusions are aligned. So do the bottom edges of N-type transistors' diffusions.

Problem Formulation: *Given a flattened SPICE netlist and the above assumptions, place P- and N-type transistors at their respective rows such that cell width, wiring density, wire length, and diffusion contour roughness are minimized and ploy gate sharing is maximized.*

We approach this problem by performing a series of tasks: transistor folding, transistor pairing, and transistor placement. Differentiating from the majority of previous work, our approach considers transistor pairing and transistor placement simultaneously. The two-row fabric in Fig. 1 enables such a possibility. This definitely increases problem complexity but will simplify transistor folding task and render a solution with a smaller cell width.

III. TRANSISTOR FOLDING, PAIRING AND PLACEMENT

A. Notation

Table I shows some symbols used for denoting constants. P_i or P_x denotes a P-type transistor and N_i or N_x denotes an N-type transistor where $1 < i,\ x < c_{num}$. Tables II and III show some 0/1 and non-0/1 integer variables respectively.

TABLE I. INTEGER CONSTANTS.

symbols	definitions
p_{num}	Number of P-type transistors.
n_{num}	Number of N-type transistors.
c_{num}	Number of columns, it is equal to $max(p_{num}, n_{num})$.
s_{num}	Number of signals.
$pw(i)$†	Width of transistor P_i.
$nw(i)$†	Width of transistor N_i.
M	A very large number.

†$pw(i)$ and $nw(i)$ will be treated as *integer variables* if they denote the sizes of sub-transistors obtained from folding a large-sized transistor.

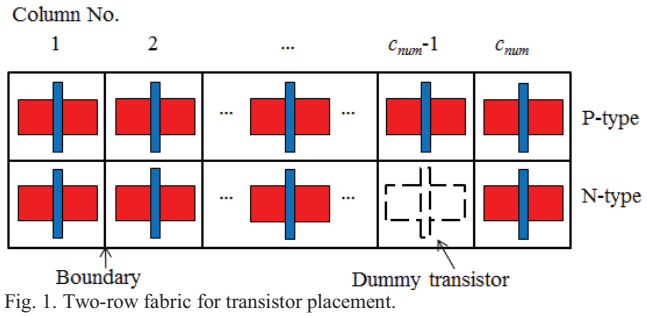

Fig. 1. Two-row fabric for transistor placement.

TABLE II. 0/1 INTEGER VARIABLES.

var	condition that sets a variable to 1
$P_{(i,j)}$	P_i is placed at column j.
$N_{(i,j)}$	N_i is placed at column j.
$O_{P(i)}$	P_i's orientation is S-G-D.
$O_{N(i)}$	N_i's orientation is S-G-D.
$A_{P(j)}$	Two P-type transistors at column j and j+1 can be abutted.
$A_{N(j)}$	Two N-type transistors at column j and j+1 can be abutted.
$G_{P(j)}$	Two P-type transistors at column j and j+1 have different transistor widths.
$G_{N(j)}$	Two N-type transistors at column j and j+1 have different transistor widths.

TABLE III. NON-0/1 INTEGER VARIABLES.

var	definitions
$W_{P(j)}$	Width of a P-type transistor placed at column j.
$W_{N(j)}$	Width of an N-type transistor placed at column j.

B. Transistor Folding

Transistor folding divides a large-sized transistor into several sub-transistors connected in parallel as shown in Fig. 2. The basic issues of transistor folding include the number of sub-transistors being generated and the relative positions with respect to their duals on the other row. These issues have been addressed on the premise that transistor pairs must be formed prior to transistor placement. Such a requirement makes transistor folding nontrivial [2]. If a large-sized transistor is divided into an even number of sub-transistors, the two ends of a sub-chain formed by the sub-transistors will be both either sources (as shown in Fig. 2(b)) or drains. In such a case, the underlying transistor chain might be broken. On the other hand, if a large-sized transistor is divided into an odd number of sub-transistors, one end of the sub-chain is a source and the other end is a drain as shown in Fig. 2(c). In this situation, the transistor chain will remain connected. In our work, we divide a large-sized transistor into a smallest number of sub-transistors. If the number of sub-transistors is even, we add an isolation gate to join the broken sub-chains, or we divide it into an odd number of smaller sub-transistors, i.e., having one more sub-transistor.

One problem yet to be addressed is about the sizes of sub-transistors. In order to minimize diffusion contour roughness,

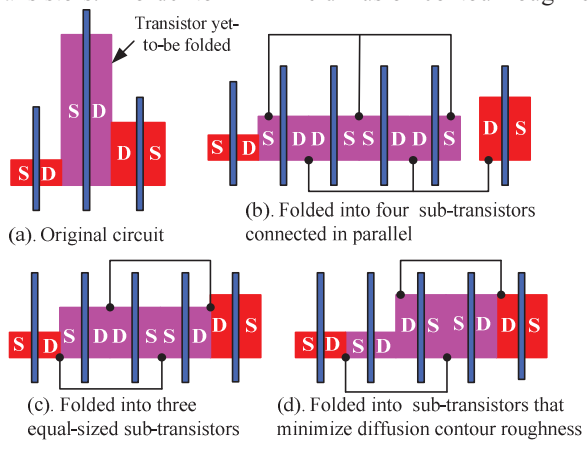

Fig. 2. Folding a large-sized transistor into sub-transistors.

we let our ILP formulation determine the sizes of sub-transistors. Let the width of a large-sized P-type transistor g be denoted by W_g. Supposed it is folded into m sub-transistors numbered from 1 to m. We add $W_g = \sum_{i=1}^{m} pw(i)$ into our ILP formulation where $pw(i)$ is an integer variable denoting the width of the i-th sub-transistor.

C. Transistor Placement

Formulas (1) ~ (4) specify the constraints for placement of P- and N-type transistors.

$$\sum_{i=1}^{c_{num}} P_{(i,j)} = 1, \quad 1 \le j \le c_{num} \tag{1}$$

$$\sum_{j=1}^{c_{num}} P_{(i,j)} = 1, \quad 1 \le i \le c_{num} \tag{2}$$

$$\sum_{i=1}^{c_{num}} N_{(i,j)} = 1, \quad 1 \le j \le c_{num} \tag{3}$$

$$\sum_{j=1}^{c_{num}} N_{(i,j)} = 1, \quad 1 \le i \le c_{num} \tag{4}$$

D. Transistor Pairing

Formulas (1)~(4) enable free transistor pairing, but pairing two transistors P_x and N_y with the same gate signal is highly encouraged, i.e., setting $pair_{(P_x,N_y)} = 1$ in (5).

$$pair_{(P_x,N_y)} = \sum_{j=1}^{c_{num}} P_{(x,j)} N_{(y,j)} \ , \quad \forall\, P_x \text{ and } N_y \text{ with the same gate signal} \tag{5}$$

The 0/1 variable $pair_{(P_x,N_y)}$ is included in the objective function. We prefer $pair_{(P_x,N_y)} = 1$. Nevertheless, this formulation allows us to generate a solution with fewer transistor chains without pairing a P- and an N-type transistor with the same gate signal. Since poly wire cannot bend, upper metal wires should be employed to connect the gate signal. This unique feature renders a solution with a smaller cell width at the expense of larger wire length. This is a good deal because cell area is more important than wire length.

E. Transistor Abutment

Transistor abutment (or source/drain diffusion sharing) determines the number of transistor chains and thus cell width. Formula (6) determines whether an abutment between the two P-type transistors placed at column j and $j+1$ can be formed.

$$B_{(P_x)} = \left\{ P_{x_i} \middle| \ x < x_i \le c_{num},\ SD_{(P_x)} \cap SD_{(P_{x_i})} \ne \Phi \right\},$$

$$A_{P(j)} = \sum_{x=1}^{c_{num}-1} \sum_{x_i,i=1}^{|B_{(P_x)}|} (P_{(x,j)} P_{(x_i,j+1)} O_{(x,x_i)} + P_{(x,j+1)} P_{(x_i,j)} O_{(x_i,x)}) \tag{6}$$

where $SD_{(P_x)}$ is the set of signals connected to the source or drain of P_x, $B_{(P_x)}$ is the set of transistors P_{x_i} that can abut P_x with diffusion sharing for $x < x_i \le c_{num}$, $|B_{(P_x)}|$ is the cardinality of $B_{(P_x)}$, and $O_{(x,x_i)}$ accounts for the orientations of P_x and P_{x_i}. If the source of P_x abuts the source of P_{x_i}, then

$$O_{(x,x_i)} = (1 - O_{P(x)}) O_{P(x_i)}$$

$$O_{(x_i,x)} = (1 - O_{P(x_i)}) O_{P(x)}$$

Formulations for $O_{(x,x_i)}$ and $O_{(x,x_i)}$ of the remaining cases can be similarly obtained. So does the one for $A_{N(j)}$. Hence, the total number of abutments is

$$Abut\# = \sum_{j=1}^{c_{num}-1} Abut(j) \tag{7}$$

where $Abut(j) = A_{P(j)} A_{N(j)}$.

F. Computing Horizontal Wire Length

Let T_k denote the set of transistors connected to signal k. We calculate the horizontal wire length of signal k by finding its right end $Sig_k.R$ and left end $Sig_k.L$ as shown in Fig. 3, where $T_k = \{P_3, N_2, P_1, P_6, N_5\}$. Formula (8) performs such a calculation.

$$Sig_k.len = Sig_k.R - Sig_k.L \tag{8}$$

where $Sig_k.R = \max_{\forall N_y, P_x \in T_k} (S_{P_x}, S_{N_y})$ and $Sig_k.L = \min_{\forall N_y, P_x \in T_k} (S_{P_x}, S_{N_y})$.

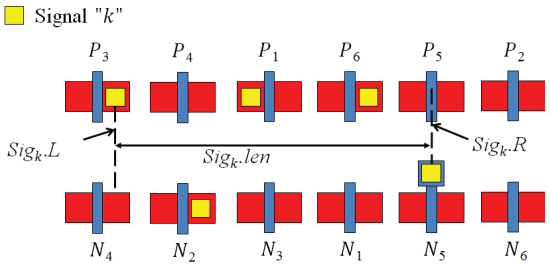

Fig. 3. Estimated horizontal wire length of signal k.

Referring to Fig. 3, we can calculate the position S_{P_x} of signal k connected to any transistor $P_x \in T_k$ at column j using (9).

$$S_{P_x} = \left\{ ofs_{(P_x,k)} + \sum_{j=2}^{c_{num}} 3(j-1) P_{(x,j)} \ \middle| \ P_x \in T_k \right\} \tag{9}$$

For simplicity we assume a transistor's horizontal dimension takes three units of length, one unit respectively for source, gate, and drain. Referring to Fig. 4, $ofs_{(P_x,k)}$ is an offset of the signal's position relative to the left end of the underlying transistor. If signal k is connected to the source of P_x whose orientation is S-G-D, i.e., $O_{P(x)} = 1$, then $ofs_{(P_x,k)} = 0$. If $O_{P(x)} = 0$, then $ofs_{(P_x,k)} = 2$. Formulas (10)~(12) summarize offset value calculations for signal k.

$$ofs_{(P_x,k)} = 2(1 - O_{P(x)}) \quad \textit{for } k \textit{ tapped to a source} \tag{10}$$

$$ofs_{(P_x,k)} = 2O_{P(x)} \quad \textit{for } k \textit{ tapped to a drain} \tag{11}$$

$$ofs_{(P_x,k)} = 1 \quad \textit{for } k \textit{ tapped to a gate} \tag{12}$$

The position S_{N_y} of a signal k connected to $N_y \in T_k$ at column j can be calculated similarly.

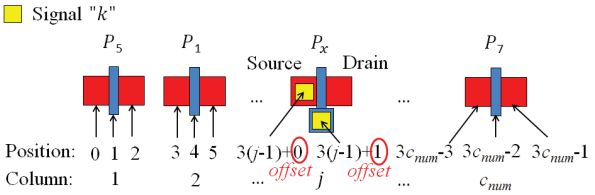

Fig. 4. Offset values for signal k connected to P_x.

The above $Sig_k.len$ is simply an estimated value because its calculation does not account for the influence of abutments on wire length. Formula (13) calculates the exact horizontal wire length of signal k.

$$Sig_k.len = Sig_k.R - Sig_k.L + \sum_{j=1}^{c_{num}-1} LB_{(j,Sig_k)} (1 - 2 \times Abut(j)) \tag{13}$$

where 0/1 variable $LB_{(j,Sig_k)} = 1$ indicates that signal k crosses the boundary between columns j and $j+1$. The term $1 - 2 \times Abut(j)$ equals -1 if there is an abutment between columns j and $j+1$. It equals 1 otherwise. Formula (13) gives exact wire length but is also more computationally intensive. We find that (12) is sufficiently good.

G. Wiring Density

Wiring density is crucial to routability under a cell height constraint. Given a transistor placement shown in Fig. 5, we draw three vertical cut lines at the source, gate and drain of each transistor pair. Wiring density can be obtained by finding a cut line which crosses the maximum number of signals. Let 0/1 variable $Cros_{(n,Sig_k)} = 1$ denote that a cut line n crosses signal k. For example, in Fig. 5 $Cros_{(0,Sig_a)} = 0$, $Cros_{(6,Sig_a)} = 1$, and $Cros_{(3j-2,Sig_b)} = 1$. $Cros_{(n,Sig_k)}$ is given in (14). The two 0/1 variables b_1 and b_2 are both set to 1 if cut line n crosses signal k.

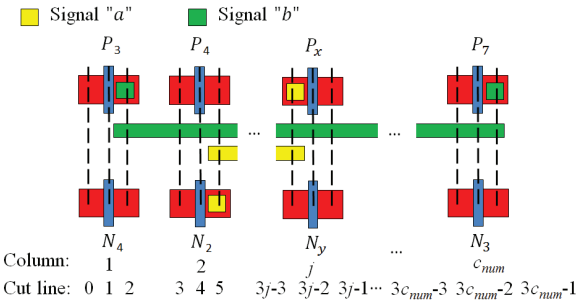

Fig. 5. Counting the number of signals crossing vertical cut lines.

$$
\begin{aligned}
& Cros_{(n,Sig_k)} = b_1 \times b_2 \qquad (14)\\
& M \times b_1 + (Sig_k.L - 1) \geq n \ , \\
& M \times (1 - b_1) + n > (Sig_k.L - 1) \ , \\
& M \times b_2 + n \geq (Sig_k.R + 1) \ , \\
& M \times (1 - b_2) + (Sig_k.R + 1) > n \ , \\
& \qquad 0 \leq n \leq 3c_{num} - 1
\end{aligned}
$$

Wiring density can be obtained by (15).

$$WD = \max_{n=0,\ldots,3c_{num}-1} \sum_{k=1}^{s_{num}} Cros_{(n,Sig_k)} \qquad (15)$$

H. Diffusion Contour Roughness

Diffusion contour roughness will increases the process variation of transistor width. It is defined as (*number of right-angle turns* – 4)/2. The roughness value of the solution in Fig. 2(c) is two whereas it is one in Fig. 2(d). To consider roughness, we employ a 0/1 variable $G_{P(j)}$ ($G_{N(j)}$) to denote whether the two neighboring P-type (N-type) transistors at columns j and $j+1$ have the different diffusion height. Here, we assume that a dummy transistor's diffusion height can be arbitrarily set equal to either the height of its neighbor on the right or the height of its neighbor on the left. In other words, a dummy transistor can have an L-shape diffusion. Since a dummy transistor must be turned off, its width variation does not cause a problem. Hence, an L-shape diffusion on a dummy transistor does not contribute to contour roughness. In Fig. 6 $G_{P(1)} = 1$, $G_{P(2)} = 0$, $G_{N(3)} = 0$ and $G_{N(4)} = 0$ given that N_6 is a dummy transistor.

Let $W_{P(j)}$ denote the width of the P-type transistor placed at column j. Then

$$W_{P(j)} = \sum_{i=1}^{p_{num}} P_{(i,j)} \times pw(i) \ , \quad 1 \leq j \leq c_{num} \qquad (16)$$

$W_{N(j)}$ can be calculated similarly. Let 0/1 variable $eql_{(W_{P(j)},W_{P(j+1)})} = 1$ if $W_{P(j)} = W_{P(j+1)}$. Formula (17) determines its value.

$$
\begin{aligned}
& M \times (1 - eql_{(W_{P(j)},W_{P(j+1)})}) - dif_{(W_{P(j)},W_{P(j+1)})} \geq 0 \ , \\
& M \times eql_{(W_{P(j)},W_{P(j+1)})} + dif_{(W_{P(j)},W_{P(j+1)})} \geq 1 \ , \qquad (17)
\end{aligned}
$$

where $dif_{(W_{P(j)},W_{P(j+1)})} = |W_{P(j)} - W_{P(j+1)}|$ and $1 \leq j \leq c_{num}$.

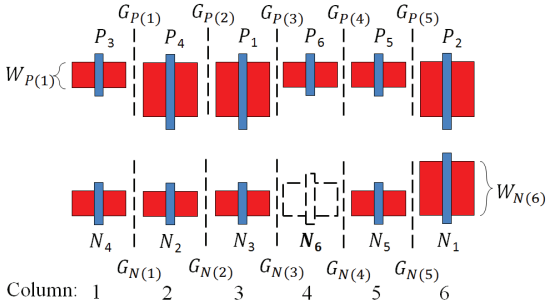

Fig. 6. Variables used to handle diffusion roughness.

To properly handle dummy transistors, we set the width (diffusion height) of a dummy transistor to -1. Hence, we have $dif_{(W_{P(j)},W_{P(j+1)})} > W_{P(j)} + W_{P(j+1)}$ if any P-type transistor at column j or $j+1$ is a dummy transistor. A 0/1 variable $Dum_{p(j,j+1)}$ in (18) denotes such a situation.

$$
\begin{aligned}
& M \times Dum_{p(j,j+1)} - dif_{(W_{P(j)},W_{P(j+1)})} + (W_{P(j)} + W_{P(j+1)}) \geq 0 \\
& M \times (1 - Dum_{p(j,j+1)}) + dif_{(W_{P(j)},W_{P(j+1)})} \\
& \qquad -(W_{P(j)} + W_{P(j+1)}) \geq 1 \ , 1 \leq j \leq c_{num}. \qquad (18)
\end{aligned}
$$

Clearly, if $dif_{(W_{P(j)},W_{P(j+1)})} > W_{P(j)} + W_{P(j+1)}$, $Dum_{p(j,j+1)}$ is forced to one. Then, $G_{P(j)}$ can be calculated using (19).

$$
\begin{aligned}
& G_{P(j)} = (1 - Dum_{p(j,j+1)}) \left(1 - eql_{(W_{P(j)},W_{P(j+1)})}\right), \\
& \qquad 1 \leq j \leq c_{num} \qquad (19)
\end{aligned}
$$

Similarly, we can obtain formulas for calculating $G_{N(j)}$. The above formulation is also valid when two or more dummy transistors are placed next to each other.

I. Complete Formulation and Optimization Approach

The whole ILP formation is given in (20). The objective function consists of three terms weighed by coefficients C_1, C_2 and C_3. The first term counts the number of transistor pairs, each of which has the same signal connected to the gates of its two transistors. *GS* denotes the set of transistor pairs of this sort. The second term denotes the total wire length. The third term is the diffusion contour roughness.

$$
\begin{aligned}
& \text{Minimize: } -C_1 \left(\sum_{\forall (P_x,N_y) \in GS} pair_{(P_x,N_y)} \right) \qquad (20)\\
& + C_2 \left(\sum_{k=1}^{s_{num}} Sig_k.len \right) + C_3 \left(\sum_{j=1}^{c_{num}-1} G_{P(j)} + G_{N(j)} \right)
\end{aligned}
$$

Subject to (1), (2), (3), (4), (21), and (22).

In order to reduce the ILP formulation complexity, number of transistor chains and wiring density are formulated

as constraints. Hence, we have (21) where B_c is a lower bound on chain number estimated by the method in [18]. Our approach could obtain an optimal solution with the number of chains smaller than B_c because it renders a higher degree of freedom for transistor pairing. $Abut\#$ is given by (7).

$$chain_{num} = c_{num} - Abut\# \leq B_c \quad (21)$$

Wiring density is also specified as a constraint given in (22). This is typical for designing a standard cell. WD is given by (15). B_d is a user specified value limited by cell height.

$$WD \leq B_d \quad (22)$$

We start with the lower bound B_c on chain number. If an ILP solver can obtain a feasible solution under this lower bound, we decrease B_c by 1. This process is repeated until B_c cannot be reduced further. On the other hand, we increase B_c by 1 and repeat this process until a solution is obtained. Note that nonlinear terms in our formulation can be easily linearized [17, 19].

J. Simplified ILP Formulation

Computation can be speeded up by either grouping serial/parallel transistors or pre-pairing P- and N-type transistors in large-drive inverters and buffers. The latter is trivial. Hence, we will focus on grouping transistors. Here, we only state some lemmas without providing a proof.

Lemma 1. Grouping an odd number of parallel transistors does not influence the minimum number of transistor chains given by an optimal solution.

Lemma 2. Grouping an even number of parallel transistors may influence the minimum number of transistor chain given by an optimal solution unless there exists a transistor which can abut the transistor at one of the two ends of the sub-chain.

Lemma 3. Grouping serial transistors does not influence the number of the chains given by an optimal solution..

IV. Experimental Results

Experiments are run on an Intel Xeon 2.13GHz CPUs (32 cores) with 256 GBs memory. We use *Cplex* to solve our ILP formulations. Our ILP approach has been part of an automatic layout generator used to create CMOS standard cells. We use it to perform transistor pairing and placement of 185 logic cells. These logic cells include commonly used ones with driving capabilities of 1X, 2X, and 4X. We also manually perform transistor pairing and placement of these cells. The auto generated cells and the handcrafted cells use the same number of routing tracks, poly pitch, M1 pitch and M2 pitch. Bends on poly, M1, and M2 wires are not permitted. All wires are on the grids defined by their pitches respectively. Isolation transistors (gates) are inserted between two adjacent chains so that only one diffusion strip is used per transistor type. The way of setting the diffusion height of a dummy transistor also applies to setting the diffusion height of an isolation transistor. The coefficients in (20) are set to $C_1 : C_3 : C_2 = 500 : 25 : 1$. This encourages more gate alignments under chain number and wiring density constraints. An industrial 90nm process technology is used for layout design.

Table IV makes a comparison of cell widths. The unit for cell width is one poly pitch. $A - H > 0$ means the width of an auto generated cell (A) is larger than that of its handcrafted counterpart (H). No auto generated cells are wider than their handcrafted counterparts and 30.8% of auto generated cells are narrower than their handcrafted counterparts. Since our approach is able to increase diffusion abutments by allowing misalignment of P- and N-type transistors both connected to the same gate signal, we can obtain narrower cells. Clearly, this cannot be easily done manually. This situation occurs in cells like AOX222X1, AO22X1, AOI222X1, etc. For example, Fig. 7(a) is a handcrafted layout which uses 7 poly tracks (6 transistor pairs each having same poly gate signal and 1 pair of isolation transistors, excluding the two on the cell boundary). Fig. 7(b) is an auto generated layout (routed by an in-house tool) which uses 6 poly tracks (4 transistor pairs each having same poly gate signal and 2 transistor pairs with misaligned poly gates). The expense for width reduction is a larger wire length as shown in Fig. 7(b).

TABLE IV. COMPARISON OF CELL WIDTHS.

	***A-H* > 0**	***A-H* = 0**	***A-H* = -1**	***A-H* = -2**
# of cells	0	128	52	5

TABLE V. COMPARISON OF HORIZONTAL WIRE SPAN.

	A* > *H	***A* = *H***	***A* < *H***
# of cells	33	75	80

Table V shows that the horizontal wire span of 155 auto generated cells are not larger than that of their counterparts. The horizontal wire span of a net equals the column number of the right most terminal minus the column number of the left most terminal of a net. Two probable reasons can explain why some of auto generated cells have larger horizontal wire spans. First, handcrafting is allowed to permute serially connected transistors or transistor groups [15] and hence may obtain a solution with shorter horizontal wire span. Second, our program can separate a P- and N-type transistor with same poly gate signal in order to obtain a solution with fewer transistor chains, but we need an extra horizontal wire to connect the separated poly gates.

Table VI shows speedup for dealing with a large circuit. *#c* denotes the number of chains. *Feas* denotes the runtime for finding the first feasible solution. *Opt* denotes the runtime for finding an optimal solution. *#T* denotes the number of transistors in a cell after transistor folding. As one can see, speedup is very significant.

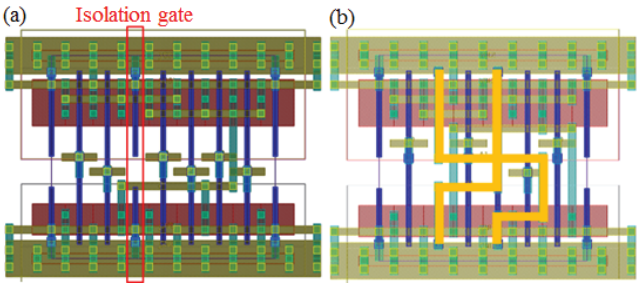

Fig. 7. Tradeoff between wire length and transistor chains of AOI222X1, (a) with handcrafted layout and (b) with auto generated layout.

Table VII compares the number of transistor chains obtained by our ILP approach with that obtained by the bipartite graph method in [7]. Our approach achieves a smaller number of chains for four circuits. Transistor pairing

for the circuits in Fig. 8 is difficult. Our approach considers transistor pairing and placement at the same time so that it can obtain a better solution. Although it takes more runtime, it is still viable. It minimizes not only the number of chains, but also wiring density, wire length, diffusion contour roughness, and poly gate misalignments, which are not done in [7].

TABLE VI. RUNTIME (IN SEC).

		Original formulation			Simplified formulation		
Cell name	*#T*	*Feas*	*Opt*	*#c*	*Feas*	*Opt*	*#c*
INVX32	82	794	10072	1	5	5	1
BUFX32	86	830	11620	1	41	42	1
DFFRBX1	32	132	10230	3	87	693	3
DFFRBX2	36	94	11704	2	110	1596	2
DFFRBX4	42	92	15360	2	92	1401	2
FA1X1	28	140	1886	2	21	100	2
FA1X2	30	101	2169	1	12	49	1
FA1X4	38	1184	16280	1	34	109	1

TABLE VII. A COMPARISON OF CHAIN NUMBER BETWEEN OURS AND [7].

		[7]		Ours	
Circuit	#T	#c	runtime (sec)	#c	runtime (sec)
[7, Fig. 1]	10	1	< 1	1	1
[16, p.198]	12	2	< 1	2	2
[16, p.334]	24	2	< 1	2	17
[7, Fig. 5(a)]	28	3	< 1	2	43
Fig. 8(a)	16	3	< 1	2	25
Fig. 8(b)	12	3	< 1	2	1
Fig. 8(c)	20	6	< 1	4	132
Fig. 8(d)	13	4	< 1	4	131

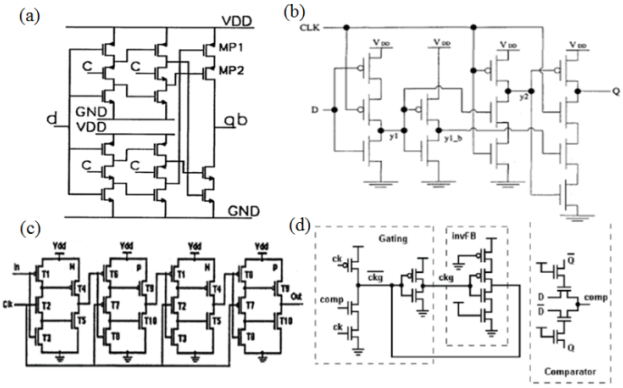

Fig. 8. Circuits tested in Table VII.

Figure 9 shows two layouts of Fig. 8(b) obtained by our approach. Figure 9(a) is generated without considering diffusion contour roughness whereas Fig. 9(b) does consider diffusion contour roughness (giving a larger weighing coefficient to diffusion contour roughness). The P- and N-diffusion contour roughness values of the layout in Fig. 9(a) are 3 and 3 respectively. They are respectively 1 and 1 for the layout in Fig. 9(b). However, the horizontal wire length used in Fig. 9(b) is 15.82um whereas it is only 12um in Fig. 9(a). Our approach enables a tradeoff among these objectives, but a designer is obviously at the helm of its degree.

V. CONCLUSIONS

The ILP approach presented in this paper could obtain a smaller number of transistor chains than the well-known bipartite graph approach. About 31% of the 185 cells created by it had smaller widths and no cells whose widths were larger than their handcrafted counterparts. It worked together with an in-house router as a standard cell layout generator integrated into a commercial layout editor to provide a full solution to standard cell layout generation. Although we have proposed an approach to significantly reducing runtime, an effective extension to handling more complicate circuits or multiple-cell-row layout generation is still a challenging task.

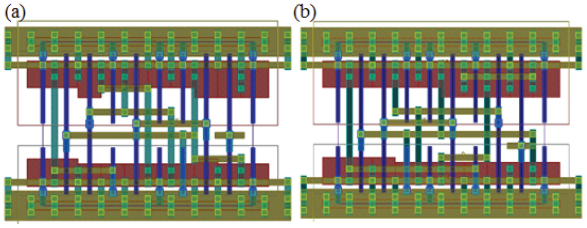

Fig. 9. Tradeoff between wire length and diffusion roughness.

REFERENCES

[1] V. Axelrad and M.C. Smayling, "16nm with 193nm immersion lithography and double exposure," SPIE, vol. 7641, pp.107-118, Feb 2010.

[2] J. Kim and S. M. Kang, "An efficient transistor folding algorithm for row-based CMOS layout design," in Proc. DAC, pp. 456–459, Jun. 1997.

[3] J. Cortadella, "Area-optimal transistor folding for 1-D gridded cell design", IEEE Trans. CAD, vol. 32, no. 11, pp. 1708-1721, Nov. 2013.

[4] M. Guruswamy, D. Dulitz, S. Raman, V. Chiluvuri, A. Fernande, and L. G. Jones, "Cellerity: A fully automatic layout synthesis system for standard cell libraries," in Proc. DAC, pp. 327–332, 1997.

[5] C. Lazzari, L. Anghel, and R. Reis, "A transistor placement technique using genetic algorithm and analytical programming," In Proc. IFIP WG.5 Conference on VLSI System-on-Chip, pp. 559–564, 2005.

[6] R. L. Maziasz and J. P. Hayes, "Exact width and height minimization of CMOS cells," in Proc. DAC, pp. 487–493, 1991.

[7] C. Y. Hwang, Y. C. Hsieh, Y. L. Lin, and Y. C. Hsu, "A fast transistor-chaining algorithm for CMOS cell layout," IEEE Trans. CAD, Vol. 9. no 7, pp. 781-786, July 1990.

[8] R. Bar-Yehuda, J. A. Feldman, R. Y. Pinter, and S. Wimer, "Depth-first search and dynamic programming algorithms for efficient CMOS cell generation," IEEE Trans. CAD, vol. 8, no. 7, pp. 737–743, July 1989.

[9] A. Gupta and J. P. Hayes, "Width minimization of two-dimensional CMOS cells using integer programming," in Proc. ICCAD, pp. 660–667, 1996.

[10] A. Gupta and J. P. Hayes, "Optimal 2-D cell layout with integrated transistor folding," in Proc. ICCAD, pp. 128–135, Nov. 1998.

[11] T. Iizuka, M. Ikeda, and K. Asada, "Exact minimum-width transistor placement without dual constraint for CMOS cells," in Proc. GLVLSI, pp. 74–77, 2005.

[12] T. Iizuka, M. Ikeda, and K. Asada, "Exact minimum-width multi-row transistor placement for dual and non-dual CMOS cells," in Proc. ISCAS, pp. 21–24, 2006.

[13] A. J. Velasco, X. Marin, R. P. Llopis, and J. Carrabina, "A combined pairing and chaining algorithm for CMOS layout generation," in Proc. IEEE European Conference on Design and Test, pp. 609-612, 1996.

[14] P. H. Wu, M. P. Lin, T. C. Chen, T. Y. Ho, Y. C. Chen, S. R. Siao, and S. H. Lin, "1-D cell generation with printability enhancement," IEEE Trans. CAD, vol. 32, no. 3, pp. 419-432, Mar. 2013.

[15] X. Chen, and J. Zhu, "Transistor permutation for better transistor chaining," 8th IEEE Int. Conf. ASICON., pp.1276-1279, Oct. 2009.

[16] N. Weste, and K. Eshraghian, *Principles of CMOS VLSI Design*. Reading, MA: Addison-Wesley, 1985.

[17] Y. C. Lin, *Optimization Problems in Computer-Aided Design of VLSI Circuits*. PhD. Thesis, Chung Yuan Christian University, 2001.

[18] S. Wimmer, R. Y. Pinter, and J. A. Feldman, "Optimal chaining of CMOS transistors in a functional cell," IEEE Trans. CAD, vol. CAD-6, pp. 795-801, Sept. 1987.

[19] F. A. Aloul, A. Ramani, I. L. Markov, and K. A. Sakallah, "Generic ILP versus specialized 0-1 ILP: An update," in Proc. ICCAD, pp. 450–457, 2002.