Placement Mitigation Techniques for Power Grid Electromigration

Wei Ye[1], Yibo Lin[1], Xiaoqing Xu[1], Wuxi Li[1], Yiwei Fu[2], Yongsheng Sun[2], Canhui Zhan[2], and David Z. Pan[1]

[1]ECE Department, University of Texas at Austin, Austin, TX, USA
[2]Hisilicon Technologies Co., Ltd., Shenzhen, China

***Abstract*—In advanced technology nodes, power grid metal wires are prone to electromigration (EM) failures due to small wire sizes and high unidirectional current densities. Power grid EM failures usually happen around weak power grid connections delivering current to high power-consuming regions. Previously, power grid EM was mostly addressed at the post-routing stage, which may be too late for a large number of EM violations in modern designs. In this paper, we propose a new set of incremental placement techniques to mitigate power grid EM, including cell move, single row placement, and single tile placement. Experimental results demonstrate the proposed placement techniques can effectively reduce EM violations with negligible wirelength and placement density impacts.**

I. Introduction

As VLSI technology continues to scale, EM has become a major reliability concern for chip design. High current densities lead to the movement of metal atoms, and such EM effect causes the growth of opens and shorts in metal wires over time. Two reasons for the rise of the current densities in metal wires are the continuous increase in transistor densities and aggressive scaling of interconnects. Consequently, the number of EM violations is growing as well as the difficulty of EM design closure. Power grid wires are more susceptible to EM failures than signal wires as they carry large unidirectional currents that cannot benefit from the healing effect of bidirectional currents [1].

EM checking tools calculate current densities in metal wires and detect EM violations in power grids with given design rules; then these violations are fixed with engineering change order (ECO) efforts. However, traditional fixing approaches such as spacing large-current cells and widening metal wires are not effective enough to handle the ever-growing number of violations in power grids. The methodology of "EM-analysis-then-fix" is becoming obsolete at advanced nodes [2]. Therefore, it is necessary to mitigate EM degradation in power grids at earlier design stages, such as placement. The locations of standard cells and the corresponding current distribution are determined during placement stage and the placement solution can directly affect the final quality of EM design closure.

Power grid consists of horizontal power rails connecting standard cells together, and these rails are connected with wider vertical power stripes. As illustrated in Fig. 1(a), *power tile* is the region between two adjacent VDD (or VSS) power stripes and the adjacent power rails [2], and the chip region is partitioned into multiple power tiles. It is observed that lower-level metal layers of power grids are more susceptible to EM failures due to smaller wire width, and EM violations are most likely to occur around weak power grid connections, which deliver current to high power-consuming regions. [2] proposes a global placement problem with a bin-packing formulation to constrain power consumption of each power tile, so that high-power cells are forced to spread across the placement region and current densities are flattened over the chip.

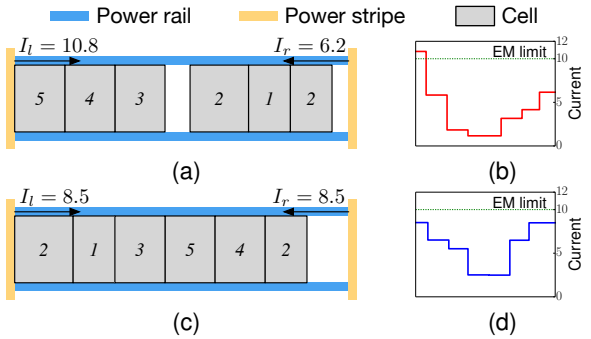

Fig. 1: (a) An initial cell placement in a power tile and (b) the corresponding current distribution in the power rail; (c) An EM-friendly placement and (d) the corresponding current distribution.

However, placement with globally balanced current density does not guarantee EM friendliness. As illustrated in Fig. 1, the metal wire segments touching vias on both sides carry the largest currents in a power rail. They feed all the cells in the tile and are the weakest points to EM. Current densities of these segments may still exceed the current limit even if the total current density of the power tile is below the threshold. Fig. 1(a) shows a placement within a power tile, and the number on each cell denotes its normalized current. Suppose the DC current limit for power rail is 10. Fig. 1(b) shows the simulation result of the current distribution on the power rail. The total current drawn by all the cells is 17, which is less than the power tile total current limit 20. But an EM violation occurs on the left side because the current exceeds the EM limit. The placement shown in Fig. 1(c) guarantees that the maximum current in the power tile will not exceed the EM current limit.

Placement is typically divided into three stages, global placement, legalization and detailed placement [3]. Global placement determines the rough locations of cells considering objectives such as wirelength, routability, and timing. After global placement, the placement is legalized by removing overlaps and aligning cells to placement sites. Detailed placement further refines the solution locally. Prior works [4]–[6] optimized cell density and pin density at detailed placement stage. However, mitigation of power grid EM during detailed placement is a different and more complicated problem. Globally balancing current density

This work is supported in part by Hisilicon Technologies Co., Ltd.

978-1-5090-6023-8/17/$31.00

over chip cannot completely resolve EM violations because the maximum current constraint for power tiles is more strict than the total current constraint. Therefore, as illustrated in Fig. 1, besides determining which cells to be placed in the power tile, we need to figure out the order and spacing of these cells under the EM current limit. In this work, we propose a series of detailed placement techniques to mitigate power grid EM. The proposed methods can reduce EM violations effectively with negligible impacts on wirelength and placement density. Our main contributions are summarized as follows:

- We present an incremental placement flow to address power grid EM violations while optimizing wirelength and placement density. To the best of our knowledge, this is the first detailed placement flow for power grid EM mitigation.
- A nested dynamic programming (DP)-based single row placement algorithm is developed to optimize wirelength under the total current constraint of power tiles.
- A mixed integer linear programming (MILP) model is formulated to solve single tile placement under the maximum current constraint. Besides, a set of hybrid techniques is proposed to improve runtime while maintaining very comparable performance.

The rest of this paper is organized as follows. Section II illustrates the specific constraints and gives the formulation of the detailed placement problem. Section III provides a detailed explanation of the proposed algorithms. Section IV demonstrates the effectiveness of our approaches with comprehensive results, followed by conclusion in Section V.

II. Preliminaries

Hsu et al. [2] propose an average power-based model to evaluate power grid static EM at placement stage. With given supply voltage, we use the DC current limit I_{limit} of power rail metal wires to evaluate power grid EM violations. For a standard cell, we consider the sum of the dynamic current and leakage current at this stage, which is calculated as:

$$I = \alpha \cdot C \cdot V_{\text{DD}} \cdot f + I_{\text{leak}},$$

where α is the cell activity factor, V_{DD} is the supply voltage and f is the system clock frequency. C is the sum of the load capacitance and the output pin capacitance. Load capacitance further includes downstream gate capacitance and interconnect capacitance. Since nets have not been routed at this stage, we use half-perimeter wirelength (HPWL), which is widely adopted in placement [3], to estimate interconnect capacitance.

Fig. 2 demonstrates how we calculate the maximum current in the local power rails within a power tile. P_l and P_r are the left and right endpoints of the VDD power rail. d_i^l and d_i^r are the distances from the midpoint of the i-th cell to P_l and P_r, respectively. R_i^l and R_i^r are the wire resistances of the corresponding metal segments, which are proportional to d_i^l and d_i^r. The following equations hold:

$$I_i^l \cdot R_i^l = I_i^r \cdot R_i^r, \quad I_i^l + I_i^r = I_i.$$

Thus,

$$I_i^l = \frac{d_i^r}{d_i^l + d_i^r} I_i, \quad I_i^r = \frac{d_i^l}{d_i^l + d_i^r} I_i. \tag{1}$$

The currents drawn by all the cells in the power tile from P_l and P_r are computed as:

$$I^l = \sum_i I_i^l, \quad I^r = \sum_i I_i^r, \quad I^l + I^r = \sum_i I_i. \tag{2}$$

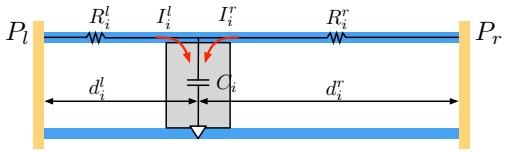

Fig. 2: The model for current calculation in a power tile.

Since the cells in the tile only draw current via P_l and P_r, the peak current that occurs in the local power rail is $\max\{I^l, I^r\}$.

Definition 1 (EM Violation)**.** *There is an EM violation in the power tile if* $\max\{I^l, I^r\} > I_{\text{limit}}$.

According to Eq. (1) and Eq. (2), it is necessary to know the current and location of each cell in the power tile to compute I^l and I^r. Therefore, once a cell is relocated, we update its current because the load capacitance is also changed. When a cell is moved to a new power tile, we need to predict whether this movement will cause an EM violation in the new power tile. Since we may not decide the new location of the cell immediately, we derive a total current constraint of the power tile to estimate the EM violation. Combining Eq. (2) with the property that an EM-friendly power tile satisfies $I^l \le I_{\text{limit}}$ and $I^r \le I_{\text{limit}}$, there is a relaxed constraint on the total current of power tile as follows:

$$\sum_i I_i = I^l + I^r \le 2I_{\text{limit}}. \tag{3}$$

Satisfying the above constraint is a necessary condition for the power tile to be free from EM violation. When the total current of the cells in the power tile is less than $2I_{\text{limit}}$, cell placement in the power tile further determines I^l and I^r.

In this work, we follow ICCAD 2013 placement contest [7] to use scaled half-perimeter wirelength (sHPWL) defined below to quantify the quality of placements:

$$\text{sHPWL} = \text{HPWL} \cdot (1 + \text{ABU}),$$

where HPWL is the wirelength metric, and average bin utilization (ABU) is used to evaluate placement density [8]. The bin used for ABU density calculation contains multiple rows of power tiles.

Problem 1 (EM-Aware Detailed Placement)**.** *Given an initial legalized detailed placement and the EM DC current limit* I_{limit}*, we seek a legal placement to minimize the number of EM violations and further reduce* sHPWL.

III. Algorithms

In this section, we introduce the three placement techniques for reducing EM violations and describe the incremental placement flow.

A. Cell Move

The major objective of the first technique is to achieve cell current balance among power tiles from a global scope. We use the cell move approach [4], [5] and try to move cells out from current-overfilled tiles to other tiles which can accommodate them and help improve sHPWL.

Power tiles with total currents greater than the threshold $2I_{\text{limit}}$ will definitely have EM violations and such violations cannot be fixed by local permutation. In addition, an EM violation may exist for a power tile even if its total current density is below the threshold. Therefore, we define a current threshold parameter t_c and the tiles with total cell current greater

than $2I_{\text{limit}} \cdot t_c$ are regarded as the source tiles for cell move. The cell with the largest current will be moved to a target tile in the search region [9] that has enough area and current capacity. Note that the total current of the target tile after the cell movement should be less than the predefined $2I_{\text{limit}} \cdot t_c$ to avoid cell move loops. We sort the candidate target tiles according to their distances to the optimal region [10] and the tile with the minimum cost will be chosen. The cost of moving cell i to target tile j is defined as:

$$\text{cost}(i,j) = \Delta\,\text{sHPWL}(i,j) + \beta \cdot \text{density}(j),$$

where $\Delta\,\text{sHPWL}(i,j)$ denotes the sHPWL change and $\text{density}(j)$ denotes the total current density of tile j after the cell movement. Both of them are normalized in the scale of site half-perimeter.

We only decide the target tile for the cell to move into by the above procedure. The cell is temporarily put in the center of the target tile, and its precise location and possible overlaps will be solved by the subsequent techniques. We iteratively repeat the above procedure until we cannot find cells to move or the maximum number of iterations is reached.

B. Single Row Placement

After cell move, we perform the ordered single row placement that minimizes wirelength under the total current constraint for each power tile in a row. The ordered single row placement for minimizing wirelength has been well-studied [6], [11]–[13]. Under the maximum displacement m for each cell, the problem can be transferred to the shortest path problem, and a DP-based algorithm is able to solve it in $O(m^2 n)$ [6], [14]. However, the additional constraint of total current makes the problem more complicated, which cannot be solved by the above approaches. We define the ordered single row placement problem under the total current constraint in Problem 2. We provide our main SINGLEROWDP algorithm in Theorem 1 which invokes SINGLETILEDP in Theorem 2. Note that our entire algorithm is still optimal even if we replace SINGLETILEDP by other algorithms that are able to output optimal solutions, which makes our single row placement algorithm widely applicable. The straightforward way to implement SINGLETILEDP has quadratic dependence in m, but we can achieve linear dependence in m by using some standard tricks [15]. For any positive integers n, m, we use $[n]$ to denote set $\{1, 2, \cdots, n\}$, and $[n, m]$ to denote set $\{n, n+1, \cdots, m\}$. To give a more general formulation, we use *cost* to denote wirelength and *value* to denote current.

Problem 2 (Fixed Order Single Row Placement)**.** *Given n ordered cells, M locations and B tiles, m denotes the maximum displacement and L_i denotes a set of feasible locations*[1] *where the i-th cell can be placed, i.e., $\max_{i\in[n]} L_i = m$. Let $c_{i,j}$ and $v_{i,j}$ denote the cost and value corresponding to placing the i-th cell at the j-th location, $\forall i \in [n], j \in L_i$. Let $v_{\max}(= 2I_{\text{limit}} \cdot t_c)$ denote the value threshold. The goal is to find a feasible, non-overlapping placement that keeps the initial order ($\forall i \in [n-1], \pi(i) < \pi(i+1)$, where $\pi(i) \in L_i$ is the new location for i-th cell), such that the value (current) constraint is satisfied, and the total cost (wirelength) is minimized.*

[1]Note that L_i is a set of consecutive integers (i.e., $L_i = [x, x+1, \cdots, y-1, y]$) in the problem as we claimed. Our algorithm is also working for the general case that L_i contains gaps.

Algorithm 1

```
1:  procedure SINGLEROWDP(c, v)                                   ▷ Theorem 1
2:      for j = 1 → B do
3:          Q_j ← {i | L_i ∩ J_j ≠ ∅, i ∈ [n]}
4:          for i ∈ Q_j do
5:              L̂_i^j ← L_i ∩ J_j
6:      for j = 1 → B; i_1 ∈ Q_j; i_2 ≥ i_1, i_2 ∈ Q_j do
7:          if Σ_{i=i_1}^{i_2} v_{i,j} ≤ v_max then
8:              f̂_{i_1,i_2,j} ← SINGLETILEDP(c, L̂^j, i_1, i_2)
9:      for j = 1 → B; i_2 ∈ Q_j do
10:         S_{i_2,j} ← {i_1 | i_1 ∈ Q_j, i_1 ≤ i_2, f_{i_1,j-1} ≠ ∞, f̂_{i_1+1,i_2,j} ≠ ∞}
11:         f_{i_2,j} ← min_{i_1 ∈ S_{i_2,j}} (f_{i_1,j-1} + f̂_{i_1+1,i_2,j})
12:     return min_{j∈[B]} f_{n,j}
13: procedure SINGLETILEDP(c, L, i_1, i_2)                        ▷ Theorem 2
14:     for k = i_1 → i_2 do
15:         τ(k) to be the last location of L_k
16:         for l ∈ L_k do
17:             if l − 1 ∉ L_{k−1} then
18:                 l' ← τ(k − 1)
19:             else
20:                 l' ← l − 1
21:             g_{k,l} ← min(g_{k,l−1}, g_{k−1,l'} + c_{k,l})
22:     return g_{i_2,τ(i_2)}
```

Theorem 1 (Single Row Dynamic Programming)**.** *There is an algorithm (Procedure* SINGLEROWDP *in Algorithm 1) running in $O(Bt^3m + Bn)$ time*[2] *that is able to output a placement $\pi : [n] \to [M]$ such that $\forall b \in [B], \sum_{i,\pi(i)\in J_b} v_{i,\pi(i)} \le v_{\max}$ holds, and $\sum_{i\in[n]} c_{i,\pi(i)}$ is minimized, where J_b denote the set of locations belong to tile $b \in [B]$, and t denote the maximum number of cells per tile.*

Proof. Let $f_{i,j}$ denote the cost that all the first i cells are placed in the first j tiles if there is no violation over all the first j tiles, otherwise $f_{i,j} = \infty$. Let $\widehat{f}_{i_1,i_2,j}$ denote the cost that for placing from the i_1-th cell to the i_2-th cell to tile j if there is no violation, otherwise $\widehat{f}_{i_1,i_2,j} = \infty$. The total running time consists of three parts. The first part (lines 2–5) is computing all Q_j and $\widehat{L}_i^j$ in $O(Bn)$ time, where Q_j is the set of cells that can be placed in tile j, and $\widehat{L}_i^j$ denotes the feasible locations for cell i in tile j. Before we define $t = \max_{j\in[B]} |Q_j|$. The second part (lines 6–8) is from calling SINGLETILEDP $O(Bt^2)$ times and the running time of SINGLETILEDP is $O(tm)$. Thus, the running time for the second part is $O(Bt^3m)$. The third part (lines 9–11) is dominated by computing set $S_{i_2,j}$ $O(Bt)$ times, and computing $S_{i_2,j}$ takes $O(t)$ time. Overall, the running time is $O(Bt^3m + Bn)$. The correctness can be proved by induction. Let $S_{i_2,j}$ denote a set of possible states i_1 such that i_2 is coming from i_1. For each iteration, we update $f_{i_2,j}$ by taking the minimum from $|S_{i_2,j}|$ states. For each state, suppose we transform from some $i_1 \in S_{i_2,j}$, the cost contains two parts: the first part is the cost of placing all the first i_1 cells in the first $j-1$ tiles, i.e., $f_{i_1,j-1}$; the second part is the cost of placing cells $i_1+1, \cdots, i_2$ in the tile j, i.e., $\widehat{f}_{i_1+1,i_2,j}$. □

Remark 1. *For simplicity, our* DP *algorithms (in Algorithm 1) demonstrate the case where each cell has the same unit site, all the tiles have the same length, and the maximum displacement for each cell is the same. It is easy to extend it to the general*

[2]Our current result assumes that $v_{i,j} = v_{i,j'}$ for any j, j' in the same tile. Our algorithm can be extended to the case without that assumption, the running time becomes $O(Bt^3m \cdot v_{\max} + Bn)$.

setting. We also omit the details of backtracking to output the optimal solution.

Theorem 2 (Single Tile Dynamic Programming)**.** *Given t cells and a tile with ℓ locations, let L_i denote a set of feasible locations for cell $i \in [t]$ and $m = \max_i |L_i|$. Let $c_{i,j}$ denote the cost of placing cell i at the j-th location. There is an optimal algorithm*[3] *(Procedure* SINGLETILEDP *in Algorithm 1) running in $O(tm)$ that is able to output a placement $\pi : [t] \to [\ell]$ such that $\sum_{i\in[t]} c_{i,\pi(i)}$ is minimized.*

Proof. Let $g_{k,l}$ denote the optimal cost of cells $i_1, \cdots, k$ being placed in the first l locations of the tile. L_k denotes a set of feasible locations that cell k can be placed, we use $\tau(k)$ denote the last location of L_k. Because we have two nested for loops, one is going through all the cells, and the other is going through all the feasible locations of the cell, thus the running time is $O(\sum_{i=i_1}^{i_2} |L_i|) = O((i_2 - i_1 + 1)m) = O(tm)$. By induction, we can show that the optimal solution is $g_{i_2,\tau(i_2)}$. In each iteration, we update the $g_{k,l}$ by taking the minimum of two states. One is placing cell k at location l, whose cost is $g_{k-1,l'} + c_{k,l}$. Note that we cannot set l' to be $l-1$ directly, because it is possible that $l-1$ is not a feasible location for cell $k-1$. The other state is not placing cell k at location l, whose cost is $g_{k,l-1}$, and $g_{k,l-1} = \infty$ if $l - 1 \notin L_k$. □

Our SINGLETILEDP algorithm finds the optimal solution by checking the states whether the current cell is placed at a certain location. Thus, the time complexity of it is $O(tm)$, which is the same as the work in [16] by pruning solution spaces. In most cases, the single row placement can help reduce the number of the current-overfilled tiles to zero. However, in some extreme cases with tight maximum displacement or existence of blockages, it may fail because the cells cannot be shifted much in the row.

C. Single Tile Placement

After the steps mentioned above, we determine the cells in each power tile. We now present the single tile placement which helps address the maximum current violation in a power tile if the total current constraint has been satisfied.

Problem 3 (Single Tile Placement)**.** *Given the cells within a power tile, find a non-overlapping placement for these cells so that* HPWL *is minimized under the constraint that the maximum current in the power tile is less that the EM current limit I_{limit}.*

In this section, we use $\mathcal{C}$ to denote the set of cells in the power tile and $\mathcal{N}$ to denote the set of nets. Let $\mathcal{L}$ denote set of all possible site locations in the tile. For the i-th cell in $\mathcal{C}$, we use W_i to denote its width and I_i to denote the its current. W denotes the width of the entire power tile. Let $p_{i,k}$ denote the horizontal distance between the i-th cell and the pin corresponding to the i-th cell associated with the k-th net.

1) MILP Formulation: A power tile with an EM violation usually contains more than ten cells and thus the sliding window approach for cell reordering [17] cannot be applied. Placement problems using mixed integer programming (MIP) [18] and MILP [19], [20], by contrast, are more scalable by applying branch-and-cut approach. Hence, we propose an MILP formulation to determine the order and locations of cells in the power tile and consider the no-overlap and maximum current constraints simultaneously. Since cells are only moved inside the power tile, the y-coordinate of each cell is fixed. Assuming the cells in other tiles is also fixed for the time, the total HPWL can be formulated by the sum of the difference between the left and right boundary of the bounding box for each net. We define the MILP model minimizing the total HPWL as follows:

$$\min_{\mathbf{x},\mathbf{y},\mathbf{l},\mathbf{r}} \quad \sum_{k\in\mathcal{N}} (\mathbf{r}_k - \mathbf{l}_k) \tag{4a}$$

$$\text{s.t.} \quad \mathbf{x}_{i,j}, \mathbf{y}_{i,j} \in \{0,1\}, \ \forall i \in \mathcal{C}, j \in \mathcal{L}, \tag{4b}$$

$$\sum_{j\in\mathcal{L}} \mathbf{x}_{i,j} = 1, \ \forall i \in \mathcal{C}, \tag{4c}$$

$$\sum_{i\in\mathcal{C}} \mathbf{y}_{i,j} \le 1, \ \forall j \in \mathcal{L}, \tag{4d}$$

$$\sum_{j\in\mathcal{L}} \mathbf{y}_{i,j} = W_i, \ \forall i \in \mathcal{C}, \tag{4e}$$

$$\sum_{j=j'}^{j'+W_i} \mathbf{y}_{i,j} - W_i \cdot x_{i,j'} \ge 0, \ \forall i \in \mathcal{C}, \forall j' \in \mathcal{L}, \tag{4f}$$

$$\sum_{i\in\mathcal{C}} \sum_{j\in\mathcal{L}} I_i \cdot (j + W_i/2) \cdot \mathbf{x}_{i,j} \le I_{\text{limit}} \cdot W, \tag{4g}$$

$$\sum_{i\in\mathcal{C}} \sum_{j\in\mathcal{L}} I_i \cdot (W - j - W_i/2) \cdot \mathbf{x}_{i,j} \le I_{\text{limit}} \cdot W, \tag{4h}$$

$$\mathbf{l}_k \le \sum_{j\in\mathcal{L}} j \cdot \mathbf{x}_{i,j} + p_{i,k} \le \mathbf{r}_k, \forall i \in \mathcal{C}, \forall k \in \mathcal{N}. \tag{4i}$$

Formulation (4) is optimized over four kinds of variables, where $x_{i,j}, y_{i,j}$ are binary variables and l_k, r_k are continuous variables. If the lower left corner of cell i locates at site j then $x_{i,j} = 1$, otherwise $x_{i,j} = 0$. If cell i occupies site j then $y_{i,j} = 1$, otherwise $y_{i,j} = 0$. Constraint (4c) makes sure that each cell is placed at only one location, and constraint (4d) guarantees that cells do not overlap. Constraints (4e) and (4f) ensure that each cell occupies the same number of total sites as its width, and all occupied site locations for the cell are contiguous. According to Eq. (1) and Eq. (2), we calculate the currents in the leftmost and rightmost power rail segments and force them to be smaller than the maximum current limit in constraint (4g) and (4h). The left boundary l_k and right boundary r_k of the bounding box of each net are defined in constraint (4i).

2) Speedup Techniques: The aforementioned MILP formulation is optimal as it considers the orders of cells and spacing simultaneously, but may suffer from long runtime overhead. Here we propose a set of speedup techniques that breaks the single tile placement into two phases, where cells are reordered and shifted subsequently.

Fig. 3 illustrates the fast way to solve the single tile place-

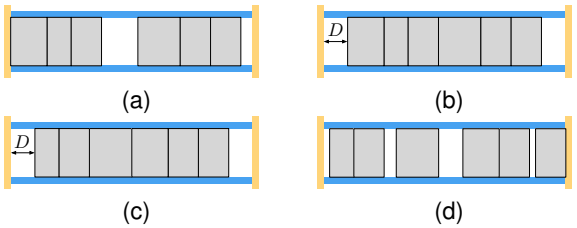

Fig. 3: Speedup techniques for single tile placement. (a) The initial placement, (b) the placement after packing cells, (c) the placement after cell reordering and (d) the final placement after SINGLETILEDP.

[3]The running time is optimal, because the input size (the number of feasible locations) is already $\Omega(tm)$.

ment problem. Given the cells in the problematic power tile (Fig. 3(a)), we pack all the cells to the center (Fig. 3(b)) and run the MILP algorithm shown in Formulation (5) to determine the order of cells and ignore whitespaces in the tile temporarily under the maximum current constraint (Fig. 3(c)). After that, the tile-based ordered placement SINGLETILEDP (in Algorithm 1) is run to determine the locations of cells if the wirelength can be improved further under the maximum current constraint. The final placement is shown in Fig. 3(d).

Let D denote the distance of the leftmost cell to left power stripe, we define FASTMILP as follows:

$$\min_{\mathbf{z},\mathbf{s},\mathbf{l},\mathbf{r}} \sum_{k\in\mathcal{N}} (\mathbf{r}_k - \mathbf{l}_k) \tag{5a}$$

$$\text{s.t. } \mathbf{z}_{i,j} \in \{0,1\},\ \forall i \neq j \in \mathcal{C}, \tag{5b}$$

$$\mathbf{z}_{i,j} + \mathbf{z}_{j,i} = 1,\ \forall j > i \in \mathcal{C}, \tag{5c}$$

$$\mathbf{z}_{i,j} + \mathbf{z}_{j,k} - \mathbf{z}_{i,k} \leq 1,\ \forall i \neq j \neq k \in \mathcal{C}, \tag{5d}$$

$$\mathbf{s}_i = \sum_{j\neq i} \mathbf{z}_{j,i} \cdot W_j + D,\ \forall i \in \mathcal{C}, \tag{5e}$$

$$\sum_{i\in\mathcal{C}} I_i \cdot (\mathbf{s}_i + W_i/2) \leq I_{\text{limit}} \cdot W, \tag{5f}$$

$$\sum_{i\in\mathcal{C}} I_i \cdot (W - \mathbf{s}_i - W_i/2) \leq I_{\text{limit}} \cdot W, \tag{5g}$$

$$\mathbf{l}_k \leq \mathbf{s}_i + p_{i,k} \leq \mathbf{r}_k, \forall i \in \mathcal{C}, \forall k \in \mathcal{N}. \tag{5h}$$

Formulation (5) is optimized over four kinds of variables, where $\mathbf{z}_{i,j}$ is a binary variable, and $\mathbf{s}_i$, $\mathbf{l}_k$ and $\mathbf{r}_k$ are continuous variables. We use variable $\mathbf{z}_{i,j}$ to represent the relative order of cell i and j. If cell i is on the left of cell j then $\mathbf{z}_{i,j} = 1$, else $\mathbf{z}_{i,j} = 0$. Variable $\mathbf{s}_i$ denotes the placement site of lower left corner of that cell. For any three cells i, j and k, we also need to make sure that if cell i is on the left of cell j and cell j is on the left of cell k, then cell i must be on the left of cell k, which is guaranteed by constraint (5d). Constraint (5e) transfers the relative orders of a group of cells to their site locations. The maximum current limit constraint is addressed by constraint (5f) and constraint (5g). Constraint (5h) is identical to constraint (4i) in Formulation (4) to formulate HPWL.

Notice that the number of binary variables in Formulation (4) is $2 \cdot |\mathcal{C}| \cdot |\mathcal{L}|$, and the number of binary variables in Formulation (5) is $|\mathcal{C}|^2 - |\mathcal{C}|$. It is observed that $|\mathcal{L}| \gg |\mathcal{C}|$ in our benchmarks, which is the reason for that solving Formulation (5) is much faster than Formulation (4) in practice. Although these speedup techniques cannot guarantee an optimal solution of the single tile placement problem, experimental results demonstrate that it can achieve noticeable runtime speedup without sacrificing too much performance.

There are some corner cases where the maximum current constraint cannot be satisfied by any cell placement within the tile, even if the total current of the tile is less than the threshold. Our MILP models become infeasible in this situation and will report that no feasible solution can be found.

D. Overall Flow

The proposed detail placement flow for power grid EM mitigation is shown in Fig. 4. The first two stages are cell move and single row placement to reduce the total currents in current-overfilled tiles. The third stage is single tile placement to reduce the maximum current in each of power tiles with EM violations. Single tile placement has two available algorithms, including the original MILP algorithm and a series of speedup techniques (FASTMILP and SINGLETILEDP).

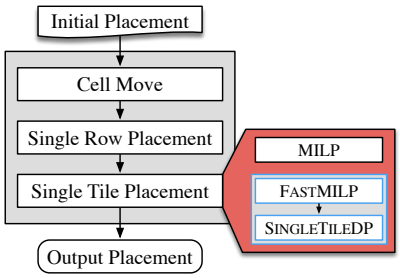

Fig. 4: Overall flow of our detailed placement techniques.

TABLE I: Benchmark Characteristics

Design	#cells	#nets	#blk	Density	Target util.	Disp.(um)
vga_lcd	165K	165K	0	68.94%	70%	10
b19	219K	219K	0	44.85%	70%	20
leon3mp	649K	649K	0	72.02%	75%	30
leon2	794K	795K	0	84.19%	90%	40
mgc_edit_dist	131K	133K	13	67.26%	70%	30
mgc_matrix_mult	155K	159K	16	59.31%	65%	30
netcard	959K	961K	12	66.29%	70%	50

IV. EXPERIMENTAL RESULTS

Our placement framework was implemented in C++ and run on a 3.40 GHz Linux machine with 32 GB memory. Since EM violations are more noticeable at advanced nodes, we validated our algorithms on the set of benchmarks from [16], which integrated NanGate $15nm$ standard cell library [21] into ICCAD 2014 placement benchmarks [22] and used RippleDP [5] to generate the initial detailed placements. Table I presents the characteristics of this set of benchmarks. ICCAD 2014 placement contest defines two maximum displacement limits for each design, and we chose the smaller one for less perturbation to the original placements, as listed in column "Disp. (um)". GUROBI [23] was used as the MILP solver. We set the user-defined parameters t_c and β to 0.7 and 5. Note that for some extremely dense power tiles, it takes a long time to find the optimal solution. Thus, we set a time limit for each run of the MILP solver to 200s.

We set the supply voltage, operating temperature and clock frequency to 0.88V, 125°C and 1GHz in the experiments. Cell current was calculated from the NLDM file in NanGate 15nm standard cell library [21]. We studied the typical values of the metal width of power grids at 16-nm nodes and set power rail and power stripe wire width to 0.09um and 0.32um, respectively. The power tile width was set to 5.76um. The EM DC current limit under this setting was 0.067mA. To demonstrate the effectiveness of the proposed flow, we set the EM DC current limit tighter to 0.026mA for benchmarks *b19*, *mgc_edit_dist* and *mgc_matrix_mult* with small initial violation numbers (≤ 40).

Table II shows the detailed placement results. "MILP flow" and "Fast flow" denote the flows using the original MILP algorithm (Section III-C1) and the speedup techniques (Section III-C2) respectively in the single tile placement step. "EM vio." denotes the number of EM violations. "Δ HPWL" and "Δ sHPWL" denote the change in wirelength and scaled wirelength. "CPU" denotes the total runtime in second. The MILP flow is effective to fix 96% of the EM violations on average and its impacts on wirelength (0.31%) and placement density (0.2%) are very small. Furthermore, compared with the MILP flow, the fast flow is at least 8× faster while achieving comparable results,

TABLE II: Result comparsion between two flows.

Design	Initial			MILP flow				Fast flow			
	EM vio. #	HPWL $\times 10^6$	sHPWL $\times 10^6$	EM vio. #	ΔHPWL %	ΔsHPWL %	CPU s	EM vio. #	ΔHPWL %	ΔsHPWL %	CPU s
vga_lcd	127	1.421	1.874	0	0.076	0.012	52.72	0	0.073	0.009	7.99
b19	363	0.965	1.139	2	0.279	0.113	32.93	4	0.290	0.086	15.04
leon3mp	738	5.340	6.837	3	0.125	0.135	254.22	7	0.125	0.131	99.13
leon2	2076	13.092	14.487	35	0.561	0.637	4445.77	39	0.558	0.626	405.26
mgc_edit_dist	100	1.525	1.880	0	0.185	-0.273	5.66	0	0.185	-0.273	5.46
mgc_matrix_mult	144	0.912	1.126	0	0.149	-0.534	80.01	0	0.151	-0.533	7.14
netcard	2424	14.570	20.189	197	0.773	1.304	5314.47	206	1.040	1.430	678.35
avg.	853.1	5.404	6.790	33.9	0.307	0.199	1455.11	36.6	0.346	0.211	174.05
ratio	1			0.040			1	0.043			0.12

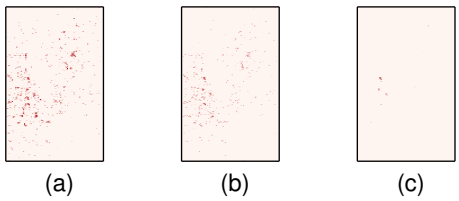

Fig. 5: The EM violation map of design *leon2*. The tiles with violations are marked in red. (a) The initial design, (b) after single row placement, and (c) after single tile placement.

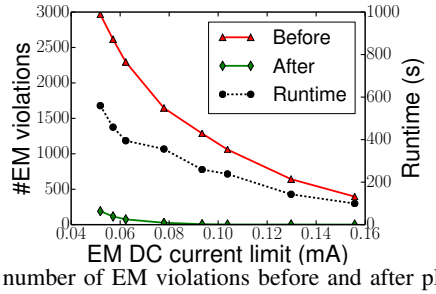

Fig. 6: The number of EM violations before and after placement and runtime vs. EM DC current limit for design *leon2*.

regarding EM violation reduction (95.4%), wirelength (0.35%), and density (0.21%). It is worth mentioning that the fast flow performs better than the MILP flow for the *vga_lcd* benchmark. It is because although the MILP approach theoretically gives an optimal solution to the power tile placement, the optimal solutions to the local sub-problems do not necessarily lead to the final globally optimal solution.

The key ideas of our work are first reducing the total currents for current-overfilled power tiles by cell move and single row placement, and further reducing the maximum current in each power tile by single tile placement. Fig. 5 demonstrates that the two types of placement techniques are indispensable for EM violation reduction. The first two techniques for total current reduction can remove a part of the EM violations, but more EM violations are fixed by single tile placement. As shown in Fig. 6, the initial number of EM violations is largely dependent on the EM current limit, and our placement flow can achieve zero EM violations in a wide range of current limits. The proposed placement techniques are designed to serve as an incremental placement step that can be used in ECO stage as well. It can be easily integrated into the existing physical design flow to eliminate EM violations iteratively.

V. Conclusion

This work presents a set of detailed placement mitigation techniques to handle power grid EM, including cell move, single row placement, and single tile placement. Experimental results show that these techniques achieve high-quality placement results regarding EM violation reduction, total wirelength, and placement density.

References

[1] B.-K. Liew, N. W. Cheung, and C. Hu, "Projecting interconnect electromigration lifetime for arbitrary current waveforms," *IEEE TED*, 1990.

[2] M.-K. Hsu, N. Katta, H. Y.-H. Lin, K. T.-H. Lin, K. H. Tam, and K. C.-H. Wang, "Design and manufacturing process co-optimization in nano-technology," in *ICCAD*, 2014.

[3] C. J. Alpert, D. P. Mehta, and S. S. Sapatnekar, *Handbook of Algorithms for Physical Design Automation*. CRC press, 2008.

[4] S. Popovych, H.-H. Lai, C.-M. Wang, Y.-L. Li, W.-H. Liu, and T.-C. Wang, "Density-aware detailed placement with instant legalization," in *DAC*, 2014, pp. 122:1–122:6.

[5] W.-K. Chow, J. Kuang, X. He, W. Cai, and E. F. Y. Young, "Cell density-driven detailed placement with displacement constraint," in *ISPD*, 2014.

[6] T. Taghavi, C. Alpert, A. Huber, Z. Li, G.-J. Nam, and S. Ramji, "New placement prediction and mitigation techniques for local routing congestion," in *ICCAD*, 2010.

[7] M.-C. Kim, N. Viswanathan, Z. Li, and C. Alpert, "ICCAD-2013 CAD contest in placement finishing and benchmark suite," in *ICCAD*, 2013.

[8] M.-C. Kim, N. Viswanathan, C. J. Alpert, I. L. Markov, and S. Ramji, "MAPLE: multilevel adaptive placement for mixed-size designs," in *ISPD*, 2012, pp. 193–200.

[9] Y. Lin, B. Yu, X. Xu, J.-R. Gao, N. Viswanathan, W.-H. Liu, Z. Li, C. J. Alpert, and D. Z. Pan, "MrDP: Multiple-row detailed placement of heterogeneous-sized cells for advanced nodes," in *ICCAD*, 2016.

[10] M. Pan, N. Viswanathan, and C. Chu, "An efficient and effective detailed placement algorithm," in *ICCAD*, 2005, pp. 48–55.

[11] J. Vygen, "Algorithms for detailed placement of standard cells," in *DATE*, 1998, pp. 321–324.

[12] A. B. Kahng, P. Tucker, and A. Zelikovsky, "Optimization of linear placements for wirelength minimization with free sites," in *ASPDAC*, 1999.

[13] U. Brenner and J. Vygen, "Faster optimal single-row placement with fixed ordering," in *DATE*, 2000, pp. 117–121.

[14] B. Yu, X. Xu, J.-R. Gao, Y. Lin, Z. Li, C. Alpert, and D. Z. Pan, "Methodology for standard cell compliance and detailed placement for triple patterning lithography," *IEEE TCAD*, 2015.

[15] J. Kleinberg and E. Tardos, *Algorithm design*. Pearson Education, 2006.

[16] Y. Lin, B. Yu, Y. Zou, Z. Li, C. J. Alpert, and D. Z. Pan, "Stitch aware detailed placement for multiple e-beam lithography," in *ASPDAC*, 2016.

[17] A. E. Caldwell, A. B. Kahng, and I. L. Markov, "Optimal partitioners and end-case placers for standard-cell layout," *IEEE TCAD*, 2000.

[18] S. Cauley, V. Balakrishnan, Y. C. Hu, and C.-K. Koh, "A parallel branch-and-cut approach for detailed placement," *ACM TODAES*, 2011.

[19] S. Li and C.-K. Koh, "Mixed integer programming models for detailed placement," in *ISPD*, 2012, pp. 87–94.

[20] K. Han, A. B. Kahng, and H. Lee, "Scalable detailed placement legalization for complex sub-14nm constraints," in *ICCAD*, 2015, pp. 867–873.

[21] "NanGate FreePDK15 Open Cell Library," http://www.nangate.com/?page_id=2328, 2015.

[22] M.-C. Kim, J. Hu, and N. Viswanathan, "ICCAD-2014 CAD contest in incremental timing-driven placement and benchmark suite," in *ICCAD*, 2014.

[23] Gurobi Optimization Inc., "Gurobi optimizer reference manual," http://www.gurobi.com, 2014.