UNIVERSIDADE FEDERAL DE SANTA CATARINA
CENTRO TECNOLÓGICO
CURSO DE GRADUAÇÃO EM CIÊNCIAS DA COMPUTAÇÃO

Arthur João Lourenço

GRCMO: Global Routing Optimization Through Median-Based Cell Movement in an Open-Source RTL-to-GDSII Flow.

Florianópolis
2024

Arthur João Lourenço

GRCMO: Global Routing Optimization Through Median-Based Cell Movement in an Open-Source RTL-to-GDSII Flow.

Relatório de Trabalho de Conclusão de Curso do Curso de Graduação em Ciências da Computação do Centro Tecnológico da Universidade Federal de Santa Catarina para a obtenção do título de Bacharel em Ciências da Computação.
Orientador: Prof. José Luís Almada Güntzel, Dr.
Coorientador: Tiago Augusto Fontana, Dr.

Florianópolis
2024

Ficha catalográfica gerada por meio de sistema automatizado gerenciado pela BU/UFSC.
Dados inseridos pelo próprio autor.

Lourenço, Arthur João
   GRCMO: global routing optimization through median-based cell movement in an open-source RTL-to-GDSII flow / Arthur João Lourenço ; orientador, José Luís Güntzel, coorientador, Tiago Augusto Fontana, 2024.
   55 p.

   Trabalho de Conclusão de Curso (graduação) - Universidade Federal de Santa Catarina, Centro Tecnológico, Graduação em Ciências da Computação, Florianópolis, 2024.

   Inclui referências.

   1. Ciências da Computação. 2. Eletronic Design Automation. 3. Placement. 4. Routing. 5. Routing with cell movement. I. Güntzel, José Luís. II. Fontana, Tiago Augusto. III. Universidade Federal de Santa Catarina. Graduação em Ciências da Computação. IV. Título.

Arthur João Lourenço

GRCMO: Global Routing Optimization Through Median-Based Cell Movement in an Open-Source RTL-to-GDSII Flow.

Este Trabalho de Conclusão de Curso foi julgado adequado para obtenção do Título de "Bacharel em Ciências da Computação" e aprovado em sua forma final pelo Curso de Graduação em Ciências da Computação.

Florianópolis, 4 de dezembro de 2024.

Banca Examinadora:

ACKNOWLEDGEMENTS

Gostaria de agradecer a todos que contribuíram para a realização deste trabalho de conclusão de curso. Primeiramente, agradeço minha família, que sempre estiveram comigo, em especial aos meus pais, Andrea e Mauricio por todo apoio que me deram para que eu pudesse conquistar tudo até aqui e também aos meus irmãos Victor e Leonardo, muito obrigado por todas as brigas e discussões que ajudaram a construir o meu caráter. Vocês todos são a minha inspiração e sem vocês nada disso seria possível.

A minha namorada Mariana, que foi minha parceira durante estes 5 anos de faculdade. Obrigado por ter escutado todas as minhas reclamações, por me falar que tudo ia dar certo mesmo quando eu achava que nunca ia conseguir me formar, por todos os papos cabeças que tivemos no kwid, pelas napolipapos depois de semanas longas de provas e por todos os bons e maus momentos que passamos na UFSC. A sua companhia deixou tudo mais fácil, amo tu!

Aos amigos que eu fiz neste curso, vocês foram uma parte essencial da minha jornada, seja passando horas estudando para as provas de org ou ficando madrugadas inteiras debugando o EPOS, cada momento contribuiu para minha formação como cientista da computação, e por isso sou profundamente grato. Meu agradecimento também aos meus colegas do laboratório ECL bem como à Professora Laleh Behjat e sua equipe do laboratório de VLSI da Universidade de Calgary, que me ajudaram durante todo o desenvolvimento deste trabalho, sem vocês este trabalho não seria o mesmo.

Agradeço ao meu orientador, Professor Dr. José Luís Güntzel, e ao meu coorientador, Dr. Tiago Fontana, por toda a dedicação, pelos valiosos conselhos e pelo apoio durante este trabalho. Suas contribuições foram essenciais não apenas para a realização deste projeto, mas também para minha formação acadêmica e profissional. Por fim, agradeço à banca avaliadora por dedicar seu tempo para ler e avaliar este trabalho!

A todos vocês, muito obrigado.

ABSTRACT

In the physical design of VLSI circuits, the placement and routing steps are responsible for finding cell positions and interconnecting cell pins. They significantly impact the circuit layout quality. Placement and routing were originally solved separately in a divide-and-conquer approach to cope with design complexity, resulting in a decoupling that often leads to poor solutions. To mitigate this problem, a few techniques seeking to integrate those steps have recently emerged. Unfortunately, some of those techniques do not consider the detailed placement information, making it difficult to evaluate their outcomes. In this final course project, we present GRCMO, a technique that optimizes Global Routing by moving a few cells to their medians, respecting the detailed placement rules, and rerouting the affected nets. GRCMO was implemented in one of the most compleat open-source platforms, OpenROAD, making its use within a complete RTL-to-GDSII flow possible. Experimental results using the ISPD 2018 Contest circuits showed that, compared to the standard global routing solutions from OpenROAD, GRCMO is able to reduce the estimated wirelength by 0.43%, on average, and from 0.5% up to 0.7% for the biggest circuits, requiring less than 20 iterations and moving 2.61% of the cells, on average.

Keywords: Physical Design, Placement, Global Routing, Legalization, Routing with Cell Movement, open source.

RESUMO

No design físico de circuitos VLSI, as etapas de posicionamento (*placement*) e roteamento (*routing*) são responsáveis por determinar as posições das células e interconectar os pinos das células. Essas etapas têm um impacto significativo na qualidade do layout do circuito. Originalmente, o posicionamento e o roteamento eram resolvidos separadamente em uma abordagem de divisão e conquista para lidar com a complexidade do design, o que resultava em um desacoplamento que frequentemente levava a soluções de baixa qualidade. Para mitigar esse problema, recentemente surgiram algumas técnicas que buscam integrar essas etapas. Neste projeto final de curso, apresentamos o GRCMO, uma técnica que otimiza o roteamento global movendo algumas células para suas medianas, implementada em uma das plataformas de código aberto mais completas, OpenROAD, tornando possivel o seu uso em um fluxo *RTL-to-GDSII* completo. Os resultados experimentais utilizando os circuitos do Concurso ISPD 2018 mostraram que, em comparação com as soluções padrão de roteamento global do OpenROAD, GRCMO é capaz de reduzir o comprimento estimado do fio em 0,43%, em média, e entre 0,5% e 0,7% para os maiores circuitos, necessitando de menos de 20 iterações e movendo, em média, 2,61% das células.

Palavras-chave: Design Físico, Posicionamento, Roteamento Global, Legalização, Roteamento com Movimentação de Células, *open source*.

LIST OF FIGURES

LIST OF ALGORITHMS

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

CAD	*Computer-Aided Design*
DRV	*Design Rule Violation*
EDA	*Electronic Design Automation*
GDS	*Graphic Data Stream*
HDL	*Hardware Description Language*
ILP	*Integer Linear Programming*
RMST	*Rectilinear Minimum Spanning Tree*
RSMT	*Rectilinear Steiner Minimum Tree*
RTL	*Register Transfer Level*
ST	*Spanning Tree*
VLSI	*Very Large-Scale Integrated*

LIST OF SYMBOLS

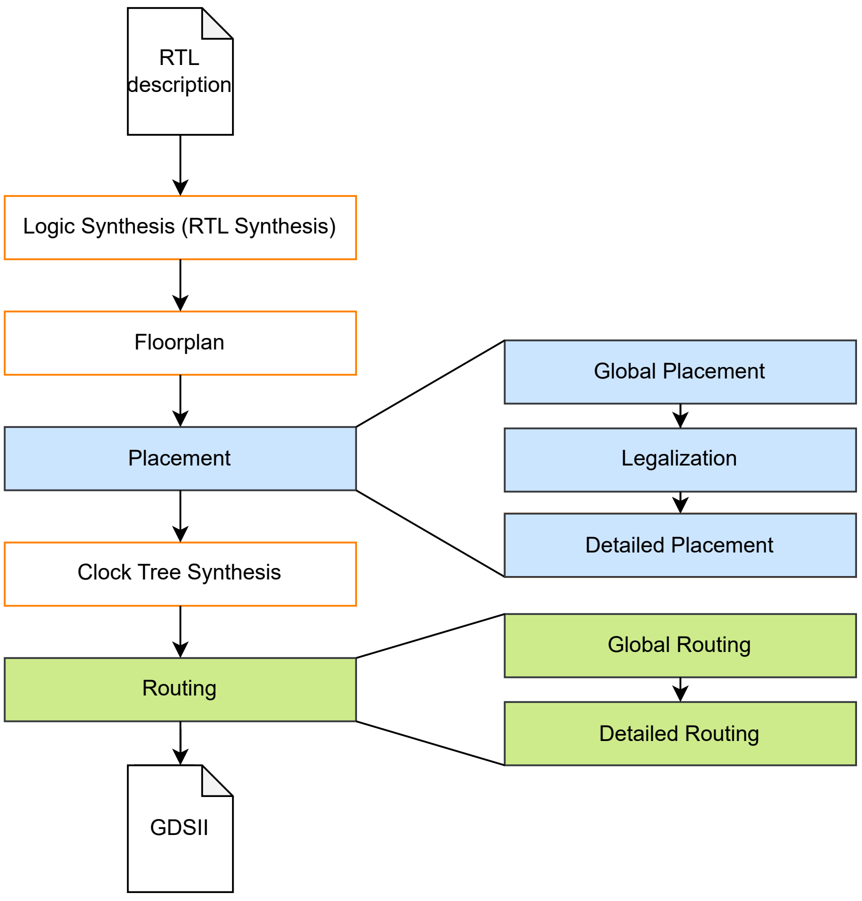
c_i	Cell i
C	Set of Cell
r_j	Row j
R	Set of Rows
w_i	Weight of cell i
n_k	Net k
$N(c_i)$	Set of nets connected to cell i

CONTENTS

1 INTRODUCTION

The design of *Very Large-Scale Integrated* (VLSI) circuits is highly complex, and such complexity continues to increase as the manufacturing technologies continue to advance towards smaller nodes (KAHNG et al., 2011). As a result, the use of specific tools for designing such circuits has become essential. These tools, called *Electronic Design Automation* (EDA), automate and make all stages of VLSI circuit design feasible. Figure 1 presents a simplified VLSI design flow, usually referred to as RTL-to-GDSII flow, where RTL stands for Register Transfer Level and GDS Graphic Data Stream. The flow begins with an abstract description of the circuit, in terms of structure and behavior, and through successive synthesis stages, progressively adds information until reaching a description that will be sent for manufacturing. At each synthesis stage, one or more EDA tools are associated.

Figure 1 – RTL-to-GDSII flow for VLSI circuit design.



Source: Adapted from Kahng et al. (2011).

Although the VLSI design flow is usually illustrated as a linear sequence of separate steps, the result of each step depends on the results of the previous ones. Therefore, a poor placement solution, for example, will negatively impact the quality of routing and may even make routing impossible. In such cases, it would be necessary to go back in the flow to redo the placement.

In light of this problem, EDA researchers from industry and academia proposed the topic of "Routing Optimization with Cell Movement" as one of the challenges in the 2020 and 2021 editions of the international *Computer-Aided Design* (CAD) Contest@ICCAD (HU; YANG, et al., 2020) (HU; YU, et al., 2021), demonstrating the industry's strong interest in finding new solutions with the integration of different steps of the VLSI design flow.

These contests stimulated a few researches in techniques to integrate the placement and routing steps, such as Starfish (WANG et al., 2021), ATLAS (ZANG et al., 2022), CR&P CRP2.0 (AGHAEEKIASARAEE et al., 2022), CRP2.0 (AGHAEEKIASARAEE et al., 2023), ILP-GRC (FONTANA; AGHAEEKIASARAEE; NETTO; ALMEIDA; GANDH, et al., 2021) and ILPGRC (FONTANA; AGHAEEKIASARAEE; NETTO; ALMEIDA; GANDHI, et al., 2023). Although the aforementioned works bring contributions to the routing with cell movement problem, they also have some limitations. For instance, it is difficult to evaluate the outcomes of Starfish, ATLAS or ILP-GRC solutions, since they do not legalize the circuits. In addition, all those techniques are implemented as standalone tools, lacking an integration with a complete open-source flow that could allow going from RTL to GDSII.

Given such gaps, this work proposes a new technique to improve global routing solutions based on cell movement called GRCMO. The proposed technique is implemented fully within the open-source platform OpenROAD (KAHNG, 2022), which is currently one of the most prominent open-source tool kits for VLSI design.

1.1 OBJECTIVES

In this context, this final course project developed GRCMO, a global routing optimization technique that moves cells and reroutes affected nets. Such technique receives as input a global routing solution and its goal is to reduce the circuit total wirelength. GRCMO was implemented using the infrastructure of the open-source OpenROAD project.

Specific objectives

- Propose and implement a routing optimization with cell movement technique within the OpenROAD, enhancing the tool flow;
- Evaluate the results obtained when the proposed technique is applied;

- Compare the obtained results with the conventional OpenROAD flow, using benchmarks from the EDA community.

1.2 CONTRIBUTIONS

With the objectives mentioned above, our work produced the following contributions:

- A post global routing cell movement technique focused on reducing wirelength, generating a valid new global routing and placement solution;
- GRCMO fully integrated with the OpenROAD tool;
- An average of 0.43% reduction on estimated wirelength is achieved when the technique is applied to the ISPD28 (MANTIK et al., 2018) contest benchmark.

These contributions are also reported in a paper that was accepted for oral presentation and inclusion in the proceedings of the 16th IEEE Latin American Symposium on Circuits and Systems (LASCAS 2025).

1.3 STRUCTURE

The next chapters are organized as follows. First, Chapter 2 introduces the necessary background. Chapter 3 discusses the most recent related work on global routing optimization with cell movement. Chapter 4 introduces the proposed technique GRCMO. First, we present the global routing with cell movement challenges and possible improvements, then all the steps of GRCMO are detailed. Chapter 5 presents the experiments conducted with the proposed technique. Finally, Chapter 6 draws the conclusions of this work.

2 BACKGROUND

In this chapter, we present some theoretical concepts necessary for a better understanding of this work. First, we introduce the VLSI flow and explain the basic concepts of the global routing step. Then, the concepts of the median of a cell and the *Rectilinear Steiner Minimum Tree* (RSMT) are presented. Finally, we present the Abacus legalization algorithm.

2.1 VLSI DESIGN FLOW

The VLSI design flow makes use of predefined layouts for complex blocks and simpler structures, called macro blocks and cells (also known as standard cells), respectively. These cells are organized into standard cell libraries, which contain a set of layouts for different logic components (such as OR, NOR, NAND gates) and sequential elements (like flip-flops and latches).

Logic synthesis — or RTL synthesis, as it is referred to in the industry — takes as input a description of the circuit in RTL using a *Hardware Description Language* (HDL) such as Verilog or VHDL, and converts such description into an equivalent one, but at the logic level. Initially, this description undergoes a series of technology-independent logic optimizations. Then, the logic expressions are mapped onto a network of combinational and sequential elements, using the layouts defined in the chosen standard cell library. Another round of optimizations is then performed. The result of this step is a file containing a list of logic and sequential elements and their respective connections, referred to as a netlist. The physical synthesis stage consists of the following steps shown in Figure 1: floorplanning, placement, clock tree synthesis and routing. Due to the high complexity the placement and routing steps are further divided into global placement, legalization, detailed placement, global routing, and detailed routing (KAHNG et al., 2011).

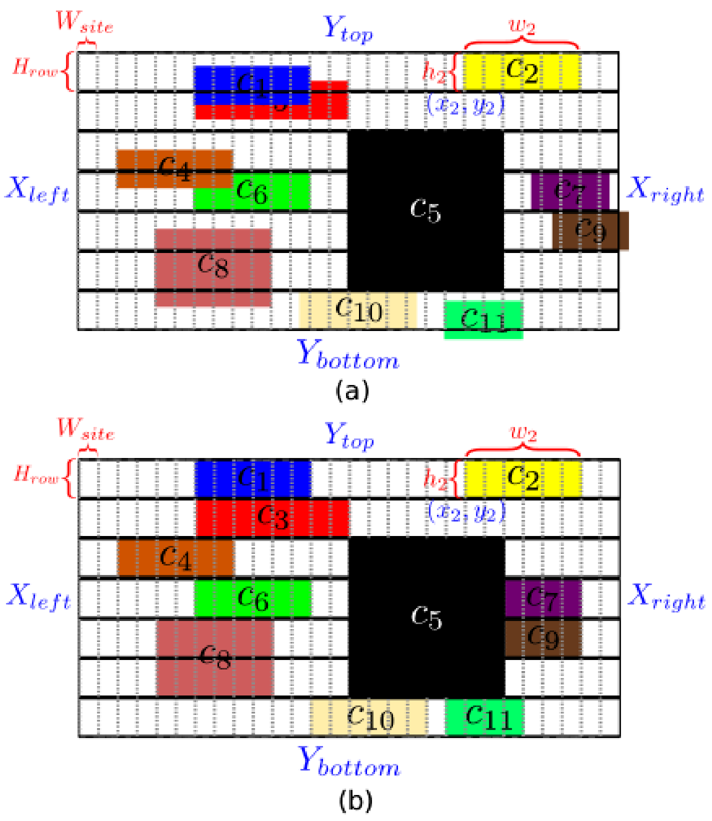
During floorplanning, the chip dimensions are determined, macro blocks are positioned, and the input/output (I/O) pin locations are defined. In addition, very complex modules are partitioned to facilitate the subsequent steps.

In the Placement step, the circuit is divided horizontally and vertically into rows and sites, respectively, and each cell is placed aligned with this subdivision (NETTO, 2022). But global placement assigns to the cells approximate positions on the chip's surface, with no concern for cell overlaps or misalignment with the rows and sites of the layout.

To solve these problems a Legalization step is applied, which moves cells to ensures that all cells are fully contained within the chips boundary, aligned with the rows and sites and have no overlaps. During the legalization step it is important to try to preserve the global placement solution quality, so the main goal of this step is to fix all the placement violations while minimizing the cells displacement. The final stage of placement, called detailed placement, performs a series of optimizations such as reducing cell density and total connection length of the nets.

Figure 2a illustrates a placement with violations, containing cells overlaps (c4 and c6, c1 and c3), misalignments to rows defined by Hrow (c8), misalignments to sites defined by Wsite (c7 and c10), and cells that are outside of chip boundaries (c9 and c11). Figure 2b presents a legalized solution derived from the previous one.

Figure 2 – Example of a circuit before legalization (a) and after legalization (b).



Source: Netto (2022).

The clock tree synthesis stage is responsible for determining the topology of the clock distribution network for all sequential elements in the circuit (flip-flops and latches), while also meeting the specified timing constraints for the clock.

Once detailed placement is completed and the clock tree is generated, the global routing is responsible to perform all the connections of cells listed on the netlist. For this task, current technologies offer up to 12 metal layers above the transistors, all metal layers are divided into a grid of large rectangles called GCells. Each GCell has a limited number of wire segments that can pass through it, that is determined by the number of metal tracks available in the GCell. Furthermore, the tracks have a preferred direction depending on the metal layer, which alternates between adjacent metal layers. Thus, global routing involves finding a sequence of GCells that must be traversed by every net to enable their connections, while respecting the capacities of the GCells and minimizing the total connection length, using a wirelength estimation metric. In this work, such metric estimation is the length of the segments generated by the global routing solution.

Figure 3a shows an example of the division of GCells grid for three metal layers: GCells are presented by solid rectangles, whereas the tracks available in each layer are represented by dashed lines. In figure 3a, the odd layers have a horizontal preferred direction, while in even layers the preferred direction is vertical.

To solve the global routing problem the GCells are modeled as a graph, where each GCell is a node, neighbor GCells connected by a track have an edge connecting their respective nodes. For the modeling of this graph there are 2 main approaches: Use a 3D graph or use a 2D graph with an extra step to define the layer of the guides.

In a 3D graph, neighbor GCells in the same layer have an edge connecting their respective node, therefore the edges follow the preferred direction of each layer. Alongside with the neighborhood in the same metal layers, GCells on adjacent layers that are aligned also have an edge connecting their nodes. Figure 3b shows the graph modeling of the routing space presented in 3a, where the edges between nodes in the same layer are represented by solid lines and edges between aligned nodes in adjacent layers (vias) are represented by dashed lines.

In the 2D graph solution the GCells and the pins a projected onto a 2D plane and the vertical and horizontal edges of all the layers are combined to form a grid. Figure 3 (c) shows a 2D grid graph modeling of the (a), all the GCells aligned between layers are combined into 1 node and are connected by horizontal and vertical edges. With this approach we only find the 2D topology of the nets connection, therefore another step must be performed to determine the layers of the nets connections.

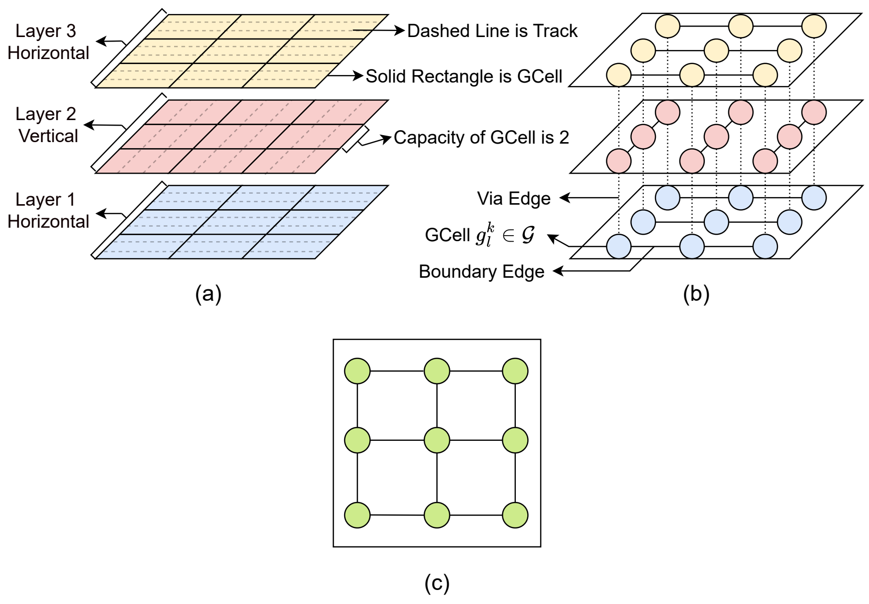
Finally, detailed routing is responsible for determining the specific track within each GCell that will be used to complete the routing of each connection, and some final optimizations are performed.

2.2 MEDIAN OF A CELL

As presented in Goto (1981), the authors define the median of a module as a point where the total wirelength for the nets associated to this module is minimum, that is, the position for this module that generates the shortest connections to its nets. Goto (1981) proposed an iterative method to find the median position of a module. In this work, we will use a heuristic to find an approximation for the median, inspired by Fontana, Aghaeekiasaraee, Netto, Almeida, Gandhi, et al. (2023). Figure 4 illustrates how the optimal region for moving the cell is determined.

In Figure 4, the cell under analysis, C3, is connected to five other cells: C1, C2 C4, C5 and C6, through three nets: red, green and yellow, whose smallest enclosing rectangles (called bounding boxes) are shown in red, green, and yellow, respectively. The optimal region is the median of these bounding boxes, which is outlined by the purple rectangle.

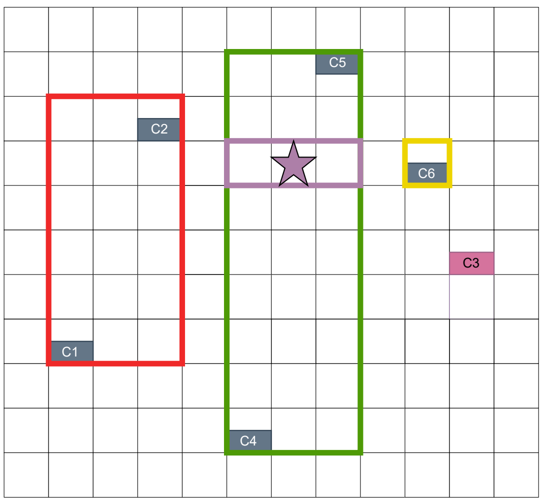
Figure 3 – Global routing GCell modeling.



Source: Adapted from Fontana (2023).

We then define the median of C3, as the midpoint of this region, represented by the purple star.

Figure 4 – Example of finding the median GCell for cell C3.



Source: Adapted from Fontana, Aghaeekiasaraee, Netto, Almeida, Gandhi, et al. (2023).

2.3 RECTILINEAR STEINER MINIMUM TREE (RSMT)

As stated in Section 2.1, during the global routing phase, the circuit is represented by a graph of GCells. Using this representation, graph algorithms can be employed to determine the sequence of GCells that route a net with the minimum distance. One such algorithm is the RSMT, which finds the routing with the minimal wirelength in a 2D GCell graph model.

A RSMT is based on the graph concept of a *Spanning Tree* (ST). According to Kao (2008), a ST is a set of edges in a graph that contains no cycles and connects to a set of defined nodes. In a weighted graph, a ST is minimal if the sum of the edge weights is minimized. When the graph's edges are weighted by the rectilinear distance between nodes, the resulting minimal spanning tree is known as a *Rectilinear Minimum Spanning Tree* (RMST).

A Steiner Tree (KAHNG et al., 2011) generalizes the concept of a Spanning Tree by allowing additional nodes, called Steiner Points, to be introduced in the predefined set of nodes. These points help reduce the total edge cost, making the tree's total wirelength lower than that of a standard RMST. If this Steiner Tree is minimal and the graph's edges are weighted by rectilinear distance, it is referred to as a RSMT.

Kahng et al. (2011) state that, although RSMTs are optimal in terms of wirelength, they are not always the best choice for routing a net. Instead, they are commonly used to estimate the routing length or provide an initial routing solution for the net (LIU et al., 2013) (CHANG et al., 2010) (MOFFITT, 2008).

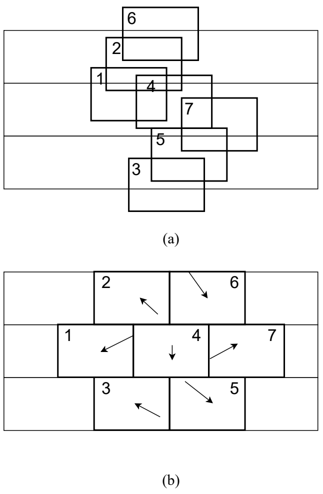
2.4 ABACUS ALGORITHM

The Abacus legalization algorithm was presented by Spindler, Schlichtmann, and Johannes (2008) as a fast and simple legalization algorithm for standard cells in a circuit, with the goal to minimize the cells displacement. Figure 5 shows an example of a global placement solution for a set of cells (a), and the same cells after applying Abacus (b), with the displacement of the cells shown by the arrows.

Algorithm 1 shows the pseudo-code of the Abacus. It receives as an entry a global placement solution and starts by sorting the cells according to their x-position. Each cell c_i in the sorted set of cells C is then legalized. When legalizing a cell, abacus tries to place the cell in every row r_j of the circuit R, using the function PlaceRow, that places and legalizes c_i in r_j. After placing the cell in r_j, a cost is computed for moving the cell to the row, the simplest cost considers only the displacement of the cell. Then, the cell is placed on the row that produced the minimum cost.

The placement of a cell in a row is the core of the algorithm, here Abacus starts by placing the cell on the row with the same x-position it had in the global placement solution. Then, all the cells with overlaps are placed side by side and clustered. When

Figure 5 – Example of a circuit legalization by Abacus.



Source: Adapted from Spindler, Schlichtmann, and Johannes (2008).

cells are clustered, the created clustered is shifted to the left to minimize the cost of the cells. After shifting the cluster, it is necessary to check if new overlaps occur. If so, the new cells are added to cluster and the cluster is shifted once more. This is done until no overlaps occur. In the end, abacus is able to generate a legalized solution with a small displacement of all cells.

Algorithm 1: Abacus legalization

```
   Input  : Illegal placement
   Output : Legal placement
1  Sort cells $C$ according to x-position;
2  foreach $c_i \in C$ do
3      $cost_{best} \leftarrow \infty$;
4      foreach $r_j \in R$ do
5          Insert $c_i$ into $r_j$;
6          PlaceRow $r_j$ (trial);
7          Determine $cost$;
8          if $cost < cost_{best}$ then
9              $cost_{best} \leftarrow cost$;
10             $r_{best} \leftarrow r_j$;
11         end
12     end
13     Insert $c_i$ into $r_{best}$;
14     PlaceRow $r_{best}$ (final);
15 end
```

3 RELATED WORK

The following sections present the most recent works focused on integrating the stages of Global Routing with Placement, using cell movement during Global Routing.

3.1 STARFISH

In this, work Wang et al. (2021) propose a technique named Starfish, which moves cells and reroutes the affected nets using an A*-based algorithm to find the shortest routes to the previous routing topology. The position of the cells are chosen from a lookup table that contains the best candidate positions and improvements estimation. The experimental results were evaluated using the ICCAD CAD Contest 2020 benchmarks.

3.2 ATLAS

In Zang et al. (2022), the technique ATLAS is presented. In this technique, as a first step cluster of cells are moved to optimize the number of vias, on a second stage of movements individual cells are moved. In both stages the movements go to the median and the affected nets are rerouted using the search algorithm A*. ATLAS was tested using the ICCAD CAD Contest 2020 and 2021 benchmarks, achieving better performances than the first place candidates of the contest.

3.3 CR&P AND CRP2.0

The works CR&P (AGHAEEKIASARAEE et al., 2022) and CRP2.0 (AGHAEEKIASARAEE et al., 2023), focus on solving the problem of highly congested areas, to alleviate these regions a set of cells inside them are moved, an *Integer Linear Programming* (ILP) model is used to legalized these movements. Finally, after all movements were applied, new routing is made with a pattern routing algorithm and an A* algorithm. Furthermore, CRP2.0 speeds up the technique using a caching approach.

The results of both CR&P and CRP2.0 were analyzed after the Detailed Routing using the benchmarks from the contests ISPD 2018 and 2019, presenting reduction on the total number of vias and the total wirelength.

3.4 ILP-GRC AND ILPGRC

In Fontana, Aghaeekiasaraee, Netto, Almeida, Gandh, et al. (2021), an ILP model for routing with cell movement is proposed, which serves as the core of the technique named ILP-GRC$_{V0}$. For every cell a set of candidate positions and candidate routing are generated, the positions candidate are the cells original position and the median of that cell, then for both these positions the routing candidate are generated as all pattern

Table 3 – Related Routing with Cell movement works.

Work	**Year**	**Legalize?**	**Evaluation after**	**Objective**	**Advantage**
Starfish	2021	No	Global Routing	wl	Lookup table
ILP-GRC	2021	No	Global Routing	wl	ILP routing
ATLAS	2022	No	Global Routing	wl	Clustering
CRP	2022	Yes	Detailed Routing	wl	ILP detailed place
CRP2.0	2023	Yes	Detailed Routing	wl	Speedup
ILPGRC	2023	Yes	Detailed Routing	via & wl	Paralelizing
GRCMO	**2024**	**Yes**	**Global Routing**	**wl**	**Integrated within a complete flow**

routing possible. Finally, the ILP model decides which candidate position and routing will be assigned for every cell. The results were analyzed using the ISPD 2018 competition benchmarks, showing better results than commercial tools in terms of total wirelength.

In the extension of the work, ILPGRC (FONTANA; AGHAEEKIASARAEE; NETTO; ALMEIDA; GANDHI, et al., 2023), a circuit partitioning method using a technique called "Checkered paneling" is proposed to make the ILP model execution parallelizable, reducing execution time for large circuits. Additionally, this new work proposes a legalization for the circuit based on clusters of GCells, aiming to minimize disturbance to non-moved cells. The ILPGRC technique was evaluated using the ISPD18 and ISPD19 contest benchmarks, demonstrating an improvement in the total number of vias after Detailed Routing.

3.5 SUMMARY

In this chapter we reviewed other works related to the routing with cell movement problem. Table 3 summarizes the works, highlighting their objectives and characteristics.

All works have as an objective to reduce wirelength, apart from ILPGRC that has as objective to reduce the number of vias and wirelength. Starfish, Atlas and ILP-GRC do not have a legalization step for their placement solution, and thus can not perform a detailed routing, making it difficult to accurately evaluate their results. CR&P and CRP2.0 have a legalization step using an ILP solver and analyze their results after detailed routing. The extension of ILP-GRC, ILPGRC, also has a legalization step and proceeds the flow to the end of the detailed routing. However, none of the aforementioned works are integrated with a complete VLSI design flow.

With this, our proposed technique was implemented within the OpenROAD tool, one of the most complete open-source VLSI design flow. Our technique objective is to reduce the estimated wirelength, that is evaluated after global routing, however we apply a legalization step, allowing to perform detailed routing on future works.

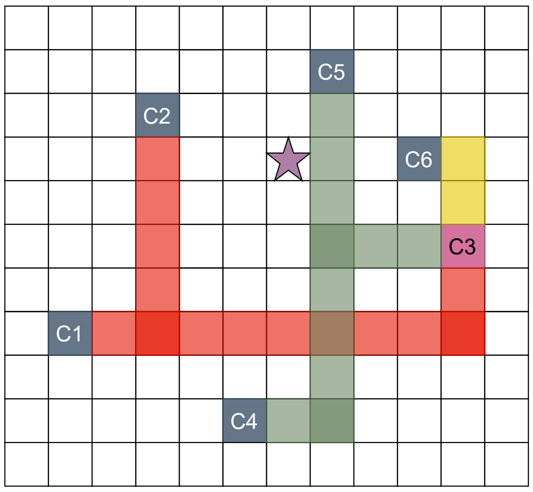
4 METHODOLOGY

This chapter first presents a formulation for the routing with cell movement problem, its challenges and impacts on the designs. Then, the proposed technique is introduced and every steps are detailed. Finally, we explain how the technique was integrated with the OpenROAD tool.

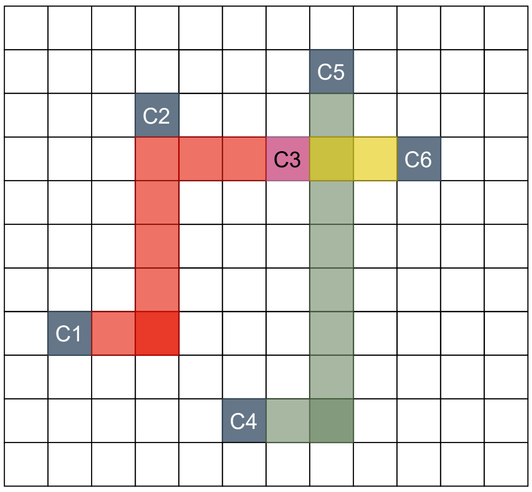
4.1 ROUTING WITH CELL MOVEMENT PROBLEM FORMULATION

As previously presented, due to its complexity, the integrated circuit design flow is divided into stages, which are handled separately. Despite being separate, the results of one stage strongly depend on the outcomes of the previous stages. Because of this, studies have been conducted to achieve better integration between some of these stages. In particular, the integration of Global Routing with Placement gained prominence following the 2020 and 2021 editions of the international CAD Contest@ICCAD (HU; YANG, et al., 2020) (HU; YU, et al., 2021), which featured "Routing with Cell Movement" as one of the proposed challenges.

Figure 6 – Example of a routing optimization by moving cell C3 to its median.



(a) Initial routing solution, total length is 35 GCells.



(b) Routing solution after moving cell C3, total length is 27 GCells.

Source: The author.

The global routing stage receives a legalized placement solution as input. Therefore, the global routing solution must adhere to the placement rules along with its own routing rules. Specifically, a valid global routing solution must ensure that all cells are aligned with the circuit's rows and sites, that there are no overlaps between cells, be fully contained within the circuit boundaries and that all nets are connected to their pins through a sequence of GCells. Additionally, the number of segments passing through a GCell must not exceed its capacity. In a standard global routing flow, placement rules are generally not a concern, as cells have fixed positions. However, in this work, we propose a technique to improve an initial global routing solution by relocating a set of cells. After moving a cell, it is necessary to ensure that the cell's new position respects placement rules. Moreover, to comply with routing rules after the movements, all nets with disrupted connections must be rerouted in accordance with GCell capacities.

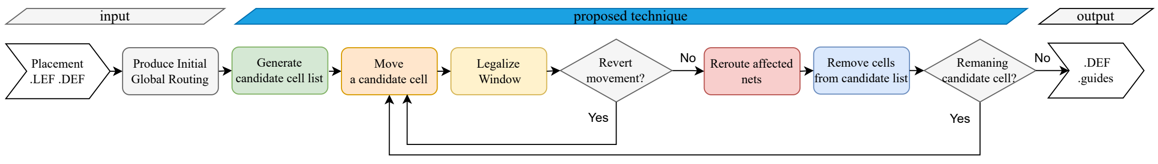
An example of a routing optimization can be seen in Figure 6. 6a shows an initial routing solution for 3 nets: red, green and yellow that are connected to a cell C3, and the median GCell of C3 is marked by a purple star. This initial routing solution has a total lenght of 35 GCells. In 6b, a new routing solution is shown after moving C3 to its median. Now, the new total length is 27, that is, we are able to reduce the total length by 8 GCells.

4.2 PROPOSED TECHNIQUE

This section presents the proposed technique to solve the problem described in the previous section. Figure 7 shows an explanatory diagram of this technique. As input, the technique receives an initial Global Routing solution and two files in standard formats used in the industry, the Library Exchange Format (LEF) and the Design Exchange Format (DEF). The .LEF file describes the technology that will be used for the chip fabrication, while the .DEF file contains the physical characteristics of the chip and its elements. The technique then applies the following steps to the initial solution:

1. Generate candidate cell list;
2. Move a candidate cell;
3. Legalize the window;
4. Reroute affected nets;
5. Remove cells from candidate list.

Figure 7 – Diagram of the proposed technique.



Source: The author.

The flow of the technique begins by selecting the cells to move and determining the cells new (target) positions. The cells are stored in the candidate cell list according to their potential to improve the routing solution. With the candidate list generated, we start moving the cells, initially the first cell on the candidate list, that is, the cell with the highest potential to improve the routing solution, is moved to its target position. Then a window, that is 3 rows high and 30 GCells wide, around the new position of the cell is legalized using the Abacus algorithm (SPINDLER; SCHLICHTMANN; JOHANNES, 2008). At the end of the legalization, we check if the position of the cell is still better then its original. If it is worse, we revert the movement. If the position after legalization is still considered beneficial, we confirm the movement and reroute the nets affected by the movement and legalization. At last, we remove the moved cell from the candidate list. These steps are better detailed in the following subsections.

Once the candidate cell list is empty the .DEF file information is updated with the cells new positions and the global routing result is written in the .guides file.

4.2.1 Generation of the candidate cell list

GRCMO flow begins by generating the candidate cell list, a structure that contains the cells with the highest potential to improve global routing solution if moved. In order to determine this potential, we first need to assign a new position for each cell and then compare such position with the original one. Considering the objective of reducing the total estimated wirelength of the circuit, in this work we take inspiration from Pan, Viswanathan, and Chu (2005) and define the target position for each cell as its median GCell, which is found using the methods presented in Section 2.2.

Once the target position is defined for all cells, we estimate the potential of improvement for every cell. This potential is computed comparing the estimated RSMT wirelength for nets connected to that cell for the original and the target position. With this, the cells that have a RSMT smaller if the cell is moved will be defined as cells with highest potential. Equation (1) formulates this potential as a weight function, in which the the weight w_i of cell c_i is the sum of the difference between the RSMT wirelength before and after moving the cell to its median GCell, considering every net n_k in $N(c_i)$, the set of nets that connects to c_i.

$$w_i = \sum_{n_k \in N(c_i)} RSMTWL_{before}(n_k) - RSMTWL_{after}(n_k) \tag{1}$$

Then, the candidate cell list is composed of the cells that have a weight greater than 0. In order to focus the movements on the cells that offer the best improvements on wirelength, the candidate list is ordered by the cell's weights so the cells with the highest weights are on the list head.

4.2.2 Movement of a candidate cell

With the candidate cell list generated, we start moving the cells one by one, until the candidate list is empty. Each cell taken from the candidate cell list is then moved to its determined target position.

4.2.3 Legalization of a moved cell

The movements as described in the previous section do not take into consideration the placement rules. Hence, as discussed in section 4.1, a legalization step is required so as to ensure that all cells have legal positions and thus the technique gives a valid global routing solution.

Due to the fact that in the original global routing solution all cells were placed in legalized positions, it is necessary to legalize only the moved cell. In light of this, to minimize the displacement, we apply the Abacus algorithm (SPINDLER; SCHLICHTMANN; JOHANNES, 2008) inside a window that is 3 rows high and 30 GCells wide, with the median GCell as the center. This size was chosen after some testing with few different window dimensions. Since the cell is in the center of the window we ensure that after the legalization will be moved at most 1 row up or down, and 15 GCells to the side.

Also, with a smaller window we limit the number of cells other than the chosen cell to be affected by the legalization, helping reduce the displacement. If the legalization algorithm fails to achieve a solution, it means that there is not enough space in the legalization window to fit the moved cell. In this case, we reverse the movement, placing the cell back in its original position.

Even though the legalization window limits the displacement of the moved cell, it can still be displaced too far from its median GCell. Therefore, a check is performed to confirm that the movement is still beneficial. This is done by computing the cells weight again, comparing the RSMT wirelength of the nets connected to the moved cell, in both the cell's original position and its new position after the legalization. If the new weight of the cell is smaller than 0, the movement after legalization is worse than the cell in its original position and we revert the movement, returning the cell to its original position. Otherwise, we confirm the movement and proceed to the next step in the flow.

4.2.4 Rerouting of affected nets

If the moved cell still has a weight higher than 0 (zero) after the legalization step, the cell has its exact position defined and we can proceed to rerouting the nets connected to it. In addition, during the legalization process, other cells may have had their positions adjusted to accommodate the cell that was moved to its legalized position. Therefore, it is necessary to identify all the nets that were impacted by the movement of the candidate cell. These nets will be rerouted using the Incremental Routing tool from OpenROAD, whose functionality will be explained in Section 4.3.

4.2.5 Remove cells from candidate list

Once the affect nets are reroute, the candidate cell list is updated. Here the moved cell is removed from the list, also the cells that are part of one of nets connected to the moved cell are removed. This second set of cells need to be removed because we want to preserve routing improvements obtained for the nets connected to the moved cell, if another cell in these nets is moved, these improvements can be lost. On top of that, the weight of these second set of cells are no longer valid, since its median GCell might depend on the moved cells old position.

At the end of this step, if the candidate list is not empty the technique returns to the second step, where a new candidate cell will be selected, moved, legalized and rerouted. When all the cells from the candidate list are removed and the list is empty, the technique proceeds to generated the files with the results, the .DEF file is updated and a .guide file is written with the global routing solution.

4.3 INTEGRATION WITH OPENROAD

Alongside with proposing a routing technique with cell movement, another objective of this work is to implement this technique within an open-source software environment. The goal is to facilitate future research related to this topic, encompassing new cell selection criteria, movement strategies, net rerouting, among other aspects. In this context, the present work is integrated into the OpenROAD EDA design environment (OPENROAD, 2024). OpenROAD has a comprehensive set of tools that entirely supports the physical design flow of a circuit, in addition to libraries and data structures that assist in the development of new techniques for the various stages of the circuit design flow.

To perform this integration, a new tool was developed within the OpenROAD project, called GRCMO. This tool has direct access to key data structures, allowing it to manipulate circuit elements such as cells, nets, and their characteristics. GRCMO uses these structures to carry out the steps of the technique presented in the previous section. During the cell selection and movement stages, OpenROAD's structures for handling these components and nets are employed to calculate the weight and median position of the cells. Specially on the weights as the RSMT are generated using the OpenROAD implementation of the FLUTE algorithm (CHU, 2004). Additionally, the cell structure provides the necessary support for moving and storing information related to the cell's new position and its pins.

For the legalization step, OpenROAD does not have an implementation of the Abacus algorithm. For this reason, an implementation of this algorithm was used, developed

Table 4 – Overview of tools used by step and who developed.

Step	Tool	Developed by
Genarate candidade cell list	**Cell median**	**Author**
	Flute	OpenROAD
	Cell weight	**Author**
Move candidate cell	None	None
Revert movement?	Flute	OpenROAD
	Cell weight	**Author**
Legalize window	Abacus	Felipe Savi
Reroute affected nets	FastRoute	OpenROAD
Remove cells from candidate list	None	None

by Felipe Savi, a colleague from the ECL lab, within the OpenROAD project. This implementation receives the coordinates that define an area of the circuit where legalization will be performed. Then, using the cell structures and the physical characteristics of the circuit, such as rows, columns, and boundaries, it applies the legalization algorithm.

In addition to the structures mentioned above, another important integration between the GRCMO tool implemented in this work and the OpenROAD project is with the Incremental Routing tool, already available in the project. As mentioned in the previous section, after the cells are moved and their new positions legalized, it is necessary to reroute all the nets that had their connections invalidated by the movements made. The Incremental Routing tool is essential for executing this step. It uses the FastRoute algorithm (PAN; XU, et al., 2012) to perform a new Global Routing exclusively for the nets provided to it, in this case the affected nets, while preserving the initial routing of the remaining nets. Thus, by using this tool, the necessary changes are made for the affected nets without disrupting the rest of the circuit's routing.

Table 4 shows the tools that where used by each step and who developed it. On the first column is the step, on the second column the tool and on the final column who developed. In total 2 of the used tools where from the OpenROAD: Flute and Incremental Routing. And 3 were implemented during this work: Cell median, Cell weight (by the author) and Abacus (by Felipe Savi).

5 RESULTS

This chapter discusses the initial experiments conducted to evaluate the technique proposed in this work. Section 5.1 details how this initial experiment was carried out, while Section 5.2 presents the results obtained from conducting these experiments. Finally, section 5.3 brings a deeper analysis of the techniques results on the 3 most complex test cases.

5.1 EXPERIMENTAL METHODOLOGY

The experiments were performed in a Linux 22.04.4 LTS machine with AMD Ryzen® 5-5500 CPU running at 2.10 GHz and 16GB of RAM (3200 MHz).

The evaluation of the proposed technique was made using the benchmark from the ISPD18 (MANTIK et al., 2018) Contest. This benchmark is composed of 10 circuits placed and legalized with different specifications such as technology node, number of cells and number of nets. Table 5 presents the detailed information of each circuit and Table 6 comments the description of each circuit. These benchmarks have LEF and DEF files with a placement solution, and the circuits use one of two technology nodes: 45nm or 32nm. This benchmark suite also provides technology information for circuits such as standard, macro, and IO cells. Finally, cells have complex pin shapes such as L, Z, and others. However, for this benchmark no power or timing information is provided. In this work, we did not evaluate circuit test 10 because it has 100% density and, therefore, no space for moving cells. We did not use the ICCAD CAD Contest 2020 (HU; YANG, et al., 2020) or ICCAD CAD Contest 2021 (HU; YU, et al., 2021) benchmarks because they do not provide row or site information, making it impossible to legalize the circuits.

Figure 8 presents the experimental flow, we begin with the placement from the ISPD 2018 contest and the initial Global Routing that is generated by the OpenROAD tool using the FastRoute algorithm. The technique is then iteratively applied 20 times. The results regarding estimated wirelength, moved cells, and estimated number of vias

Table 5 – ISPD 2018 CAD contest benchmarks. Source: Mantik et al. (2018).

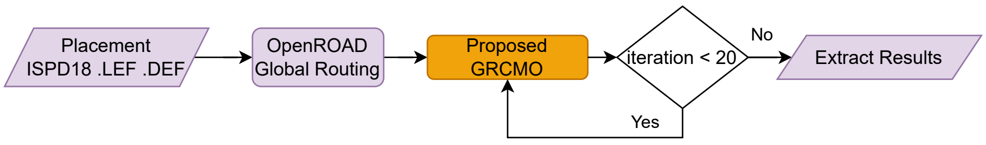
Design	**# Cells**	**# Blk**	**# Net**	**# Pin**	**# Layer**	**Die size**	**Density (%)**	**Tech. node**
test1	8879	0	3153	0	9	0.20x0.19mm2	85	45nm
test2	35913	0	36834	1211	9	0.65x0.57mm2	57	45nm
test3	35973	4	36700	1211	9	0.99x0.70mm2	65	45nm
test4	72094	0	72401	1211	9	0.89x0.61mm2	89	32nm
test5	71954	0	72394	1211	9	0.93x0.92mm2	92	32nm
test6	107919	0	107701	1211	9	0.86x0.53mm2	99	32nm
test7	179865	16	179863	1211	9	1.36x1.33mm2	90	32nm
test8	191987	16	179863	1211	9	1.36x1.33mm2	90	32nm
test9	192911	0	178857	1211	9	0.91x0.78mm2	91	32nm
test10	290386	0	182000	1211	9	0.91x0.87mm2	100	32nm

Table 6 – ISPD 2018 CAD contest benchmarks characteristics. Source: Mantik et al. (2018).

Design	Comments
test1	Standard cell netlist only.
test2	Standard cell netlist with IO pins.
test3	Standard cell netlist with IO pins and block macros.
test4	Design has Metal2 OBS in some of its standard cells.
test5	Design has Metal2 OBS, Metal2 Power/Ground pins, and routing direction is reversed.
test6	Design has Metal2 OBS, Metal2 Power/Ground pins, and reversed routing direction, but without any block macro.
test7	Quad-core design with Metal2 OBS and Metal2 Power/Ground pins as blockage.
test8	Quad-core design with Metal2 to Metal3 OBS and Metal2 to Metal4 Power/Ground pins as blockage.
test9	Quad-core design with Metal2 to Metal3 OBS and Metal2 to Metal4 Power/Ground pins as blockage, no block macro, higher utilization.
test10	Quad-core design with Metal2 to Metal3 OBS and Metal2 to Metal4 Power/Ground pins as blockage, no block macro, extra congested area.

are gathered after each iteration using the OpenROAD tools. The original OpenROAD Global Routing solution is the baseline for the initial contest placement solution.

Figure 8 – Experimental flow.



Source: The author.

5.2 EXPERIMENTAL RESULTS

Table 7 – Experimental results for the ISPD18 benchmarks.

ISPD18 Designs	Cells		Estimated Wirelength (um)			Estimated # VIAs		
	#(K)	Moved %	OpenROAD WL	GRCMO WL	dif WL %	OpenROAD #EV	GRCMO #EV	dif #EV %
test1	3	0.40	124946	124921	-0.02	12213	12225	0.10
test2	3	5.15	2031357	2021869	-0.47	177783	183169	3.03
test3	36	1.16	2223886	2212728	-0.50	178118	183775	3.18
test4	72	3.24	3255627	3237828	-0.55	396694	401632	1.24
test5	72	4.00	3388105	3374124	-0.41	410784	419464	2.11
test6	108	0.29	4494295	4490235	-0.09	628805	630960	0.34
test7	180	3.38	8062032	8018601	-0.54	1282271	1304140	1.71
test8	192	2.94	8093235	8051835	-0.51	1302119	1325827	1.82
test9	193	2.98	6948508	6897316	-0.74	989156	1011347	2.24
Avg		**2.61**			**-0.43**			**1.75**

Table 7 presents the results of the experiments. The first column lists the ISPD 2018 benchmarks, followed by the total number of cells and the percentage of moved cells. The columns labeled *OpenROAD WL* and *GRCMO WL* show the estimated wirelength, in microns, for the Global Routing results obtained by baseline and by the proposed technique, respectively. The column *dif WL %* shows the difference, in percentage, between GRCMO and the baseline results, therefore a negative result means a reduction in the estimated total wirelength. The next columns present the baseline for the number of estimated vias, the number of estimated vias achieved by GRCMO and the percentage difference between GRCMO and the baseline results.

The results show that the proposed technique improved the the estimated total wirelength for all circuits, reducing it on average by 0.43%, while moving 2.61% of the cells, on average. Circuit test9 had the best improvement, reaching a 0.74% reduction and moving almost 3% cells. The lowest improvement was achieved on Test1, the smallest circuit, where we barely affected the design, moving only 0.4% of cells and reaching 0.02% reduction. Similar to this, on test6 only 0.3% of the cells were moved leading to an almost insignificant estimated wirelength reduction, this circuit is highly dense and most of the movements were rejected due to the lack of space to legalize them. Other than these outliers, the rest of the circuit shows similar estimated wirelength reduction, with the percentage of moved cells varying between 1% and 5%.

The estimated number of vias was degraded by GRCMO, increasing on average by 1.75%, the circuits with the highest increase being test2 and test3 with 3. 03% and 3. 18%, respectively. Following what happened with wirelength, due to the low percentage of cells moved, circuits test1 and test6 had the smallest changes on the estimated number of vias.

Its worth noting that GRCMO was able to finish with no overflow on the GCells usage and no placement violations for all the designs.

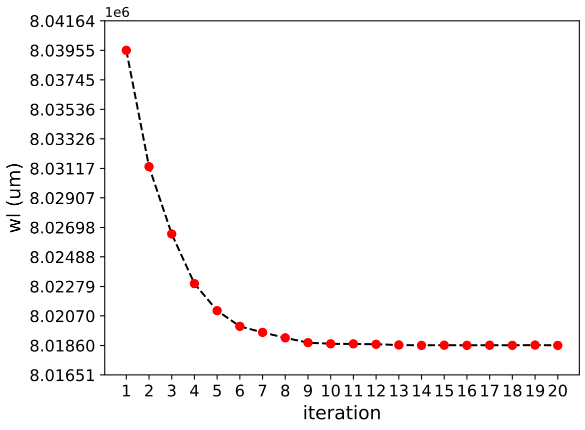
5.3 RESULTS ANALYSIS

For a better understanding of the results, we make a deeper analysis of the results for the circuits test7, test8 and test9. We choose these circuits because, as shown in Tables 5 and 6, they have similar specifications, but with increasing complexity.

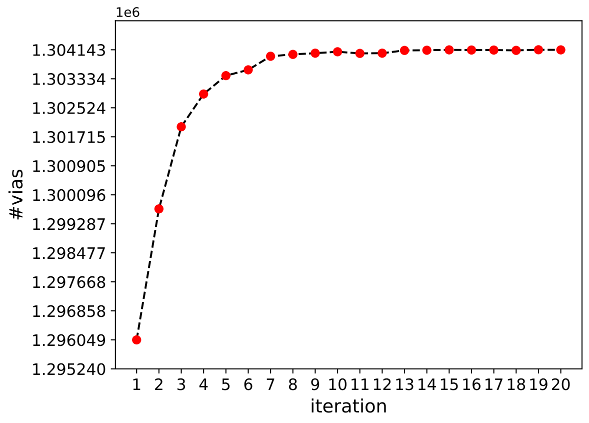
5.3.1 Impacts of Iteration

First, we look at Figures 9, 10 and 11 present the evolution of the technique results throughout the iteration respectively for test7, test8 and test9. with Figures 9a, 10a and 11a presenting the evolution of the estimated wirelength and Figures 9b, 10b and 11b the evolution of the estimated number of vias. The x-axis represents the iterations, and the y-axis the metric in question

Figure 9 – Evolution of estimated wirelength and number of vias through the iterations for test7.



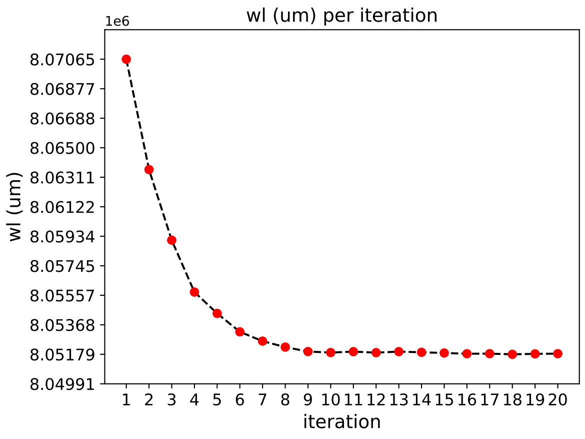
(a) Evolution of estimated wirelength.



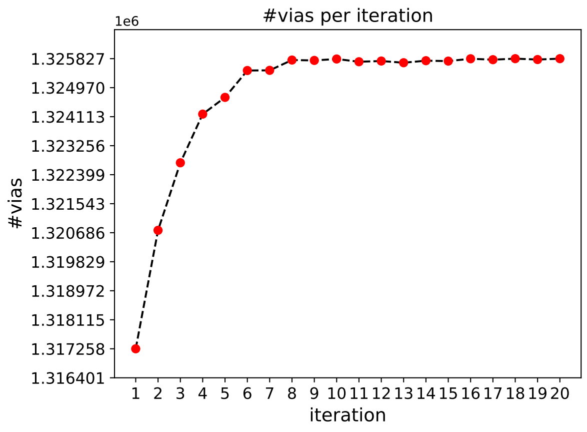
(b) Evolution of estimated number of vias.

Source: The author.

Figure 10 – Evolution of estimated wirelength and number of vias through the iterations for test8.



(a) Evolution of estimated wirelength.



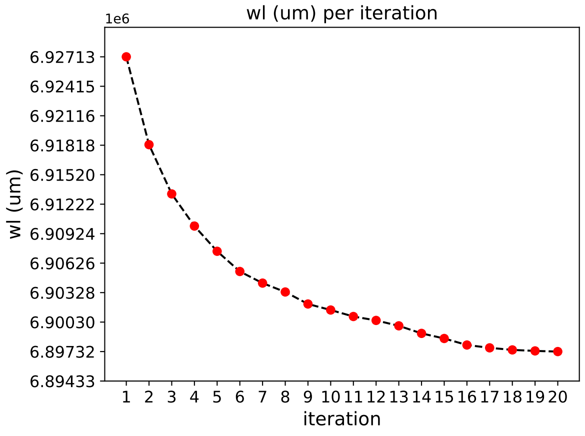
(b) Evolution of estimated number of vias.

Source: The author.

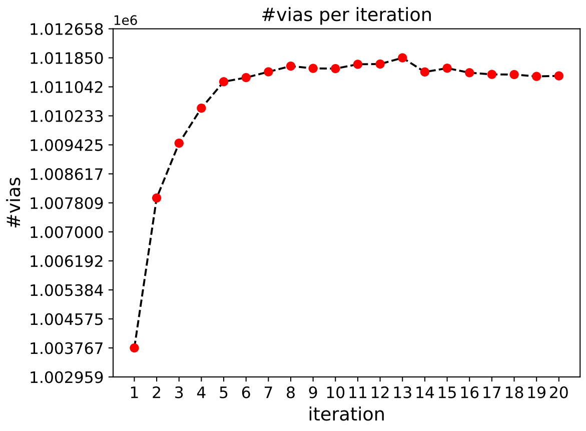
Figure 9a, 10a and 11a shows that the estimated wirelength keeps decreasing as the iterations progress. For test8 and test7, the first 6 iterations are responsible for the most improvements and the following iterations barely affect the results. For test9, the although the first 6 iterations are responsible for the most significant gains, the following iteration still improve the estimated wirelength.

In Figures 9b, 10b and 11b, the estimated number of vias follows a similar behavior as with the wirelength, with the estimated number of vias increasing with each iteration. Just as before, for test7 and test8 the most significant increases in the estimated vias happens in the first 6 iteration and different than on wirelength, for test9 it also stabilizes after 6 iterations.

Figure 11 – Evolution of estimated wirelength and number of vias through the iterations for test9.



(a) Evolution of estimated wirelength.



(b) Evolution of estimated number of vias.

Source: The author.

In conclusion, applying the technique iteratively amplified the results of the first iteration, highlighting its impact beyond a single pass. In the case of estimated wirelength, the results improved with the progression of the iterations, for the estimated number of vias, that degraded after the first iteration, the metric kept degrading as the iterations progress.

Although the technique remains effective throughout all iterations for test9, for test7 and test8 the results stabilize much earlier, around the sixth iteration. Therefore, execution time can be saved by running fewer iterations.

5.3.2 Usage per metal layer

Table 8 – Usage percentage of the available tracks per layer for ISPD18 test7.

	Metal1	**Metal2**	**Metal3**	**Metal4**	**Metal5**	**Metal6**	**Metal7**	**Metal8**	**Metal9**
OpenROAD	17.72%	38.27%	32.86%	29.73%	16.13%	11.34%	3.56%	1.15%	0.93%
GRCMO	17.30%	36.57%	31.44%	29.32%	17.22%	14.00%	4.41%	3.16%	1.08%
Diff	**-0.42**	**-1.70**	**-4.31**	**-1.42**	**1.09**	**2.66**	**0.85**	**2.01**	**0.15**

Table 9 – Usage percentage of the available tracks per layer for ISPD18 test8.

	Metal1	**Metal2**	**Metal3**	**Metal4**	**Metal5**	**Metal6**	**Metal7**	**Metal8**	**Metal9**
OpenROAD	17.49%	38.58%	32.87%	30.36%	16.49%	15.60%	3.78%	1.96%	0.82%
GRCMO	17.07%	36.95%	31.54%	29.81%	17.53%	18.09%	4.52%	4.44%	1.18%
Diff	**-0.42**	**-1.63**	**-1.33**	**-0.55**	**1.04**	**2.49**	**0.74**	**2.48**	**0.36**

The previous sections showed us that after applying our technique, the estimated number of vias increased for all test cases, in order to better understand this we studied the usage of the guides per metal layer before and after applying GRCMO. Tables 8, 9 and 10 shows the percentage of used tracks over the total available tracks obtained by

Table 10 – Usage percentage of the available tracks per layer for ISPD18 test9.

	Metal1	**Metal2**	**Metal3**	**Metal4**	**Metal5**	**Metal6**	**Metal7**	**Metal8**	**Metal9**
OpenROAD	48.79%	67.17%	31.47%	39.72%	3.89%	8.31%	0.32%	3.24%	0.03%
GRCMO	46.82%	64.32%	31.29%	39.73%	5.66%	11.08%	0.81%	4.77%	0.07%
Diff	**-1.97**	**-1.85**	**-0.18**	**0.01**	**1.77**	**2.77**	**0.49**	**1.53**	**0.04**

the OpenROAD tool for the circuits test7, test3 and test9 respectively. The first lines show the usage for the OpenROAD routing, the second lines shows the usage for GRCMO routing finally the third line shows the difference between the second and the first line.

When we look at these tables we see that after applying GRCMO the usage on the lower metal layers (Metal 1 to 4) decreases or is almost the same for all the test cases. On the upper layer (Metal 5 to 9) the usage increases, especially on Metal 6 and Metal 8 where the highest increases appear. With a higher number of guides in the upper layers, more vias are necessary for the guides to reach the pins. .

From Table 7, test9 has the highest percentage increase in the estimated number of vias when compared to test7 and test8. This can be observed in the tables above as the usage on the metal layers 7 and 9 doubles for test9, while for test7 and test8 the increase is not as proportionally high.

One possible explanation for the increase on the usage on the higher metal layers is that, after GRCMO moves a cell, the nets to be rerouted need to go throw areas with high congestion forcing them to routed to the upper layers to avoid generating overflow. However, we cannot draw any conclusions, as we do not have the necessary data to confirm our assumptions. In a future work, to improve GRCMO, further investigation must be conducted to identify the origin of such problem.

6 CONCLUSION

This work aimed to propose and develop a Global Routing technique that includes cell movement, with support for cell relocation, legalization, and rerouting the affected nets. Another objective was the implementation of the proposed technique within an open-source design environment, enabling it to serve as a foundation for further investigation of different approaches to perform Global Routing with cell movement.

The cell movement routing problem was introduced, which offers a circuit design approach that diverges from the traditional flow where stages are solved separately. A literature review identified related works that implement various cell movement techniques during the Global Routing stage, particularly those showing favorable results for metrics such as total wirelength and the number of vias. This review also highlights a gap for developing new techniques for this problem and a lack of integration of these techniques within complete circuit design flows in open-source tools.

In light of this, the present work proposed a technique that, starting from an initial Global Routing solution, selects a list of candidate cells for relocation, moves them to their medians, legalizes this new position, and finally reroutes the nets impacted by the cell movement. The technique has been fully integrated into the open-source OpenROAD design flow.

The experiments show that the proposed technique can improve the OpenROAD Global Routing solution, reducing the estimated total wirelength by 0.43% and moving 2.6% of cells on average, at the cost that the estimated number of vias increased by 1.7%.

The proposed technique enhances the OpenROAD Global Routing step, enabling cell movements during routing to further optimize the estimated wirelength. Also, the technique's integration with the OpenROAD environment enables further exploration of new ways of moving cells and selecting candidates for rerouting nets for the routing and placement problem.

As future work, we intend to apply the detailed routing step to the solutions generated by the proposed technique. Since global routing provides just an approximate solution, studying its effects on the final detailed routing would allow us to evaluate more accurate metrics for wirelength and the number of vias. Additionally, with the results from detailed routing, we will be able to assess the impact of the technique on *Design Rule Violation* (DRV).

Another direction for future research would be to extend the technique to account for routing congestion when moving cells. This adjustment could help the routing engine find solutions with reduced wirelength and fewer vias. If a cell is moved into a congested area, the routing may need to detour or use upper layers, increasing complexity. By considering congestion, we could potentially reduce DRVs as well.

REFERENCES

AGHAEEKIASARAEE, Erfan et al. CR&P: An Efficient Co-operation between Routing and Placement. In: 2022 Design, Automation & Test in Europe Conference & Exhibition (DATE). [S.l.: s.n.], 2022. P. 772–777. DOI: 10.23919/DATE54114.2022.9774530.

________. CRP2.0: A Fast and Robust Cooperation between Routing and Placement in Advanced Technology Nodes. **ACM Transactions on Design Automation of Electronic Systems (TODAES)**, Association for Computing Machinery, New York, NY, USA, 2023. ISSN 1084-4309. DOI: 10.1145/3590962.

CHANG, Yen-Jung et al. NTHU-route 2.0: a robust global router for modern designs. **IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)**, IEEE, v. 29, n. 12, p. 1931–1944, 2010.

CHU, C. FLUTE: fast lookup table based wirelength estimation technique. In: IEEE/ACM International Conference on Computer Aided Design, 2004. ICCAD-2004. [S.l.: s.n.], 2004. P. 696–701. DOI: 10.1109/ICCAD.2004.1382665.

FONTANA, Tiago Augusto. **Improving the VLSI Circuit Design Flow Through Cell Movements During Global Routing**. 2023. PhD thesis – Universidade Federal de Santa Catarina. Disponível em: https://repositorio.ufsc.br/handle/123456789/252480.

FONTANA, Tiago Augusto; AGHAEEKIASARAEE, Erfan; NETTO, Renan; ALMEIDA, Sheiny Fabre; GANDH, Upma, et al. ILP-Based Global Routing Optimization with Cell Movements. In: 2021 IEEE Computer Society Annual Symposium on VLSI (ISVLSI). [S.l.: s.n.], 2021. P. 25–30. DOI: 10.1109/ISVLSI51109.2021.00016.

FONTANA, Tiago Augusto; AGHAEEKIASARAEE, Erfan; NETTO, Renan; ALMEIDA, Sheiny Fabre; GANDHI, Upma, et al. ILPGRC: ILP-Based Global Routing Optimization With Cell Movements. **IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems**, p. 1–1, 2023. DOI: 10.1109/TCAD.2023.3305579.

GOTO, S. An efficient algorithm for the two-dimensional placement problem in electrical circuit layout. **IEEE Transactions on Circuits and Systems**, v. 28, n. 1, p. 12–18, 1981. DOI: 10.1109/TCS.1981.1084903.

HU, Kai-Shun; YANG, Ming-Jen, et al. ICCAD-2020 CAD contest in routing with cell movement. In: 2020 IEEE/ACM International Conference on Computer-Aided Design (ICCAD). Virtual Event, USA: Association for Computing Machinery, 2020. P. 1–4. DOI: 10.1145/3400302.3415738.

HU, Kai-Shun; YU, Tao-Chun, et al. 2021 ICCAD CAD Contest Problem B: Routing with Cell Movement Advanced: Invited Paper. In: 2021 IEEE/ACM International

Conference On Computer Aided Design (ICCAD). [S.l.: s.n.], 2021. P. 1–5. DOI: 10.1109/ICCAD51958.2021.9643568.

KAHNG, Andrew B. Leveling Up: A Trajectory of OpenROAD, TILOS and Beyond. In: PROCEEDINGS of the 2022 International Symposium on Physical Design. Virtual Event, Canada: Association for Computing Machinery, 2022. (ISPD '22), p. 73–79. DOI: 10.1145/3505170.3511479.

KAHNG, Andrew B. et al. **VLSI Physical Design: From Graph Partitioning to Timing Closure**. 1st. [S.l.]: Springer Publishing Company, Incorporated, 2011. ISBN 9789048195909.

KAO, M.Y. **Encyclopedia of Algorithms**. [S.l.]: Springer, 2008. (Encyclopedia of Algorithms). ISBN 9780387307701. Disponível em: https://books.google.com.br/books?id=i3S9_GnHZwYC.

LIU, Wen-Hao et al. NCTU-GR 2.0: Multithreaded collision-aware global routing with bounded-length maze routing. **IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)**, IEEE, v. 32, n. 5, p. 709–722, 2013.

MANTIK, Stefanus et al. ISPD 2018 Initial Detailed Routing Contest and Benchmarks. In: PROCEEDINGS of the 2018 International Symposium on Physical Design. Monterey, California, USA: Association for Computing Machinery, 2018. (ISPD '18), p. 140–143. DOI: 10.1145/3177540.3177562.

MOFFITT, Michael D. MaizeRouter: Engineering an Effective Global Router. **IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems**, v. 27, n. 11, p. 2017–2026, 2008. DOI: 10.1109/TCAD.2008.2006082.

NETTO, Renan Oliveira. **Algorithm Selection Framework for Legalization Using Deep Convolutional Neural Networks**. 2022. PhD thesis – Universidade Federal de Santa Catarina. Disponível em: https://repositorio.ufsc.br/handle/123456789/231275.

OPENROAD. **OpenROAD Project**. [S.l.]: GitHub, 2024. https://github.com/The-OpenROAD-Project/OpenROAD.

PAN, Min; VISWANATHAN, N.; CHU, C. An efficient and effective detailed placement algorithm. In: ICCAD-2005. IEEE/ACM International Conference on Computer-Aided Design, 2005. [S.l.: s.n.], 2005. P. 48–55. DOI: 10.1109/ICCAD.2005.1560039.

PAN, Min; XU, Yue, et al. FastRoute: An Efficient and High-Quality Global Router. **VLSI Design**, Hindawi Limited, London, GBR, v. 2012, Jan. 2012. ISSN 1065-514X. DOI: 10.1155/2012/608362. Disponível em: https://doi.org/10.1155/2012/608362.

SPINDLER, Peter; SCHLICHTMANN, Ulf; JOHANNES, Frank M. Abacus: Fast Legalization of Standard Cell Circuits with Minimal Movement. In: PROCEEDINGS of the 2008 International Symposium on Physical Design. Portland, Oregon, USA: Association for Computing Machinery, 2008. (ISPD ’08), p. 47–53. DOI: `10.1145/1353629.1353640`.

WANG, Fangzhou et al. Starfish: An Efficient P&R Co-Optimization Engine with A*-based Partial Rerouting. In: IEEE. IEEE/ACM International Conference On Computer Aided Design (ICCAD). [S.l.: s.n.], 2021. P. 1–9.

ZANG, Xinshi et al. ATLAS: A Two-Level Layer-Aware Scheme for Routing with Cell Movement. In: IEEE/ACM International Conference On Computer Aided Design (ICCAD). [S.l.: s.n.], 2022. P. 1–7.

Appendix

APPENDIX A – UNDERGRADUATE THESIS PAPER

GRCMO: Global Routing Optimization Through Median-Based Cell Movement in an Open-Source RTL-to-GDSII Flow

Arthur Lourenco[1]

[1]Departamento de Informática e Estatıstica – Centro Tecnológico
Universidade Federal de Santa Catarina (UFSC)

Abstract. *In the physical design of VLSI circuits, the placement and routing steps are responsible for finding cell positions and interconnecting cell pins. They significantly impact the circuit layout quality. Placement and routing were originally solved separately in a divide-and-conquer approach to cope with design complexity, resulting in a decoupling that often leads to poor solutions. To mitigate this problem, a few techniques seeking to integrate those steps have recently emerged. In this final course project, we present GRCMO, a technique that optimizes Global Routing by moving a few cells to their medians, implemented in one of the most compleat open-source platforms, OpenROAD. Experimental results using the ISPD 2018 Contest circuits showed that, compared to the standard global routing solutions from OpenROAD, GRCMO is able to reduce the estimated wirelength by 0.43%, on average, and from 0.5% up to 0.7% for the biggest circuits, requiring less than 20 iterations and moving 2.61% of the cells, on average.*

Resumo. *No design físico de circuitos VLSI, as etapas de posicionamento (placement) e roteamento (routing) são responsáveis por determinar as posições das células e interconectar os pinos das células. Essas etapas têm um impacto significativo na qualidade do layout do circuito. Originalmente, o posicionamento e o roteamento eram resolvidos separadamente em uma abordagem de divisão e conquista para lidar com a complexidade do design, o que resultava em um desacoplamento que frequentemente levava a soluções de baixa qualidade. Para mitigar esse problema, recentemente surgiram algumas técnicas que buscam integrar essas etapas. Neste projeto final de curso, apresentamos o GRCMO, uma técnica que otimiza o roteamento global movendo algumas células para suas medianas, implementada em uma das plataformas de código aberto mais completas, OpenROAD, tornando possivel o seu uso em um fluxo RTL-to-GDSII completo. Os resultados experimentais utilizando os circuitos do Concurso ISPD 2018 mostraram que, em comparação com as soluções padrão de roteamento global do OpenROAD, GRCMO é capaz de reduzir o comprimento estimado do fio em 0,43%, em média, e entre 0,5% e 0,7% para os maiores circuitos, necessitando de menos de 20 iterações e movendo, em média, 2,61% das células.*

1. Introduction

The design of *Very Large-Scale Integrated* (VLSI) circuits is highly complex, and this complexity continues to increase as the manufacturing technologies continue to advance towards smaller nodes [Kahng et al. 2011]. As a result, the use of specific tools for designing such circuits has become essential. These tools, called *Electronic Design Automation* (EDA), automate and make all stages of VLSI circuit design feasible. Figure 1 presents the so-called VLSI design flow, which begins with an abstract description of the circuit, in terms of structure and behavior, and through successive synthesis stages, progressively adds information until reaching a description that will be sent for manufacturing [Kahng et al. 2011]. At each synthesis stage, one or more EDA tools are associated.

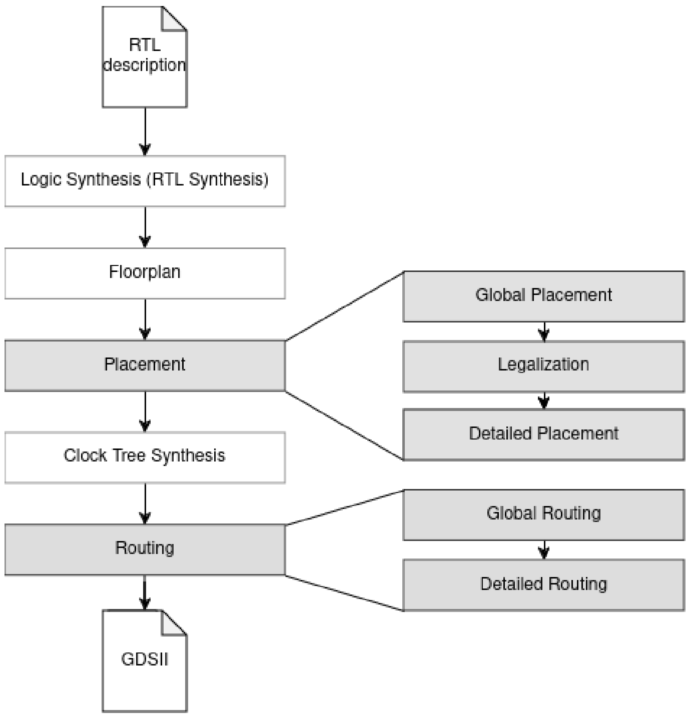


Figure 1. Simplified design flow for VLSI circuits.

Although the VLSI design flow is usually illustrated as a linear sequence of separate steps, the result of each step depends on the results of the previous ones. Therefore, a poor placement solution, for example, will negatively impact the quality of routing and may even make routing impossible. In such cases, it would be necessary to go back in the flow to redo the placement.

In light of this problem, researchers in the EDA industry proposed the topic of 'Routing Optimization with Cell Movement' as one of the challenges in the 2020 and 2021 editions of the international *Computer-Aided Design* (CAD) Contest@ICCAD [Hu et al. 2020] [Hu et al. 2021], demonstrating the industry's strong interest in finding new solutions with the integration of different steps of the VLSI design flow.

These contests stimulated various researches in techniques to integrate the placement and routing steps, such as [Wang et al. 2021], [Zang et al. 2022], [Aghaeekiasaraee et al. 2022], [Aghaeekiasaraee et al. 2023], [Fontana et al. 2021] and [Fontana et al. 2023]. Although the aforementioned works bring contributions to the routing with cell movement problem, they also have some limitations. For instance, it is difficult to evaluate the outcomes of [Wang et al. 2021], [Zang et al. 2022] or [Fontana et al. 2021] works, since they do not legalize the circuits. In addition, all those techniques are implemented as standalone tools, lacking an integration with a complete open-source flow that could allow going from *Register Transfer Level* (RTL) to *Graphic Data Stream* (GDS).

Given such gaps, this work proposes a new technique to improve global routing solutions based on cell movement called GRCMO. The proposed technique is implemented fully on the open-source platform OpenROAD [Kahng 2022], which is currently one of the most prominent open-source tool kits for VLSI design

2. Related Works

The following sections present the most recent works focused on integrating the stages of Global Routing with Placement, using cell movement during Global Routing.

In [Wang et al. 2021] authors propose a technique named Starfish, which moves cells and reroutes the affected nets using an A* based algorithm to find the shortest routes to the previous routing topology. The position of the cells are chosen from a lookup table that contains the best candidate positions and improvements estimation. The experimental results were evaluated using the ICCAD CADContest 2020 benchmarks.

In [Zang et al. 2022], the technique ATLAS is presented. In this technique, as a first step cluster of cells are moved to optimize the number of vias, on a second stage of movements individual cells are moved. In both stages the movements go to the median and the affected nets are rerouted using the search algorithm A*. ATLAS was tested using the ICCAD CADContest 2020 and 2021 benchmarks, achieving better performances than the first place candidates of the contest.

The works CR&P [Aghaeekiasaraee et al. 2022] and CRP2.0 [Aghaeekiasaraee et al. 2023], focus on solving the problem of highly congested areas, to alleviate these regions a set of cells inside them are moved, an *Integer Linear Programming* (ILP) model is used to legalized these movements. Finally, after all movements were applied, new routing is made with a pattern routing algorithm and an A* algorithm. Furthermore, CRP2.0 speeds up the technique using a caching approach.

The results of both CR&P and CRP2.0 were analyzed after the Detailed Routing using the benchmarks from the contests ISPD 2018 and 2019, presenting reduction on the total number of vias and the total wirelength.

In [Fontana et al. 2021], an ILP model for routing with cell movement is proposed, which serves as the core of the technique named ILP-GRC$_{V0}$. For every cell a set of candidate positions and candidate routing are generated, the positions candidate are the cells original position and the median of that cell, then for both these positions the routing candidate are generated as all pattern routing possible. Finally, the ILP model decides which candidate position and routing will be assigned for every cell. The results were analyzed using the ISPD 2018 competition benchmarks, showing better results than commercial tools in terms of total wirelength.

In the extension of the work, ILPGRC [Fontana et al. 2023], a circuit partitioning method using a technique called ”Checkered paneling” is proposed to make the ILP model execution parallelizable, reducing execution time for large circuits. Additionally, this new work proposes a legalization for the circuit based on clusters of GCells, aiming to minimize disturbance to non-moved cells. The ILPGRC technique was evaluated using the ISPD18 and ISPD19 contest benchmarks, demonstrating an improvement in the total number of vias after Detailed Routing.

Starfish, Atlas and ILP-GRC do not have a legalization step for their placement solution, and thus can not perform a detailed routing, making it difficult to accurately evaluate their results. CR&P and CRP2.0 have a legalization step using an ILP solver and analyze their results after detailed routing. The extension of ILP-GRC, ILPGRC, also has a legalization step and proceeds the flow to the end of the detailed routing. However, none of the aforementioned works are integrated with a complete VLSI design flow.

With this, our proposed technique was implemented within the OpenROAD tool, one of the most complete open-source VLSI design flow. Our technique objective is to reduce the estimated wirelength, that is evaluated after global routing, however we apply a legalization step, allowing to perform detailed routing on future works.

3. Proposed Technique

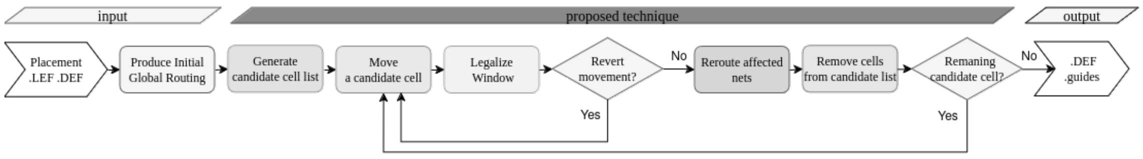


Figure 2. Diagram of the proposed technique

This section presents the proposed technique to solve the global routing with cell movement problem. Figure 2 shows an explanatory diagram of this technique. As input, the technique receives an initial Global Routing solution and two files in standard formats used in the industry, the Library Exchange Format (LEF) and the Design Exchange Format (DEF). The .LEF file describes the technology that will be used for the chip fabrication, while the .DEF file contains the physical characteristics of the chip and its elements.

GRCMO flow begins by generating the candidate cell list, a structure that contains the cells with the highest potential to improve global routing solution if moved. In order to determine this potential, we first need to assign a new position for each cell and then compare such position with the original one. Considering the objective of reducing the total estimated wirelength of the circuit, in this work we take inspiration from [Pan et al. 2005] and define the target position for each cell as its median GCell.

Figure 3 shows an example of how to find the median of a cell, the cell under analysis, C3, is connected to five other cells: C1, C2 C4, C5 and C6, through three nets: red, green and yellow, whose smallest enclosing rectangles (called bounding boxes) are shown in red, green, and yellow, respectively. The optimal region is the median of these bounding boxes, which is outlined by the purple rectangle. We then define the median of C3, as the midpoint of this region, represented by the purple star.

Once the target position is defined for all cells, we estimate the potential of improvement for every cell. This potential is computed comparing the estimated *Rectilinear Minimum Spanning Tree* (RSMT) wirelength [Kahng et al. 2011] for nets connected to that cell for the original and the target position. With this, the cells that have a RSMT

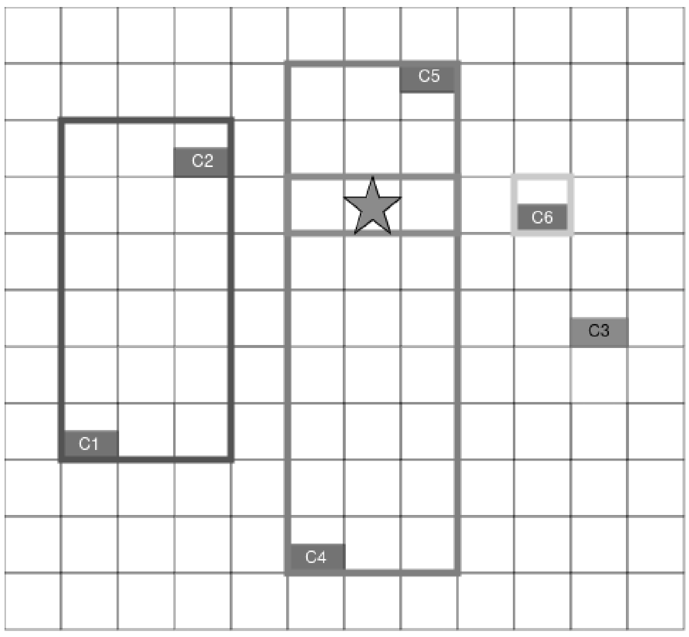


Figure 3. Example of finding the median GCell for cell C3.

smaller if the cell is moved will be defined as cells with highest potential. Equation 1 formulates this potential as a weight function, in which the the weight w_i of cell c_i is the sum of the difference between the RSMT wirelength before and after moving the cell to its median GCell, considering every net n_k in $N(c_i)$, the set of nets that connects to c_i.

$$w_i = \sum_{n_k \in N(c_i)} RSMTWL_{before}(n_k) - RSMTWL_{after}(n_k) \tag{1}$$

Then, the candidate cell list is composed of the cells that have a weight greater than 0. In order to focus the movements on the cells that offer the best improvements on wirelength, the candidate list is ordered by the cell's weights so the cells with the highest weights are on the list head.

With the candidate cell list generated, we start moving the cells one by one, until the candidate list is empty. The cell with the highest weight is taken from the candidate cell list is then moved to its determined target position. With the cell in its new position, we apply the Abacus algorithm [Spindler et al. 2008] inside a window that is 3 rows high and 30 GCells wide, with the median GCell as the center. This size was chosen after some testing with few different window dimensions. Since the cell is in the center of the window we ensure that after the legalization will be moved at most 1 row up or down, and 15 GCells to the side. If the legalization algorithm fails to achieve a solution, it means there is not enough space in the legalization window to fit the moved cell. In this case, we reverse the movement, placing the cell back in its original position.

Even though the legalization window limits the displacement of the moved cell, it can still be displaced too far from its median GCell. Therefore, a check is performed to confirm that the movement is still beneficial. This is done by computing the cells weight again, comparing the RSMT wirelength of the nets connected to the moved cell, in both the cell's original position and its new position after the legalization. If the new weight of the cell is smaller than 0, the movement after legalization is worse than the cell in its original position and we revert the movement, returning the cell to its original position. Otherwise, we confirm the movement and proceed to the next step in the flow.

Once the moved cell has a legal position defined, we can proceed to rerouting the nets connected to it. In addition, during the legalization process, other cells may have had their positions adjusted to accommodate the cell that was moved to its legalized position. Therefore, it is necessary to identify all the nets that were impacted by the movement of the candidate cell. These nets will be rerouted using the Incremental Routing tool from OpenROAD, that runs a FastRoute algorithm for routing [Pan et al. 2012].

Once the affect nets are reroute, the candidate cell list is updated. Here the moved cell is removed from the list, also the cells that are part of one of nets connected to the moved cell are removed. This second set of cells need to be removed because we want to preserve routing improvements obtained for the nets connected to the moved cell, if another cell in these nets is moved, these improvements can be lost. On top of that, the weight of these second set of cells are no longer valid, since its median GCell might depend on the moved cells old position.

At the end of this step, if the candidate list is not empty the technique returns to the second step, where a new candidate cell will be selected, moved, legalized and rerouted. When all the cells from the candidate list are removed and the list is empty, the technique proceeds to generated the files with the results, the .DEF file is updated and a .guide file is written with the global routing solution.

4. Methodology and Experimental Results

The experiments were performed in a Linux 22.04.4 LTS machine with AMD Ryzen® 5-5500 CPU running at 2.10 GHz and 16GB of RAM (3200 MHz).

The evaluation of the proposed technique was made using the benchmark from the ISPD18 [Mantik et al. 2018] Contest. This benchmark is composed of 10 circuits placed and legalized with different specifications such as technology node, number of cells and number of nets. Table 1 presents the detailed information of each circuit. These benchmarks have LEF and DEF files with a placement solution, and the circuits use one of two technology nodes: 45nm or 32nm. This benchmark suite also provides technology information for circuits such as standard, macro, and IO cells. Finally, cells have complex pin shapes such as L, Z, and others. However, for this benchmark no power or timing information is provided. In this work, we did not evaluate circuit test 10 because it has 100% density and, therefore, no space for moving cells. We did not use the ICCAD CAD Contest 2020 [Hu et al. 2020] or ICCAD CAD Contest 2021 [Hu et al. 2021] benchmarks because they do not provide row or site information, making it impossible to legalize the circuits.

Figure 4 presents the experimental flow, we begin with the placement from the ISPD 2018 contest and the initial Global Routing that is generated by the OpenROAD tool using the FastRoute [Pan et al. 2012] algorithm. The technique is then iteratively applied 20 times. The results regarding estimated wirelength, moved cells, and estimated number of vias are gathered after each iteration using the OpenROAD tools. The origi-

Table 1. ISPD 2018 CAD contest benchmarks. Source [Mantik et al. 2018].

Design	# Cells	# Blk	# Net	# Pin	# Layer	Die size	Density (%)	Tech. node
test1	8879	0	3153	0	9	0.20x0.19mm2	85	45nm
test2	35913	0	36834	1211	9	0.65x0.57mm2	57	45nm
test3	35973	4	36700	1211	9	0.99x0.70mm2	65	45nm
test4	72094	0	72401	1211	9	0.89x0.61mm2	89	32nm
test5	71954	0	72394	1211	9	0.93x0.92mm2	92	32nm
test6	107919	0	107701	1211	9	0.86x0.53mm2	99	32nm
test7	179865	16	179863	1211	9	1.36x1.33mm2	90	32nm
test8	191987	16	179863	1211	9	1.36x1.33mm2	90	32nm
test9	192911	0	178857	1211	9	0.91x0.78mm2	91	32nm
test10	290386	0	182000	1211	9	0.91x0.87mm2	100	32nm

nal OpenROAD Global Routing solution is the baseline for the initial contest placement solution.

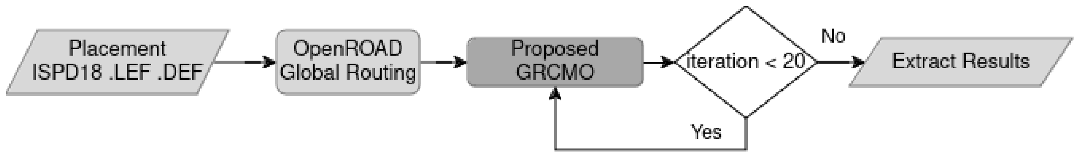


Figure 4. Experimental flow

Table 2. Experimental results for the ISPD18 benchmarks.

ISPD18 Designs	Cells		Estimated Wirelength (um)			Estimated # VIAs		
	#(K)	Moved %	OpenROAD WL	GRCMO WL	dif WL %	OpenROAD #EV	GRCMO #EV	dif #EV %
test1	3	0.40	124946	124921	-0.02	12213	12225	0.10
test2	3	5.15	2031357	2021869	-0.47	177783	183169	3.03
test3	36	1.16	2223886	2212728	-0.50	178118	183775	3.18
test4	72	3.24	3255627	3237828	-0.55	396694	401632	1.24
test5	72	4.00	3388105	3374124	-0.41	410784	419464	2.11
test6	108	0.29	4494295	4490235	-0.09	628805	630960	0.34
test7	180	3.38	8062032	8018601	-0.54	1282271	1304140	1.71
test8	192	2.94	8093235	8051835	-0.51	1302119	1325827	1.82
test9	193	2.98	6948508	6897316	-0.74	989156	1011347	2.24
Avg		2.61			-0.43			1.75

Table 2 presents the results of the experiments. The first column lists the ISPD 2018 benchmarks, followed by the total number of cells and the percentage of moved cells. The columns labeled *OpenROAD WL* and *GRCMO WL* show the estimated wire-length, in microns, for the Global Routing results obtained by baseline and by the proposed technique, respectively. The column *dif WL %* shows the difference, in percentage, between GRCMO and the baseline results, therefore a negative result means a reduction in the estimated total wirelength. The next columns present the baseline for the number of estimated vias, the number of estimated vias achieved by GRCMO and the percentage difference between GRCMO and the baseline results.

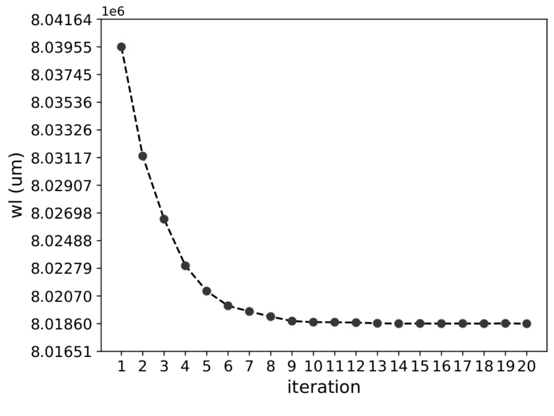
The results show that the proposed technique improved the the estimated total wirelength for all circuits, reducing it on average by 0.43%, while moving 2.61% of the cells, on average. Circuit test9 had the best improvement, reaching a 0.74% reduction and moving almost 3% cells. The lowest improvement was achieved on Test1, the smallest circuit, where we barely affected the design, moving only 0.4% of cells and reaching 0.02% reduction. Similar to this, on test6 only 0.3% of the cells were moved leading to an almost insignificant estimated wirelength reduction, this circuit is highly dense and most of the movements were rejected due to the lack of space to legalize them. Other than these outliers, the rest of the circuit shows similar estimated wirelength reduction, with the percentage of moved cells varying between 1% and 5%.

The estimated number of vias was degradated by GRCMO, increasing on average by 1.75%, the circuits with the higher increase being test2 and test3 with 3.03% and 3.18% respectively. Following what happened with wirelength, due to the low percentage of moved cells, the circuits test1 and test6 had the smallest changes on the estimated number of vias.

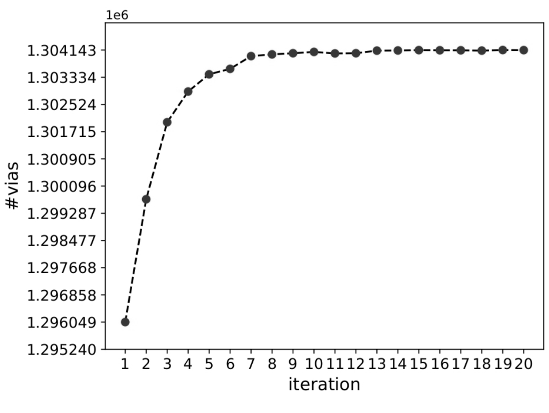
GRCMO was able to finish with no overflow on the GCells usage and no placement violations for all the designs.

For a better understanding of the results we make a deeper analysis of the results for the circuits test7, test8 and test9. We choose these circuits because, as shown in Table 1, they have similar specifications, but with increasing complexity.

First, we look at Figures 5, 6 and 7 present the evolution of the technique results throughout the iteration respectively for test7, test8 and test9. with Figures 5a, 6a and 7a presenting the evolution of the estimated wirelength and Figures 5b, 6b and 7b the evolution of the estimated number of vias. The x-axis represents the iterations, and the y-axis the metric in question



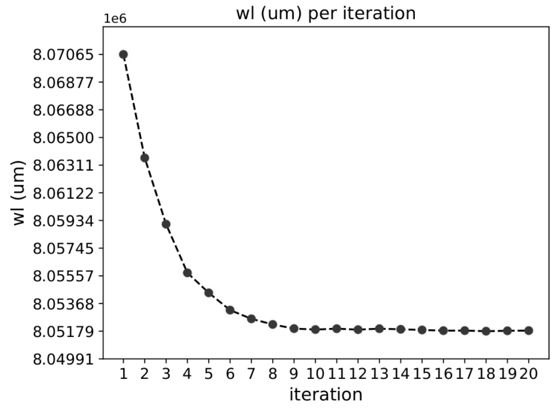
(a) Evolution of estimated wirelength.



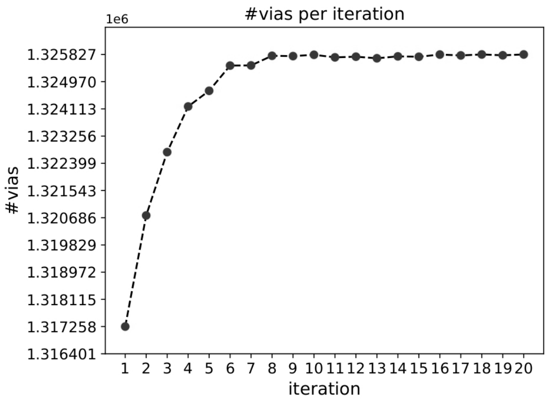
(b) Evolution of estimated number of vias.

Figure 5. Evolution of estimated wirelength and number of vias through the iterations for test7.

Figure 5a, 6a and 7a shows that the estimated wirelength keeps decreasing as the iterations progress. For test8 and test7, the first 6 iterations are responsible for the most

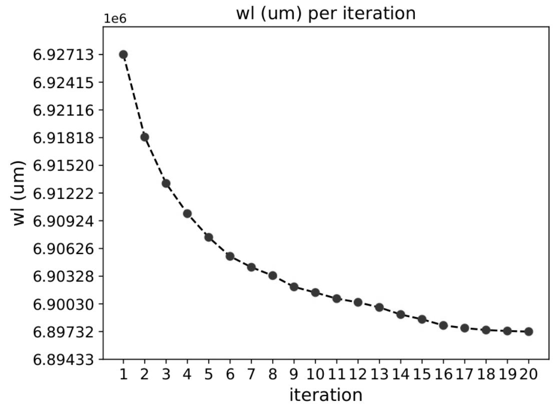


(a) Evolution of estimated wirelength.

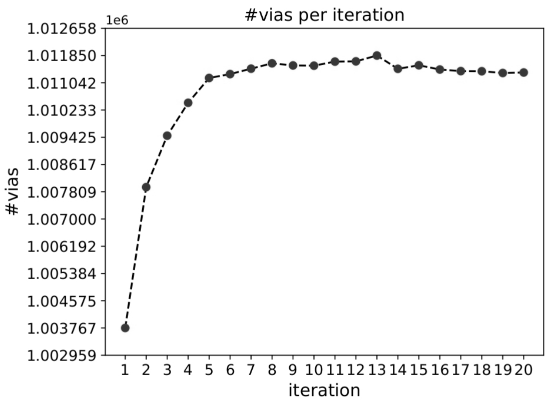


(b) Evolution of estimated number of vias.

Figure 6. Evolution of estimated wirelength and number of vias through the iterations for test8.



(a) Evolution of estimated wirelength.



(b) Evolution of estimated number of vias.

Figure 7. Evolution of estimated wirelength and number of vias through the iterations for test9.

improvements and the following iterations barely affect the results. For test9, the although the first 6 iterations are responsible for the most significant gains, the following iteration still improve the estimated wirelength.

In Figures 5b, 6b and 7b, the estimated number of vias follows a similar behavior as with the wirelength, with the estimated number of vias increasing with each iteration. Just as before, for test7 and test8 the most significant increases in the estimated vias happens in the first 6 iteration and different than on wirelength, for test9 it also stabilizes after 6 iterations.

In conclusion, applying the technique iteratively amplified the results of the first iteration, highlighting its impact beyond a single pass. In the case of estimated wirelength, the results improved with the progression of the iterations, for the estimated number of vias, that degraded after the first iteration, the metric kept degrading as the iterations progress.

Although the technique remains effective throughout all iterations for test9, for test7 and test8 the results stabilize much earlier, around the sixth iteration. Therefore, execution time can be saved by running fewer iterations.

Table 3. Usage percentage of the available tracks per layer for ISPD18 test7.

	Metal1	Metal2	Metal3	Metal4	Metal5	Metal6	Metal7	Metal8	Metal9
OpenROAD	17.72%	38.27%	32.86%	29.73%	16.13%	11.34%	3.56%	1.15%	0.93%
GRCMO	17.30%	36.57%	31.44%	29.32%	17.22%	14.00%	4.41%	3.16%	1.08%
Diff	-0.42	-1.70	-4.31	-1.42	1.09	2.66	0.85	2.01	0.15

Table 4. Usage percentage of the available tracks per layer for ISPD18 test8.

	Metal1	Metal2	Metal3	Metal4	Metal5	Metal6	Metal7	Metal8	Metal9
OpenROAD	17.49%	38.58%	32.87%	30.36%	16.49%	15.60%	3.78%	1.96%	0.82%
GRCMO	17.07%	36.95%	31.54%	29.81%	17.53%	18.09%	4.52%	4.44%	1.18%
Diff	-0.42	-1.63	-1.33	-0.55	1.04	2.49	0.74	2.48	0.36

Table 5. Usage percentage of the available tracks per layer for ISPD18 test9.

	Metal1	Metal2	Metal3	Metal4	Metal5	Metal6	Metal7	Metal8	Metal9
OpenROAD	48.79%	67.17%	31.47%	39.72%	3.89%	8.31%	0.32%	3.24%	0.03%
GRCMO	46.82%	64.32%	31.29%	39.73%	5.66%	11.08%	0.81%	4.77%	0.07%
Diff	-1.97	-1.85	-0.18	0.01	1.77	2.77	0.49	1.53	0.04

Table 2 showed us that after applying our technique, the estimated number of vias increased for all test cases, in order to better understand this we studied the usage of the guides per metal layer before and after applying GRCMO. Tables 3, 4 and 5 shows the percentage of used tracks over the total available tracks obtained by the OpenROAD tool for the circuits test7, test3 and test9 respectively. The first lines show the usage for the OpenROAD routing, the second lines shows the usage for GRCMO routing finally the third line shows the difference between the second and the first line.

When we look at these tables we see that after applying GRCMO the usage on the lower metal layers (Metal 1 to 4) decreases or is almost the same for all the test cases. On the upper layer (Metal 5 to 9) the usage increases, especially on Metal 6 and Metal 8 where the highest increases appear. With a higher number of guides in the upper layers, more vias are necessary for the guides to reach the pins. .

From Table 2, test9 has the highest percentage increase in the estimated number of vias when compared to test7 and test8. This can be observed in the tables above as the usage on the metal layers 7 and 9 doubles for test9, while for test7 and test8 the increase is not as proportionally high.

One possible explanation for the increase on the usage on the higher metal layers is that, after GRCMO moves a cell, the nets to be rerouted need to go throw areas with high congestion forcing them to routed to the upper layers to avoid generating overflow. However, we cannot draw any conclusions, as we do not have the necessary data to confirm our assumptions. In a future work, to improve GRCMO, more studies must be done to identify the cause of such problem.

5. Conclusion

This work aimed to propose and develop a Global Routing technique that includes cell movement, with support for cell relocation, legalization, and rerouting the affected nets. Another objective was the implementation of the proposed technique within an open-source design environment, enabling it to serve as a foundation for further investigation of different approaches to perform Global Routing with cell movement.

A literature review identified related works that implement various cell movement techniques during the Global Routing stage, particularly those showing favorable results for metrics such as total wirelength and the number of vias. This review also highlights a gap for developing new techniques for this problem and a lack of integration of these techniques within complete circuit design flows in open-source tools.

In light of this, the present work proposed a technique that, starting from an initial Global Routing solution, selects a list of candidate cells for relocation, moves them to their medians, legalizes this new position, and finally reroutes the nets impacted by the cell movement. The technique has been fully integrated into the open-source OpenROAD design flow.

The experiments show that the proposed technique can improve the OpenROAD Global Routing solution, reducing the estimated total wirelength by 0.43% and moving 2.6% of cells on average, at the cost that the estimated number of vias increased by 1.7%.

The proposed technique enhances the OpenROAD Global Routing step, enabling cell movements during routing to further optimize the estimated wirelength. Also, the technique's integration with the OpenROAD environment enables further exploration of new ways of moving cells and selecting candidates for rerouting nets for the routing and placement problem.

As future work, we can cite applying the detailed routing step to the solutions generated by the proposed technique. Since global routing only provides an approximate solution, studying its effects on the final detailed routing would allow us to evaluate more accurate metrics for wirelength and the number of vias. Additionally, with the results from detailed routing, we could assess the impact of the technique on *Design Rule Violation* (DRV).

Another direction for future research would be to extend the technique to account for routing congestion when moving cells. This adjustment could help the routing engine find solutions with reduced wirelength and fewer vias. If a cell is moved into a congested area, the routing may need to detour or use upper layers, increasing complexity. By considering congestion, we could potentially reduce DRVs as well.

References

Aghaeekiasaraee, E., Tabrizi, A. F., Fontana, T. A., Netto, R., Almeida, S. F., Gandhi, U., Güntzel, J. L., Westwick, D., and Behjat, L. (2023). Crp2.0: A fast and robust cooperation between routing and placement in advanced technology nodes. *ACM Transactions on Design Automation of Electronic Systems (TODAES).*

Aghaeekiasaraee, E., Tabrizi, A. F., Fontana, T. A., Netto, R., Almeida, S. F., Gandhi, U., Güntzel, J. L., Westwick, D., and Behjat, L. (2022). Cr&p: An efficient co-operation between routing and placement. In *2022 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 772–777.

Fontana, T. A., Aghaeekiasaraee, E., Netto, R., Almeida, S. F., Gandh, U., Tabrizi, A. F., Westwick, D., Behjat, L., and Güntzel, J. L. (2021). Ilp-based global routing optimization with cell movements. In *2021 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*, pages 25–30.

Fontana, T. A., Aghaeekiasaraee, E., Netto, R., Almeida, S. F., Gandhi, U., Behjat, L., and Güntzel, J. L. (2023). Ilpgrc: Ilp-based global routing optimization with cell movements. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, pages 1–1.

Hu, K.-S., Yang, M.-J., Yu, T.-C., and Chen, G.-C. (2020). Iccad-2020 cad contest in routing with cell movement. In *Proceedings of the 39th International Conference on Computer-Aided Design*, ICCAD '20, pages 1–4, New York, NY, USA. Association for Computing Machinery.

Hu, K.-S., Yu, T.-C., Yang, M.-J., and Shen, C.-F. C. (2021). 2021 iccad cad contest problem b: Routing with cell movement advanced: Invited paper. In *2021 IEEE/ACM International Conference On Computer Aided Design (ICCAD)*, pages 1–5.

Kahng, A. B. (2022). Leveling up: A trajectory of openroad, tilos and beyond. In *Proceedings of the 2022 International Symposium on Physical Design*, ISPD '22, page 73–79, New York, NY, USA. Association for Computing Machinery.

Kahng, A. B., Lienig, J., Markov, I. L., and Hu, J. (2011). *VLSI Physical Design: From Graph Partitioning to Timing Closure*. Springer Publishing Company, Incorporated, 1st edition.

Mantik, S., Posser, G., Chow, W.-K., Ding, Y., and Liu, W.-H. (2018). Ispd 2018 initial detailed routing contest and benchmarks. In *Proceedings of the 2018 International Symposium on Physical Design*, ISPD '18, page 140–143, New York, NY, USA. Association for Computing Machinery.

Pan, M., Viswanathan, N., and Chu, C. (2005). An efficient and effective detailed placement algorithm. In *ICCAD-2005. IEEE/ACM International Conference on Computer-Aided Design, 2005.*, pages 48–55.

Pan, M., Xu, Y., Zhang, Y., and Chu, C. (2012). Fastroute: An efficient and high-quality global router. *VLSI Design*, 2012.

Spindler, P., Schlichtmann, U., and Johannes, F. M. (2008). Abacus: Fast legalization of standard cell circuits with minimal movement. In *Proceedings of the 2008 International Symposium on Physical Design*, ISPD '08, page 47–53, New York, NY, USA. Association for Computing Machinery.

Wang, F., Liu, L., Chen, J., Liu, J., Zang, X., and Wong, M. D. (2021). Starfish: An efficient p&r co-optimization engine with a*-based partial rerouting. In *IEEE/ACM International Conference On Computer Aided Design (ICCAD)*, pages 1–9. IEEE.

Zang, X., Wang, F., Liu, J., and Wong, M. D. (2022). Atlas: A two-level layer-aware scheme for routing with cell movement. In *IEEE/ACM International Conference On Computer Aided Design (ICCAD)*, pages 1–7.

APPENDIX B – SOURCE CODE

The code developed is available at the ECL lab fork of the OpenROAD project: **https://github.com/eclufsc/OpenROAD_ecl/tree/rcm_Arthur**. To run the code you need to clone the ropository to your machine with the flag *–recursive*, and checkout to the branch *rcm_Arthur*, then you must follow the instruction to compile the OpenROAD tool, these isntructions are available at: **https://github.com/The-OpenROAD-Project/OpenROAD/blob/master/docs/user/Build.md**. With the OpenROAD tool compiled you will run the OpenROAD tool, then load a design using the command *read_lef file.lef* and after *read_def file.def*, finally to run GRCMO you will use the command *rcm::run_cmro*.

1. Clone ECL OpenROAD for with the *–recursive* flag;
2. Checkout to the branch *rcm_Arthur*;
3. Follow the OpenROAD instruction to compile the tool;
4. Run the tool;
5. Read the design using the commands:

   read_lef file.lef;

   read_def file.def;
6. Run GRCMO with the command *rcm::run_cmro*;