UNIVERSIDADE FEDERAL DE SANTA CATARINA
CENTRO TECNOLÓGICO
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Tiago Augusto Fontana

Improving the VLSI Circuit Design Flow Through Cell Movements During Global Routing

Florianópolis
2023

Tiago Augusto Fontana

Improving the VLSI Circuit Design Flow Through Cell Movements During Global Routing

Tese submetida ao Programa de Pós-Graduação em Ciência da Computação para a obtenção do título de Doutor em Ciência da Computação.
Orientador: Prof. José Luís Almada Güntzel, Dr.
Coorientador: Profa. Laleh Behjat, Dra.

Florianópolis
2023

Ficha de identificação da obra elaborada pelo autor,
através do Programa de Geração Automática da Biblioteca Universitária da UFSC.

Fontana, Tiago Augusto
Improving the VLSI Circuit Design Flow Through Cell Movements During Global Routing / Tiago Augusto Fontana ; orientador, José Luís Almada Güntzel, coorientadora, Laleh Behjat, 2023.
90 p.

Tese (doutorado) - Universidade Federal de Santa Catarina, Centro Tecnológico, Programa de Pós-Graduação em Ciência da Computação, Florianópolis, 2023.

Inclui referências.

1. Ciência da Computação. 2. Electronic Design Automation. 3. Integer Linear Programming. 4. Physical Design. 5. Routing with cell movement. I. Güntzel, José Luís Almada. II. Behjat, Laleh. III. Universidade Federal de Santa Catarina. Programa de Pós-Graduação em Ciência da Computação. IV. Título.

Tiago Augusto Fontana

Improving the VLSI Circuit Design Flow Through Cell Movements During Global Routing

O presente trabalho em nível de doutorado foi avaliado e aprovado por banca examinadora composta pelos seguintes membros:

Prof. Rafael de Santiago, Dr.
Universidade Federal de Santa Catarina

Gracieli Posser , Dra.
Cadence Design Systems, Inc.

Renato Fernandes Hentschke, Dr.
Synopsys, Inc.

Certificamos que esta é a **versão original e final** do trabalho de conclusão que foi julgado adequado para obtenção do título de Doutor em Ciência da Computação.

ICP-Edu
Documento assinado digitalmente
Marcio Bastos Castro
Data: 21/11/2023 08:43:40-0300
CPF: ***.707.720-**
Verifique as assinaturas em https://v.ufsc.br

Prof. Márcio Bastos Castro, Dr.
Coordenador do Programa

ICP-Edu
Documento assinado digitalmente
Jose Luis Almada Guntzel
Data: 21/11/2023 07:33:16-0300
CPF: ***.780.670-**
Verifique as assinaturas em https://v.ufsc.br

Prof. José Luís Almada Güntzel, Dr.
Orientador

Florianópolis, 2023.

This thesis represents the culmination of my diligent efforts,
spanning a period of five years.

ACKNOWLEDGEMENTS

I would like to acknowledge my parents, Marli Dileta Secco Fontana (*in memoriam*) and Waldir Fontana, for supporting me on this journey through the University since 2011. I would also like to express appreciation to my wife Cecília Fernandes Contreras, for her love, encouragement, and patience during this Ph.D. course.

I want to express my deepest appreciation to Dr. José Luís Almada Güntzel, my supervisor during my academic journey, especially during my Ph.D. studies, for his consistent and invaluable support that has been instrumental in my research and the completion of this thesis. I would also like to express my gratitude to Dr. Laleh Behjat, my co-supervisor, for her valuable insights and guidance, which have greatly contributed to this work. Additionally, I would like to express my sincere gratitude to Dr. Behjat for the warm reception I received during my scholarship at her laboratory at the University of Calgary, Canada.

I would also like to acknowledge my colleagues, especially Renan Oliveira Netto, Erfan Aghaeekiasaraee, Sheiny Fabre Almeida, and Upma Gandhi, for all the discussions and help during my research.

Finally, I would like to thank the committee members Gracieli Posser, Renato Fernandes Hentschke, and Rafael de Santiago for giving up their time to evaluate this work.

“Oh, home, let me come home
Home is wherever I’m with you ..."
(Edward Sharpe)

RESUMO

O projeto de circuitos integrados *Very Large-Scale Integration* (VLSI) atuais contitui-se em tarefas extremamente complexa e por isso, deve seguir um fluxo rigoroso utilizando ferramentas computacionais sofisticadas denominadas *Electronic Design Automation* (EDA). Devido à complexidade dos circuitos contemporâneos, a síntese física tornou-se um passo crucial para alcançar o fechamento do projeto. Neste contexto, os processos de posicionamento e roteamento são partes fundamentais da síntese física, uma vez que impactam diretamente o desempenho, a área, o consumo de energia e a confiabilidade do circuito. Para lidar com tal complexidade dos circuitos VLSI modernos, as etapas de posicionamento e roteamento são normalmente abordadas separadamente, aplicando-se uma abordagem de "dividisão e conquista". Infelizmente, devido ao aumento contínuo da complexidade das regras de projeto, a convergência de soluções pode sofrer desalinhamento, e os efeitos de um posicionamento insatisfatório serão notados tão somente durante o roteamento, quando o posicionamento seria considerado fixo. Esta tese apresenta uma técnica chamada ILPGRC, que significa "ILP-Based Global Routing Optimization With Cell Movements". O núcleo da técnica ILPGRC é composto por um modelo de Programação Linear Inteira (*Integer Linear Programming* - ILP) que simultaneamente move células e re-roteia as interconexões. A técnica ILPGRC permite a realocação de células que podem levar a problemas de roteamento, sem comprometer a qualidade em relação ao número de VIAs (*Vertical Interconnection Access*), comprimento das interconexões e violações de regras de projeto (*Design Rule Violations* - DRVs). Esta tese também propõe uma estratégia de particionamento chamada *Checkered Paneling*, que reduz o tamanho de entrada do modelo ILP, tornando esta abordagem escalável. A estratégia de *Checkered Paneling* também permite a execução de vários modelos ILP em paralelo, proporcionando aceleração para circuitos grandes. Além disso, esta tese apresenta uma abordagem baseada em *cluster* de GCells para legalizar a solução com o mínimo de perturbação. O método proposto é testado nos *benchmarks* da competição ACM/IEEE International Symposium on Physical Design (ISPD) 2018 e 2019 dentro de um fluxo de síntese física composto por ferramentas acadêmicas de posicionamento e roteamento do estado da arte. Os resultados após o roteamento detalhado mostram que a técnica ILPGRC pode reduzir, em média, o número de VIAs em 4,69%, com menos de 1% de impacto no comprimento das interconexões. Além disso, ILPGRC reduz o número de DRVs na maioria dos casos, sem deixar interconexões incompletas. Comparando ILPGRC com o trabalho estado da arte CRP 2.0, ILPGRC reduz o número de VIAs em 5,61% em média, enquanto a CRP 2.0 atinge apenas 3,59% de redução, assumindo a mesma referência, com um impacto semelhante no comprimento das interconexões. Esta comparação indica que a técnica ILPGRC é a melhor abordagem para otimizar a solução de roteamento detalhado através de movimentos de células durante o roteamento global, funcionando como uma etapa adicional entre o roteamento global e o roteamento detalhado no fluxo de projeto físico.

Palavras-chave: Electronic Design Automation. Physical Design. Integer Linear Programming. Placement. Routing. Routing with cell movement.

RESUMO ESTENDIDO

Introdução

O projeto de circuitos integrados VLSI (*Very Large-Scale Integration*) atuais constitui-se em uma tarefa extremamente complexa que deve seguir metodologias rigorosas utilizando ferramentas sofisticadas denominadas EDA (*Electronic Design Automation*) (Papa, 2010; Kahng et al., 2011). Devido a tal complexidade, a síntese física tornou-se crucial para alcançar o fechamento do projeto. Além disso, para satisfazer requisitos de tempo de lançamento no mercado cada vez mais curtos, os fluxos de projeto de circuitos VLSI contam com bibliotecas de componentes pré-projetados (chamados de *standard cells*) e blocos de propriedade intelectual.

O fluxo começa com a captura do comportamento do sistema em um alto nível de abstração. Então, o sistema é particionado em software e hardware. A seguir, a arquitetura de hardware é definida e descrita em RTL (*Register Transfer Level*) usando uma linguagem de descrição de hardware (em inglês, *Hardware Description language* - HDL) como VHDL ou Verilog. Tal descrição é traduzida para o nível lógico, resultando em uma lista de portas lógicas, elementos sequenciais (*latches* e/ou *flip-flops*) e interconexões, geralmente chamada de *netlist*. Na etapa de projeto físico, as portas lógicas e os elementos sequenciais da *netlist* são associados às descrições geométricas das associações de transistores, ou seja, às células disponíveis na biblioteca, que serão posicionadas e interligadas, formando uma descrição completa das máscaras que são utilizadas para fabricar o Circuito Integrado (CI). Tais descrições ainda deverão ser verificadas antes de serem enviadas para fabricação. A verificação segue regras que capturam as limitações físicas do processo de fabricação. Por exemplo, todos os fios devem estar a uma distância mínima e ter uma largura mínima absoluta. Depois de fabricado, o CI é testado e encapsulado.

No projeto físico, duas etapas se destacam por suas complexidades: posicionamento (em inglês, ***placement***) e roteamento (em inglês, ***routing***). A etapa de **posicionamento** é responsável por determinar posições de todas as células na superfície bidimensional (2D) do circuito. Para lidar com a complexidade dos circuitos atuais contendo até milhões de células, a etapa de posicionamento é geralmente dividida em três subetapas (Kahng et al., 2011):

- **Posicionamento global**, que encontra a posição inicial para todas as células no circuito desconsiderando os tamanhos e restrições das células, permitindo assim sobreposições e desalinhamentos de células;
- **Legalização**, responsável por alinhar todas as células às linhas e colunas da grade de roteamento do circuito e por remover as sobreposições de células;
- **Posicionamento detalhado**, que otimiza as posições de algumas células, uma vez que o processo de legalização pode degradar algumas métricas, tais como atraso crítico, densidade e comprimento do fio.

Roteamento é o processo de interligação de todos os elementos do circuito, sendo uma etapa essencial no fluxo de projeto de CIs. Tal importância advém do fato de que, à medida que a tecnologia foi evoluindo, as dimensões foram reduzidas para valores abaixo do mícron. Com isso, o atraso das interconexões tornou-se mais relevante do que os atrasos dos componentes lógicos, se tornando assim o principal responsável pelo desempenho dos circuitos (Liu et al., 2018; Liu et al., 2013; Cho; Pan, 2007). Como resultado, as interconexões críticas devem ser roteadas nas camadas superiores de metal para reduzir

a resistência e, portanto, os seus atrasos. Porém, o número de *tracks*[1] nas camadas superiores é muito limitado. Particularmente, menos *tracks* de metal estão disponíveis nas camadas superiores do que nas inferiores, o que complica bastante o roteamento, tornando-o uma das etapas mais desafiadoras no fluxo do projeto físico. Portanto, a qualidade de um roteador global influencia profundamente a temporização, a potência e a densidade de um circuito (Chang et al., 2010).
Geralmente, as etapas de posicionamento e de roteamento são tratadas como dois problemas separados e resolvidos usando uma abordagem de divisão e conquista. Devido à separação entre essas etapas, os efeitos de um posicionamento inadequado podem ser amplificados durante o roteamento, a ponto de o circuito ser considerado impossível de ser roteado e o posicionamento precisar ser refeito. Isso irá resultar em significativo aumento no tempo de projeto, com consequências no custo final. Uma solução alternativa reside em permitir movimentos de células durante o roteamento. Esses movimentos podem aliviar o congestionamento, permitindo que o roteador conclua todas as interconexões. Mover células durante o roteamento também pode reduzir o comprimento total dos fios, bem como o número de VIAs, resultando em menor consumo de energia e possivelmente otimizando o desempenho do circuito. Recentemente, o roteamento global com movimentação de células ganhou destaque ao ser tema de duas competições de CAD no ICCAD em 2020 (Hu et al., 2020) e em 2021 (Hu et al., 2021). Após estas competições, foram publicados alguns trabalhos que propõem alterações no posicionamento das células durante o roteamento, dentre os quais citam-se: Fontana et al. (2021), Wang et al. (2021), Huang et al. (2021), Zou et al. (2022), Aghaeekiasaraee et al. (2022), Aghaeekiasaraee et al. (2023), Fontana et al. (2023).

Objetivos

O principal objetivo deste trabalho é melhorar a qualidade do fluxo de projeto físico através de movimentos de células durante o roteamento global, em vez de tratar o posicionamento e o roteamento como etapas separadas. Nesse contexto, este trabalho explora os impactos da movimentação de células nas métricas de qualidade tradicionais, como comprimento das interconexões, número de VIAs e DRVs. Para isso, esta tese apresenta uma técnica chamada **ILPGRC**, que significa "ILP-Based Global Routing Optimization With Cell Movements". O núcleo da técnica ILPGRC é composto por um modelo de Programação Linear Inteira (em inglês, *Integer Linear Programming* - ILP) que simultaneamente move células e re-roteia as interconexões.
As principais contribuições desta tese são as seguintes:

- Desenvolvimento de uma formulação de Programação Linear Inteira (ILP) que simultaneamente move células e re-roteia interconexões visando otimizar o roteamento e reduzir o número de violações de regras de projeto (em inglês, *Design Rule Violations* - DRV).
- Elaboração de uma estratégia de particionamento de regiões dinâmica e hierárquica, denominada *Checkered Paneling*, que reduz o tamanho de entrada do modelo ILP e permite a sua paralelização. Esta estratégia permite também ordenar as interconexões de acordo com a sua área, tratando primeiro as interconexões menores.

[1] *Tracks* são as linhas metálicas utilizadas para realizar as conexão. Geralmente, possuem largura mínima, mas sempre respeitando o espaçamento mínimo para outras *tracks* adjacentes. Tanto a largura mínima quanto o espaçamento entre as *tracks* dependem da camada metálica e do nodo tecnológico.

- Criação de uma abordagem baseada em *cluster* para legalizar a solução com o mínimo de perturbação.
- Determinação de um fluxo de síntese física baseado em ferramentas acadêmicas para trabalhos que envolvam as etapas de posicionamento e/ou roteamento. Este fluxo foi determinado avaliando-se doze fluxos diferentes construídos pela combinação de três opções de posicionamento, duas ferramentas de roteamento global e duas ferramentas de roteamento detalhado. Esta tese considera as duas melhores combinações dessas ferramentas como referências para as comparações.
- Avaliação da eficácia das técnicas propostas após a etapa de roteamento detalhado por meio de um fluxo de projeto acadêmico usando ferramentas de posicionamento e roteamento de última geração e circuitos das competições CAD do ISPD 2018 (Mantik et al., 2018) e ISPD 2019 (Liu et al., 2019). As informações provenientes do roteamento detalhado permitem uma análise profunda e realista, incluindo algumas métricas como VIAs e *off-track* VIAs, *off-track wires, wrong-way wires, metal shorts, min-areas, spacing rules, e open nets*. Isto contrasta com a maioria dos trabalhos relacionados, que limitam o seu processo de avaliação a informações de roteamento global.

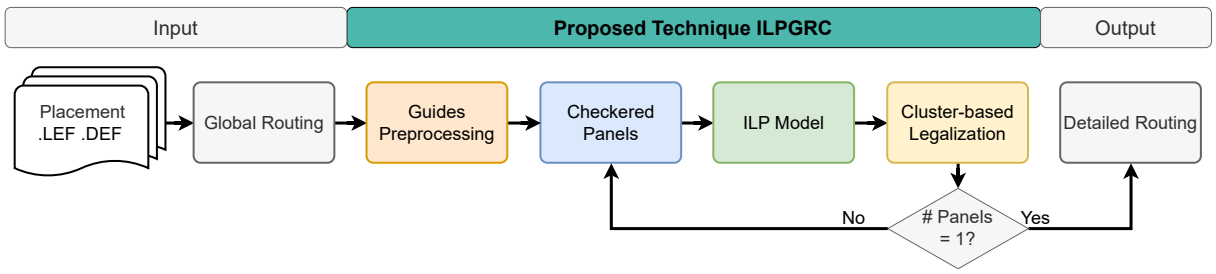
As principais contribuições desta tese foram publicadas em três artigos:

1. ISVLSI 2021 : "ILP-Based Global Routing Optimization with Cell Movements".
2. JICS 2022 : "Towards a Reference Place and Route Flow for Academic Research".
3. TCAD 2023 : "ILPGRC: ILP-Based Global Routing Optimization With Cell Movements".

Metodologia

O fluxograma da técnica proposta (ILPGRC) é apresentado na Figura 1. As entradas são o arquivo de tecnologia (.lef), o arquivo de projeto (.def) e a solução inicial do roteamento global (.guide). Library Exchange Format (LEF) e Design Exchange Format (DEF) são dois formatos industriais para descrever um projeto. O arquivo LEF define os parâmetros de fabricação para uma tecnologia e uma biblioteca de modelos de células. O arquivo DEF define os componentes relevantes para o projeto físico de um CI, incluindo a lista das interconexões (*netlist*) e as restrições de projeto. Como saída, ILPGRC produz um novo arquivo DEF contendo a posição de todos os componentes e um arquivo .GUIDE que contém a solução do roteamento global.

Figura 1 – Fluxograma da técnica proposta (ILPGRC).

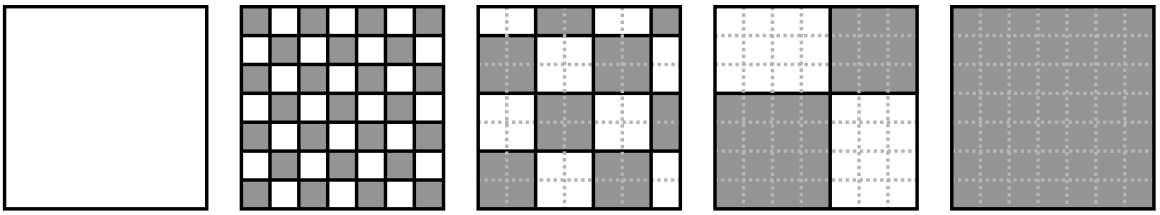


Fonte: O autor.

O primeiro passo de ILPGRC consiste em pré-processar a solução de roteamento global de entrada, de modo a mapear os retângulos-guia (*Guides*) originais para a estrutura GCell adotada. Em seguida, o *layout* do circuito é particionado pela técnica *Checkered*

Paneling, conforme ilustrado na Figura 2. O tamanho do painel em cada nível subsequente é definido como sendo o dobro do tamanho do painel no nível anterior. O processamento termina quando todo o *layout* é coberto por um único painel. Dentro de um determinado nível, cada painel está associado a uma cor (preto ou branco). Painéis pertencentes ao mesmo nível e cor serão executados em paralelo. É importante observar que um painel não possui vizinhos da mesma cor imediatamente a sua direita ou esquerda. Tal característica visa garantir que uma célula que cruze a borda do painel não seja mapeada simultaneamente para duas *threads*. Consequentemente, o ILPGRC é determinístico entre execuções sequenciais e paralelas, ou seja, execuções sequenciais e paralelas produzirão o mesmo resultado.

Figura 2 – Exemplo da aplicação de *Checkered Panels* para um dado *layout*.



(a) *Layout* (b) Nível 1 (c) Nível 2 (d) Nível 3 (e) Nível 4

Fonte: O autor.

Posteriormente, o modelo ILP é construído e resolvido para cada *panel* de um dado nível. Note que dado um *panel* arbitrário, o modelo ILP somente considerará as células e interconexões que estão totalmente dentro da região do mesmo. As demais células e interconexões serão tratadas como fixas. Outro ponto importante é que cada interconexão pertencerá a somente um *panel* e nível, sendo considerada fixa nos demais *panels* e níveis. A solução do modelo ILP fornece a posição ótima para cada célula e suas respectivas interconexões. Porém, tal posição resultante do modelo ILP pode não estar legalizada, ou seja, pode haver sobreposição com outras células do circuito. Por isso, a última etapa da técnica proposta é a legalização baseada em *clusters* de GCells. Por fim, ao final de cada iteração de nível de *panels*, o circuito encontra-se legalizado e completamente roteado.

Resultados e Discussão

Nesta tese, os impactos da técnica proposta foram avaliados após o roteamento detalhado. Tal estratégia constitui-se em diferencial em relação aos trabalhos correlatos, uma vez que ela fornece uma avaliação mais precisa do impacto da técnica sobre o fluxo de projeto completo, e não apenas sobre a etapa de roteamento global. Quando comparado com a melhor combinação de ferramentas acadêmicas do estado de arte (ISPD + CUGR + TritonRoute), ILPGRC reduz o número de VIAs em $4,69\%$, movendo apenas $1,98\%$ das células e degradando o comprimento das interconexões em menos de 1%, em média. Além disso, o roteador detalhado não relatou interconexões abertas após a técnica proposta, enquanto o número de DRVs foi reduzido em dois dos três casos. A estratégia *Checkered Paneling* também permitiu a execução de vários modelos Integer Linear Programming (ILP) em paralelo, proporcionando aceleração de até 6.4 vezes para o circuíto ispd18_test10.

Comparando com a segunda melhor combinação de ferramentas acadêmicas do estado de arte (DreamPlace + CUGR + TritonRoute), o ILPGRC reduz o número de VIAs em $5,53\%$, movendo $3,26\%$ das células e degradando o comprimento das interconexões em

0,61%, em média. ILPGRC também atingiu soluções sem DRVs em 14 dos 17 circuitos testados, enquanto o fluxo de referência só produziu 9 soluções sem DRVs.
No experimento que comparou ILPGRC com a técnica CRP2.0, considerada estado da arte, até então, houve uma redução média de 2% maior no número de VIAs quando a ILPGRC foi utilizada, ao passo que o impacto médio no comprimento das interconexões foi de apenas 0,25%. O fluxo com CRP 2.0 falhou para um circuitos ispd19_test4, ao passo que fluxo incorporando a ILPGRC foi concluído com sucesso para todos os circuitos. No que se refere aos DRVs, dos 20 circuitos testados, CRP2.0 produziu 14 ao passo que ILPGRC produziu 16. Sendo assim, a técnica proposta nesta tese, ILPGRC, superou em qualidade a técnica que até então era o estado da arte.
ILPGRC é escalonável devido à estratégia proposta de *Checkered Paneling*. Essa estratégia foi capaz de acelerar a técnica em até 6x usando oito threads, mantendo sob controle o número de variáveis do modelo ILP. Experimentos adicionais foram executados para avaliar a qualidade dos resultados em diferentes cenários onde toda a técnica ILPGRC foi iterada cinco vezes no mesmo circuito, mais candidatos de movimento foram gerados, e considerados diferentes valores nas constantes de penalidade dos diferentes níveis de metal. Executando o ILPGRC cinco vezes, é possível reduzir ligeiramente o número de VIAs para alguns circuitos. Gerar mais candidatos não ajudou a melhorar a qualidade dos resultados. Isso ocorre porque a mediana da GCell é o ponto ótimo para um movimento, considerando todas as redes conectadas a uma célula. Portanto, o modelo ILP preferirá mover as células para este ponto mediano em vez das GCells vizinhas. O terceiro experimento mostra que as constantes de peso não possuem muito impacto na qualidade final pdos resultados.

Considerações Finais
Esta tese propôs, desenvolveu e avaliou ILPGRC, uma técnica baseada em ILP para otimizar o roteamento global. ILPGRC move células e re-roteia redes simultaneamente, mantendo a legalidade do circuito e evitando violações das regras de projeto. Isto contrasta com a abordagem tradicional que considera o posicionamento fixo durante as etapas de roteamento. Esta tese também propôs uma estratégia de *Checkered Paneling*, a qual reduz o tamanho de entrada do modelo ILP, tornando assim a abordagem ILPGRC escalável. A estratégia *Checkered Paneling* também permite a execução de vários modelos ILP em paralelo, proporcionando aceleração para grandes circuitos. A técnica ILPGRC permite que o roteador mova células para otimizar diferentes objetivos de roteamento, como o número total de VIAs, comprimento das interconexões e DRVs. ILPGRC pode auxiliar outros algoritmos de roteamento no trabalho colaborativo com os algoritmos de posicionamento, tornando tanto o posicionamento quanto o roteamento mais ágeis e eficientes.
Três experimentos suplementares foram fornecidos para avaliar o ILPGRC em um fluxo diferente usando DreamPlace, comparando-o com o CRP 2.0, e com a versão anterior da técnica ILPGRC publicada na conferência ISVLSI 2021. ILPGRC conseguiu produzir melhores resultados em comparação com todos os trabalhos e fluxos destes experimentos. Portanto, a técnica ILPGRC deve ser considerada como a técnica de otimização de roteamento com movimentação de células estado da arte.

Palavras-chave: Electronic Design Automation. Physical Design. Integer Linear Programming. Placement. Routing. Routing with cell movement.

ABSTRACT

The design of current Very Large-Scale Integrated (VLSI) circuits is an extremely complex task and, therefore, must follow a strict flow using sophisticated computational tools referred to as Electronic Design Automation (EDA). Due to the complexity of contemporary circuits, physical synthesis has become a crucial step for achieving design closure. In this context, the placement and routing processes are key parts of the physical synthesis since they directly impact the circuit performance, area, power consumption, and reliability. To handle the high complexity of modern VLSI circuits, placement and routing steps are typically tackled separately by applying a divide-and-conquer approach. Unfortunately, due to the continuous increase of design rules complexity, the convergence of solutions can suffer from misalignment, and the effects of an unsatisfactory placement will be noticed only during routing when the placement would be considered fixed. This thesis presents a technique called **ILPGRC**, which stands for "ILP-Based Global Routing Optimization With Cell Movements". The core of ILPGRC is composed of an Integer Linear Programming (ILP) model that simultaneously moves cells and reroutes the nets. ILPGRC enables the relocation of cells that can lead to routing issues without compromising the quality concerning the number of VIAs, wirelength, and Design Rule Violations (DRVs). This thesis also proposes a partitioning strategy named Checkered paneling, which reduces the input size of the ILP model, making this approach scalable. The Checkered paneling strategy enables the execution of multiple ILP models in parallel, providing a speedup for large circuits. Additionally, this thesis presents a GCell cluster-based approach to legalize the solution with minimum disturbance and displacement. The proposed method is tested on the ACM/IEEE International Symposium on Physical Design (ISPD) 2018 and 2019 contest benchmarks within a physical synthesis flow composed of state-of-the-art place and route academic tools. The results after the detailed routing show that ILPGRC can reduce, on average, the number of VIAs by 4.69% with less than 1% impact on wirelength. Furthermore, ILPGRC reduces the number of DRVs in most cases with no open nets left. Comparing ILPGRC with the so-far state-of-the-art work CRP 2.0, ILPGRC reduces, on average, the number of VIAs by 5.61% while CRP 2.0 only achieves 3.59% assuming the same baseline, with a similar impact in the wirelength. This comparison indicates that ILPGRC is the best approach to optimize the detailed routing solution through cell movements during global routing, working as an additional step between global routing and detailed routing in the physical design flow.

Keywords: Electronic Design Automation; Physical Design; Integer Linear Programming; Placement; Routing; Routing with cell movement.

LIST OF FIGURES

LIST OF TABLES

LIST OF ALGORITHMS

LIST OF SYMBOLS

b_i	Pin blockage
$\mathcal{C}$	Set of Cells
$C(n_i)$	Set of cells connected to n_i
c_i	Cell i
$cost_{i,j}$	Cost of n_i using candidate j
$D(g_i^k)$	Demand of GCell g_i^k
$\Delta(g_m, g_n)$	Distance between g_m and g_n
$\mathcal{G}$	Set of GCells
$\mathcal{G}(n_i)$	GCells crossed by net n_i
g_i^k	GCell i of metal layer k
H_{row}	Height of circuit row
$L(c_i)$	Set of possible locations of c_i
$loc(p)$	Location of Pin p
$m_{i,j}$	Position Candidate
$max(g_i)$	Max corner of GCell g_i
$min(g_i)$	Min corner of GCell g_i
$\mathcal{N}$	Set of Nets
$\mathcal{N}(c_i)$	Nets connected to c_i
$N(m_{ij})$	Set of nets of $m_{i,j}$
n_i	Net i
$\mathcal{P}$	Set of Pins
$\mathcal{P}(n_i)$	Pins belonging to net n_i
p_i	Pin i
$R(m_{i,j})$	Set of routing candidates of $m_{i,j}$
$R(g_k)$	Set of net candidates routed through g_k
$r_{i,j}$	Routing candidate
$S(g_i^k)$	Supply of GCell g_i^k
$\mathcal{V}$	Set of Vias
$\mathcal{V}(n_i)$	Vias belonging to net n_i
via_{ij}	Number of vias of candidate j of net i
X_{left}	Left limit of circuit rows
X_{right}	Right limit of circuit rows
$x(c_i)$	Coordinate X of cell c_i
W_{site}	Width of circuit site
$w(c_i)$	Width of cell c_i
wl_{ij}	Wirelength of candidate j of net i

LIST OF SYMBOLS

Y_{bottom}	Lower limit of circuit rows
Y_{top}	Upper limit of circuit rows
$y(c_i)$	Coordinate Y of cell c_i
α_k	Wirelength weight of metal layer k
β_k	Via weight of metal layer k
ω	Via factor

LIST OF ABBREVIATIONS AND ACRONYMS

BFS Breadth-First Search. 33

DLM Discrete Lagrange Multipliers. 31, 33
DP Dynamic Programming. 32, 33
DR Detailed Routing. 26, 39, 64
DRV Design Rule Violation. 28, 77, 78

EDA Electronic Design Automation. 23, 28, 31

GR Global Routing. 25, 26, 27, 28, 29, 30, 31, 32, 33, 64, 77

HDL Hardware Description Language. 23
HPWL Half-Perimeter Wirelength. 35

IC Integrated Circuit. 24, 25, 27
ILP Integer Linear Programming. 11, 12, 28, 30, 32, 33, 34, 35, 39, 41, 51, 53, 54, 56, 58, 60, 61, 62, 65, 67, 68, 69, 71, 72, 75, 76, 77, 78

LA Layer Assignment. 26, 27, 31, 32, 33, 34

NVM Negotiation Based Layer Assignment. 33

RMST Rectilinear Minimum Spanning Tree. 33
RRR Rip-up and Reroute. 32, 33
RSMT Rectilinear Steiner Minimum Tree. 27, 31, 33
RTL Register Transfer Level. 23

SLR Systematic Literature Review. 89

TNS Total Negative Slack. 34

VHDL VHSIC Hardware Description Language. 23
VHSIC Very High Speed Integrated Circuits. 19
VIA Vertical Interconnect Access. 9, 10, 11, 12, 14, 25, 27, 28, 29, 32, 34, 35, 38, 39, 42, 48, 49, 50, 53, 56, 57, 65, 66, 67, 68, 70, 72, 73, 74, 75, 76, 77, 78
VLSI Very Large-Scale Integrated. 23, 75

CONTENTS

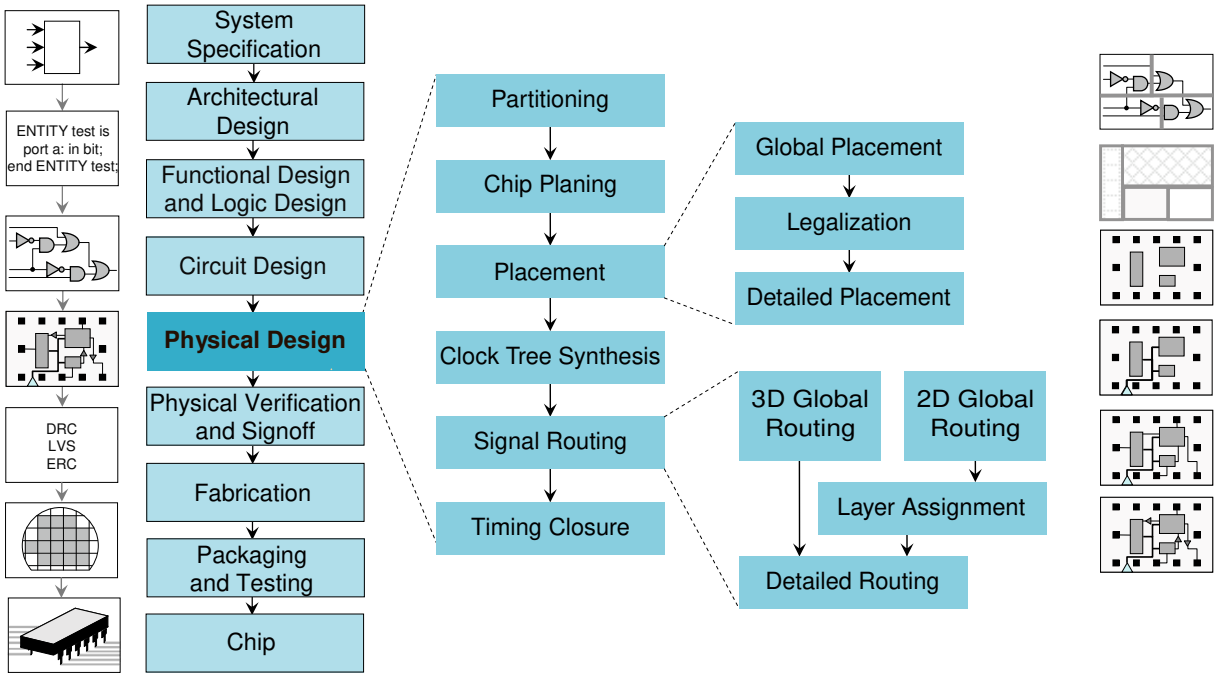
1 INTRODUCTION

The design of contemporary Very Large-Scale Integrated (VLSI) circuits is an extremely complex task that must follow stringent methodologies using sophisticated tools referred to as Electronic Design Automation (EDA) (Papa, 2010; Kahng et al., 2011). Due to such complexity, physical synthesis has become crucial for achieving design closure. In addition, to satisfy the ever-shorter time-to-market requirement, the standard VLSI circuit design flows rely on libraries of pre-designed components (the so-called standard cells) and intellectual property blocks.

Figure 1 summarizes the most important steps of the traditional VLSI design flow, which relies on the standard cells methodology. Notice that although the flow presented is linear for simplicity, during its execution usually one or more steps may have to be repeated, giving rise to loops in the flow.

The flow begins by capturing the system behavior at a high level of abstraction. Then, the system is partitioned into software and hardware. Next, the hardware architecture is defined, and the Register Transfer Level (RTL) is described using a Hardware Description Language (HDL) such as VHDL or Verilog. Such description is translated to the logic level, resulting in a list of logic gates, sequential elements (latches and/or flip-flops), and interconnections, usually referred to as netlist. In the physical design step, the logic gates and sequential elements of the netlist are associated with geometric descriptions of transistor associations, i.e., the cells, which will be positioned and interconnected,

Figure 1 – Major steps of the VLSI design flow. The main physical design steps are illustrated in the middle, whereas the right-hand side details the placement and the signal routing step.



Source: adapted from Kahng et al. (2011)

forming a complete description of the masks that are used to manufacture the Integrated Circuit (IC). Such descriptions must still be verified before being sent to manufacturing. Verification follows rules that capture the physical limitations of the manufacturing process. For example, all wires must be at a minimum distance and have an absolute minimum width. Once manufactured, the IC is tested and packaged.

The dominant steps of the physical design flow are detailed in the central portion of Figure 1. First, the netlist is partitioned, and the topological circuit planning is performed. Then, the standard cells are positioned and legalized in the placement step. After that, the clock network (sometimes called the clock tree) is synthesized. Then, the circuit signals are routed using the metal layers available in the target fabrication technology. Finally, the time closure step verifies and optimizes the circuit timing.

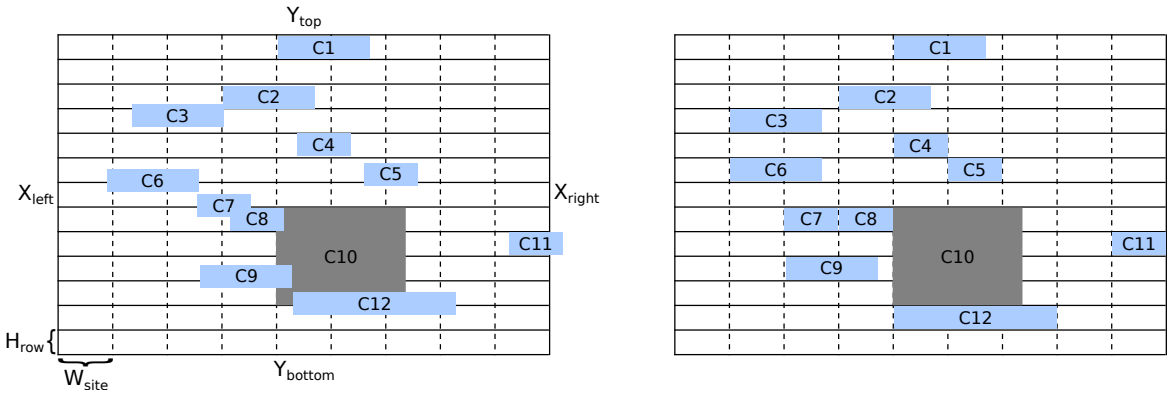
The **placement** step is responsible for positioning all cells in the two-dimensional (2D) circuit surface. This surface is divided into rows (horizontal slices) and sites (vertical columns). Figure 2 depicts the circuit surface where the solid horizontal lines represent the rows with height equal to H_{row} and the dashed vertical lines represent the sites with width equal to W_{site}. To tackle the complexity of modern circuits containing up to millions of cells, the placement step is usually divided into three sub-steps (Kahng et al., 2011), as shown on the right-hand side of Figure 1:

- *Global Placement* finds the initial position for all cells in the circuit by disregarding the cells' sizes and constraints, thus allowing cell overlaps and misalignments. A simplified example of the global placement is depicted in Figure 2(a) where the blue rectangles represent the cells and the gray rectangle represents a macroblock or fixed cell. In this example, it is possible to observe three types of placement violations: 1) cell overlaps ($C7$ and $C8$, $C9$ and $C10$, $C10$ and $C12$); 2) cells not aligned with circuit sites and rows ($C3$ to $C9$, $C11$, $C12$); 3) cells outside the circuit boundaries ($C11$).
- *Legalization* aligns all cells to circuit sites and rows and removes cell overlaps. This should be done relocating a subset of cells C' while minimizing the placement perturbation (displacement). A common displacement metric is the sum of all Manhattan distances[1] between the cell location before and after the legalization. Figure 2(b) depicts the result of a legalization algorithm on the example presented in Figure 2(a). The legality constraints are formalized in Chapter 4 by Equation 4.2 to 4.7.
- *Detailed Placement* is the final placement step and optimizes some cells' positions since the legalization process could degrade some metrics, such as timing, density, and wirelength.

Signal routing, which is the process of interconnecting all elements of the circuit, is an essential step in the flow. This is because as technology has scaled down into deep sub-micron, the delay of the interconnections became more important than the delays of

[1] Given two points (x, y) and (x', y'), the Manhattan distance is given by $|x - x'| + |y - y'|$.

Figure 2 – Example of (a) Global Placement and (b) Legalization step. Blue rectangles represent cells, and the gray rectangle represents a macroblock.

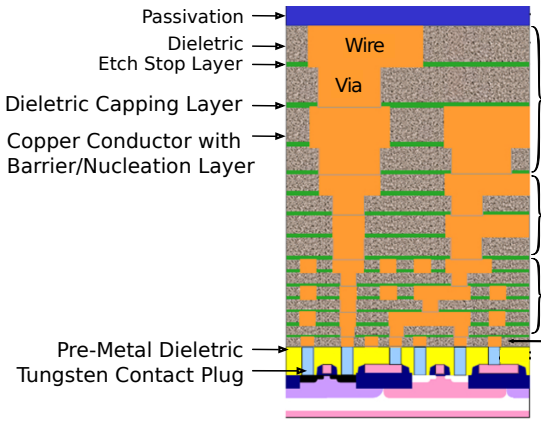


(a) (b)

Source: the author.

the gates thus being the main responsible for the performance of the circuits (Liu et al., 2018; Liu et al., 2013; Cho; Pan, 2007). As a result, the critical nets should be routed in the upper layers to reduce the resistance of the interconnections and hence their delays. However, as it can be deduced from Figure 3, the number of tracks[2] in the upper layers is very limited. Particularly, fewer metal tracks are available in the upper layers than in the lower ones, which greatly complicates the routing, making it one of the most challenging steps in the physical design flow. Therefore, the quality of a global router deeply influences the timing, power, and density of a chip (Chang et al., 2010).

Figure 3 – The cross-section of IC interconnection stack in advanced technologies (Schaller, 2004), where wires and VIAs on top metal layers are much wider and much less resistive than those on lower metals. The normalized pitch lengths of different metal layers are listed in the table.



Layer	Pitch
M10	1.00
M9	1.00
M8	0.63
M7	0.63
M6	0.63
M5	0.63
M4	0.63
M3	0.33
M2	0.35
M1	0.33

Source: adapted from Hsu et al. (2014) and Liu et al. (2018).

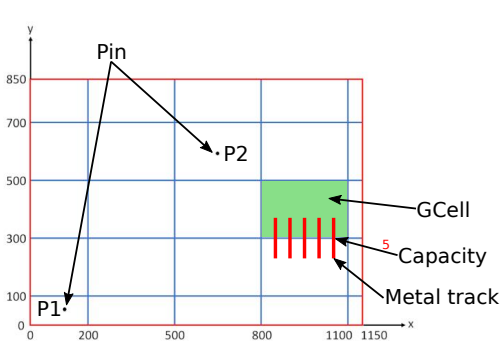
The routing process is divided into two steps: Global Routing and Detailed Routing. The Global Routing (GR) step allocates routing resources that are used for intercon-

[2] Tracks are the metal lines used to make a connection, generally with minimum width but always respecting the minimum spacing to other adjacent tracks. Both minimum width and spacing between tracks depend on the metal layer and technology node.

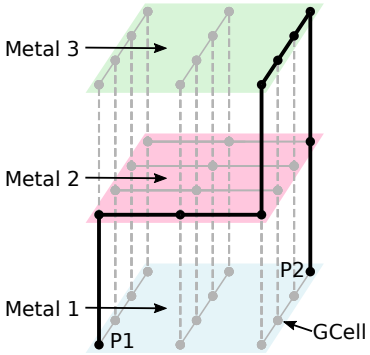
nections, and Detailed Routing (DR) is responsible for assigning routes to specific routing tracks within the GR resources (Kahng et al., 2011).

In the GR step, the circuit area is partitioned into GCells (Grid Cells), as illustrated in Figure 4(a). Each GCell has two attributes: metal layer and capacity. The former associates the GCell with a specific metal layer, whereas the latter defines the number of metal tracks available on that metal layer and the preferred direction that connections should follow. It is important to note that the preferred routing direction is alternated between metal layers. For instance, if tracks run in the horizontal direction in even metal layers then tracks run in the vertical direction in the odd layers. The attribute capacity represents the number of metal tracks available per GCell in a given metal layer and the preferred direction. In Figure 4(a), the green GCell has a capacity equal to five. The goal of GR is to find, for each net, the set of GCells through which it is possible to establish a connection between pins belonging to the same net, as shown in Figure 4(b).

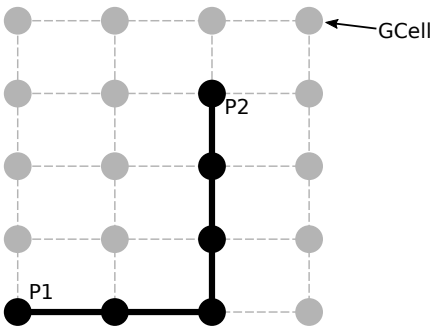
Figure 4 – Example of Global Routing.



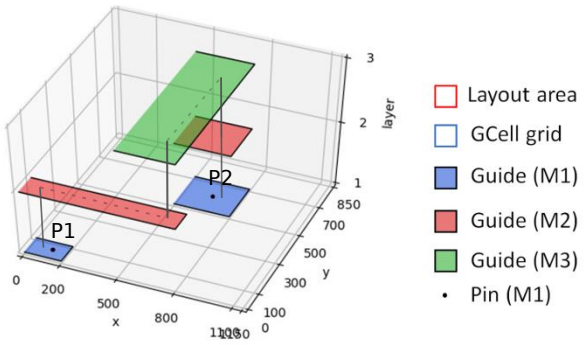
(a) 2D view of a net with two pins.



(b) 3D grid graph where dots represent vertices and dashed lines represent edges between vertices. The solid line is a 3D GR.



(c) 2D grid graph where dots represent the vertices and dashed lines represent the edge between vertices. The solid line is a 2D GR.



(d) 3D view of a valid GR solution.

Source: (a) and (d) adapted from Volkov & Dolgov (2019); (b) and (c) the author.

There are two main ways to perform the GR step: using a 3D data structure or using a 2D data structure, followed by a second step called Layer Assignment (LA) as shown in Figure 1. The 3D GR determines the metal layers during the path search. For

example, Figure 4(b) shows the 3D GR approach applied to a two-pin interconnection (P1 and P2) presented in Figure 4(a). Note that this approach does not require an additional step to determine a layer for each segment. However, due to the high complexity of contemporary designs, full 3D routing normally results in longer execution times than the 2D approach followed by the LA approach (Livramento et al., 2017; Chang et al., 2010). Figure 4(c) presents the 2D GR approach for the example shown in Figure 4(a). In this scenario, the position of each pin is projected onto the 2D surface. Then, a path-search algorithm is performed to find the set of GCells that completes the interconnection. Finally, a LA step is accomplished to determine the layer of each segment. Figure 4(d) shows the result of GR for the two-pin net represented in Figure 4(a).

The example presented in Figure 4 shows the GR flow in a comprehensive though oversimplified way. Indeed, it overlooks several difficulties found in this step, such as congestion, total wirelength, and timing. Congestion occurs when the number of interconnections passing through the same GCell surpasses this GCell capacity, which is common during GR of real ICs. That is why congestion has been addressed by several works found in the literature, such as Gao, Wu & Wang (2008), Ozdal & Wong (2009), Roy & Markov (2008), Chang et al. (2010), Pan et al. (2012).

Besides reducing the congestion, an efficient GR technique should improve the interconnection total wirelength. Some works, such as of Moffitt (2008), Cho & Pan (2007), Cho et al. (2009), Liu et al. (2013), minimize the wirelength using a spatial search algorithm like A* (Hart; Nilsson; Raphael, 1968) or using Rectilinear Steiner Minimum Tree (RSMT). However, these works do not consider any timing information in their metrics, which makes them inaccurate.

With the advancement of technology, the wire delay became more prominent than the gate delays (Kahng et al., 2011; Held et al., 2017; Tu; Chow; Young, 2017). Thereby, routing became, along with placement, the two most important steps determining the whole circuit performance. Most of the recent works on routing focus on wirelength reduction while ignoring the number of VIAs (Liu et al., 2020; Chen et al., 2020; Kahng; Wang; Xu, 2018; Kahng; Wang; Xu, 2020; Kahng; Wang; Xu, 2021). This increases the mismatch between global and detailed routing.

Yield, reliability, product performance, and cost are the cornerstones of a successful IC manufacturing technology. In such a context, a large number of VIAs can significantly reduce the *reliability* of the circuit (Kahng et al., 2011). Additionally, the delay is proportional to wire resistance. Hence, reducing the number of VIAs can result in less wire resistance.

Moreover, placement and routing are usually treated as two separate problems and solved using a divide-and-conquer approach. Because of the separation between these steps, the effects of a poor placement can be amplified during routing to the extent that the circuit placement is deemed unroutable, and the placement needs to be redone. This means a significant increase in design time and costs. An alternative solution is

to allow cell movements during routing. Those movements can alleviate the congestion, allowing the router to route all the nets. Moving cells during routing can also reduce the total wirelength and number of VIAs, resulting in less power consumption and possibly optimizing the circuit performance. Recently, global routing with cell movement came into focus as it was the subject of two ICCAD CAD Contests: 2020 (Hu et al., 2020) and 2021 (Hu et al., 2021). Since these contests, some recent works proposed to enable changes in cells' placements during the routing, namelly: Fontana et al. (2021), Wang et al. (2021), Huang et al. (2021), Zou et al. (2022), Aghaeekiasaraee et al. (2022), Aghaeekiasaraee et al. (2023), Fontana et al. (2023).

1.1 THESIS HYPOTHESIS AND CONTRIBUTIONS

As a main research question, it is claimed that the quality of the physical design flow may be improved by optimizing the placement through selective cell movements during GR instead of treating placement and routing as separated steps. Therefore, this work explores the impacts of moving cells on traditional quality metrics such as wirelength, number of VIAs, and Design Rule Violations (DRVs).

As developments of this hypothesis, we can list the following hypothesis:

- **Hypothesis 1:** An Integer Linear Programming (ILP) model can be defined to choose the best position of cells and their respective routing segments to optimize the circuit placement and routing simultaneously.
- **Hypothesis 2:** Given the computational complexity of solving ILP models, a partitioning technique should lead to a better runtime for ILP-based optimization methods for EDA tools. This approach systematically sorts and breaks the data dependency between the circuit's nets. This is crucial since an ILP model encompassing all circuit cells and interconnections can involve millions of variables and billions of constraints, thus leading to prohibitive runtime.
- **Hypothesis 3:** The real impact of placement and routing optimization techniques can only be appropriately measured after the detailed routing solution is completed.

This thesis presents a technique called **ILPGRC**, which stands for "ILP-Based Global Routing Optimization With Cell Movements". The core of ILPGRC is composed of an ILP model that simultaneously moves cells and reroutes the nets. The major contributions of this thesis are as follows:

- Developing an Integer Linear Programming (ILP) formulation that simultaneously moves cells and reroutes nets targeting routing optimization and DRV reduction.
- Designing a dynamic and hierarchical region-based partitioning strategy, named Checkered Paneling, which reduces the input size of the ILP model and enables

parallelization. This strategy also allows sorting the nets according to their area, making it possible to treat small nets first.
- Designing a cluster-based approach to legalize the solution with minimum disturbance and displacement.
- Establish an academic state-of-the-art baseline flow for works in the Placement and/or Routing steps. This flow was settled by evaluating twelve different flows built by combining three placements, two Global Routing, and two Detailed Routing tools. This thesis considers the two best combinations of these tools as baselines.
- Evaluating the effectiveness of the proposed techniques after the Detailed Routing step through an academic design flow using state-of-the-art placers and routers and benchmark circuits from ISPD 2018 (Mantik et al., 2018) and ISPD 2019 (Liu et al., 2019) contests. The detailed routing data allows a deep and realistic analysis, including some metrics such as off-track VIAs and wires, wrong-way wires, metal shorts, min-areas, spacing rules, and open nets. This contrasts most of the related work, which limit their evaluation process to the global routing data.

The main contributions of this thesis were published in three research papers:

1. ISVLSI 2021 : "ILP-Based Global Routing Optimization with Cell Movements" (Fontana et al., 2021).
2. JICS 2022 : "Towards a Reference Place and Route Flow for Academic Research" (Fontana et al., 2022).
3. TCAD 2023 : "ILPGRC: ILP-Based Global Routing Optimization With Cell Movements" (Fontana et al., 2023).

The details on these and other publications achieved during the development of this thesis as well as the list of awards received, can be found in Appendix A.

1.2 THESIS STRUCTURE

The remaining chapters of this thesis are organized as follows.

The related works to this thesis are discussed in **Chapter 2**. First, a review of the most relevant works on Global Routing is presented, followed by a review of placement works that use GR engines to estimate the interconnections to decide the best position for each cell. Lately, all works that move cells during the GR step are reviewed.

In **Chapter 3** it is established an academic state-of-the-art baseline flow for works in the Placement and/or Routing steps. First, the justification for the benchmark set choice is presented. Next, the main characteristics of the evaluator (ISPD evaluator) used in this work are presented. Then, the best combination of placement and routing academic tools is investigated by an experimental exploration of twelve different flows.

Chapter 4 formalizes the GR problem, bringing all the equations needed to route a circuit fully.

Chapter 5 outlines the proposed framework. First, the chapter presents an overview of the proposed technique's steps. Then, the preprocessing guide step is detailed. Next, the Checkered Paneling strategy is presented. Then, the ILP model is proposed to move cells and reroute nets simultaneously. Finally, the GCells cluster-based legalization approach is presented.

Chapter 6 presents the experimental results. First, the methodology and experimental setup are presented, followed by the default values of GCells, panels, and constants used in this thesis. Then, four sets of experiments are shown: 1) an overall quality experiment in relation to the best placement and routing academic flow established in Chapter 3. This experiment discusses the workload, runtime, impact of more iterations, impact of more candidates, and the constant values. 2) The ILPGRC technique is evaluated over the second-best academic placement and routing tool combinations. 3) After, in the third set of experiments, ILPGRC is compared with the so far state-of-the-art technique CRP 2.0. 4) In the last analysis, the current version of ILPGRC is compared with its previous version published in the ISVLSI 2021 conference, renamed as ILPGRC$_{V0}$.

The research conclusions are drawn in **Chapter 7**. It also brings suggestions for future research.

2 RELATED WORK

The main subject of this work is the "routing with cell movement" problem. This problem encompasses two classical EDA topics: Global Routing (GR) and Placement. Hence, this chapter presents the most relevant works on GR in Section 2.1. The works that use GR engines to estimate the interconnections to decide the best position for each cell are reviewed in Section 2.2. Lately, Section 2.3 reviews all works that move cells during the GR step.

2.1 GLOBAL ROUTING

This Section presents the selected GR works found by a Systematic Literature Review (SLR) conducted. The methodology used in such a review is described in Appendix B. The Global Routing works can be classified into two groups: **2D** and **3D**. For each work, **three metrics** are evaluated: dimension, net order, and timing. The metric **dimension** corresponds to whether the main routing algorithm uses data structures in two or three dimensions. As mentioned in Chapter 1, the result of GR is a set of GCells and their layers required to route each net. Therefore, the final output of GR is always 3D. In cases where the main routing algorithm uses 2D data structures, an additional step of LA is needed to transform the 2D solution into a 3D solution. **Net Order** relates to the order in which the algorithm routes nets. This order can be arbitrary, from the shortest to the longest net, or can use the number of pins belonging to each net. The metric **timing** refers to whether the routing technique uses timing information to sort the nets or to guide the path search to improve timing.

2.1.1 2D Global Routing

A frequent solution to handle the GR problem for a net with more than two pins is decomposing it into a set of two-pin nets. This technique is called multi-pin decomposition and is used in many 2D global routing techniques, such as: MaizeRouter (Moffitt, 2008), BoxRouter (Cho; Pan, 2007; Cho et al., 2009), NTHU-Route (Gao; Wu; Wang, 2008; Chang et al., 2010), FastRoute (Xu; Zhang; Chu, 2009; Pan et al., 2012), and NCTU-GR (Dai; Liu; Li, 2011; Liu et al., 2013).

The technique of Moffitt (2008), called MaizeRouter, constructs a RSMT using FLUTE (Chu; Wong, 2007). Then, it shifts and retracts the edge to optimize congestion. The algorithm performs in 2D, and thus, requires a post step of LA. This technique does not sort nets in a specific.

Roy & Markov (2008) propose a technique based on Discrete Lagrange Multipliers (DLM), which provides a natural way to handle net weights optimization in GR. They implemented their algorithm in "Fairly Good Router" (FGR), which handles 2D and 3D

routing applications. First, the technique projects the routing instance onto a 2D grid and then aggregates the capacities of edges. The grid contains a single layer of horizontal and vertical wires connected by a layer of vias. Then, FGR routes this 2D instance and performs a LA step in the end. FGR is also capable of solving 3D problems directly by using full 3D maze routing. The authors report that the full 3D routing takes at least 50% longer. However, it can decrease the VIA counts by 2.0% and improve the total cost by 2.1%.

Cho & Pan (2007) proposes a router based on congestion-initiated box expansion called BoxRoute. BoxRouter progressively expands a box that initially covers the most congested region only but, finally, comprises the whole circuit. After every expansion, the circuit is divided into two sections: inside the box and outside the box. The wires inside the box have priority over those outside the box. BoxRouter uses different routing strategies for each section to maximize routability and minimize wirelength. However, this work did not consider circuit layers, only routing one layer per time. Posteriorly, Cho et al. (2009) extended BoxRoute to introduce the multi-layer routing. This is achieved by performing a 2D GR followed by layer assignment. The 2D global routing has two ideas: node shifting for congestion-aware Steiner tree and robust negotiation-based A* search for routing stability. The layer assignment step is done by progressive VIA/blockage-aware ILP. Nevertheless, these two works do not adopt any net ordering.

Ozdal & Wong (2009) proposed Archer: A history-based global routing algorithm. It is a Rip-up and Reroute (RRR)-based global routing algorithm that guides the routing iterations out of local optima through practical usage of congestion histories. They proposed a Lagrangian relaxation strategy bounded by wirelength that enables Steiner trees to dynamically balance the trade-off between total overflow and wirelength in history cost. The idea is to increase the costs of the routing resources that have been congested for several iterations so that only the nets that need to use these resources end up using them. This congestion cost is used to sort the net order during the RRR. Although the technique can use 3D structures, they cast the structures to 2D and perform a post-layer assignment step.

NTHU-Route (National Tsing Hua University) was first proposed in Gao, Wu & Wang (2008) and later improved by Chang et al. (2010). This router decomposes the multi-pin nets using FLUTE. Then, each two-pin net is routed as a L_shaped structure. Then, it improves the initial solution with RRR. The congested region identification method defines the order for nets to be ripped up and rerouted. This cost function helps distribute the overflow throughout the circuit. NTHU-Route applies the work "Congestion-constrained layer assignment for VIA minimization in global routing" from Lee & Wang (2008) to achieve the LA step. This LA approach uses Dynamic Programming (DP) to minimize the total number of vias.

Another routing technique that decomposes the multi-pin nets into two-pin nets is FastRoute (Pan; Chu, 2006; Pan; Chu, 2007a; Xu; Zhang; Chu, 2009; Pan et al.,

2012). The first step of FastRoute is to construct a congestion-driven via-aware Steiner topology for each net, and then a segment-shifting technique is applied. Later, the tree structures are decomposed into 2-pin nets using FLUTE. Then, a pattern routing step using an L-shape and Z-shape initializes the routing solution for each net, and the virtual capacity based on the current routing status is initialized. Next, a loop of rip-up and reroute process is executed until the overflow stops decreasing. Inside the loop, it uses pattern routing and multi-source multi-sink maze routing, and virtual capacity to reduce congestion. The virtual capacity gradually changes the capacity associated with each global edge to divert wire usage from highly congested regions to congestion-free regions. Finally, the 2D solution is extended to a full 3D solution by a spiral layer assignment algorithm using DP.

The NCTUgr (Dai; Liu; Li, 2011; Liu et al., 2013) begins with Rectilinear Minimum Spanning Tree (RMST) topologies and utilizes the RSMT topologies to guide the following monotonic routing and negotiation-based RRR. To sort all the nets, it decomposes a net using Breadth-First Search (BFS) and evaluates all sub-nets with the area and congestion. Layer assignment is then performed on all nets sequentially in the decreasing score order.

The work of Moffitt & Sze (2011) focuses on resolving the *layer compliance problem* in routing congestion evaluation and global routing, which is critical for timing closure with physical synthesis. They proposed a method of progressive projection to account for wire tags and layer directives, in which classes of nets are successively applied and locked while performing partial aggregation. After this, each cluster of nets is routed using NTHU-Route 2.0 and FastRoute.

Table 1 summarizes the most relevant features of the related works with 2D dimensions sorted in ascending order of publication. Note that none of the works presented so far use timing information in their routing metrics.

Table 1 – Related works of 2D Global Routing

Work	Year	Technique	Dimen.	Net Order	Timing	LA
MaizeRouter	2008	multi-Pin Decomposition + bounding box	2D	No	No	—
FGR	2008	DLM	2D / 3D	No	No	
BoxRouter	2007; 2009	multi-Pin Decomposition + A* + rip-up and reroute	2D	No	No	ILP
Archer	2009	Lagrangian relaxation + rip-up and reroute	2D / 3D	Congestion History	No	—
NTHU-Route	2008; 2010	FLUTE + rip-up and reroute whit cost function	2D	Congestion	No	DP
MOFFITT; SZE	2011	Cluster nets by timing clousure + NTHU-Route 2.0 and FastRoute	2D	Layers	No	—
FastRoute	2006;2007; 2009;2012	FLUTE + monotonic routing + rip-up and reroute	2D	Wirelength and #Pins	No	DP
NCTU-GR	2012; 2013	RMST + RSMT	2D	Area and Congestion	No	NVM

2.1.2 3D Global Routing

In this section, the 3D GR approaches are reviewed. Unlike the 2D techniques, 3D ones do not require an additional step to determine the metal layers of each segment. However, they usually require longer execution times than 2D ones. Notwithstanding, the popular projection-based layer assignment approach (2D works) fails to adequately

anticipate the nonuniform wire distribution that results from these directives, forcing the router to either violate constraints or incur unexpected overflow (Moffitt; Sze, 2011).

Wu, Davoodi & Linderoth (2009) proposed "GRIP: Scalable 3D Global Routing Using Integer Programming". GRIP optimizes wirelength and VIA cost without going through the LA phase. The solution procedure begins with a column generation to solve the linear programming, followed by branch-and-bound. It relaxes the integer condition to a linear condition to facilitate the solution. Furthermore, it decomposes the circuit into rectangular sub-regions to create smaller ILP-GR instances and accelerate the procedure. The technique can reduce the wirelength by 3.9%, on average, in 2D ISPD 2007 benchmarks and by 11.3%, on average, in 3D ISPD 2007 benchmarks. Note that if ILP formulation is used, it does not make sense to discuss the order of nets since the mathematical solver deals with all nets simultaneously. As in the previously discussed works, this work did not consider timing information in its metrics.

Researchers from the University of Bonn, in cooperation with IBM, developed BonnRoute (Gester et al., 2013), which was extended in other three works: Held et al. (2015), Scheifele (2016), and Held et al. (2017). BonnRoute is based on an efficient, fully polynomial approximation scheme for the min-max resource sharing problem. It produces a provably near-optimal fractional solution, rounds it, and removes the local congestion caused by rounding. This work considers only capacitance as weight and does not mention any timing models to improve the critical paths. Then, Held et al. (2015) extended this work using the multicommodity flow problem and considered the linear model for the timing and congestion as weights of the Steiner Tree. This work clusters the sink pins for bigger instances and then creates the Steiner Tree.

Scheifele (2016) extended the work of Held et al. (2015) by considering the Elmore model for timing formalizing the Congestion-Aware Minimum Elmore Delay Steiner Tree Problem. The authors reported wirelength and Worst Negative Slack (WNS) but did not report the Total Negative Slack (TNS). Besides, it is not possible to change the delay model easily because the timing model is embedded in the technique. Furthermore, the technique does not sort the nets.

Held et al. (2017) group this previous tree works together ((Gester et al., 2013; Held et al., 2015); and (Scheifele, 2016)) and adopt Lagrange multipliers to dynamically adjust delay budgets performing a trade-off between wiring congestion and delay.

Liu et al. (2020) proposed the CUGR, which comprises three steps: initial routing, multi-level 3D maze routing, and route guide generation. In the initial routing, each multi-pin net is broken down into a set of two-pin nets in the pattern routing planning step. Then, in the 3D pattern routing step, the FLUTE (Chu; Wong, 2007) algorithm is used to generate a rectilinear Steiner minimum tree (RSMT), and a dynamic programming algorithm performs pattern routing and layer assignment simultaneously. After initial routing, the nets with violations are submitted to multiple iterations of rip-up and reroute (RRR) by maze routing. The maze routing algorithm is limited to the bounding box of the

net. Such an approach is to reduce the execution time of performing maze routing on the whole 3D grid graph. The last step in the flow is to generate the guides and insert patches. Patches are extra regions of guides to alleviate some conditions, and they can be of three types: 1) *Pin Region Patching* to improve pin accessibility; 2) *Long Segment Patching* to improve the track assignment; and 3) *Violation Patching* to add more flexibility in regions that may contain violations.

Table 2 summarizes the related works with 3D dimension sorted by publication year. Note that none of the techniques found in the literature so far sort nets in their routing approaches.

Table 2 – Related works of 3D Global Routing

Work	Author/Year	Technique	Dimen.	Net Order	Timing
GRIP	2009	ILP	3D	—	No
BonnRoute	2013	Min-Max Resource Sharing Problem	3D	No	No
HELD et al.	2015	Multicommodity flow problem	3D	No	Linear
SCHEIFELE	2016	Multicommodity flow problem	3D	No	Elmore
HELD et al.	2017	Multicommodity flow problem	3D	No	Elmore
CUGR	2020	3D pattern routing step + rip-up and reroute	3D	No	No

2.2 PLACEMENT USING ROUTING ENGINES

Most placement techniques are guided by routing metrics, such as wirelength, congestion, and density. However, many rely on estimates obtained from simplified models instead of the outcome of a router. As a result, there may be a gap between the work and the final result of the route step. This section addresses the works concerning placement that use routers to guide or evaluate their placement solutions.

The first work to propose an integration of a global router in the placement stage was IPR (Pan; Chu, 2007c). The authors integrate FastRoute (Pan; Chu, 2007b) into FastPlace (Viswanathan; Pan; Chu, 2007) to achieve better routability results. FastPlace is a quadratic placer that uses Half-Perimeter Wirelength (HPWL) metric as routing estimation. IPR reduces overflow by 36%, global routing wirelength by 3.6%, and runtime by 36% compared to ROOSTER(Roy; Markov, 2007), which was the previous best academic routability-driven placer.

GRPlacer (Dai; Lu; Li, 2009) uses an embedded global router from Dai, Liu & Li (2009) to find the congested regions of the circuit. Then, it employs cell shifting and cell rearrangement to move cells in order to reduce the routed wirelength. Although this work was routability-driven, it does not directly work on total overflow for cost evaluation, which may not capture routability appropriately.

CRISP (Roy et al., 2009) employs fast global routing, NTHU-Route 2.0 (Chang et al., 2010), to choose standard cells to inflate and interactively spread for congestion reduction temporarily. This work achieved a reduction of 8.7% in VIA counts, 6.5% in globally routed wirelength, and 5.3% in routing detour.

SimPLR (Kim et al., 2011) extends CRISP using a BFG-R router to obtain the congestion map. BFG-R (Hu; Roy; Markov, 2010) is a global router based on Lagrangian relaxation. The cell bloating approach was inspired by CRISP and was used for local refinement and spread overlapped instances.

POLAR 2.0 (Lin; Chu, 2014) uses roughly legalized placement to calculate the pin locations, and then the congestion-aware pattern routing of FastRoute (Pan; Chu, 2007b) is applied to estimate the routing demand.

Table 3 summarizes the related works in the placement step that use a routing engine to guide their techniques sorted by publication year.

Table 3 – Related works in placement

Work	**Year**	**Routing**	**Objective**
IPR	2007	FastRoute	wirelength and overflow
GRPlacer	2009	Embedded	wirelength and routability
CRISP	2009	NTHU-Route 2.0	congestion
SimPLR	2011	BFG-R	routability
POLAR 2.0	2014	FastRoute	congestion

2.3 ROUTING WITH CELL MOVEMENT

Traditionally, the routing engines assume that the cells are fixed. To the best of the author's knowledge, the first work that breaks this traditional approach was SRP (He; Chow; Young, 2013). SRP is a technique that relocates cells placed under a congested region to reduce the congestion after the Global Routing step. SRP unplaces one cell at a time, and the new cell location is defined using a multi-source propagation method on each associated net of this cell. After the new location is determined, the opened sub-nets are rerouted by the maze routing algorithm. SRC was evaluated using the DAC 2012 benchmarks set and reduced the total overflow by 32.6% on average.

Recently, the ICCAD 2020 (Hu et al., 2020) and 2021 (Hu et al., 2021) CAD Contests brought the community's attention to this topic again. These two contests proposed "*routing with cell movement*" to point out that the routing engines need to be able to move some cells and reroute some nets to optimize the routing solution. As a result, some new academic works on this topic have been published lately.

To fuse routing with cell movement, Wang et al. (2021) developed a partial rerouting approach that integrates routing with cell movements called Starfish. They devised a multi-source single-target A* algorithm responsible for reconnecting relocated cells to the previous routing topology's core. In addition, Starfish introduces a dynamic threshold mechanism to enhance the benefit gained from each cell movement. Furthermore, the authors proposed a lookup table for determining optimal candidate destinations and estimating the advantage of cell movements. Nevertheless, it remains unclear how accurately

this lookup table can perform in benchmarks featuring advanced technology nodes and comprehensive routing data because this work was only evaluated in the ICCAD 2020 Contest benchmarks (Hu et al., 2020) suite.

Huang et al. (2021) use Breadth-First Search (BFS)-based approximation to reduce the optimal region for cell movement. The BFS search considers routing constraints such as layer direction, minimum layer, and overflow as the search heuristic. This work uses A* for partial rerouting after cell movement and ICCAD 2020 Contest benchmarks. The technique proposed by Zou et al. (2022) first alleviates the congestion by rerouting the circuit using a congestion-aware 3D global routing. Then, it tries to move cells using a modified version of SRP (He; Chow; Young, 2013). In the end, it applies an edge-adjusting algorithm to reduce the wirelength. As in previous works, Zou *et al.* evaluated their technique using the ICCAD 2020 Contest benchmarks.

Zhu et al. (2022) proposed a cell movement approach based on a lookup table considering routing directions and layer-based power consumption. The lookup table of wirelength is used to generate a gain map for each movable cell. Then, based on the gain map, alternately perform several rounds of cell movement and partial rip-up and rerouting. After each movement, they must re-estimate and generate a new gain map for the subsequent cells since this estimation probably becomes inaccurate. Ultimately, they moved some cells to their original positions to legalize the maximum number of moved cells imposed by the ICCAD Contest.

Zang et al. (2022) proposed a two-level layer-aware scheme named ATLAS. ATLAS first performs an Incremental 3D Global Routing to improve only the routing. Then, it groups the cells through a VIA-sharing cluster. After this, it iteratively moves cells to the median and reroutes the nets using an A*-based partial rerouting until no gain in routing is observed. Finally, a single-cell movement is performed to their original positions to satisfy the movement constraint.

Both Zhu *et al.* and ATLAS were evaluated using the ICCAD 2020 and 2021 Contest benchmarks and slightly outperformed the first-place team of the ICCAD 2021 contest.

The six aforementioned works have similar drawbacks: they only support the DAC 2012, ICCAD 2020, and ICCAD 2021 Contest benchmarks. Unfortunately, these benchmark suites are oversimplified since they rely only on GCell information to specify the locations of cells and net connectivity. In addition, there is no information about technology nodes, cell geometries, circuit rows, circuit sites, and routing tracks. Hence, it is impossible to know if a movement could be legalized and if the optimization in the global routing step will result in some optimization after the detailed routing. Therefore, by using these sets of benchmarks, it is not possible to measure the real impact of movements in the circuit after the physical design flow is finished.

Recently, CRP and its extension CRP 2.0, (Aghaeekiasaraee et al., 2022; Aghaeekiasaraee et al., 2023), use a cost function to identify critical areas and reduce the congestion

by moving some cells. Then, an ILP-based legalizer is used to generate new legal locations for the candidate cells. CRP 2.0 also introduces a caching technique to speed up the method. CRP and CRP 2.0 techniques were evaluated using the ISPD 2018 and 2019 Contest benchmarks after the detailed routing solution. CRP and CRP 2.0 are the first works that evaluate the optimization quality after the Detailed Routing step. CRP reduced the number of VIAs by 2.06% and the wirelength by 0.14% on average. CRP 2.0 reduced the number of VIAs by 3.59% and the wirelength by 0.09% on average. Unfortunately, CRP and CRP 2.0 first move a set of cells, leaving the rerouting of affected nets to the end. Consequently, rerouting may not be possible for some (or all) of those new cell locations, demanding more work to revert these changes.

Table 4 summarizes the related works presented in this section ordered by publication year. The columns bring the acronym, title, year of publication, benchmarks used in the evaluation process, and the objective for each work. None of the related work has the number of VIAs after the Detailed Routing as objective. This metric is only presented in the technique proposed by this thesis: ILPGRC. Another important observation is that only CRP, CRP2.0, and this thesis technique evaluated the results using Detailed Routing information.

Table 4 – Related works in routing with cell movement

Work	Title	Year	Benchmarks					Objective
			DAC	ICCAD		ISPD		
			2012	2020	2021	2018	2019	
SRP	SRP: simultaneous routing and placement for congestion refinement	2013	X					GR Overflow
Starfish	Starfish: An Efficient P&R Co-Optimization Engine with A*-based Partial Rerouting.	2021		X				GR Wirelength
Huang et al.	Detailed Placement and Global Routing Co-Optimization with Complex Constraints.	2021		X				GR Wirelength
Zou et al.	Incremental 3D Global Routing Considering Cell Movement and Complex Routing Constraints	2022		X				GR Wirelength
Zhu et al.	A Robust Global Routing Engine with High-accuracy Cell Movement under Advanced Constraints	2022		X	X			GR Wirelength
ATLAS	ATLAS: A Two-Level Layer-Aware Scheme for Routing with Cell Movement	2022		X	X			GR Wirelength
CRP	Cr&p: An efficient co-operation between routing and placement.	2022				X		DR Score
CRP 2.0	CRP2.0: A Fast and Robust Cooperation between Routing and Placement in Advanced Technology Nodes	2023				X	X	DR Score
ILPGRC	ILPGRC: ILP-Based Global Routing Optimization With Cell Movements	2023				X	X	DR Vias DR Wirelength

2.4 SUMMARY

In this chapter, the pertinent related works were comprehensively examined. Firstly, the key features of Global Routing methods were deepened through a comparative analysis between 2D and 3D approaches, while the limitations associated with each technique were highlighted. Following, the works that proposed placement engines guided by routing metrics were reviewed. Within this context, a comparative assessment of their objectives was conducted, considering metrics such as wirelength, congestion, and density. Finally, the focus shifted to related works in routing with cell movement. In this context, the techniques' objectives are compared along with each technique's weaknesses. Also, the limitations of the benchmarks used to validate each technique were also pointed out.

The literature gap becomes evident when the concepts from these three sets of related works are consolidated. None of the existing works simultaneously addresses the following critical points: 1) eliminating the issue of net ordering; 2) guaranteeing cell movements that improve or maintain the initial solution; 3) incorporating VIAs as primary objectives; 4) evaluating the results after DR step to ensure that the technique is effective at the end of the physical flow. ILPGRC, the technique proposed in this thesis, effectively encompasses all these aspects: the three first aspects are covered by the ILP model that moves all cells at the same time, considers the initial solution as one of the possible solutions, and incorporates VIAs in the objective function. The remaining point is addressed by the evaluation process applied in the experimental results of this thesis.

3 A REFERENCE PLACE AND ROUTE FLOW FOR ACADEMIC RESEARCH

This chapter aims to establish an academic baseline flow for works in the Placement and/or Routing steps. This chapter is based on the publication named "Towards a Reference Place and Route Flow for Academic Research" (Fontana et al., 2022).

The separation into placement and routing steps is essential to handling the massive complexity of contemporary designs. However, these two steps are strongly correlated so that small inefficiencies in the placement solution can be amplified during routing, thus deteriorating design quality and convergence. For example, a region with a high cell density can lead to pin access issues in detailed routing.

Given the aforementioned difficulties, it is possible to deduce that it is a challenge to select which placement and routing tools to use when synthesizing a circuit. This difficulty affects not only the designers who want to rely on academic tools but also the researchers. Particularly, most academic research works evaluate their results only at the target step without considering the complete place and route flow. However, as presented in Section 2.3, some recent works propose optimization techniques to promote the collaboration between routing and placement steps, and thus, their improvements should be evaluated considering the whole physical synthesis flow.

Since 2016, the IEEE CEDA Design Automation Technical Committee (DATC) has been developing a public reference design flow named DATC Robust Design Flow (RDF) (Chen et al., 2021). This flow aims to provide a foundation and backplane for academic research in the RTL-to-GDS IC implementation arena. The DATC RDF-2021 version includes the OpenROAD Flow project (Kahng; Spyrou, 2021) for the physical synthesis. However, the DATC RDF flow is simply a tool-chain integration, and no detailed analyses of those tool interactions are given. Searching to fill this gap, this chapter brings an experimental assessment of twelve flows built up from academic placers and routers to determine which one can lead to the best results so that researchers can use it as a reference. To evaluate those flows, the ISPD 2018 (Mantik et al., 2018) and ISPD 2019 (Liu et al., 2019) CAD Contest benchmarks were used, which are the most realistic academic benchmarks available with placement and routing information.

The rest of this Chapter is structured as follows. First, Section 3.1 brings the motivation to choose the ISPD CAD Contest benchmarks. Then, in Section 3.2, the details of the evaluation and the metrics are detailed. Next, Section 3.3 presents a quality evaluation of the placement and routing flows. Finally, the top three flows for each benchmark considering the ranking based on score and short violations are depicted in Figure 5. The two best flows, Contest + CUGR + TritonRoute and DreamPlace + CUGR + TritonRoute, are used as baselines for the quality evaluation of ILPGRC, the technique proposed in this thesis, in Section 6.5 and Section 6.6, respectively.

3.1 BENCHMARKS

The choice of the ISPD Contest 2018 (Mantik et al., 2018) and ISPD Contest 2019 (Liu et al., 2019) benchmarks is due to the fact that they are the most recent benchmarks that include technology files and advanced design constraints, thus allowing the generation of detailed routing solutions. In turn, the ICCAD CAD Contest 2020 (Hu et al., 2020) and ICCAD CAD Contest 2021 (Hu et al., 2021) benchmarks are overly simplified and do not have technology information and hence, were not used. For example, ICCAD 2020 and ICCAD 2021 benchmarks consider that the cells are placed in the center of GCells. Since there is no row or site information, it is not possible to produce detailed routing solutions because the placements cannot be legalized.

On the contrary, the ISPD Contest 2018 (Mantik et al., 2018) and ISPD Contest 2019 (Liu et al., 2019) benchmarks have LEF and DEF files, and the circuits use one among three technology nodes: 65nm, 45nm, and 32nm. Additionally, these benchmark suites also provide technology information for the circuits, such as standard cells, Macros, and IO Cells, and the circuits are entirely placed and legalized. Finally, cells have complex pin shapes such as L, Z, and others. Nonetheless, there is no power and timing information in these benchmarks. The main characteristics of these circuits are presented in Table 5. In this table, the first column brings the circuit names. For simplicity, the circuit names were shortened by removing the "ispd" and "test" words. For example, circuit "ispd18_test1" has been renamed "18_t1". Columns 2 to 8 in Table 5 display the technology node, number of cells, number of nets, number of I/O pins, number of macro blockages, placement density, and number of routing layers.

Note that benchmark 18_t10 is the only circuit with 100% of density. This is because, in this circuit, there are no empty spaces due to the use of filler cells. Filler cells have no logical functionality but have the VDD/VSS metal lines matching the rest of the standard cells, ensuring the connectivity of power nets. Due to the lack of empty spaces, the ILP model produces a solution for circuit 18_t10 that is precisely the same as the input placement. Therefore, a modified version of circuit 18_t10, called 18_t10_nf, was created by removing the filler cells. This way, the density of 18_t10_nf is 92%. Another interesting point is that all the circuits have 9 metal layers for routing, except circuits 19_t4 and 19_t5, which have only 5 metal layers.

3.2 ISPD EVALUATOR

The quality of the routing solution was measured using the official contest ISPD 2018 evaluator. This evaluator receives as input the DEF, Guide, and LEF files and executes the following tasks:

1. invokes Cadence Innovus (Cadence, 2020) to perform design rule and connectivity

Table 5 – Main characteristics of ISPD 18 and ISPD 19 benchmarks. From the left to right, columns bring the circuit names, technology node, number of cells, number of nets, number of I/O pins, number of macro blockages, placement density, and number of routing layers.

ISPD Circuits	Node (nm)	Cells (K)	Nets (K)	I/O Pin	Macros	Density (%)	Layers
18_t1	45	9	3	0	0	85	9
18_t2	45	36	37	1,211	0	57	9
18_t3	45	36	37	1,211	4	65	9
18_t4	32	72	72	1,211	4	89	9
18_t5	32	72	72	1,211	8	92	9
18_t6	32	108	108	1,211	0	99	9
18_t7	32	180	180	1,211	16	90	9
18_t8	32	192	180	1,211	16	90	9
18_t9	32	193	179	1,211	0	91	9
18_t10	32	290	182	1,211	0	100	9
18_t10_nf	32	290	182	1,211	0	92	9
19_t1	32	9	3	0	0	83	9
19_t2	32	72	72	1,211	4	72	9
19_t3	32	8	9	57	4	84	9
19_t6	32	180	180	1,211	16	75	9
19_t7	32	360	359	2,216	16	96	9
19_t8	32	540	538	3,221	16	79	9
19_t9	32	899	895	3,221	16	84	9
19_t10	32	899	895	3,221	16	88	9
19_t4	65	146	152	4,802	7	21	5
19_t5	65	29	29	360	6	9	5

checking,

2. generates design rule violation and connectivity reports,
3. starts an evaluation program to perform guide and track obedience checking and read the Innovus reports, and
4. generates the report table as output.

Then, the metrics used to evaluate the quality of results in this work are:

- *Number of VIAs*: the total number of VIAs after the Detailed Routing.
- *Wirelength*: the total length of wires after the Detailed Routing.
- *Off-track VIAs*: the number of VIAs whose center is misaligned with the locations of the tracks.
- *Off-track wirelength*: the length of wires placed out of the track locations.
- *Wrong way wirelength*: the length of wires routed in the non-preferred direction of the layer.
- *Shorts*: either a VIA or wire metal overlaps with another object like VIA, wire metal, blockages, or pin shapes.
- *Min Area*: specifies the minimum metal area required for polygons on each layer[1].

[1] Min Area counts the number of occurrences in which a metal shape area is less than the specified

- *Spacing*: specifies the required spacing between two objects. This metric encompasses the different types of violations which are parallel-run length, end-of-line (EOL), and cut spacing.
- *Open Nets*: if any pin in a net is disconnected, then the net will be considered as an open net.

3.3 QUALITY EVALUATION

In this section, the experimental results of the twelve different flows for the ISPD 2018 and ISPD 2019 benchmarks are presented. Tables 6 and 7 report the breakdown of the results for the detailed routing metrics on each circuit. In both tables, the circuit names (Benchmarks) are displayed in Column 1. Columns 2, 3, and 4 identify the placement, Global Routing, and Detailed Routing tools used in each flow, respectively. Then, Columns 5 and 6 present the routing metrics Wirelength (reported in millimeters) and number of Vias (reported in thousands). For each circuit, the lowest (best) values of wirelength and number of vias are highlighted in bold. Columns 7 to 11 report the routing preference metrics. These metrics are off-guide wirelength (OFGW), off-guide vias (OFGV), off-track wirelength (OFTW), off-track vias (OFTV), and wrong-way wirelength (WWW). Columns 12 to 15 bring the Design Rule Violations (DRV). Then, Columns 16 and 17 present, for each circuit, the final score and the difference to the best score, in percentage. Therefore, in Column 17, 0% identifies the flow that achieved the best score for a given circuit. The flows were ranked based on the final score and the number of short violations. This rank is presented in Column 18 of Tables 6 and 7. It was considered that the solutions with short violations should not outrank the solutions without short violations. That is why all solutions without short violations, highlighted in bold, come before the solutions that leave short violations.

Before analyzing the results in more detail, it is important to remark that the two flows that use Eh?Placer together with CUGR failed for all benchmarks. For such reason, they do not appear in Tables 6 and 7. Further investigation is necessary to determine why CUGR reports an error when applied to placements issued by Eh?Placer. In these executions, the binary of CUGR just stops the execution after the "mark fixed metal rtrees..." and "mark fixed metal batch ..." messages, but no errors are reported in the CUGR log file. In addition to these two flows, other flows have produced errors in a few circuits. These errors are reported in the line of the respective flow, falling in one of the following cases:

- *Time Out 72h*: the experiment runtime was limited to 72 hours.
- *Out of Memory*: the RAM memory utilization was limited to 64GB.

value for a layer. Therefore, it may occur multiple times for a single net.

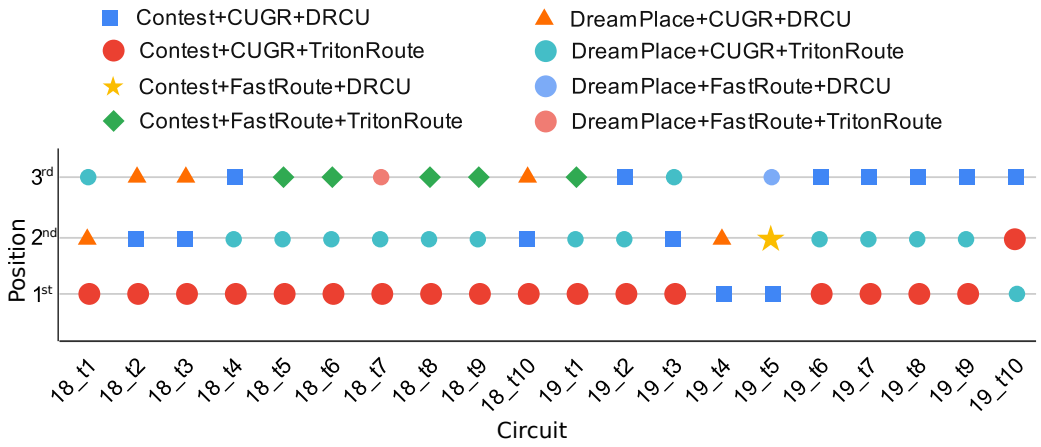
- *TritonRoute Fail*: this error occurs at the end of the "0th optimization iteration" after printing the message "post-processing ..." in the log. No error message was reported at that instant. It is possible that open nets occurred, which could not be resolved by TritonRoute.
- *Eh?Placer Fail*: Eh?Placer reported "std::bad_alloc" in the log file.
- *FastRoute Fail*: FastRoute reported the message "Routing congestion too high" in the log file.

Analyzing all results, one observes that no combination of placement and routing tools led to the best result for all circuits. Considering the top 3 solutions for each circuit, the wirelength varied by 1.7% on average, and the number of vias varied by 4.8% on average. In addition, it is important to note that at least one of the top 3 best solutions was generated by a flow that begins with the original ISPD Contests placement. This means that the original ISPD Contests placements are already well-optimized for routing solutions. It is also possible to conclude that the ISPD 2019 circuits are more difficult to synthesize than the ISPD 2018 ones due to the higher number of failing flows. Particularly, only two flows were able to generate valid solutions for circuit ispd19_test4. The reason why all the other flows failed could be because this circuit has only five metal layers available for routing and was designed with 65 nm technology.

Observing the Design Rule Violations columns of Tables 6 and 7, for each of the circuits, it is possible to note a considerable variance between the flows. Considering only wirelength and number of vias and taking into account only the Global Routing tool, we can observe that the flow that uses CUGR generated the best results for 18 out of 20 circuits. One can also notice that FastRoute could lead to the best wirelength only for circuit ispd19_test6 and produced the lowest number of vias for circuit ispd19_test10. For a given combination of Placement and Global Routing tools, it can be observed that TritonRoute leaves much fewer violations than Dr. CU.

The best three flows for each circuit are depicted in Figure 5. Observe that the flow Contest + CUGR + TritonRoute led to the best score for 17 circuits, the flow Contest + CUGR + Dr. CU led to the best score for 2 circuits, and the flow DreamPlace + CUGR + TritonRoute restulted in the best score only for circuit ispd19_test10. Finally, the conducted experiments indicate that the best combination of academic open-source tools for placement and routing is Contest + CUGR + TritonRoute, considering the ISPD Contest 2018 and ISPD Contest 2019 benchmarks.

Figure 5 – Top three flows for each benchmark considering the ranking based on score and short violations.



Source: the author.

Table 6 – Experimental results for ISPD 2018 Benchmarks.

Circuit	Flow			Routing Metrics		Routing Preference Metrics					Design Rule Violations				Score		Rank
	Placement	Global Routing	Detailed Routing	Wirelength (mm)	Vias (K)	OFGW (mm)	OFGV	OFTW (mm)	OFTV	WWW (mm)	Shorts	Min Area	Spacing	Open Nets	(K)	(%)	
ispd18_test1	Contest	CUGR	DRCU	171.7	**31.7**	0.4	195	0.2	0	2.5	28,800	0	1	0	286.1	0	6
			TritonRoute	**171.5**	35.7	2.9	438	0.3	171	3.4	0	0	0	0	302.5	5.7	**1**
		FastRoute	DRCU	181.3	35.0	71.1	19,413	0.3	33	6.7	19,326,600	1,740	212	0	1,547.4	440.9	7
			TritonRoute	175.5	38.2	89.4	25,612	0.4	220	5.7	0	0	0	0	559.7	95.6	**4**
	DreamPlace	CUGR	DRCU	177.6	36.0	1.0	605	0.1	0	2.0	0	0	7	0	305.6	6.8	**2**
			TritonRoute	176.3	38.0	3.0	2,061	0.6	196	3.5	0	0	0	0	315.6	10.3	**3**
		FastRoute	DRCU	187.6	35.0	72.8	19,213	0.3	36	6.5	13,139,000	1,771	267	0	1,582.3	453.1	8
			TritonRoute	182.6	38.7	93.6	25,831	0.4	215	5.4	0	0	0	0	579.8	102.7	**5**
	EhPlacer	FastRoute	DRCU	231.4	37.7	88.0	21,004	0.3	62	6.6	14,443,600	1,599	287	0	1,610.6	463	9
			TritonRoute	227.7	41.1	114.5	27,303	0.5	229	5.4	0	0	0	2	-	-	-
ispd18_test2	Contest	CUGR	DRCU	**3,120.5**	**316.0**	8.8	3,636	2.4	0	24.8	504,000	0	35	0	4,642.3	0	2
			TritonRoute	3,134.1	359.0	29.4	5,203	4.0	2,164	31.8	0	0	0	0	4,801.1	3.4	**1**
		FastRoute	DRCU	3,284.7	383.1	971.6	218,204	4.5	651	68.0	502,668,340	18,418	3,938	0	20,444.2	340.4	5
			TritonRoute	3,201.0	424.2	1,140.0	294,248	5.2	4,489	48.0	0	0	0	7	-	-	-
	DreamPlace	CUGR	DRCU	3,155.3	381.2	24.1	10,342	2.1	0	19.4	684,400	0	114	0	4,887.6	5.3	3
			TritonRoute	3,150.7	388.1	37.7	25,759	6.6	4,405	32.2	0	0	0	1	-	-	-
		FastRoute	DRCU	3,328.9	388.1	958.8	224,622	4.1	490	68.4	445,963,620	15,596	3,641	0	18,747.6	303.8	4
			TritonRoute	TritonRoute Fail		-	-	-	-	-	-	-	-	-	-	-	-
	EhPlacer	FastRoute	DRCU	3,284.0	394.0	999.2	227,476	3.7	676	68.6	434,541,160	18,891	4,179	0	20,688.2	345.6	6
			TritonRoute	3,184.3	436.5	1,157.7	302,482	5.8	4,673	49.3	0	0	0	4	-	-	-
ispd18_test3	Contest	CUGR	DRCU	**3,490.6**	**316.2**	23.6	4,025	2.7	0	25.1	8,287,400	0	84	0	5,192.5	0	2
			TritonRoute	3,504.6	357.1	35.2	5,072	4.8	2,197	32.4	0	0	0	0	5,277.0	1.6	**1**
		FastRoute	DRCU	3,636.3	385.8	986.6	217,716	3.9	537	68.0	667,804,940	21,042	5,013	0	23,290.9	348.5	6
			TritonRoute	TritonRoute Fail		-	-	-	-	-	-	-	-	-	-	-	-
	DreamPlace	CUGR	DRCU	3,533.2	376.6	27.3	10,578	2.4	0	19.4	64,358,800	0	184	0	5,593.0	7.7	3
			TritonRoute	3,528.8	382.4	39.7	25,502	6.7	4,430	32.2	0	0	0	2	-	-	-
		FastRoute	DRCU	3,672.5	388.6	953.2	222,050	3.9	450	69.1	564,123,420	16,726	4,030	0	20,291.9	290.8	4
			TritonRoute	3,610.8	432.0	1,111.4	299,697	5.7	4,541	49.2	0	0	0	1	-	-	-
	EhPlacer	FastRoute	DRCU	3,986.5	409.7	1,000.2	231,887	3.7	554	70.7	641,798,620	17,562	4,070	0	21,538.5	314.8	5
			TritonRoute	TritonRoute Fail		-	-	-	-	-	-	-	-	-	-	-	-
ispd18_test4	Contest	CUGR	DRCU	5,269.8	727.6	30.4	11,610	3.6	0	38.5	1,535,508	16	677	0	15,360.4	3.3	3
			TritonRoute	**5,239.8**	**719.4**	27.2	7,965	10.0	17,336	27.8	0	0	0	0	14,863.5	0	**1**
		FastRoute	DRCU	5,471.8	955.1	1,761.6	603,650	7.4	593	96.2	253,641,208	21,317	8,208	0	43,434.1	192.2	6
			TritonRoute	5,375.2	976.3	1,486.1	687,267	11.7	16,938	61.4	0	0	1	1	-	-	-
	DreamPlace	CUGR	DRCU	5,356.0	719.9	35.6	21,525	3.8	0	41.1	1,644,428	103	612	0	15,622.3	5.1	4
			TritonRoute	5,313.3	721.5	43.9	52,160	10.5	17,363	31.5	0	0	0	0	15,198.7	2.3	**2**
		FastRoute	DRCU	5,555.7	968.2	1,783.0	620,757	7.9	556	96.4	211,694,612	18,680	8,216	0	41,958.1	182.3	5
			TritonRoute	5,458.8	988.0	1,455.0	700,581	11.8	17,279	60.8	0	0	0	4	-	-	-
	EhPlacer	FastRoute	DRCU	9,262.5	1,439.9	3,743.1	1,071,534	8.8	426	295.4	8,132,654,144	18,731	37,502	0	177,097.4	1091.5	7
			TritonRoute	Time Out 72h		-	-	-	-	-	-	-	-	-	-	-	-
ispd18_test5	Contest	CUGR	DRCU	5,504.0	926.9	21.9	6,401	1.2	3	10.2	3,205,312	72	482	0	16,100.4	3.1	6
			TritonRoute	**5,481.3**	**843.4**	20.1	8,227	3.8	14,946	17.1	0	0	0	0	15,608.9	0	**1**
		FastRoute	DRCU	5,609.4	958.7	2,936.1	735,473	4.7	666	77.5	819,660,424	25,720	9,489	0	59,606.9	281.9	10
			TritonRoute	5,591.8	978.2	2,398.0	781,933	5.6	21,272	45.2	0	0	0	0	28,969.0	85.6	**3**
	DreamPlace	CUGR	DRCU	5,614.5	915.5	27.7	23,559	1.7	10	14.1	172,833,808	701	604	0	18,916.6	21.2	7
			TritonRoute	5,579.6	889.3	62.8	63,515	4.0	19,637	18.2	0	0	0	0	16,225.5	4	**2**
		FastRoute	DRCU	5,713.6	973.5	2,955.7	744,270	4.4	564	76.7	760,443,668	19,502	7,287	0	55,048.5	252.7	8
			TritonRoute	5,695.1	986.6	2,439.4	786,447	5.4	21,538	45.2	0	0	1	0	29,456.0	88.7	**4**
	EhPlacer	FastRoute	DRCU	5,887.3	969.3	3,071.6	741,542	4.3	646	72.6	713,646,584	20,817	7,643	0	56,281.5	260.6	9
			TritonRoute	5,882.7	1,005.9	2,564.7	802,004	5.7	21,412	46.8	0	0	0	0	30,613.2	96.1	**5**
ispd18_test6	Contest	CUGR	DRCU	7,118.8	1,388.1	8.3	6,059	2.4	16	14.5	42,000	106	640	0	21,072.8	2.2	6
			TritonRoute	**7,100.2**	**1,278.5**	31.5	10,082	4.2	22,899	23.8	0	0	0	0	20,627.5	0	**1**
		FastRoute	DRCU	7,267.7	1,449.6	4,421.3	1,119,359	37.6	18,245	115.4	1,210,463,600	28,627	11,483	0	80,169.0	288.7	10
			TritonRoute	7,236.5	1,464.1	3,692.4	1,169,063	5.4	31,482	65.6	0	0	0	0	41,023.6	98.9	**3**
	DreamPlace	CUGR	DRCU	7,787.0	1,398.6	19.2	27,471	2.9	20	17.7	215,210,100	1,168	671	0	26,093.6	26.5	7
			TritonRoute	7,752.2	1,381.8	92.6	76,400	4.9	28,522	24.7	0	0	0	0	22,847.7	10.8	**2**
		FastRoute	DRCU	7,973.7	1,514.9	4,615.3	1,168,527	26.2	11,203	122.2	985,468,100	17,806	8,356	0	73,295.9	255.3	8
			TritonRoute	7,924.9	1,509.4	3,866.8	1,201,688	5.5	31,301	67.5	0	0	0	0	43,749.4	112.1	**5**
	EhPlacer	FastRoute	DRCU	7,500.7	1,464.5	4,641.0	1,121,862	31.9	17,153	108.0	1,046,733,300	26,581	11,053	0	78,545.7	280.8	9
			TritonRoute	7,484.8	1,488.9	3,924.6	1,177,346	5.6	31,088	66.3	0	0	0	0	42,866.5	107.8	**4**
ispd18_test7	Contest	CUGR	DRCU	12,977.9	2,289.1	16.5	9,888	5.3	0	22.8	5,395,132	148	93	0	37,430.5	1.2	5
			TritonRoute	**12,917.7**	**2,096.0**	49.1	15,965	7.1	28,490	37.4	0	0	0	0	36,980.6	0	**1**
		FastRoute	DRCU	13,417.5	2,443.1	4,461.1	1,497,361	39.7	26,634	255.5	1,746,034,824	41,214	17,876	0	115,006.7	211	8
			TritonRoute	13,165.7	2,409.5	2,966.0	1,533,012	9.1	29,076	100.9	0	0	2	0	54,653.6	47.8	**4**
	DreamPlace	CUGR	DRCU	13,141.7	2,228.2	60.1	117,793	6.4	0	30.6	514,719,112	2,399	853	0	45,957.4	24.3	6
			TritonRoute	13,055.3	2,214.0	172.2	196,878	8.0	28,557	37.8	0	0	0	0	38,361.8	3.7	**2**
		FastRoute	DRCU	13,603.5	2,469.0	4,208.5	1,450,091	36.0	23,174	241.2	1,599,312,268	35,892	16,507	0	108,949.3	194.6	7
			TritonRoute	13,368.1	2,414.5	2,825.6	1,463,221	9.2	29,061	102.3	0	0	0	0	54,404.1	47.1	**3**
	EhPlacer	FastRoute	DRCU	24,129.9	3,364.2	7,571.4	2,124,913	184.7	13,664	649.5	38,162,386,680	30,369	77,606	0	641,775.1	1635.4	9
			TritonRoute	Time Out 72h		-	-	-	-	-	-	-	-	-	-	-	-
ispd18_test8	Contest	CUGR	DRCU	13,099.3	2,345.9	26.2	11,451	5.4	0	23.7	5,652,108	164	144	0	37,939.3	1.4	5
			TritonRoute	**13,032.5**	**2,146.5**	57.6	18,251	8.0	28,508	39.1	0	0	0	0	37,424.3	0	**1**
		FastRoute	DRCU	13,483.2	2,470.8	4,469.6	1,503,164	39.0	25,424	259.6	1,690,035,584	38,876	17,592	0	113,281.4	202.7	8
			TritonRoute	13,221.8	2,426.7	2,959.4	1,531,564	9.2	29,140	102.5	0	0	1	0	54,801.6	46.4	**3**
	DreamPlace	CUGR	DRCU	13,237.1	2,285.5	76.9	122,201	6.4	0	31.3	491,893,448	2,386	807	0	46,088.1	23.2	6
			TritonRoute	13,137.3	2,263.6	186.7	200,180	8.9	28,595	38.9	0	0	0	0	38,749.7	3.5	**2**
		FastRoute	DRCU	13,662.7	2,485.0	4,348.7	1,474,868	35.9	22,029	250.2	1,606,529,252	33,596	16,048	0	108,611.4	190.2	7
			TritonRoute	13,404.1	2,428.7	2,888.2	1,487,491	9.1	29,080	103.1	0	0	0	0	54,863.6	46.6	**4**
	EhPlacer	FastRoute	DRCU	Out of Memory		-	-	-	-	-	-	-	-	-	-	-	-
			TritonRoute	TritonRoute Fail		-	-	-	-	-	-	-	-	-	-	-	-
ispd18_test9	Contest	CUGR	DRCU	10,872.6	2,341.2	22.0	10,455	4.4	0	22.1	154,000	204	117	0	32,268.5	1.3	6
			TritonRoute	**10,808.3**	**2,149.7**	57.8	17,983	7.5	28,478	37.7	0	0	0	0	31,863.2	0	**1**
		FastRoute	DRCU	11,229.9	2,394.8	7,242.1	1,854,366	59.3	34,432	211.9	1,928,646,624	53,556	20,101	0	133,108.1	317.7	10
			TritonRoute	11,092.0	2,404.9	5,805.8	1,913,859	9.1	28,970	102.3	0	0	0	0	64,046.1	101	**3**
	DreamPlace	CUGR	DRCU	10,980.6	2,289.5	37.5	46,220	5.1	0	28.2	447,390,300	1,882	543	0	39,223.0	23.1	7
			TritonRoute	10,884.2	2,264.8	161.4	126,621	8.0	28,862	37.6	0	0	0	0	32,910.9	3.3	**2**
		FastRoute	DRCU	11,329.3	2,391.3	7,238.4	1,847,634	61.2	33,480	199.0	1,908,258,040	50,640	19,312	0	131,155.9	311.6	9
			TritonRoute	11,217.9	2,404.1	5,864.3	1,909,683	9.0	28,991	102.6	0	0	0	0	64,648.7	102.9	**4**
	EhPlacer	FastRoute	DRCU	12,282.4	2,426.9	7,718.8	1,862,499	52.9	29,320	200.5	1,873,411,300	42,861	17,660	0	130,858.6	310.7	8
			TritonRoute	12,188.4	2,482.8	6,330.6	1,960,986	9.8	29,034	106.9	0	0	0	0	69,638.3	118.6	**5**
ispd18_test10	Contest	CUGR	DRCU	13,623.6	2,496.3	137.5	27,585	6.2	0	29.0	13,705,600	230	669	0	40,548.3	2.3	2
			TritonRoute	**13,559.7**	**2,309.9**	145.7	52,557	12.9	32,519	52.4	0	0	0	0	39,626.5	0	**1**
		FastRoute	DRCU	Out of Memory		-	-	-	-	-	-	-	-	-	-	-	-
			TritonRoute	Time Out 72h		-	-	-	-	-	-	-	-	-	-	-	-
	DreamPlace	CUGR	DRCU	16,187.5	2,495.6	331.9	102,678	8.2	427	42.4	532,716,036	1,936	3,097	0	56,630.4	42.9	3
			TritonRoute	TritonRoute Fail		-	-	-	-	-	-	-	-	-	-	-	-
		FastRoute	DRCU	Time Out 72h		-	-	-	-	-	-	-	-	-	-	-	-
			TritonRoute	Time Out 72h		-	-	-	-	-	-	-	-	-	-	-	-
	EhPlacer	FastRoute	DRCU	Out of Memory		-	-	-	-	-	-	-	-	-	-	-	-
			TritonRoute	Time Out 72h		-	-	-	-	-	-	-	-	-	-	-	-

Table 7 – Experimental results for ISPD 2019 Benchmarks.

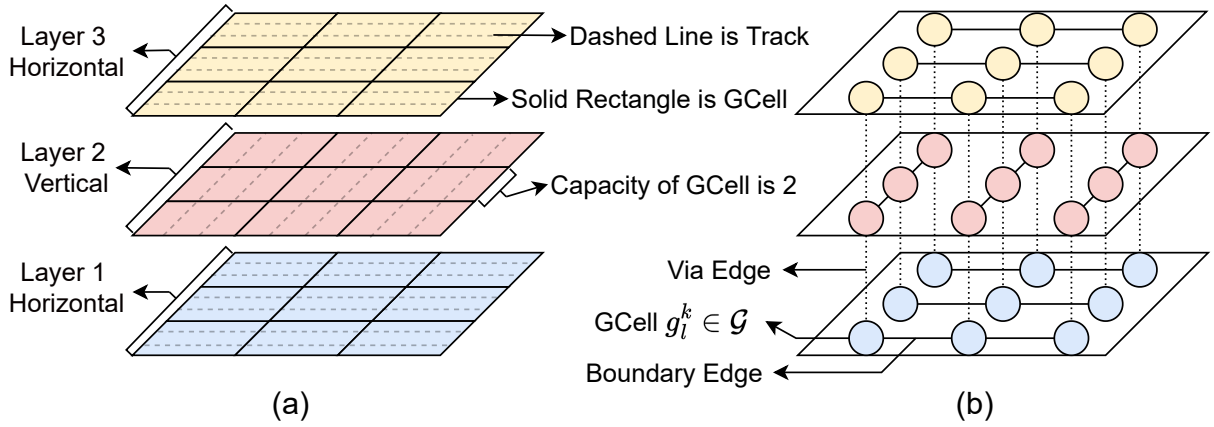
Circuit	Flow			Routing Metrics		Routing Preference Metrics					Design Rule Violations				Score		Rank
	Placement	Global Routing	Detailed Routing	Wirelength (mm)	Vias (K)	OFGW (mm)	OFGV	OFTW (mm)	OFTV	WWW (mm)	Shorts	Min Area	Spacing	Open Nets	(K)	(%)	
ispd19_test1	Contest	CUGR	DRCU	128.5	38.4	0.5	220	0.1	731	1.7	100,000	30	13	0	433.4	4.3	5
			TritonRoute	**126.0**	38.0	1.9	524	0.4	3,965	1.8	0	0	0	0	415.4	0	**1**
		FastRoute	DRCU	130.8	38.8	77.2	29,056	0.5	1,274	3.2	25,746,516	1,377	779	0	2,238.0	438.8	7
			TritonRoute	128.4	38.8	75.8	28,953	0.4	4,064	2.4	0	0	0	0	823.9	98.4	**3**
	DreamPlace	CUGR	DRCU	134.9	**37.9**	2.1	1,882	0.2	718	2.2	6,710,756	31	85	0	579.2	39.4	6
			TritonRoute	132.1	38.7	5.0	3,396	0.4	3,990	2.0	0	0	0	0	450.9	8.5	**2**
		FastRoute	DRCU	137.1	39.7	80.5	30,076	0.6	1,359	3.2	29,962,292	1,543	907	0	2,473.4	495.5	9
			TritonRoute	135.0	39.7	79.6	29,937	0.4	4,080	2.5	0	0	0	0	862.6	107.7	**4**
	EhPlacer	FastRoute	DRCU	163.2	41.1	96.0	30,938	0.4	1,182	3.3	33,117,896	1,229	810	0	2,453.1	490.6	8
			TritonRoute	161.1	41.6	93.5	31,272	0.5	4,109	2.6	0	0	0	1	-	-	-
ispd19_test2	Contest	CUGR	DRCU	4,979.7	842.3	17.7	6,372	4.6	23,390	28.2	5,145,400	1,190	2,336	0	16,232.1	11	3
			TritonRoute	**4,939.9**	798.2	65.2	19,160	14.3	132,593	32.0	0	0	0	0	14,619.6	0	**1**
		FastRoute	DRCU	5,132.3	936.4	1,078.0	481,483	7.0	26,866	72.1	804,037,992	12,443	15,185	0	44,844.4	206.7	6
			TritonRoute	TritonRoute Fail		-	-	-	-	-	-	-	-	-	-	-	-
	DreamPlace	CUGR	DRCU	5,059.5	801.0	77.3	52,073	4.6	23,458	52.4	238,509,864	1,651	3,417	0	20,501.6	40.2	4
			TritonRoute	4,992.0	**790.9**	137.2	90,630	14.9	133,119	37.3	0	0	3	0	15,196.9	3.9	**2**
		FastRoute	DRCU	5,194.1	936.1	1,059.1	487,862	6.9	27,084	72.6	799,785,424	13,632	17,460	0	46,591.6	218.7	7
			TritonRoute	TritonRoute Fail		-	-	-	-	-	-	-	-	-	-	-	-
	EhPlacer	FastRoute	DRCU	5,057.3	913.8	1,027.6	465,330	6.1	26,043	73.2	709,309,396	10,919	14,970	0	42,292.2	189.3	5
			TritonRoute	TritonRoute Fail		-	-	-	-	-	-	-	-	-	-	-	-
ispd19_test3	Contest	CUGR	DRCU	166.7	66.4	1.0	478	0.3	664	2.5	560,000	67	270	0	744.6	26.4	2
			TritonRoute	**164.3**	63.4	5.3	2,056	0.5	6,083	3.2	0	0	0	0	589.0	0	**1**
		FastRoute	DRCU	173.8	65.2	41.8	29,558	0.9	1,436	8.8	75,885,988	2,755	2,014	0	4,184.1	610.4	6
			TritonRoute	166.6	63.2	42.1	30,890	0.6	6,467	5.0	0	0	0	5	-	-	-
	DreamPlace	CUGR	DRCU	173.8	**60.9**	5.2	4,167	0.4	703	5.2	4,803,524	85	319	0	876.5	48.8	4
			TritonRoute	170.6	62.8	13.3	9,833	0.6	6,338	4.4	8,016,512	0	13	0	765.2	29.9	3
		FastRoute	DRCU	181.4	65.3	44.6	29,828	0.8	1,481	8.8	75,685,460	2,736	1,967	0	4,182.1	610	5
			TritonRoute	173.6	64.0	44.4	31,253	0.6	6,417	5.0	7,962,596	0	16	7	-	-	-
	EhPlacer	FastRoute	DRCU	FastRoute Fail		-	-	-	-	-	-	-	-	-	-	-	-
			TritonRoute	FastRoute Fail		-	-	-	-	-	-	-	-	-	-	-	-
ispd19_test4	Contest	CUGR	DRCU	5,928.7	**907.4**	77.8	24,813	2.4	2	27.1	35,990,788	142	466	0	17,945.6	0	1
			TritonRoute	5,428.1	1,050.3	899.5	286,445	31.1	3,874	259.2	35,409,532,306	0	90,561	1	-	-	**-**
		FastRoute	DRCU	FastRoute Fail		-	-	-	-	-	-	-	-	-	-	-	-
			TritonRoute	FastRoute Fail		-	-	-	-	-	-	-	-	-	-	-	-
	DreamPlace	CUGR	DRCU	**5,334.3**	926.1	725.4	175,288	3.5	7	131.2	26,140,503,316	1,764	14,953	0	354,769.4	1876.9	2
			TritonRoute	TritonRoute Fail		-	-	-	-	-	-	-	-	-	-	-	-
		FastRoute	DRCU	FastRoute Fail		-	-	-	-	-	-	-	-	-	-	-	-
			TritonRoute	FastRoute Fail		-	-	-	-	-	-	-	-	-	-	-	-
	EhPlacer	FastRoute	DRCU	FastRoute Fail		-	-	-	-	-	-	-	-	-	-	-	-
			TritonRoute	FastRoute Fail		-	-	-	-	-	-	-	-	-	-	-	-
ispd19_test5	Contest	CUGR	DRCU	967.7	137.9	1.8	1,040	0.4	21	3.7	2,949,569	6	163	0	2,845.7	0	1
			TritonRoute	**926.5**	150.9	33.3	10,184	1.3	155	15.7	2,804,046,064	0	31	0	37,942.7	1233.3	5
		FastRoute	DRCU	976.0	150.9	222.7	84,325	1.1	0	15.9	192,367,300	4,071	8,101	0	12,512.3	339.7	2
			TritonRoute	TritonRoute Fail		-	-	-	-	-	-	-	-	-	-	-	-
	DreamPlace	CUGR	DRCU	937.4	**136.9**	16.3	5,537	0.5	25	6.5	2,393,896,156	27	1,161	0	33,256.4	1068.7	4
			TritonRoute	930.0	158.0	76.1	25,044	1.6	169	19.1	2,890,065,340	0	1,287	0	39,915.3	1302.7	6
		FastRoute	DRCU	984.1	152.5	222.6	86,056	1.1	4	16.0	213,965,102	3,586	9,820	0	13,425.1	371.8	3
			TritonRoute	TritonRoute Fail		-	-	-	-	-	-	-	-	-	-	-	-
	EhPlacer	FastRoute	DRCU	EhPlacer Fail		-	-	-	-	-	-	-	-	-	-	-	-
			TritonRoute	EhPlacer Fail		-	-	-	-	-	-	-	-	-	-	-	-
ispd19_test6	Contest	CUGR	DRCU	13,241.1	2,086.8	38.8	14,132	5.3	12,206	53.9	7,100,800	1,097	1,541	0	39,187.0	4.3	3
			TritonRoute	13,116.8	1,972.3	92.6	24,966	12.2	46,333	57.3	0	0	0	0	37,587.8	0	**1**
		FastRoute	DRCU	13,572.1	2,344.3	2,916.7	1,236,163	21.8	27,598	180.6	1,883,483,896	25,974	28,825	0	106,366.7	183	5
			TritonRoute	13,347.6	2,220.2	2,064.6	1,180,111	12.8	57,808	101.3	0	3	15	3	-	-	-
	DreamPlace	CUGR	DRCU	13,305.4	1,967.4	159.8	123,064	5.6	12,250	112.1	558,908,764	3,442	3,066	0	48,947.9	30.2	4
			TritonRoute	13,136.4	**1,947.6**	210.1	176,021	12.2	54,757	68.9	0	6	16	0	38,403.2	2.2	**2**
		FastRoute	DRCU	13,617.0	2,332.1	2,770.6	1,239,857	27.9	33,548	176.0	2,056,446,124	35,526	40,227	0	118,364.8	214.9	7
			TritonRoute	13,404.3	2,197.0	2,081.6	1,165,393	12.8	58,685	103.2	0	3	6	7	-	-	-
	EhPlacer	FastRoute	DRCU	12,985.0	2,301.3	2,823.6	1,224,941	21.6	29,265	178.0	1,832,001,176	29,731	37,135	0	109,714.1	191.9	6
			TritonRoute	**12,786.0**	2,197.4	2,150.7	1,178,931	12.9	59,735	106.0	0	0	6	3	-	-	-
ispd19_test7	Contest	CUGR	DRCU	24,403.2	4,069.6	120.4	40,845	8.5	24,497	145.5	64,847,568	4,392	10,615	0	78,876.9	11.6	3
			TritonRoute	**24,198.2**	3,789.3	264.6	88,661	34.3	255,196	169.9	0	0	0	0	70,676.3	0	**1**
		FastRoute	DRCU	26,656.8	3,882.4	5,059.1	1,672,731	29.1	43,217	1,484.9	38,070,911,988	77,462	375,867	0	811,466.6	1048.1	6
			TritonRoute	TritonRoute Fail		-	-	-	-	-	-	-	-	-	-	-	-
	DreamPlace	CUGR	DRCU	24,929.0	3,737.7	720.8	331,608	9.5	24,466	437.3	1,325,446,516	9,983	19,816	0	107,435.9	52	4
			TritonRoute	24,380.9	**3,726.0**	707.9	554,977	40.3	264,367	231.0	0	0	10	0	74,023.8	4.7	**2**
		FastRoute	DRCU	26,861.0	3,877.7	5,107.7	1,670,887	29.1	43,859	1,471.5	37,245,083,408	79,271	386,301	0	807,940.9	1043.2	5
			TritonRoute	TritonRoute Fail		-	-	-	-	-	-	-	-	-	-	-	-
	EhPlacer	FastRoute	DRCU	49,293.4	4,773.7	8,797.0	2,124,200	33.7	39,101	2,978.5	203,966,053,804	69,234	996,638	0	3,276,417.3	4535.8	7
			TritonRoute	TritonRoute Fail		-	-	-	-	-	-	-	-	-	-	-	-
ispd19_test8	Contest	CUGR	DRCU	37,364.4	6,450.7	112.8	52,544	12.8	36,469	135.0	31,813,600	6,182	6,914	0	114,617.6	6.5	3
			TritonRoute	**37,073.1**	**6,176.4**	225.6	84,142	44.8	382,323	174.3	0	0	0	0	107,613.7	0	**1**
		FastRoute	DRCU	42,984.6	6,738.8	9,992.3	3,378,498	46.7	64,644	2,459.6	188,749,368,432	137,735	981,980	0	3,105,982.6	2786.2	6
			TritonRoute	37,965.3	6,907.2	7,768.3	3,976,150	56.3	424,486	347.5	0	0	9	7	-	-	-
	DreamPlace	CUGR	DRCU	37,794.7	6,225.4	307.4	280,245	13.7	36,473	289.4	2,273,847,480	17,616	17,092	0	156,049.5	45	4
			TritonRoute	37,377.2	6,241.9	590.5	547,359	47.0	403,575	203.4	0	0	14	0	110,971.9	3.1	**2**
		FastRoute	DRCU	43,186.1	6,501.1	10,113.9	3,256,179	46.0	65,140	2,353.6	181,849,673,384	144,279	1,039,681	0	3,051,842.4	2735.9	5
			TritonRoute	TritonRoute Fail		-	-	-	-	-	-	-	-	-	-	-	-
	EhPlacer	FastRoute	DRCU	44,891.4	6,451.2	10,246.9	3,149,428	41.0	58,378	2,581.5	225,566,244,400	126,959	1,136,688	0	3,643,984.9	3286.2	7
			TritonRoute	39,899.5	7,100.6	8,590.8	4,157,514	61.7	440,137	376.4	0	4	3	9	-	-	-
ispd19_test9	Contest	CUGR	DRCU	56,504.4	10,745.0	169.1	86,994	21.3	60,771	228.3	70,983,984	10,527	12,946	0	177,563.1	7.7	3
			TritonRoute	**56,021.2**	**10,265.1**	376.2	141,606	77.4	637,914	295.0	0	3	2	0	164,914.6	0	**1**
		FastRoute	DRCU	Out of Memory		-	-	-	-	-	-	-	-	-	-	-	-
			TritonRoute	TritonRoute Fail		-	-	-	-	-	-	-	-	-	-	-	-
	DreamPlace	CUGR	DRCU	57,251.9	10,392.7	477.8	452,374	22.6	60,722	492.3	4,037,655,500	30,912	32,847	0	251,685.5	52.6	4
			TritonRoute	56,560.7	10,412.6	991.6	914,673	80.7	671,099	337.5	0	0	24	0	170,672.4	3.5	**2**
		FastRoute	DRCU	Out of Memory		-	-	-	-	-	-	-	-	-	-	-	-
			TritonRoute	TritonRoute Fail		-	-	-	-	-	-	-	-	-	-	-	-
	EhPlacer	FastRoute	DRCU	Out of Memory		-	-	-	-	-	-	-	-	-	-	-	-
			TritonRoute	59,937.4	11,569.4	13,728.6	6,719,820	101.0	730,078	603.6	0	0	25	16	-	-	-
ispd19_test10	Contest	CUGR	DRCU	55,827.8	10,337.6	167.5	62,283	21.1	61,013	325.7	110,579,000	10,726	13,665	0	176,464.0	8.1	3
			TritonRoute	**55,324.7**	9,574.1	517.7	156,548	85.1	637,558	423.5	1,000	0	15	0	163,180.2	0	2
		FastRoute	DRCU	61,709.3	9,380.7	9,704.5	3,471,939	81.8	102,649	3,366.4	118,752,103,024	191,805	1,551,339	0	2,598,141.5	1492.2	5
			TritonRoute	TritonRoute Fail		-	-	-	-	-	-	-	-	-	-	-	-
	DreamPlace	CUGR	DRCU	57,261.1	9,436.4	1,236.7	601,340	23.6	61,034	956.5	2,476,987,464	28,511	45,958	0	241,910.1	48.2	4
			TritonRoute	56,003.6	9,427.7	1,446.3	1,197,385	96.9	663,260	572.3	0	0	66	0	171,092.9	4.8	**1**
		FastRoute	DRCU	62,410.5	**9,283.5**	9,855.2	3,463,843	82.3	104,980	3,344.5	121,739,103,340	205,335	1,596,340	0	2,666,942.6	1534.4	6
			TritonRoute	TritonRoute Fail		-	-	-	-	-	-	-	-	-	-	-	-
	EhPlacer	FastRoute	DRCU	Out of Memory		-	-	-	-	-	-	-	-	-	-	-	-
			TritonRoute	TritonRoute Fail		-	-	-	-	-	-	-	-	-	-	-	-

4 FORMULATION OF THE GLOBAL ROUTING PROBLEM

This chapter presents a mathematical formulation for the routing with cell movement problem. Given an initial solution where cells have been placed, and the global routing is done, the routing with cell movement problem consists in moving a set of cells to minimize some routing metrics, such as the number of VIAs and/or wirelength, by finding the regions that connect all net pins. In the global routing problem, the circuit area is partitioned into regions called GCells ($\mathcal{G}$), where the 3D routing space can be modeled as a 3D graph of GCells. In this formulation, the length of a net is measured based on the number of GCells it spans. A cell movement consists of reassigning a cell's location from one GCell to another. The features of each cell, including pin location and blockages, are defined in the standard cell library.

Figure 6 shows a representation of the 3D routing space on the left side (a) and the respective graph model on the right side (b). Figure 6.a depicts how each metal layer (k) could be subdivided into regions called GCells ($g_i^k \in \mathcal{G}$). The GCells in Figure 6.a are demarcated with the solid rectangles. In the graph, each GCell is represented by a node. The dashed lines in Figure 6.a represent the tracks for each metal layer. The number of tracks crossing each GCell defines the capacity of each GCell. In this example, the capacity of each GCell is 2, and this value is associated with each graph node shown in Figure 6.b. An edge in the graph represents the neighborhood of two GCells (Figure 6.b). This neighborhood could be in the same metal layer (solid lines) or between lower/upper adjacent metal layers (dashed lines). Note that, for each metal layer, only the neighborhood in the preferred routing direction is connected by edges. For example, only horizontal connections are made within layers 1 and 3.

Figure 6 – 3D routing space (a) and the respective graph model (b).

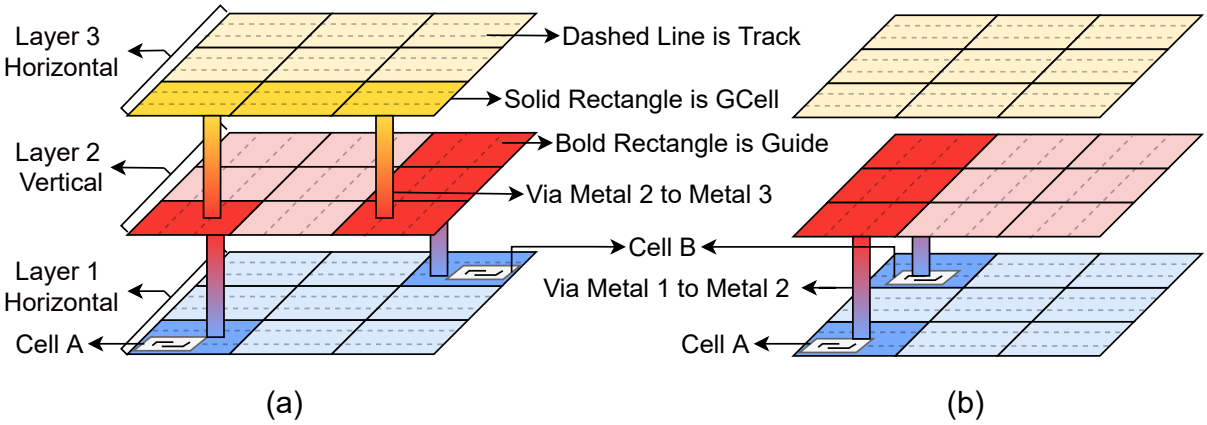


Source: the author.

Figure 7 illustrates an example of routing optimization through cell movement for a 2-pin net. Figure 7.a presents an initial solution for a single two-pin net. Assuming

that for this net, the minimum routing layer is metal 2[1], the initial routing solution is composed of 4 VIAs and 9 GCells. Figure 7.b presents the same net after moving Cell B from the top right to the top left GCell in Layer 1, thus making it possible to optimize the routing. This final routing solution is composed of 2 VIAs and 5 GCells. This simple example shows that moving cells during the routing steps may improve the solution without compromising the design constraints.

Figure 7 – Example of routing optimization through cell movement.



Source: the author.

It is necessary to keep track of placement constraints during the routing to ensure that the placement will be legalized at the end of the routing. Therefore, the routing with cell movement problem can be defined as follows:

Input: A netlist with legalized placement and Global Routing solution.
Problem: Move a set of cells to optimize the number of VIAs and wirelength, keeping a legalized placement and all nets connected.

The objective of this problem could be expressed by Equation 4.1, while Equations 4.2 to 4.7 are the placement constraints, and Equations 4.8 to 4.11 are the routing constraints.

[1] The minimum routing layer constraint, for a specific net, establishes the layer that the routing must be on or above.

$$\min \quad \sum_{n_i \in \mathcal{N}} \mid \mathcal{V}(n_i) \mid + \mid \mathcal{G}(n_i) \mid \tag{4.1}$$

$$\text{s.t.:} \quad X_{left} \leq x(c_i) \leq X_{right} - w(c_i), \quad \forall\, c_i \in \mathcal{C} \tag{4.2}$$

$$Y_{bottom} \leq y(c_i) \leq Y_{top} - h(c_i), \quad \forall\, c_i \in \mathcal{C} \tag{4.3}$$

$$x(c_i) = n \times W_{site}, \quad n \in \mathbb{N} \wedge \forall\, c_i \in \mathcal{C} \tag{4.4}$$

$$y(c_i) = m \times H_{row}, \quad m \in \mathbb{N} \wedge \forall\, c_i \in \mathcal{C} \tag{4.5}$$

$$(x(c_i) + w(c_i) < x(c_j)) \vee (x(c_j) + w(c_j) < x(c_i)), \quad \forall\, c_i, c_j \in \mathcal{C} \ \wedge\ y(c_i) = y(c_j) \ \wedge\ \forall\, i \neq j \tag{4.6}$$

$$(y(c_i) + h(c_i) < y(c_j)) \vee (y(c_j) + h(c_j) < y(c_i)), \quad \forall\, c_i, c_j \in \mathcal{C} \ \wedge\ x(c_i) = x(c_j) \ \wedge\ \forall\, i \neq j \tag{4.7}$$

$$\exists\, \mathcal{G}(n_i) \subset \mathcal{G}\ , \quad \forall\, n_i \in \mathcal{N}(c_i) \tag{4.8}$$

$$\exists g_j \in \mathcal{G}(n_i) \mid\ min(g_j) \leq loc(p_i) \leq max(g_j), \quad \forall\, p_i \in \mathcal{P}(n_i) \wedge n_i \in \mathcal{N}(c_i) \tag{4.9}$$

$$\exists\ g_n \in \mathcal{G}(n_i)\ \mid\ \Delta(g_m, g_n) = 1\ , \quad \forall\, g_n, g_m \in \mathcal{G}(n_i) \wedge n_i \in \mathcal{N}(c_i) \tag{4.10}$$

$$D(g_l^k) \leq S(g_l^k)\ , \forall\, g_j^k \in \mathcal{G}(n_i) \wedge n_i \in \mathcal{N}(c_i) \tag{4.11}$$

Equation 4.1 states that the objective of the problem is to minimize, for all nets, the total number of VIAs ($\mid \mathcal{V} \mid$) and the total wirelength ($\mid \mathcal{G} \mid$). As mentioned before, the length of a net is measured based on the number of GCells it spans. Equation 4.2 and 4.3 ensure that each Cell $c_i \in \mathcal{C}$ is placed within the circuit rows region while Equation 4.4 and 4.5 state that each Cell c_i is aligned with circuit sites and rows, respectively. The non-overlap for each pair of cells $c_i, c_j \in \mathcal{C}$ is guaranteed by Equation 4.6 if c_i and c_j are placed in the same row or by Equation 4.7 if c_i and c_j are placed in different rows.

Equation 4.8 guarantees that for every net n_i connected to Cell c_i, there is a subset of GCells allowing n_i to be fully routed, and thus, no open nets can exist. Equation 4.9 establishes that all pins p_i of a given net n_i can be assigned to a GCell g_j. Equation 4.10 states that for every GCell g_n belonging to $\mathcal{G}(n_i)$ there will be an adjacent GCell g_m. Equation 4.11 ensures that the demand for each GCell g_l located in layer k will not exceed the maximum capacity of that GCell. This takes into account the overflow concerns of each GCell.

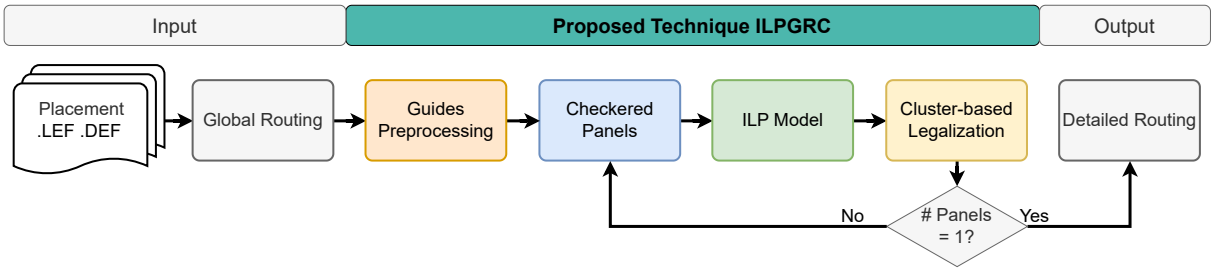
With the aforementioned constraints, it is possible to ensure that all the cells are legalized and the circuit is fully routed. Therefore, no open nets are allowed.

5 PROPOSED ILP-BASED TECHNIQUE

This chapter presents the proposed technique, named ILPGRC, for the target problem introduced in the previous section. ILPGRC comprises of four main steps: guides preprocessing, Checkered Panels construction, ILP model construction and solving, and cluster-based panel legalization. Each of these steps will be discussed in detail in subsections 5.1 to 5.4. It is important to highlight that, despite the other contributions, the most relevant contribution of this work is the ILP model that simultaneously moves cells and re-routes all nets, presented in Section 5.3, due to its elegance.

The overall flow of ILPGRC is presented in Figure 8 and in Algorithm 1 as well. The inputs to the algorithm are the technology file (.lef), the design file (.def), and the initial Global Routing solution file (.guide) of the given layout. Library Exchange Format (LEF) and Design Exchange Format (DEF) are two industrial files to describe a design. LEF file defines the elements of an IC process technology and the associated library of cell models. DEF file defines the elements of an IC design relevant to physical layout, including the netlist and design constraints.

Figure 8 – Flow of the proposed technique, ILPGRC.



Source: The author.

After loading the files that describe the circuit, the initial global routing solution is pre-processed in line 2 of Algorithm 1. More details of this preprocessing step will be presented in Section 5.1. Next, in line 3, the Checkered Panels are created, and the number of levels is stored in variable *nlev*. The motivation and details of this partitioning technique are given in Section 5.2. Each sub-part of the problem is called a panel, and each panel is associated with a level and one color.

Afterward, all panels with the same *level* and *color* are processed in parallel (lines 9 to 15). For each of these panels, the proposed ILP model for moving cells and rerouting nets is created and solved in line 10. Subsection 5.3 presents the details about this step. Then, in line 11, the panel is legalized using a version of the Abacus algorithm (Spindler; Schlichtmann; Johannes, 2008). The details of the panel legalization process are presented in Section 5.4. Later, if the panel is considered legalized, all the movements and new routing solutions are stored in variables $\mathcal{M}$ and $\mathcal{R}$ (line 13). Once all panels of the same *level* and *color* were processed, all movements and routing solutions are applied to the

Algorithm 1: RUN_ILPGRC(lef, def, guide)

```
   Input  : lef = Technology file,
            def = Design placement,
            guide = Global Routing solution
   Output: def = Design with new placement,
            guide = New Global Routing solution
 1 load_circuit(lef, def, guide);
 2 GUIDE_PREPROCESSING() ;                                  // Section 5.1
 3 nlev ← CHECKERED_PANELING() ;                            // Section 5.2
 4 foreach level ∈ nlev do
 5 |   M ← {} ;                                    // Map of movements
 6 |   R ← {} ;                                  // Map of routing guides
 7 |   foreach color ∈ {black, white} do
 8 |   |   panels ← get_panels(color, level);
   |   |   // run parallel
 9 |   |   foreach panel ∈ panels do
10 |   |   |   RUN_ILP(panel) ;                           // Section 5.3
11 |   |   |   l ← LEGALIZE(panel) ;                      // Section 5.4
12 |   |   |   if l = TRUE then
13 |   |   |   |   save_movements_and_routing(M, R);
14 |   |   |   end
15 |   |   end
16 |   end
17 |   update_database(M, R);
18 end
19 write_output_def();
20 write_output_guide();
```

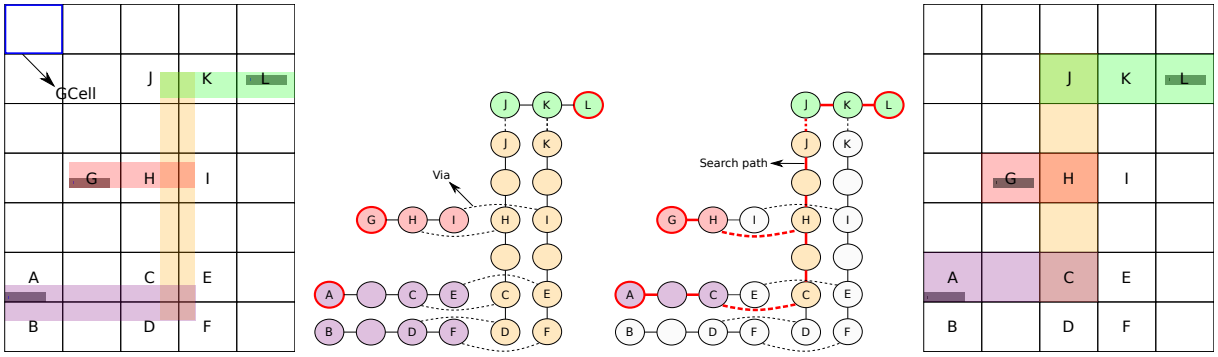
database (line 17). In the end, the new DEF and Guide files are generated as output of the algorithm (lines 19 and 20).

5.1 GUIDES PREPROCESSING

As already stated, the first step of ILPGRC consists in preprocessing the input global routing solution, so as to map the original guide rectangles to the adopted GCell structure. Figure 9.a presents an example of an original routing guide for a three-pin net: white squares represent GCells, gray rectangles represent cells, and the colored rectangles represent the guide(s). In this example, cells are placed in GCells A, G, and L. It also puts in evidence that the original guides may be entirely within a single row/column of GCells (red and green rectangles), or may span multiple rows/columns of GCells (purple and yellow rectangles). The latter case could cause cycles in the GCell graph, degrading or making unfeasible the solution of a path search algorithm. Therefore, mapping the original guides that span more than one row/column of GCells is crucial since it avoids those cycles.

The guide preprocessing step begins by mapping the original guide to a graph (Figure 9.b) where the nodes represent GCells intersected by a guide in a given metal layer, and the edges represent the neighborhood of those GCells. GCell neighborhood may be within the same metal layer (solid edges) and, thus, must respect the preferred direction in that metal layer, or may be the overlapping between GCells in two different layers (dashed edges), therefore representing a VIA. The graph nodes representing the GCells that contain the net cells are marked as terminal nodes and appear outlined in red in Figure 9.b. Once the graph has been built, the preprocessing step performs a Breadth-First Search (BFS), starting from any terminal node and progressing toward all other terminal nodes. Such a procedure determines the minimum path needed to keep this net connected. Figure 9.c shows the result for this BFS search on the example considered, where the minimum path is identified by the sequence of the colored nodes. Hence, in this example, the final guide structure has 13 GCells and 3 VIAs, as shown in Figure 9.d.

Figure 9 – Guide preprocessing steps.



(a) Original guide. (b) Guide mapped to a Graph. (c) Path search result. (d) Preprocessed guide.

Source: the author.

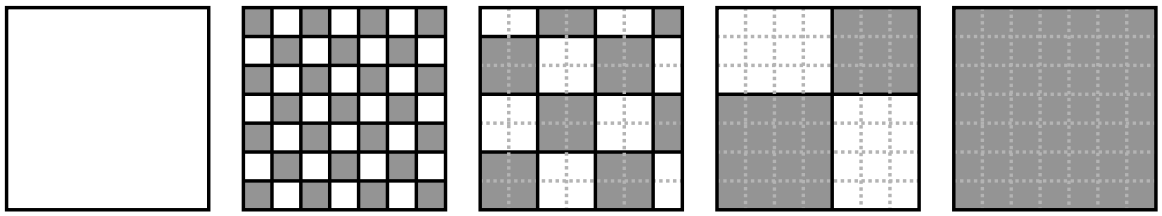
5.2 CHECKERED PANELS

Global Routing using math solvers can be time-consuming and very slow, especially for circuits with a large number of nets and/or containing nets that cross the whole circuit area. This may be worsened when routing is combined with cell movements because each cell movement generates a set of routing candidates, and therefore, the number of variables inside the ILP model is potentially huge. To circumvent these problems, the so-called Checkered Paneling strategy is proposed. Basically, this strategy partitions the circuit area and, for each partition (panel), identifies all nets that are entirely inside the partition. Then, the routing with cell movement problem is solved separately for each panel considering only the identified nets. The partitioning procedure is repeated, assuming progressively larger partitions, so as to take into account the nets that span wider regions. In summary, the Checkered Paneling strategy divides the problem into a

number of sub-problems of smaller sizes that can be solved in parallel using an ILP solver in acceptable runtimes. As an extra benefit, the Checkered Paneling helps balance the workload between the threads since the nets that are entirely within the panels of a given level present similar wirelength.

Figure 10 presents how the Checkered Paneling procedure hierarchically partitions the circuit layout into panels. Two input variables define the panel width (gcw) and height (gch), expressed in the number of GCells, in the first level. The panel size in every subsequent level is defined as being twice the panel size in the preceding level. The levels end when the whole layout is covered by a single panel (Figure 10.e). Within a given level, each panel is associated with a color (either black or white). Panels belonging to the same level and color will be executed in parallel. It is important to note that a panel will not have right or left neighbors of the same color. Such characteristic is to ensure that a cell that crosses the panel edge will not be mapped simultaneously to two threads at the same time. Consequently, ILPGRC is deterministic between sequential and parallel executions, i.e., sequential and parallel executions will produce the same result.

Figure 10 – Example of Checkered Panels for a given layout.



(a) Layout (b) Level 1 (c) Level 2 (d) Level 3 (e) Level 4

Source: the author.

Algorithm 2 presents a pseudo-code for creating the Checkered Panels and making the association between panels and nets. The algorithm receives two variables as input: the panel width (gcw) and height (gch), expressed in number of GCells. The output is the number of levels. First, in lines 1 and 2, the set of processed nets is initialized, and the maximum number of GCells in both dimensions of the circuit layout is calculated. Then, the number of levels is calculated in line 3. As the panel size in a given level is at most twice the size of the panel in the preceding level, the number of levels will be 1 plus the ceiling of logarithmic in base 2 of the maximum number of GCells. Next, for each level starting in 1, the panel width ($panel\_w$) and height ($panel\_h$) are calculate in lines 5 and 6, respectively. Note that, in level 1, the panel size will equal the input variables gcw and gch. After this, function $make\_regions(panel\_w, panel\_h)$ in line 7 partitions the circuit layout creating all the panels for this specific level. Subsequently, for each panel $p \in panels$, the procedure will identify and associate to the panel all nets that will be processed. To that purpose, the function $search\_inside$ in line 9 returns all the nets whose bounding boxes are inside the panel borders. Function $region(p)$ determines

the borders for a given panel p. This search is implemented using the spatial structure RTree (Manolopoulos et al., 2006). Note that, in line 9, all nets that are already associated with any lower-level panel (nets that are in *proc_nets* set) are removed before making the association with the current panel. Function $set\_nets(p, nets)$ in line 10 generates the mapping between all nets and panel p. After this, *proc_nets* receives the nets for this panel in line 11. Finally, line 14 returns the number of levels.

Algorithm 2: CHECKERED_PANELING(*gcw*, *gch*)

Input : *gcw* = Number of GCells to panel width,
gch = Number of GCells to panel height
Output: Number of levels

1 $proc\_nets \leftarrow \emptyset$; `// Set of processed nets`
2 $gcells \leftarrow \max(num\_gcells\_width, num\_gcells\_height)$;
3 $nlevels \leftarrow \lceil \log_2(gcells) \rceil + 1$;
4 **for** $level \leftarrow 1$ **to** $nlevels$ **do**
5 $\quad panel\_w \leftarrow gcw * 2^{level-1}$;
6 $\quad panel\_h \leftarrow gch * 2^{level-1}$;
7 $\quad panels \leftarrow make\_regions(panel\_w, panel\_h)$;
8 $\quad$ **foreach** $p \in panels$ **do**
9 $\quad\quad nets \leftarrow search\_inside(region(p)) \setminus proc\_nets$;
10 $\quad\quad set\_nets(p, nets)$;
11 $\quad\quad proc\_nets \leftarrow proc\_nets \cup nets$;
12 $\quad$ **end**
13 **end**
14 **return** *nlevels*;

5.3 ILP MODEL

This section presents the ILP model to solve the routing with cell movement problem presented in Chapter 4. For convenience, Table 8 brings all the symbols used in the ILP model. This table also categorizes the symbols between variables and parameters. Variables are all the values the ILP model will resolve, and parameters are the values that the current circuit provides to the ILP model.

Table 8 – Symbols used in the ILP model.

Symbol	**Mean**	**Type**
$r_{i,j}$	Routing candidate	Variable
$m_{i,j}$	Position Candidate	Variable
b_i	Pin blockage	Parameter
$cost_{i,j}$	Cost of n_i using candidate j	Parameter
$S(g_i)$	Supply of GCell g_i	Parameter
via_{ij}	Number of vias of candidate j of net i	Parameter
wl_{ij}	Wirelength of candidate j of net i	Parameter
α_k	Wirelength weight of metal layer k	Parameter
β_k	Via weight of metal layer k	Parameter

To solve this problem, we need to model all nets and circuit resources. For each net n_i, a set of possible routing candidates is generated. Each routing candidate j of net n_i has its respective binary variable r_{ij} to indicate which j, among all candidates of n_i, must be selected. Then, the ILP model aims to select the best set of candidates that optimizes the circuit routing. To do that, the objective function of the ILP formulation minimizes the weighted sum of each variable in Equation 5.1, where $cost_{i,j}$ denotes the cost of n_i when it is routed using candidate j.

$$min \sum_{i=1}^{n} \sum_{j=1}^{m} cost_{i,j} \times r_{ij} \tag{5.1}$$

Equation 5.2 presents the cost $cost_{i,j}$ for a net i and candidate j that is a weighted sum of the wirelength wl_{ij} and the number of VIAs via_{ij} in each metal layer k. It is important to note that, as this model is applied to the global routing step, the wirelength of a net is measured in the number of GCells the net spans. In contrast, the number of VIAs is the number of intersections between segments of the same net that are in adjacent metal layers. α_k and β_k are constant values (weights) for each metal layer k, which is used to penalize the lower metal layers, thus distributing the net segments through the upper layers so as to alleviate the congestion. Short nets are less penalized because the sum of wirelength (wl_{ij}) and the number of VIAs (via_{ij}) are smaller. Therefore, short nets are assigned to the lower layers. On the other hand, for the long nets, the sum of wirelength and VIAs is bigger, and thus, they are assigned to the upper layers. In this work, we use $\beta_k = \omega \times (\alpha_k + \alpha_{k+1})$ where ω represents the VIA factor and is equal to 1

in all experiments. We experimentally tested different values for ω, 0.001, 0.1, 1, 10, 100, and the best average VIA reduction was with $\omega = 1$. All constant values used in this work are presented in Section 6.4 in Table 9.

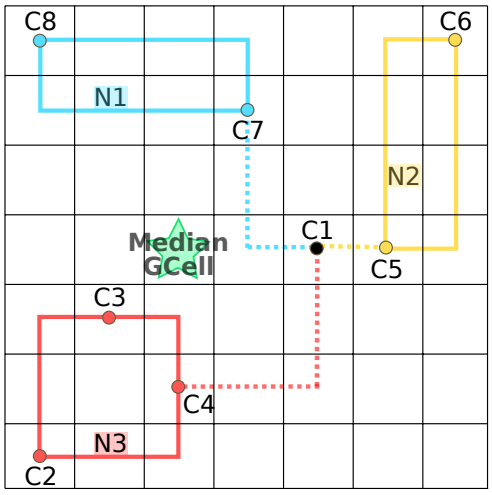
$$cost_{i,j} = \sum_{k=1}^{n} \alpha_k \times wl_{ij}^k + \sum_{k=1}^{n-1} \beta_k \times via_{ij}^k \tag{5.2}$$

Concerning cell movement, for each cell c_i, we generate a set of possible locations $L(c_i) = \{l_1(c_i), l_2(c_i), ..., l_n(c_i)\}$. For each location, there is a binary variable $m_{i,j}$ that indicates if cell c_i should be placed on that location $l_j(c_i)$. In addition, there is a variable, $l_0(c_i)$, used to keep track of each cell's initial location.

When a cell is moved, its nets must be re-routed. Therefore, for each candidate location $m_{i,j}$, a set of nets $N(m_{i,j})$ with their respective routing candidates $R(m_{i,j})$ is defined. In addition, for each net n_i we define the set of cells connected to this net as $C(n_i)$. Given that, the objective function remains the weighted sum of the net candidates, except that now we need to consider the nets resulting from the cell movements.

Figure 11 shows an example of how the candidate locations to move a cell C1, connected to 7 other cells (C2 to C8) through nets N1, N2, and N3, are generated. The bounding box of each net (represented in the figure by the blue, yellow, and red rectangles) is calculated, and the median GCell with respect to all the bounding box coordinates is selected as the candidate location. In the example of Figure 11, the median GCell of C1 is identified by the green star. This strategy was inspired by the work presented in (Pan; Viswanathan; Chu, 2005).

Figure 11 – Illustration showing how the proposed technique determines a candidate location for cell C1 by calculating the Median GCell. Each hue in this image represents the network associated to a cell (except the candidate cell), each cell is represented by colored dots. The green star indicates the median GCell.



Source: the author.

The decision to adopt the median GCell as the target position for cell movement was taken after the evaluation of 9 different movement candidates: the median point

of the connected nets, the four neighbors of the median point, and the four neighbors of the initial location of the cell. According to the conducted experiments, among all candidate movements, moving to the median point of each cell has the highest chance of reducing the wirelength of nets connected to the cell. An experiment with 5 movement candidates for each cell at the same time was also performed: the median GCell, and the four neighboring GCells of the median (N, S, W, E). Such experiment did not help to improve the quality of the results. This is due to the fact that the median GCell is the optimum point for a movement considering all nets connected to a cell. Therefore, the ILP model will prefer to move the cells for this median point instead of the neighboring GCells. By adding 4 times more movement candidates, the runtime increased by only 2.03 times, on average.

Figure 12 presents an example of the routing candidates generation for a two-pin net connecting Cell A (lower left corner) and Cell B (upper right corner) considering three position candidates. This example considers that the design uses only three metal layers for simplicity. For a two-pin net, we generate all possible combinations of L-shape patterns, i.e., the combination of all possible layers for each of the L-patterns. The four routing candidates for the initial placement (Candidate 1: $m_{B,1}$) can be visualized in the first row of Figure 12, two of them using the two possible lower L-shape: $r_{k,1}$ and $r_{k,2}$, and the other two using the two possible upper L-shape: $r_{k,3}$ and $r_{k,4}$. Considering a movement of Cell B to the top left corner (Candidate 2: $m_{B,2}$), the only viable routing candidate is $r_{k,5}$. On the other hand, if Cell B is moved to the lower right corner (Candidate 3: $m_{B,3}$), there are two possible routing candidates: $r_{k,6}$ and $r_{k,7}$ For a multi-pin net, first, the Steiner tree is generated using the Flute algorithm (Chu; Wong, 2007). Then, for each two-point segment, the same approach of two-pin nets described above is used. In the ILP model, it is ensured that for a given net, one and only one routing candidate is selected for each of the two-point segments.

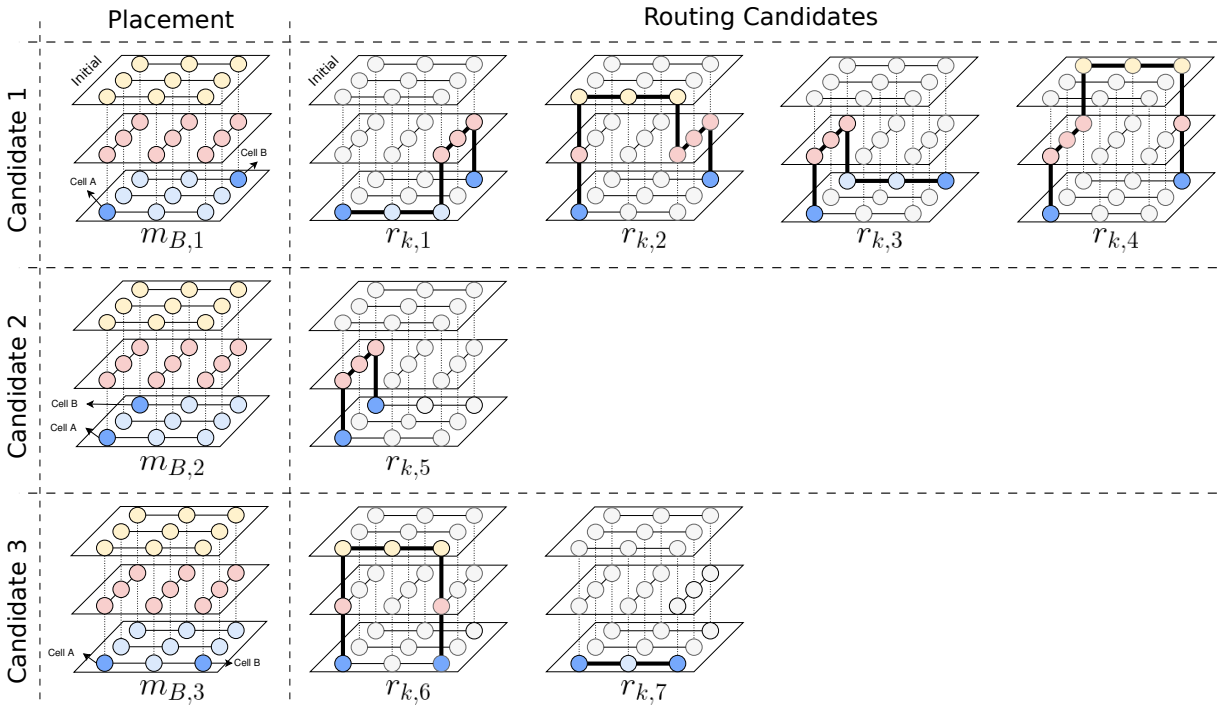
To properly route the circuit, a few constraints must be added to the ILP model. The first one states that each net must be routed by a single routing candidate, which is captured by Equation 5.3.

$$\sum r_{ij} = 1,\ \forall j\ \wedge\ i = 1...n \tag{5.3}$$

The second constraint ensures no overflow in the GCells. Each GCell g_k has a supply $S(g_k)$ representing the number of available routing tracks in the GCell g_k. The number of nets routed through a GCell cannot exceed its supply. We define $R(g_k)$ as the set of net candidates routed through GCell g_k. Then, the second constraint is modeled as in Equation 5.4, which must be defined for each GCell.

$$\sum_{r_{ij} \in R(g_k)} r_{ij} \leq S(g_k), k = 1...K \tag{5.4}$$

Figure 12 – Placement and routing candidates for a 2-pin net connecting Cell A and Cell B.



Source: the author.

Next, a few constraints should be added to ensure those movements do not invalidate circuit routing. The third constraint ensures that each cell is assigned to one candidate location, as expressed in Equation 5.5.

$$\sum m_{i,j} = 1, \ \forall j \ \wedge \ i = 1...n \tag{5.5}$$

The next constraint ties the cell candidate locations to their respective nets. This constraint is specified in Equation 5.6, and must be defined for each candidate location $m_{i,j}$ and for each net in $N(m_{i,j})$ associated with this location. For each of those nets, the sum of their candidate variables r_{kl} must equal the value of the candidate location $m_{i,j}$. As a consequence, if the cell is moved to location $l_j(c_i)$, then $m_{i,j} = 1$ and the net must use one of the candidates associated with $m_{i,j}$. Otherwise, $m_{i,j} = 0$ and the net cannot use any of those candidates. Notice that we do not need to establish this constraint for the initial location $l_0(c_i)$. This is because there are already constraints that ensure that all nets are routed someway, so if the cell is not moved, one of the initial routing candidates will automatically be used.

$$\sum_{l=1}^{m} r_{kl} = m_{i,j}, i = 1...n, j = 1...m, n_k \in N(m_{i,j}) \tag{5.6}$$

The next constraint certifies that we do not move two cells from the same net (Equation 5.7). This constraint must be specified for each net, and it is necessary to ensure we do not assign different routing candidates for the same net. For a given net, this is done by ensuring that the sum of the moved location variables of all cells connected

to that net is one. Notice that this constraint only considers the location variables that do not correspond to the initial location ($i \geq 1$). So, if one cell in the net is moved, all the other ones should remain in their initial locations.

$$\sum_{c_i \in C(n_k)} \sum_{j=1}^{m} m_{i,j} = 1, k = 1...n \tag{5.7}$$

Finally, it is necessary to update the GCell supply constraints to include cell blockages. Some cell pins impose blockages on metal layers, so the ILP formulation must consider this to avoid overflowing the GCells. In order to do that, we defined $Y(g_k)$ as the set of cell locations inside GCell g_k, and each blockage reduces the GCell supply in b_i. Then, Equation 5.8 models this constraint by considering not only the nets inside each GCell, but also the blockages.

$$\sum_{r_{ij} \in R(g_k)} r_{ij} + \sum_{m_{i,j} \in Y(g_k)} b_i \times m_{i,j} \leq S(g_k), k = 1...K \tag{5.8}$$

Algorithm 3 presents a pseudo-code for creating and solving the described ILP model. This Algorithm receives as input a circuit panel. The first two lines will query from the database the nets and cells that are associated with the such panel. Then, function *create_initial_variables*, line 3, produces all the variables representing the initial state of placement and routing for this panel. Therefore, the initial solution is also considered as a valid solution which allows taking into account the situation when the technique is not able to find a better solution for a given net or cell. In other words, this ensures that the ILP model is always feasible. It is interesting to observe that in the worst-case scenario, the initial solution may be kept for the entire panel or event for the entire circuit. The function *create_nets_candidates*(*nets*), in line 4, creates different routing solutions for the initial placement. The proposed technique generates different pattern routing solutions using different metal layers for each net.

Next, the loop between lines 5 and 13 generates different candidate positions and routing solutions for each cell in the current panel. First, the cell original location is stored in *optig_p* in line 6, and its median position is calculated by function median_position(*cell*) in line 7. The ILP variable representing this cell in the median position is generated by function create_variable(*cell*, *median_p*) in line 8. Then, the cell is placed in the median position (line 9), its connected nets are stored in variable *nets_c* in line 10, and the routing candidates for this position are generated by the function *create_nets_candidates*(*nets_c*) in line 11. Finally, the cell is restored to the initial location in line 12.

After creating all these candidates for positions and routing, the constraints presented in Equations 5.3 to 5.8 are added into the ILP model in line 14. Then, the ILP objective, Equation 5.1, is added in line 15. The ILP model is solved using CPlex (IBM,

2018), and the result is stored in variable *result* (line 16). Finally, if the result from ILP is *Optimal*, the movements and routing solutions are applied to the design in line 18.

Algorithm 3: RUN_ILP(panel)

```
   Input  : panel = Panel to create and solve the ILP
 1 nets ← get_nets(panel);
 2 cells ← get_cells(panel);
 3 create_initial_variables(cells, nets);
 4 create_nets_candidates(nets);
 5 foreach cell ∈ cells do
 6 |   orig_p ← position(cell);
 7 |   median_p ← median_position(cell);
 8 |   create_variable(cell, median_p);
 9 |   place(cell, median_p);
10 |   nets_c ← get_nets(cell);
11 |   create_nets_candidates(nets_c);
12 |   place(cell, orig_p);
13 end
14 add_constraints();
15 add_objective();
16 result ← solve_model();
17 if result = Optimal then
18 |   apply_movements_and_routing();
19 end
```

5.4 PANEL LEGALIZATION

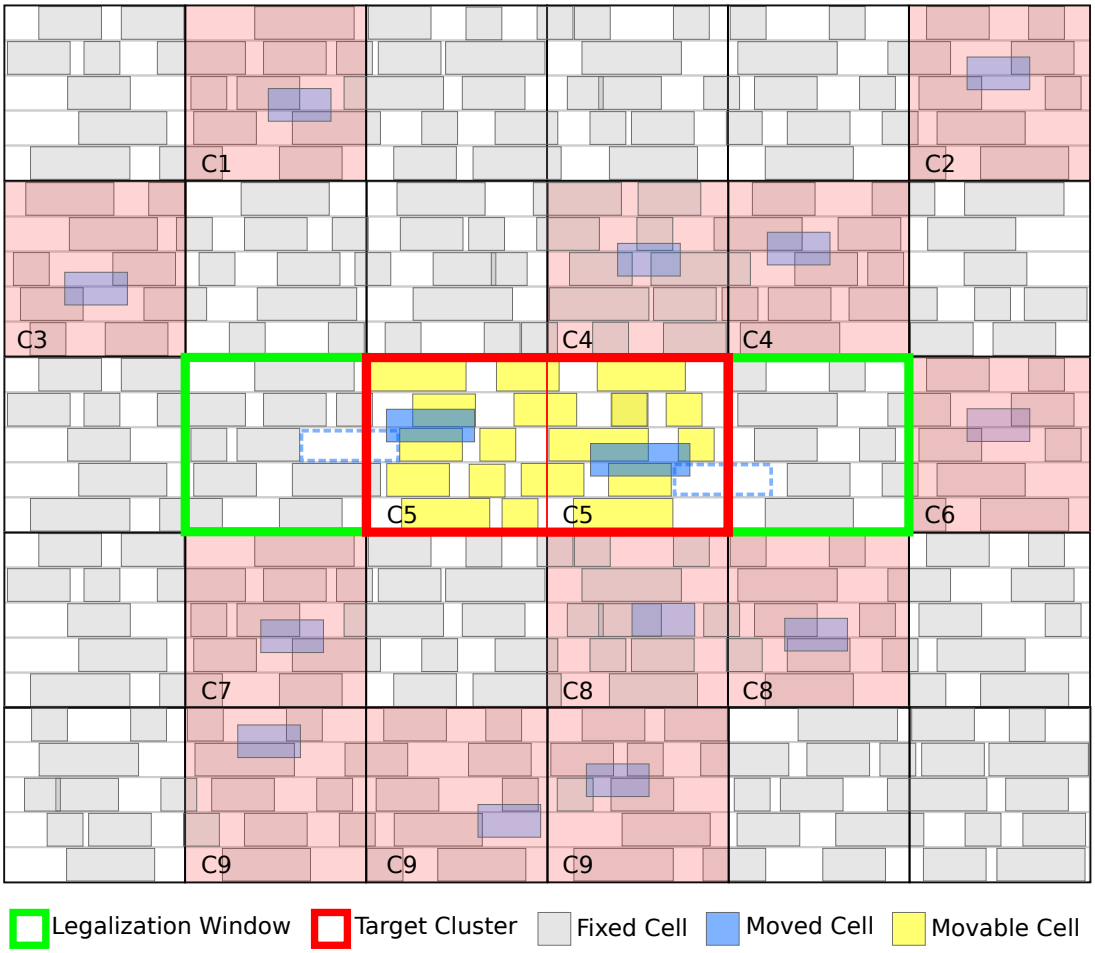
The ILP model does not consider legal positions when moving cells. This simplification in choosing the positions is fundamental to keeping a small number of variables inside the ILP model. If the ILP model would consider legal positions for each move, these positions should be seen as fixed for the subsequent moves, which would reduce the solution space. In addition, considering legalized positions, the number of variables would increase considerably. This is because each move would have to be combined with all other moves, and for each of those placement combinations, we would have new solutions and candidates for routing. Therefore a legalization step is required after solving the ILP model. This legalization step is executed by function "LEGALIZE(*panel*)" presented in line 11 of the Algorithm 1.

This step should legalize the moved cells moving as few already legalized cells as possible in order not to disturb the solution. If we call a legalization algorithm for the whole panel region, the number of affected cells could be higher. Therefore, a GCell clustered-base approach to legalize each panel is proposed. Figure 13 displays an example of a GCell cluster for a panel. For each row of GCell, neighbor GCells that are unlegalized are clustered. In this example, there are nine clusters (C1 to C9). A cluster is based

on GCells that have unlegalized cells inside. If a GCell has unlegalized cells and all neighbors are legalized, this cluster will be only one GCell (case of C1, C2, C3, C6, and C7). Otherwise, the neighboring unlegalized GCells are merged (case of C4, C5, C8, and C9). Then, picking cluster 5 (C5) as the target cluster (red bold rectangle), a legalization window is created by expanding one GCell on each side of the cluster. The green rectangle presents the legalization window in Figure 13. Only the cells that are totally inside the cluster area (colored in yellow) are considered movable for the legalization process.

Observe that the GCells belonging to the expansion area (left and right of the cluster) are always legalized because of the previous cluster step. Another importance of this expansion is that it enables us to legalize the cells using free spaces available in the neighboring legalized GCell. This helps us to reduce the displacement of legalized cells (cells that the ILP model has not moved). The blue dashed rectangle marks these free areas in Figure 13. The next step is to call the Abacus (Spindler; Schlichtmann; Johannes, 2008) algorithm to legalize each legalization window. Finally, if in the legalization process, some pin access of any cell change between GCells, this pin is reconnected to the previous GCell using the L-shape pattern route algorithm. In the end, the whole panel will be considered legalized if, and only if, all the clusters can be legalized. Otherwise, all the cells and nets belonging to this panel are restored to the state before the ILP call.

Figure 13 – Example of a GCells clustered-base approach to legalize each panel.



Source: the author.

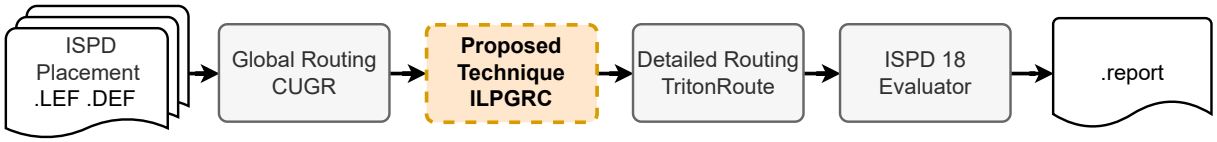
6 EXPERIMENTAL EVALUATION

In this chapter, the quality of the proposed techniques is assessed. First, the methodology and experimental setup are presented in Sections 6.1 and 6.2, followed by the default values of GCells, panels, and constants used in this thesis in Sections 6.3 and 6.4. Then, four sets of experiments are shown: **1)** Section 6.5 brings an overall quality experiment in relation to the best placement and routing academic flow established in Chapter 3. This experiment discusses the workload, runtime, impact of more iterations, impact of more candidates, and the constant values. **2)** The ILPGRC technique is evaluated over the second-best academic placement and routing tool combinations in Section 6.6. **3)** After, in Section 6.7, ILPGRC is compared with the so far state-of-the-art technique CRP 2.0. **4)** Section 6.8 brings a comparison of the current version of ILPGRC with its previous version published in the ISVLSI 2021 conference, renamed as ILPGRC$_{V0}$. Finally, Section 6.9 draws a summary of all the results.

6.1 EXPERIMENTAL METHODOLOGY

To assess the efficacy of ILPGRC and comprehend the real impact of moving cells during the GR step, this work relied on the evaluation flow shown in Figure 14, which includes the DR step. First, CUGR (Liu et al., 2020) is used to generate the GR solution for the ISPD Contest 2018 (Mantik et al., 2018) and ISPD Contest 2019 (Liu et al., 2019) initial placements. Then, the proposed technique (ILPGRC) is applied to optimize the circuits by moving cells and rerouting nets. Subsequently, TritonRoute (Kahng; Wang; Xu, 2020) is used to perform the DR step. Finally, the official ISPD Contest 2018 evaluator generates the report that allows us to investigate the quality of the solution. Section 3.2 details the evaluation process. The flow in Figure 14 generates the baseline results, but in this case, skipping the application of the proposed technique (ILPGRC). The selection process of CUGR and TritonRoute as the GR and DR tools was already justified in Chapter 3.

Figure 14 – The evaluation flow using the ISPD Contest 2018 and ISPD Contest 2019 benchmarks.



Source: the author.

6.2 EXPERIMENTAL SETUP

The developed algorithms were implemented in C++ using the open-source library Ophidian (Embedded Computing Lab, 2019). The boost library (Schling, 2011) was used for the data structure, including graphs, trees, and geometric operations. CPLEX (IBM, 2018) 12.8 library is applied to solve the ILP model. The experiments were executed on a Linux cluster with 48 cores, 2x Intel® Xeon® Gold 6240R CPU @ 2.40GHz (Cascade Lake, 2021) and 185GB RAM.

6.3 GCELLS AND PANELS DEFAULT SIZES

This section presents the GCells and Panels default sizes, which were experimentally defined. Four different sizes were tested for GCells: 4k by 5k dbu (2 rows x 50 sites), 4k by 10k dbu (2 rows x 100 sites), and 10k by 10k dbu (5 rows x 100 sites). These GCell sizes did not impact the quality of the final results. Therefore, this work considers the GCells size of 10k by 10k dbu (5 rows and 50 sites) to increase the chances of legalizing the movements without impacting the already placed cells in the same region. With a larger GCell, there are more empty spaces in the legalization area which can lead to less displacement for the placed cells. Thus, this work uses the default values:

- CUGR GCell size is 3K by 3K dbu.
- ILPGRC GCell size is 10K by 10K dbu.
- ILPGRC Checkered panels in level 1 are 5 by 5 GCells (50K by 50K dbu).

The only exceptions for these values are circuits ispd19_test4 and ispd19_test5, which use a 65 nm technology node. For these two circuits:

- CUGR GCell size is 2K by 2K dbu.
- ILPGRC GCell size is 8K by 8K dbu.
- ILPGRC Checkered panels in level 1 are 5 by 5 GCells (40K by 40K dbu).

The use of the above-mentioned values solves the following issue: if the default values are used to route circuits ispd19_test4 and ispd19_test5, TritonRoute incorrectly reports that the guides are not connected (no pin access found), and ends the execution.

6.4 CONSTANT VALUES USED IN THE ILP MODEL

The constant values used in Equation 5.2 of the ILP model are presented in Table 9. These values penalize the lower metal layers and keep the net nearest to the provided initial solution. Note that the values for VIAs are greater than those used in routing wires. Therefore the ILP model will prioritize solutions that use fewer VIAs, which

is the focus of this work. All these values were obtained empirically through extensive experimentation.

Table 9 – Constant values used in the ILP model

Metal	Const.	Value	VIA	Const.	Value
Metal 1	α_1	10K	VIA 1	β_1	20K
Metal 2	α_2	10K	VIA 2	β_2	11K
Metal 3	α_3	1K	VIA 3	β_3	2K
Metal 4	α_4	1K	VIA 4	β_4	1.1K
Metal 5	α_5	100	VIA 5	β_5	200
Metal 6	α_6	100	VIA 6	β_6	110
Metal 7	α_7	10	VIA 7	β_7	20
Metal 8	α_8	10	VIA 8	β_8	11
Metal 9	α_9	1			

6.5 QUALITY EVALUATION OF ILPGRC

Table 10 brings the results obtained by using the ILPGRC technique and the baseline results (i.e., without using ILPGRC), employing the evaluation flow depicted in Figure 14. For each circuit listed in the leftmost column, the table brings the technology node, the number and the percentage of movements, the total number of VIAs, the number of off-track VIAs, the total wirelength, the off-track wirelength, the wrong way wirelength, the number of metal short violations, the number of min area violations, the number of metal spacing violations, and the number of open nets. To make the comparison easy, the columns labeled as Basel bring the absolute number of VIAs and wirelength obtained by the baseline flow, whereas the columns labeled as ILPGRC Imp % show the percentage of improvement achieved when the ILPGRC technique is applied. Thus, a positive value in any of these columns means that ILPGRC has improved the given metric concerning the baseline result.

Table 10 – Experimental results for ISPD 2018 and ISPD 2019 benchmarks evaluated after the detailed routing solution.

Circuits ISPD	Node (nm)	Movements		Vias				Wirelength (mm)						Design Rule Violations (DRVs)							
				Total		Off Track		Total		Off Track		Wrong Way		Shorts		Min Area		Spacing		Open Nets	
		#	%	Basel. (K)	ILPGRC Imp. %	Basel.	ILPGRC Imp. %	Basel.	ILPGRC Imp. %	Basel.	ILPGRC Imp. %	Basel.	ILPGRC Imp. %	Basel.	ILPGRC	Basel.	ILPGRC	Basel.	ILPGRC	Basel.	ILPGRC
18_t1	45	642	7.23	36	**1.92**	171	**9.36**	172	-0.46	0.35	**72.76**	3.37	**8.14**	0	0	0	0	0	0	0	0
18_t2		922	2.57	360	**1.93**	2,164	**6.24**	3,134	-0.58	3.99	**63.83**	31.81	**13.32**	0	0	0	0	0	0	0	0
18_t3		1,042	2.90	363	**1.54**	2,197	**6.87**	3,505	-0.48	4.83	**57.20**	32.36	**14.08**	0	0	0	0	0	0	0	0
18_t4	32	2,264	3.14	725	**2.68**	17,336	-0.72	5,240	-0.39	9.98	**51.22**	27.84	**24.31**	0	0	0	0	0	0	0	0
18_t5		1,011	1.41	860	**8.53**	14,946	-0.58	5,481	-0.39	3.82	**76.68**	17.13	-2.64	0	0	0	0	0	0	0	0
18_t6		108	0.10	1,278	**8.20**	22,899	-1.31	7,100	-0.38	4.19	**63.62**	23.79	-6.62	0	0	0	0	0	0	0	0
18_t7		1,544	0.86	2,096	**6.59**	28,490	**0.13**	12,918	-0.30	7.07	**60.20**	37.40	**1.36**	0	0	0	0	0	0	0	0
18_t8		1,399	0.73	2,147	**6.20**	28,508	**0.13**	13,033	-0.25	7.95	**59.27**	39.06	**2.86**	0	0	0	0	0	0	0	0
18_t9		1,227	0.64	2,150	**7.08**	28,478	**0.09**	10,808	-0.44	7.49	**66.47**	37.70	**3.89**	0	0	0	0	0	0	0	0
18_t10_nf		1,267	0.44	2,308	**7.83**	32,184	**1.54**	13,542	-0.12	12.18	**65.37**	51.80	**14.30**	0	0	0	0	0	0	0	0
19_t1		99	1.11	38	**4.54**	3,965	**0.53**	126	-0.70	0.44	**17.03**	1.81	**5.11**	0	0	0	0	0	0	0	0
19_t2		985	1.37	798	**1.62**	132,593	**0.57**	4,940	-0.36	14.28	**12.55**	32.04	**16.96**	0	0	0	0	0	0	0	0
19_t3		343	4.14	63	**1.66**	6,083	**2.98**	164	-0.42	0.48	**17.06**	3.17	**13.11**	0	0	0	0	0	0	0	0
19_t6		2,041	1.13	1,972	**2.19**	46,333	**4.29**	13,117	-0.05	12.20	**28.89**	57.30	**18.16**	0	0	0	0	0	0	0	0
19_t7		2,263	0.63	3,789	**2.91**	255,196	**1.11**	24,198	-0.05	34.29	**42.32**	169.90	**19.47**	0	0	0	0	0	0	0	0
19_t8		5,755	1.07	6,176	**5.19**	382,323	**0.83**	37,073	-0.48	44.78	**38.48**	174.34	-2.67	0	0	0	0	0	0	0	0
19_t9		11,135	1.24	10,265	**5.88**	637,914	**0.79**	56,021	-0.48	77.37	**35.66**	295.01	-2.74	0	0	3	**0**	2	**0**	0	0
19_t10		10,951	1.22	9,574	**2.61**	637,558	**0.92**	55,325	-0.30	85.11	**39.45**	423.47	**14.00**	1	39	0	0	15	46	0	0
19_t4*	65	9,424	6.44	1,050	**11.74**	3,874	**71.17**	5,428	-6.40	31.11	**80.47**	259.16	**81.00**	170K	**3**	0	0	90K	**5**	1	**0**
19_t5*		1,881	6.50	151	**0.12**	155	**10.97**	927	-3.28	1.30	**41.52**	15.65	**47.74**	3K	**7**	0	0	31	**0**	0	0
Average			1.98		**4.69**		**5.61**		-0.83		**48.28**		**14.47**								
Avg. 18			2.00		**5.25**		**2.17**		-0.38		**63.66**		**7.30**								
Avg. 19			2.48		**3.85**		**9.42**		-1.25		**35.34**		**21.01**								

The results in Table 10 show that ILPGRC performed the best in the total number of VIAs for all circuits. The average reduction in the total number of VIAs is 4.69%, which was reached by moving only 1.98% of cells. This improvement comes from the path cost considered in Equation 5.2 and from the multiple routing candidates generated by function "*create_nets_candidates*" (lines 4 and 11 in Algorithm 3). This function generates multiple routing paths for each net. Each of these paths is routed through different metal layers, and as a consequence, they have different costs. Therefore, the ILP model tries to minimize the whole routing cost (Equation 5.1) by selecting, whenever it is possible, the routing paths that use the lower metal layers. It is worth noting that applying ILPGRC has enabled TritonRoute to reduce the off-track VIAs by 5.61% on average.

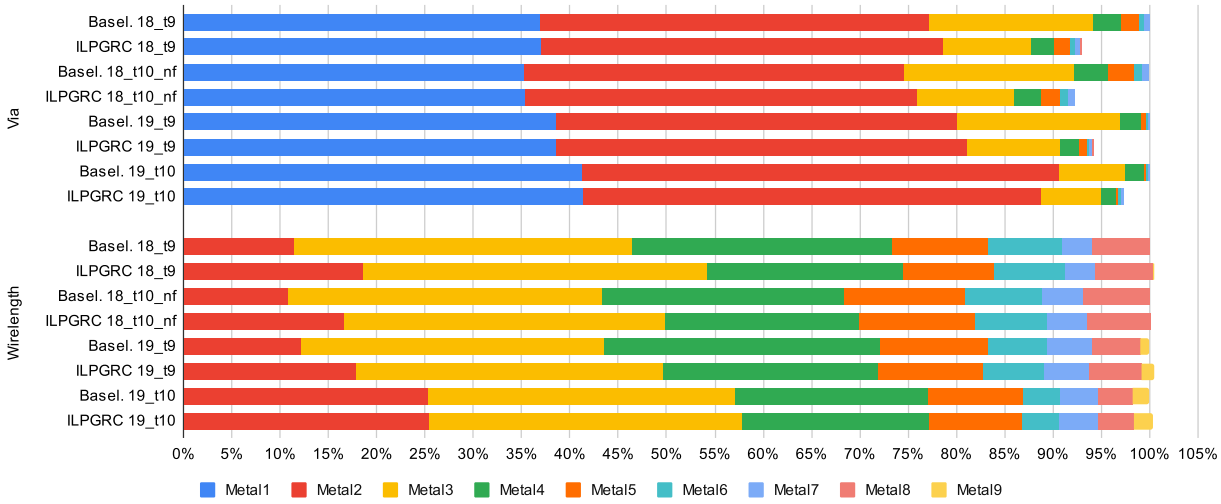
The ILPGRC technique achieves free DRV solutions in 17 out of 20 benchmarks, outperforming the reference flow by one benchmark (19_t9). It eliminates open nets and min area violations, with only a few shorts and spacing violations remaining. As a result, ILPGRC offers an improved solution after detailed routing, requiring minimal intervention from CAD engineers.

Moving our attention to the wirelength metric, we can observe significant improvements in the off-track and wrong-way wirelength when the ILPGRC was applied. This is highly correlated with the fact that the ILPGRC technique does not allow a net to be routed outside the tracks or in the non-preferred direction. Finally, although the ILPGRC technique was not able to lead TritonRoute to reduce wirelength, it is important to note that the wirelength was increased by only 0.83% on average, and in most cases (15 out of 20 benchmarks), the degradation in wirelength is less than 0.5%.

The counterpart of reducing the total number of VIAs can be an increase in the wirelength in the lower layers. The VIA and Wirelength distributions for the baseline and ILPGRC for circuits 18_t9, 18_t10_nf, 19_t0, and 19_t10 are presented in Figure 15. These values were extracted from the DEF file generated by TritonRoute using a commercial tool. The bars shorter than 100% mean that the ILPGRC could improve this metric for the specific circuit. In contrast, bars longer than 100% mean that ILPGRC degrades this metric for the given circuit.

Considering the VIA distribution, we can observe that the percentage of VIAs in Metal 1 is very similar for both ILPGRC and baseline. On the other hand, the difference in the number of VIAs comes from the upper layers. Considering the wirelength distribution, we can observe that both approaches result in the same length of metal in all layers. Therefore, we can conclude that ILPGRC is able to reduce the number of VIAs without impacting the circuit congestion.

Figure 15 – VIAs and Wirelength distribution after Detailed Routing for circuits ispd18_test9, ispd18_test10_nf, ispd19_test9, ispd19_test10.



Source: the author.

6.5.1 Workload and Runtime Analysis

The Checkered Paneling strategy is essential to make the ILPGRC technique viable since it reduces the problem size and breaks data dependency, thus enabling parallel execution. The workload of the largest design available in the conducted experiments (ispd19_test10) is presented in Table 11. Circuit ispd19_test10 contains 899K cells and 895K nets in total. For each level of the Checkered panels, this table presents the number of panels inside the level and the average, minimum, and maximum number of cells and nets.

We can note that there are less than 10 ILP instances with more than 1K cells and 5K nets. For all the other ILP instances (5,584), the number of cells and nets is smaller. Therefore, the circuit partitioning makes it possible to build and solve the ILP model in an acceptable runtime. It is important to mention that, in contrast with an ILP model for global routing, which must consider only the net candidates, in ILPGRC, the ILP model must consider all candidate nets and cells entirely inside the panel.

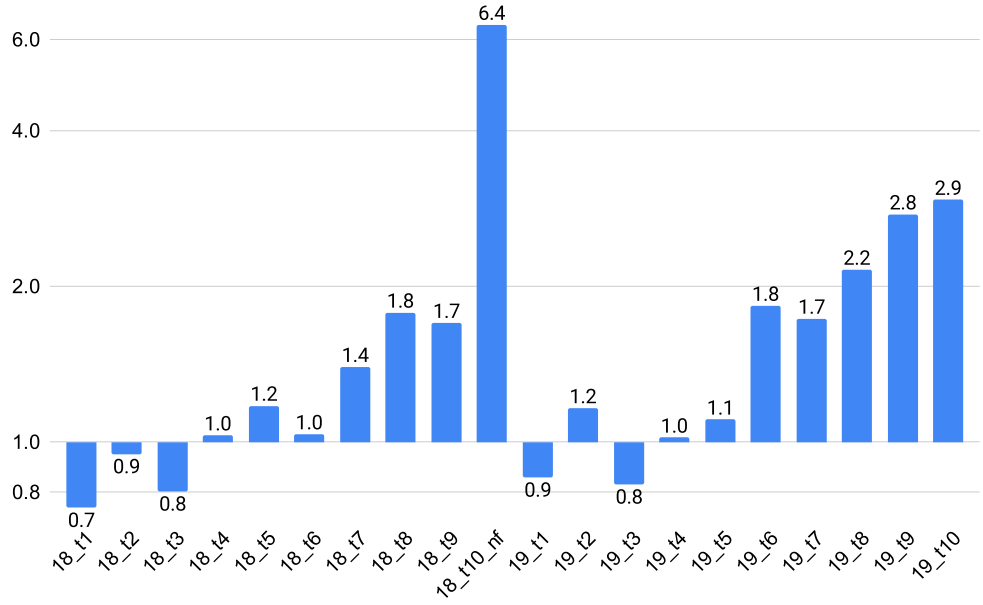
Building upon the insights garnered from the previous workload analysis, it is worth noting that another benefit of the Checkered Paneling strategy is the deterministic solution for all levels. This is because the parallelization only occurs inside the same level and color, and no data dependency is created because regions do not overlap each other.

Figure 16 displays the speedup achieved when the Checkered panels are run in parallel using 8 threads, assuming the sequential execution as a baseline. The speedup difference among the circuits is very noticeable, ranging from less than 1x for the smaller circuits to more than 6x in the case of circuit 18_t10_nf. As shown in Table 12, for the smaller circuits such as 18_t1, 18_t2, 18_t3, 19_t1, and 19_t3 the runtime of the parallel

Table 11 – The number of cells and nets in the ILP instances for circuit 19_t10

		average	min	max			average	min	max
level 1	cells	8.5	0	163	level 5	cells	224	18	404
# panels 4174	nets	134.2	1	318	# panels 20	nets	1722.6	284	2706
level 2	cells	2.2	0	32	level 6	cells	487.2	108	745
# panels 1040	nets	109	1	232	# panels 6	nets	3761	1047	5466
level 3	cells	14.6	0	99	level 7	cells	660.5	218	1103
# panels 280	nets	297.3	3	602	# panels 2	nets	4873.5	1529	8218
level 4	cells	91	6	199	level 8	cells	-	-	831
# panels 70	nets	920.5	251	1461	# panels 1	nets	-	-	7469

Figure 16 – Speedup when using parallel mode with 8 threads (y-axis) for each circuit (x-axis).



Source: the author.

version is greater than that of the sequential version. This is because the overhead time to create and synchronize the threads supersedes the benefits of running ILP in parallel. On the contrary, for the larger circuits, with more than 290K cells and 182K nets (18_t10_nf, and 19_t7 to 19_t10), the time to create and synchronize the threads is less significant compared to the time to create and solve the ILP model.

6.5.2 Impact of more iterations

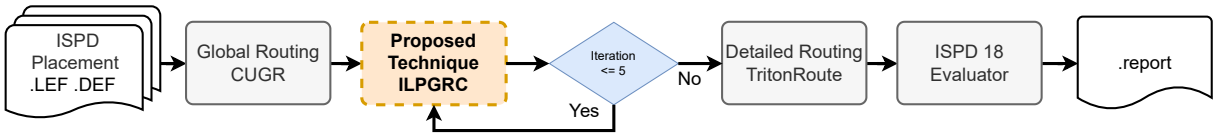
In order to investigate the impact of multiple iterations on the quality of results, a new experiment in which the (complete) ILPGRC technique is applied five times to each circuit was designed, following the flow shown in Figure 17.

Table 13 contains the improvement comparison between a single execution (ILP-GRC) and iterating five times (5X ILPGRC). By running ILPGRC 5 times, it is possible

to slightly reduce the number of VIAs for some circuits, but the average (3.97%) is worse than a single iteration (4.69%). Additionally, the number of design rule violations is reduced in the design ispd19_test10.

The runtime comparison between the original approach and the iterative one is presented in Table 14. Note that the runtimes of the iterative version (5X ILPGRC) are much longer (5.56 times on average) than those of the single execution of the ILPGRC. Therefore, we can conclude that the iterative version is not worth it.

Figure 17 – The evaluation flow using the ISPD Contest 2018 and ISPD Contest 2019 benchmarks.



Source: the author.

6.5.3 Impact of more movement candidates

This section brings attention to an experiment named Neighbor of Median Candidates (NMC). In this experiment, the four neighboring GCells of the median GCell (N, S, W, E) are also considered as movement candidates. With this, for each cell, the ILP model will contain 5 movement candidates. The main objective here is to explore the quality of results and runtime trade-off of creating more candidates in the ILP model. Table 15 presents the quality of results for this experiment, while the runtime in seconds is shown in Table 16.

We can observe in Table 15 that generating more candidates did not help to improve the QoR. This is due to the fact that the median GCell is the optimum point for

Table 12 – Runtime in seconds of the ILPGRC in sequential and parallel mode. The best runtime for each circuit is highlighted in bold.

Circuits ISPD	Sequential	Parallel	Circuits ISPD	Sequential	Parallel
18_t1	**22**	29	19_t1	**24**	28
18_t2	**45**	48	19_t2	127	**109**
18_t3	**49**	62	19_t3	**12**	14
18_t4	112	**109**	19_t4	157	**154**
18_t5	135	**115**	19_t5	30	**27**
18_t6	190	**184**	19_t6	614	**335**
18_t7	552	**396**	19_t7	1172	**677**
18_t8	673	**378**	19_t8	2175	**1011**
18_t9	510	**300**	19_t9	4928	**1791**
18_t10_nf	2644	**413**	19_t10	4605	**1568**

Table 13 – Results comparing ILPGRC with 5X ILPGRC. The best result for each circuit is highlighted in bold.

Circuits ISPD	Node (nm)	Movements				Vias (Imp. %)				Wirelength (Imp. %)						Design Rule Violations (DRVs)							
		ILPGRC		5X ILPGRC		Total		Off Track		Total		Off Track		Wrong Way		Shorts		Min Area		Spacing		Open Nets	
		#	%	#	%	ILPGRC	5X ILPGRC	ILPGRC	5X ILPGRC	ILPGRC	5X ILPGRC	ILPGRC	5X ILPGRC	ILPGRC	5X ILPGRC	ILPGRC	5X ILPGRC	ILPGRC	5X ILPGRC	ILPGRC	5X ILPGRC	ILPGRC	5X ILPGRC
18_t1		642	7.23	728	8.20	**1.92**	**1.92**	**9.36**	**9.36**	**-0.46**	**-0.46**	**72.76**	**72.76**	**8.14**	**8.14**	0	0	0	0	0	0	0	0
18_t2	45	922	2.57	1,095	3.05	**1.93**	1.86	6.24	**6.47**	**-0.58**	-0.63	63.83	**64.63**	13.32	**13.37**	0	0	0	0	0	0	0	0
18_t3		1,042	2.90	1,268	3.52	**1.54**	1.50	**6.87**	**6.87**	**-0.48**	-0.54	**57.20**	53.58	**14.08**	13.76	0	0	0	0	0	0	0	0
18_t4		2,264	3.14	2,662	3.69	**2.68**	2.65	-0.72	-0.41	**-0.39**	-0.43	**51.22**	51.14	24.31	**24.66**	0	0	0	0	0	0	0	0
18_t5		1,011	1.41	1,075	1.49	**8.53**	**8.53**	-0.58	-0.75	**-0.39**	-0.41	**76.68**	75.59	-2.64	-3.16	0	0	0	0	0	0	0	0
18_t6		108	0.10	108	0.10	**8.20**	**8.20**	-1.31	-1.31	**-0.38**	**-0.38**	**63.62**	**63.62**	-6.62	-6.62	0	0	0	0	0	0	0	0
18_t7		1,544	0.86	1,656	0.92	6.59	**6.60**	**0.13**	0.11	**-0.30**	-0.31	60.20	**60.38**	1.36	**1.58**	0	0	0	0	0	0	0	0
18_t8		1,399	0.73	1,549	0.81	**6.20**	6.17	**0.13**	0.11	**-0.25**	**-0.25**	59.27	**59.54**	2.86	**3.03**	0	0	0	0	0	0	0	0
18_t9		1,227	0.64	1,267	0.66	**7.08**	7.06	**0.09**	0.05	**-0.44**	-0.45	**66.47**	65.89	3.89	**4.08**	0	0	0	0	0	0	0	0
18_t10_nf		1,267	0.44	1,381	0.48	**7.83**	7.75	**1.54**	1.42	**-0.12**	-0.13	**65.37**	64.39	**14.30**	13.79	0	0	0	0	0	0	0	0
19_t1	32	99	1.11	120	1.35	**4.54**	4.41	0.53	**0.63**	-0.70	**-0.64**	17.03	**18.98**	**5.11**	3.94	0	0	0	0	0	0	0	0
19_t2		985	1.37	1,085	1.50	**1.62**	1.59	**0.57**	**0.57**	**-0.36**	**-0.36**	**12.55**	12.11	**16.96**	16.65	0	0	0	0	**0**	6	0	0
19_t3		343	4.14	398	4.81	1.66	**1.76**	2.98	**3.52**	**-0.42**	-0.45	17.06	**19.45**	**13.11**	12.15	0	0	0	0	0	0	0	0
19_t6		2,041	1.13	2,575	1.43	**2.19**	2.12	**4.29**	4.09	**-0.05**	-0.08	28.89	**28.92**	18.16	**18.21**	0	0	0	0	0	0	0	0
19_t7		2,263	0.63	2,418	0.67	**2.91**	2.89	**1.11**	1.09	**-0.05**	-0.06	**42.32**	42.29	**19.47**	19.43	0	0	0	0	0	0	0	0
19_t8		5,755	1.07	6,611	1.23	**5.19**	4.97	0.83	**0.86**	-0.48	**-0.45**	**38.48**	37.38	-2.67	-2.89	0	0	0	0	0	0	0	0
19_t9		11,135	1.24	12,261	1.36	**5.88**	5.80	0.79	**0.83**	**-0.48**	-0.50	**35.66**	35.56	-2.74	-2.72	0	0	0	0	0	0	0	0
19_t10		10,951	1.22	11,800	1.31	**2.61**	2.60	0.92	**0.94**	**-0.30**	-0.31	**39.45**	39.11	**14.00**	13.84	39	**31**	0	0	**46**	**46**	0	0
19_t4*	65	9,424	6.44	Triton Fail		**11.74**		**71.17**		**-6.40**		**80.47**		**81.00**		3		0		5		0	
19_t5*		1,881	6.50	3,040	10.51	0.12	**-2.92**	**10.97**	-31.61	-3.28	**3.20**	**41.52**	40.58	**47.74**	44.12	7	1k	0	0	0	20	0	0
Average			1.98		2.76	**4.69**	3.97	**5.61**	0.15	-0.83	**-0.19**	**48.28**	47.68	**14.47**	10.28								
Avg. 18			2.00		2.29	**5.25**	5.22	**2.17**	2.19	**-0.38**	-0.40	**63.66**	63.15	**7.30**	7.26								
Avg. 19			2.48		3.23	**3.85**	2.58	**9.42**	-2.12	-1.25	**0.04**	**35.34**	30.49	**21.01**	13.64								

Table 14 – Runtime in seconds of the ILPGRC. The best runtime for each circuit is highlighted in bold.

Circuits ISPD	1X	5X	Circuits ISPD	1X	5X
18_t1	**29**	78	19_t1	**28**	87
18_t2	**48**	107	19_t2	**109**	358
18_t3	**62**	128	19_t3	**14**	20
18_t4	**109**	290	19_t4	**154**	439
18_t5	**115**	337	19_t5	**27**	57
18_t6	**184**	472	19_t6	**335**	1,546
18_t7	**396**	1,657	19_t7	**677**	3,950
18_t8	**378**	1,748	19_t8	**1,011**	8,631
18_t9	**300**	1,286	19_t9	**1,791**	20,309
18_t10_nf	**413**	10,338	19_t10	**1,568**	23,449

a movement considering all nets connected to a cell. Therefore, the ILP model will prefer to move the cells to this median point instead of the neighboring GCells. The runtime increased by only 2.03 times, on average, adding 4 times more movement candidates. This can be due to the sparsity of the model.

Table 15 – Experimental results of ISPD 2018 and ISPD 2019 benchmarks were evaluated after the detailed routing solution. ILPGRC NMC column means more candidates of movement are generated.

Circuits ISPD	Node (nm)	Movements				Vias (Imp. %)				Wirelength (Imp. %)						Design Rule Violations (DRVs)							
		ILPGRC		ILPGRC NMC		Total		Off Track		Total		Off Track		Wrong Way		Shorts		Min Area		Spacing		Open Nets	
		#	%	#	%	ILPGRC	ILPGRC NMC	ILPGRC	ILPGRC NMC	ILPGRC	ILPGRC NMC	ILPGRC	ILPGRC NMC	ILPGRC	ILPGRC NMC	ILPGRC	ILPGRC NMC	ILPGRC	ILPGRC NMC	ILPGRC	ILPGRC NMC	ILPGRC	ILPGRC NMC
18_t1		642	7.23	1,229	13.84	**1.92**	**1.92**	**9.36**	**9.36**	**-0.46**	**-0.46**	**72.76**	**72.76**	**8.14**	**8.14**	0	0	0	0	0	0	0	0
18_t2	45	922	2.57	1,489	4.15	**1.93**	1.55	6.24	**8.13**	**-0.58**	-0.86	63.83	**64.76**	13.32	**13.99**	0	0	0	0	0	0	0	0
18_t3		1,042	2.90	1,773	4.93	**1.54**	1.16	**6.87**	6.83	**-0.48**	-0.79	**57.20**	52.40	**14.08**	13.31	0	0	0	0	0	0	0	0
18_t4		2,264	3.14	3,920	5.44	**2.68**	2.45	**-0.72**	-1.12	**-0.39**	-0.52	**51.22**	50.39	24.31	**24.84**	0	0	0	0	0	0	0	0
18_t5		1,011	1.41	4,237	5.89	**8.53**	8.50	**-0.58**	-1.05	**-0.39**	-0.45	76.68	**77.13**	-2.64	**-2.53**	0	0	0	0	0	0	0	0
18_t6		108	0.10	721	0.67	**8.20**	8.19	**-1.31**	-1.44	-0.38	**-0.37**	63.62	**63.65**	-6.62	**-6.46**	0	0	0	0	0	0	0	0
18_t7		1,544	0.86	5,318	2.96	6.59	**6.71**	**0.13**	0.10	**-0.30**	-0.46	60.20	**61.05**	1.36	**1.51**	0	0	0	0	0	0	0	0
18_t8		1,399	0.73	5,028	2.62	6.20	**6.66**	**0.13**	0.10	**-0.25**	-0.39	59.27	**62.66**	2.86	**4.03**	0	0	0	0	0	0	0	0
18_t9		1,227	0.64	5,006	2.59	**7.08**	7.05	**0.09**	0.06	**-0.44**	-0.63	**66.47**	66.36	**3.89**	3.87	0	0	0	0	0	0	0	0
18_t10_nf		1,267	0.44			**7.83**		**1.54**		**-0.12**		**65.37**		**14.30**		0		0		0		0	
19_t1	32	99	1.11	451	5.08	4.54	**4.60**	0.53	**0.66**	**-0.70**	-0.74	17.03	**18.21**	**5.11**	4.61	0	0	0	0	0	0	0	0
19_t2		985	1.37	2,673	3.71	**1.62**	0.99	0.57	**0.58**	**-0.36**	-0.74	**12.55**	12.21	**16.96**	16.80	0	0	0	0	**0**	3	0	0
19_t3		343	4.14	1,553	18.75	**1.66**	1.24	2.98	**3.37**	**-0.42**	-2.00	**17.06**	16.41	13.11	**13.28**	0	0	0	0	0	0	0	0
19_t6		2,041	1.13	5,996	3.33	**2.19**	2.01	**4.29**	3.94	**-0.05**	-0.39	**28.89**	28.64	**18.16**	17.76	0	0	0	0	0	0	0	0
19_t7		2,263	0.63	10,265	2.85	**2.91**	2.90	**1.11**	1.06	**-0.05**	-0.14	**42.32**	42.17	**19.47**	19.40	0	0	0	0	0	0	0	0
19_t8		5,755	1.07	18,795	3.48	**5.19**	4.34	**0.83**	0.75	**-0.48**	-0.88	**38.48**	37.00	**-2.67**	-3.22	0	0	0	0	0	0	0	0
19_t9		11,135	1.24	37,255	4.14	**5.88**	5.67	**0.79**	0.74	**-0.48**	-0.87	**35.66**	35.39	**-2.74**	-3.04	0	0	0	0	0	0	0	0
19_t10		10,951	1.22	38,729	4.31	**2.61**	2.48	**0.92**	0.88	**-0.30**	-0.63	**39.45**	39.13	**14.00**	13.38	**39**	48	0	0	46	**35**	0	0
19_t4*	65	9,424	6.44			**11.74**		**71.17**		**-6.40**		**80.47**		**81.00**		**3**		**0**		**5**		**0**	
19_t5*		1,881	6.50	3,263	11.28	**0.12**	-1.10	**10.97**	7.74	**-3.28**	-4.03	**41.52**	40.28	47.74	**50.64**	**7**	11	0	0	**0**	100	0	0
Average			1.98		5.43	**4.69**	3.74	**5.61**	2.26	**-0.83**	-0.85	**48.28**	46.70	**14.47**	10.57								
Avg. 18			2.00		4.31	**5.25**	4.91	**2.17**	2.33	**-0.38**	-0.55	**63.66**	63.46	**7.30**	6.74								
Avg. 19			2.48		6.56	**3.85**	2.57	**9.42**	2.19	-1.25	**-1.16**	**35.34**	29.94	**21.01**	14.40								

Table 16 – Runtime in seconds of the ILPGRC and ILPGRC NMC. The best runtime for each circuit is highlighted in bold.

Circuits ISPD 2018	ILPGRC	ILPGRC NMC	Circuits ISPD 2019	ILPGRC	ILPGRC NMC
18_t1	**29**	75	19_t1	**28**	89
18_t2	**48**	77	19_t2	**109**	199
18_t3	**62**	96	19_t3	**14**	25
18_t4	**109**	189	19_t4	**154**	314
18_t5	**115**	216	19_t5	**27**	40
18_t6	**184**	294	19_t6	**335**	676
18_t7	**396**	695	19_t7	**677**	1,325
18_t8	**378**	725	19_t8	**1,011**	2,451
18_t9	**300**	549	19_t9	**1,791**	5,212
18_t10_nf	**413**	521	19_t10	**1,568**	5,026

6.5.4 Impact of Layer Weight

If a high cost is given to the lower layers, the GR/DR engines assign long segments to the upper layers. This approach helps in leaving resources for short segments in lower layers, possibly alleviating the pin access problem as well. For example, adopting an extremely high cost in the first two metal layers (horizontal and vertical) will possibly allow VIAs only for pin access in these layers.

To understand how the layer weight distribution affects the quality of results, two new experiments were performed, changing the metal weights (constant α of the cost function). The new metal weights are printed in Table 17. In these experiments, the VIA Factor (ω) is equal to 1. Two scenarios were evaluated: W2 and W3. W2 aims to understand how the ILP model will perform if all metal layers have the same weight, and W3 aims to be a counterpoint of the original experiment (that uses W1). The results of these experiments for circuit 18_t10_nf are stamped in Table 18. We can observe that the best improvement in VIAs and the lowest degradation in wirelength are achieved by the experiment with W1 metal weights (original experiment). It is worth noting that the only scenario producing design rule violations is the one without metal weights (W2).

Table 17 – Metal Weights for each metal layer

Metal	Const.	W1	W2	W3
Metal 1	α_1	10,000	1	1
Metal 2	α_2	10,000	1	1
Metal 3	α_3	1,000	1	10
Metal 4	α_4	1,000	1	10
Metal 5	α_5	100	1	100
Metal 6	α_6	100	1	100
Metal 7	α_7	10	1	1,000
Metal 8	α_8	10	1	1,000
Metal 9	α_9	1	1	10,000

Table 18 – Results for circuit i18_t10_nf changing the wirelength weights.

Metal Weight (α)	Via Factor (ω)	Movements #	Movements %	Vias (Imp. %) Total	Vias (Imp. %) Off Track	Wirelength (Imp. %) Total	Wirelength (Imp. %) Off Track	Wirelength (Imp. %) Wrong Way	DRVs Shorts	DRVs min area	DRVs Spacing	DRVs Open Nets
W1	1	1267	0.44	**7.83**	**1.54**	**-0.12**	65.37	**14.30**	0	0	0	0
W2	1	1038	0.36	6.55	1.37	-0.20	**67.47**	7.77	5	0	0	0
W3	1	1133	0.39	7.36	1.43	-0.22	65.29	8.92	0	0	0	0

6.6 COMPARING ILPGRC IN A DIFFERENT FLOW

Another experiment was conducted to assess the effectiveness of the ILPGRC technique in an alternative scenario. This experiment relied on the evaluation flow in Figure 14, except that DreamPlace (Lin et al., 2020) is used to generate new placement solutions for the ISPD Contests 2018 and 2019 circuits. The flow DreamPlace + CUGR + TritonRoute resulted in the second best placement to detailed routing flow reported in Chapter 3 and published in Fontana et al. (2022). Table 19 presents the results for this experiment. The columns labeled as DrPl bring the absolute values for the new baseline (DreamPlace + CUGR + TritonRoute) whilst the columns labeled as ILPGRC Imp % show the percentage of improvement achieved when the ILPGRC technique is applied. The rest of the table is organized in a similar fashion as Table 10. The baseline flow fails for three circuits: 18_t10_nf, 19_t3, and 19_t4. In particular, 18_t10_nf was not legalized after the execution of DreamPlace, whereas 19_t3 and 19_t4 resulted in a time-out of 48 hours when performing TritonRoute.

Table 19 – Experimental results for ISPD 2018 and ISPD 2019 benchmarks compared with DreamPlace (DrPl) + CUGR + TritonRoute. The results were evaluated after the detailed routing solution.

Circuits ISPD	Node (nm)	Movements #	Movements %	VIAs Total DrPl (K)	VIAs Total ILPGRC Imp. %	VIAs Off Track DrPl	VIAs Off Track ILPGRC Imp. %	Wirelength (mm) Total DrPl	Wirelength (mm) Total ILPGRC Imp. %	Wirelength (mm) Off Track DrPl	Wirelength (mm) Off Track ILPGRC Imp. %	Wirelength (mm) Wrong Way DrPl	Wirelength (mm) Wrong Way ILPGRC Imp. %	DRVs Shorts DrPl	DRVs Shorts ILPGRC	DRVs Min Area DrPl	DRVs Min Area ILPGRC	DRVs Spacing DrPl	DRVs Spacing ILPGRC	DRVs Open Nets DrPl	DRVs Open Nets ILPGRC
18_t1	45	893	10.06	38	**4.85**	196	**26.02**	176	-1.62	0.58	**83.85**	3.50	**19.52**	0	0	0	0	0	0	0	0
18_t2		1,180	3.29	388	**5.29**	4,405	**56.19**	3,151	-1.36	6.57	**72.76**	32.25	**22.72**	0	0	0	0	0	0	1	**0**
18_t3		1,287	3.58	382	**4.92**	4,430	**55.58**	3,529	-0.94	6.67	**60.02**	32.21	**18.44**	0	0	0	0	0	0	2	**0**
18_t4		3,194	4.43	722	**1.29**	17,363	-1.70	5,313	-0.27	10.49	**53.74**	31.47	**36.64**	0	0	0	0	0	0	0	0
18_t5		4,976	6.92	889	**12.49**	19,637	**20.97**	5,580	-0.51	4.03	**78.59**	18.17	**3.79**	0	0	0	0	0	0	0	0
18_t6		720	0.67	1,382	**13.06**	28,522	**18.59**	7,752	-0.56	4.89	**64.69**	24.67	-1.28	0	0	0	0	0	0	0	0
18_t7		3,001	1.67	2,214	**10.97**	28,557	**0.29**	13,055	-0.52	8.03	**62.60**	37.85	**3.62**	0	0	0	0	0	0	0	0
18_t8		2,899	1.51	2,264	**10.71**	28,595	**0.35**	13,137	-0.39	8.92	**63.05**	38.90	**6.01**	0	0	0	0	0	0	0	0
18_t9		2,716	1.41	2,265	**11.05**	28,862	**1.32**	10,884	-0.66	8.01	**65.29**	37.61	**6.31**	0	0	0	0	0	0	0	0
18_t10_nf		Fail	Fail	Fail	Fail	Fail	Fail	Fail	Fail	Fail	Fail	Fail	Fail	Fail	Fail	Fail	Fail	Fail	Fail	Fail	Fail
19_t1	32	147	1.66	39	**4.18**	3,990	**2.21**	132	-0.93	0.42	**16.12**	1.97	**11.50**	0	0	0	0	0	0	0	0
19_t2		1,397	1.94	791	-1.63	133,119	**0.43**	4,992	-0.64	14.89	**12.06**	37.34	**22.77**	0	0	0	0	3	3	0	0
19_t3		Fail	Fail	Fail	Fail	Fail	Fail	Fail	Fail	Fail	Fail	Fail	Fail	Fail	Fail	Fail	Fail	Fail	Fail	Fail	Fail
19_t6		3,091	1.72	1,948	**0.81**	54,757	**18.26**	13,136	-0.48	12.22	**30.34**	68.90	**27.30**	0	0	6	**0**	16	**0**	0	0
19_t7		5,236	1.46	3,726	**1.28**	264,367	**4.35**	24,381	**0.00**	40.34	**50.82**	230.98	**39.24**	0	0	0	0	10	**0**	0	0
19_t8		9,191	1.70	6,242	**4.77**	403,575	**4.33**	37,377	-0.72	46.99	**38.90**	203.45	**5.59**	0	0	0	0	14	**0**	0	0
19_t9		16,815	1.87	10,413	**5.62**	671,099	**4.21**	56,561	-0.64	80.70	**36.99**	337.54	**5.06**	0	0	0	0	24	**0**	0	0
19_t10		20,185	2.24	9,428	**0.99**	663,260	**3.91**	56,004	-0.31	96.94	**46.42**	572.25	**32.28**	0	97	0	0	66	146	0	0
19_t4*	65	Fail	Fail	Fail	Fail	Fail	Fail	Fail	Fail	Fail	Fail	Fail	Fail	Fail	Fail	Fail	Fail	Fail	Fail	Fail	Fail
19_t5*		2,306	7.97	158	**3.35**	169	-17.16	930	**0.15**	1.57	**50.53**	19.06	**49.13**	5K	**1K**	0	0	1K	1K	0	0
Average			3.26		**5.53**		**11.66**		-0.61		**52.16**		**18.16**								
Avg. 18			3.46		**8.29**		**19.73**		-0.76		**67.18**		**12.86**								
Avg. 19			3.07		**2.42**		**2.57**		-0.45		**35.27**		**24.11**								

The results in Table 19 show that applying ILPGRC resulted in VIA reduction for all circuits, except 19_t2. The average reduction is 5.53% and was achieved by moving on average 3.26% of cells. As in the previous experiment, applying ILPGRC has enabled TritonRoute to reduce the off-track VIAs, in this case, by 11.66% on average.

Regarding DRVs, the baseline flow generated violations for nine circuits: 18_t2, 18_t3, 19_t2, 19_t5, 19_t6, 19_t7, 19_t8, 19_t9, and 19_10. On the other hand,

ILPGRC flow produced violations in only three circuits: 19_t2, 19_t5, and 19_10. The baseline produced open nets in circuits 18_t2 and 18_t3; no open nets were reported when ILPGRC was applied.

ILPGRC improves the wirelength in two circuits (19_t5 and 19_t7) while not increasing it by a maximum of 1.62%. On average, the wirelength is increased by only 0.61%. Additionally, ILPGRC resulted in significant improvements in the off-track and wrong-way wirelength in this experiment. This behavior shows that the ILPGRC technique leads TritonRoute to choose better routing paths for the nets.

6.7 COMPARING ILPGRC WITH CRP 2.0

To the best of the author's knowledge, CRP (Aghaeekiasaraee et al., 2022) and its respective extension CRP 2.0 (Aghaeekiasaraee et al., 2023) are the only available works able to do routing with cell movement using realistic benchmarks with technology node information. The data of CRP 2.0 was acquired from their authors, enabling the comparison between CRP 2.0 and ILPGRC after the detailed routing solutions.

Table 20 presents how ILPGRC performs using the same initial routing and baseline of CRP 2.0 technique (Aghaeekiasaraee et al., 2023). The values in this table are the percentages of improvement in relation to the baseline. So, the higher values are better, and the best value for each circuit and metric is highlighted in bold.

We can see that, on average, ILPGRC is superior in reducing VIAs but slightly degrades wirelength, which suggests that ILPGRC is more aggressive in the VIA reduction. ILPGRC can produce a valid solution for all benchmarks, whereas CRP 2.0 fails in the 19_t4 benchmark. Also, ILPGRC only generates a few DRVs that can be easily solved by hand, while CRP 2.0 leaves 24k shorts in circuit 18_t10. It is worth noting that most of the gain from CRP 2.0 comes from the circuits 19_t5, 19_t6, 19_t7, 19_t8, 19_t9, 19_t10, while for the other circuits CRP 2.0 only achieves slight gains. Furthermore, ILPGRC is superior in off-track and Wrong Way Wirelengh.

Table 20 – Results comparing ILPGRC with CRP 2.0

Circuits ISPD	Node (nm)	Vias (Imp. %)				Wirelength (Imp. %)						Design Rule Violations (DRVs)							
		Total		Off Track		Total		Off Track		Wrong Way		Shorts		Min Area		Spacing		Open Nets	
		ILPGRC	CRP 2.0	ILPGRC	CRP 2.0	ILPGRC	CRP 2.0	ILPGRC	CRP 2.0	ILPGRC	CRP 2.0	ILPGRC	CRP 2.0	ILPGRC	CRP 2.0	ILPGRC	CRP 2.0	ILPGRC	CRP 2.0
18_t1	45	**4.16**	0.21	**1.95**	0.65	-1.03	**-0.05**	**80.33**	-0.37	**8.88**	-1.16	0	0	0	0	0	0	0	0
18_t2		**5.08**	1.18	**5.32**	-2.71	-0.76	**-0.2**	**76.55**	-5	**17.71**	-7.23	0	0	0	0	0	0	0	0
18_t3		**4.71**	1.21	**7.00**	-0.84	-0.48	**-0.13**	**69.74**	-7.23	**16.73**	-7.75	0	0	0	0	1	**0**	0	0
18_t4	32	**2.18**	-0.05	-0.55	**0.21**	-0.28	**0.01**	**46.76**	-1.64	**25.73**	-1.89	0	0	0	0	0	0	0	0
18_t5		**6.83**	0.78	-0.77	**-0.11**	-0.43	**-0.01**	**70.05**	-3.61	**-3.69**	-6.75	0	0	0	0	0	0	0	0
18_t6		**6.64**	0.14	-1.38	**-0.14**	-0.43	**0.04**	**71.14**	-0.78	-4.89	**-1.23**	0	0	0	0	0	0	0	0
18_t7		**5.73**	0.98	**0.13**	-0.09	-0.34	**0.03**	**70.59**	-4.7	**1.23**	-8.93	0	0	0	0	0	0	0	0
18_t8		**5.60**	1.07	**0.14**	-0.12	-0.34	**0.09**	**70.14**	-2.23	**2.84**	-7.68	0	0	0	0	0	0	0	0
18_t9		**5.74**	1.17	**0.13**	-0.05	-0.52	**-0.03**	**74.57**	-1	**2.82**	-7.51	0	0	0	0	0	0	0	0
18_t10			**0.1**		**-0.56**		**0**		**-3.44**		**-2.61**		24K		0		0		0
18_t10_nf		**6.38**		**2.10**		**-0.20**		**72.92**		**12.18**		**1**		0		0		0	
19_t1		**8.87**	-2.45	**1.72**	**1.32**	-0.64	**0.22**	-19.38	**17.53**	0.23	**14.18**	0	0	0	0	0	0	0	0
19_t2		**2.41**	-0.59	**0.62**	-0.32	-0.16	**0.01**	**23.32**	**14.79**	**19.21**	15.7	0	0	0	0	0	0	0	0
19_t3		**2.17**	0.79	**2.93**	0.38	-0.07	**0.34**	**26.01**	11.46	**9.30**	7.32	0	0	0	0	0	0	0	0
19_t6		**6.36**	-1.41	**5.09**	-1.39	**0.06**	-0.22	**36.78**	**34.77**	18.42	**20.75**	0	0	0	0	0	0	0	0
19_t7		**3.42**	**16.14**	**1.11**	0.46	0.05	**0.58**	**42.65**	6.84	**19.47**	-6.18	0	0	0	0	0	0	0	0
19_t8		8.76	**11.61**	**1.36**	**0.92**	-0.52	**0.76**	**-76.64**	**14.15**	-8.95	**9.82**	0	0	0	0	**0**	5	0	0
19_t9		9.59	**11.51**	**1.21**	**0.9**	-0.56	**0.95**	**-62.75**	**12.56**	-8.73	**8.19**	0	0	0	0	**0**	8	0	0
19_t10		2.93	**18.29**	0.93	**1**	-0.19	**0.84**	**39.83**	9.06	**13.77**	-10.76	53	**0**	0	0	**54**	73		0
19_t4*	65	**7.90**	Fail	**-19.60**	Fail	**2.58**	Fail	**41.75**	Fail	**2.12**	Fail	**31**	Fail	**0**	Fail	**159**	Fail	**0**	Fail
19_t5*		6.74	**7.62**	19.31	**37.97**	**1.08**	**-1.49**	**59.86**	34.51	**23.91**	-10.47	74	**0**	**0**	3	**49**	**2**	0	0
Average		**5.61**	3.59	**1.44**	**1.97**	-0.16	**0.09**	**40.71**	6.61	**8.41**	-0.22								
Avg. 18		**5.31**	0.68	**1.41**	-0.38	-0.48	**-0.02**	**70.28**	-3	**7.95**	-5.27								
Avg. 19		**5.92**	**6.83**	**1.47**	**4.58**	0.16	**0.22**	**11.14**	**17.3**	**8.87**	5.4								

6.8 COMPARING WITH ISVLSI 2021

In this section, ILPGRC is compared with its predecessor, renamed as ILPGRC$_{V0}$, published in the 2021 IEEE Computer Society Annual Symposium on VLSI (Fontana et al., 2021). The main differences between ILPGCR and ILPGRC$_{V0}$ rely on the three following aspects: the ILP objective function, the partitioning strategy, and the legalization step. The objective function of the ILPGRC$_{V0}$ ILP model only considers the wirelength and does not consider the number of VIAs in each interconnection. The partitioning strategy of ILPGRC$_{V0}$, named paneling, divides the whole design region into horizontal slices. In addition, there is no legalization of the movements between the partition levels. With this, the final solution of ILPGRC$_{V0}$ is not legalized and, thus, cannot be processed by a detailed router. Finally, the ISVLSI worr (Fontana et al., 2021) only evaluates the results using global routing information instead of after the detailed routing step.

Toward a fair and complete comparison between the ILPGRC and ILPGRC$_{V0}$, it is necessary to evaluate the results from both techniques after the detailed routing. The required steps to make this comparison possible are depicted in Figure 18. With the solution generated by the ILPGRC$_{V0}$ binary, the circuit was legalized using a commercial tool. After the legalization, the new placement was rerouted with CUGR because of the disturbance in cell positions generated by the legalization process. Finally, having the legalized placement and the global routing solution, the same evaluation process used for ILPGRC was applied: TritonRoute was used to generate the detailed routing, and the ISPD 18 evaluator generated all the reports.

Figure 18 – ILPGRC$_{V0}$ evaluation flow.

Source: the author.

A comparison between the percentage improvement of ILPGRC and ILPGRC$_{V0}$ is presented in Table 21. All values are percentage improvements of the baseline (contest placement + CUGR + TritonRoute). Thus, a positive value in any of these columns means the technique has improved the given metric concerning the baseline result.

We can observe that the ILPGRC performed better than ILPGRC$_{V0}$ in all circuits and considering all metrics. Even using a commercial tool to legalize the circuits, the legalization embedded into ILPGRC leads to better results. Another important observation is the relevance of evaluating the results in a complete physical design flow. The ISVLSI work reports a wirelength reduction of 15.85% and a VIA reduction of 60.64% on average. However, only with this complete evaluation is it possible to determine the real impact of techniques in the flow. This discrepancy between the results emphasizes that

it is impossible to guarantee that a good impact in the Global Routing step will produce a better result after the detailed routing step.

Table 21 – Results comparing the current version of ILPGRC with its previous version published in the ISVLSI 2021 conference (Fontana et al., 2021), renamed as ILPGRC$_{V0}$. The best result for each circuit is highlighted in bold.

Circuits ISPD	Node (nm)	VIAs (Imp. %)				Wirelength (Imp. %)					
		Total		Off Track		Total		Off Track		Wrong Way	
		ILPGRC	ILPGRC$_{V0}$	ILPGRC	ILPGRC$_{V0}$	ILPGRC	ILPGRC$_{V0}$	ILPGRC	ILPGRC$_{V0}$	ILPGRC	ILPGRC$_{V0}$
18_t1		**1.92**	-4.22	9.36	**9.94**	**-0.46**	-5.76	**72.76**	-4.27	**8.14**	8.45
18_t2	45	**1.93**	0.03	**6.24**	1.57	-0.58	**-0.01**	**63.83**	-1.25	**13.32**	0.41
18_t3		**1.54**	0.01	**6.87**	1.27	-0.48	**-0.05**	**57.20**	2.01	**14.08**	0.32
18_t4		**2.68**	-0.07	-0.72	**0.14**	-0.39	**-0.06**	**51.22**	1.49	**24.31**	0.41
18_t5		**8.53**	0.08	**-0.58**	-1.21	-0.39	**-0.06**	**76.68**	0.39	-2.64	**-0.29**
18_t6		**8.20**	-0.37	-1.31	**-0.12**	-0.38	**-0.23**	**63.62**	0.52	-6.62	**-0.27**
18_t7	32	**6.59**	-0.03	**0.13**	0.04	-0.30	**-0.08**	**60.20**	-1.31	**1.36**	-0.53
18_t8		**6.20**	0.04	**0.13**	0.02	-0.25	**-0.09**	**59.27**	0.94	**2.86**	-0.20
18_t9		**7.08**	-0.03	**0.09**	0.03	-0.44	**-0.06**	**66.47**	0.68	**3.89**	0.42
18_t10_nf		**7.83**	-0.13	**1.54**	-0.23	**-0.12**	-0.34	**65.37**	0.10	**14.30**	-0.48
Avg. 18		**5.25**	-0.47	**2.17**	1.14	**-0.38**	-0.67	**63.66**	-0.07	**7.30**	0.82

6.9 SUMMARY

In this chapter, the results of the proposed flow on ISPD 2018 and 2019 benchmarks are presented. ILPGRC reduces the number of VIAs by 4.69%, moving only 1.98% of the cells and degrading the wirelength to less than 1%, on average. In addition, the detailed router reported no open nets after the proposed technique, while the number of DRVs was reduced in two out of three cases. The Checkered paneling strategy also enables the execution of multiple ILP models in parallel, providing a speedup for large circuits.

Three supplementary experiments were provided to evaluate the ILPGRC in a different flow using DreamPlace, comparing it with CRP 2.0, and comparing it with the previous ILP version published in the ISVLSI 2021 conference, named ILPGRC$_{V0}$. ILPGRC was able to produce better results compared with all the works and flows in these experiments. Therefore, ILPGRC should be considered the state-of-the-art router with cell movement technique.

7 CONCLUSIONS

This thesis proposed, developed, and evaluated ILPGRC, an ILP-based technique to optimize Global Routing (GR). ILPGRC simultaneously moves cells and routs nets while maintaining the legality of the circuit and avoiding design rule violations. This contrasts with the traditional approach that considers fixed placement during routing steps. This thesis also proposed a Checkered Paneling strategy, which reduces the input size of the Integer Linear Programming (ILP) model, thus making the ILPGRC approach scalable. The Checkered Paneling strategy also enables the execution of multiple ILP models in parallel, providing a speedup for large circuits. ILPGRC allows the router to move cells to optimize different routing objectives, such as the total number of VIAs, wirelength, and Design Rule Violations (DRVs). ILPGRC can assist other routing algorithms in working collaboratively with the placement algorithms, making both placement and routing more agile and efficient.

The source code for the proposed technique, including the Checkered Paneling strategy and the cluster-based approach to legalize the solution, is open-source and can be accessed at the Ophidian Library gitlab (Embedded Computing Lab, 2023). The main contributions of this thesis can be summarized as follows:

- Proposal, implementation, and validation of an Integer Linear Programming formulation that simultaneously moves cells and re-routes nets targeting routing optimization and DRV reduction.
- Proposal and development of a dynamic and hierarchical region-based partitioning strategy, named Checkered Paneling, that reduces the input size of the ILP model and enables parallelization. This strategy also sorts the nets according to their area, where small nets will be touched first.
- Establishment of an academic state-of-the-art baseline flow for works in the Placement and Routing steps. This flow was settled by evaluating twelve different flows built by combining three placements, two Global Routing, and two Detailed Routing tools.
- Proposal and development of a cluster-based approach to legalize the solution with minimum disturbance and displacement.
- Demonstration of the effectiveness of the proposed technique on benchmarks that include detailed routing information (ISPD 2018 and ISPD 2019), thus allowing the evaluation of metrics such as off-track VIAs and wires, wrong-way wires, metal shorts, min-areas, spacing rules, and open nets. This contrasts most of the related work, which limits the improvement evaluation to the global routing step.

In this thesis, the impacts of the proposed technique were evaluated after the detailed routing. This is unlike other works in the area but can provide a more accurate picture of the impact of the work. The experimental results show that ILPGRC reduces

the number of VIAs by 4.69%, moving only 1.98% of the cells and while keeping the wirelength increase to less than 1%, on average. In addition, the detailed router reported no open nets after the proposed technique, while the number of DRVs was reduced in two out of three cases.

ILPGRC is scalable due to the proposed Checkered panel's strategy. This strategy was able to speed up the technique up to 6x using eight threads while keeping the number of variables and contains in the ILP model under control. Additional experiments were provided to assess the quality of results in different scenarios where the whole ILPGRC technique was iterated five times in the same design, and more movement candidates were generated, and different values in the Layer Weight constants. By running ILPGRC five times, it is possible to reduce the number of VIAs for some circuits slightly. Generating more candidates did not help improve the quality of results. This is because the median GCell is the optimum point for a movement, considering all nets connected to a cell. Therefore, the ILP model will prefer to move the cells to this median point instead of the neighboring GCells. The third experiment shows that the Weight constants do not differ much in the final via reductions.

Three supplementary experiments were provided to evaluate the ILPGRC in a different flow using DreamPlace, comparing it with CRP 2.0, and with the previous ILP version published in the ISVLSI 2021 conference (named as ILPGRC$_{V0}$). ILPGRC was able to produce better results compared with all the works and flows in these experiments. Therefore, ILPGRC should be considered the state-of-the-art router with cell movement technique.

7.1 FUTURE WORK

Future work includes the following possibilities. First, the overall quality of the routing solutions could be enhanced by including more types of routing candidates, such as A* and Maze routing solutions for each net. By considering more types of routing candidates, the solution space of the ILP model will be larger. Consequently, the ILP model should be capable of selecting better paths for the nets, and the results will be improved.

As a second possible future work, the cell selection step could target cells in congested areas to move them to non-congested areas. Reducing the hot spots of the circuit should improve the number of DRVs in the Detailed Routing step. Such improvement may help ILPGRC achieve DRV-free solutions for all benchmarks.

Finally, the proposed ILP model can be extended to include timing information. With timing information, the model can try to move cells connected to cells belonging to the critical paths. This will reduce the capacitance associated with the critical cells, thus improving the circuit timing.

BIBLIOGRAPHY

AGHAEEKIASARAEE, E. et al. Cr&p: an efficient co-operation between routing and placement. In: IEEE. **Design, Automation & Test in Europe Conference & Exhibition (DATE)**. [S.l.], 2022. p. 772–777.

AGHAEEKIASARAEE, E. et al. Crp2. 0: A fast and robust cooperation between routing and placement in advanced technology nodes. **ACM Transactions on Design Automation of Electronic Systems (TODAES)**, ACM New York, NY, 2023.

CADENCE. **Innovus**. 2020. https://www.cadence.com/ko_KR/home/tools/digital-design-and-signoff/soc-implementation-and-floorplanning/innovus-implementation-system.html.

CHANG, Y.-J. 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.

CHEN, G. et al. Dr. cu: Detailed routing by sparse grid graph and minimum-area-captured path search. **IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)**, v. 39, n. 9, p. 1902–1915, 2020.

CHEN, J. et al. Datc rdf-2021: Design flow and beyond iccad special session paper. In: IEEE. **IEEE/ACM International Conference On Computer Aided Design (ICCAD)**. 2021. p. 1–6. Disponível em: https://doi.org/10.1109/ICCAD51958.2021.9643553.

CHO, M. et al. Boxrouter 2.0: A hybrid and robust global router with layer assignment for routability. **ACM Transactions on Design Automation of Electronic Systems (TODAES)**, ACM, v. 14, n. 2, p. 32, 2009.

CHO, M.; PAN, D. Z. Boxrouter: a new global router based on box expansion and progressive ilp. **IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)**, IEEE, v. 26, n. 12, p. 2130–2143, 2007.

CHU, C.; WONG, Y.-C. Flute: Fast lookup table based rectilinear steiner minimal tree algorithm for vlsi design. **IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)**, IEEE, v. 27, n. 1, p. 70–83, 2007.

DAI, K.-R.; LIU, W.-H.; LI, Y.-L. Efficient simulated evolution based rerouting and congestion-relaxed layer assignment on 3-d global routing. In: IEEE. **2009 Asia and South Pacific Design Automation Conference (DAC)**. [S.l.], 2009. p. 570–575.

DAI, K.-R.; LIU, W.-H.; LI, Y.-L. Nctu-gr: Efficient simulated evolution-based rerouting and congestion-relaxed layer assignment on 3-d global routing. **IEEE Transactions on very large scale integration (VLSI) systems**, IEEE, v. 20, n. 3, p. 459–472, 2011.

DAI, K.-R.; LU, C.-H.; LI, Y.-L. Grplacer: Improving routability and wire-length of global routing with circuit replacement. In: **IEEE/ACM International Conference on Computer-Aided Design (ICCAD)**. [S.l.: s.n.], 2009. p. 351–356.

Elsevier. **Scopus**. 2019. https://www2.scopus.com.

Embedded Computing Lab. **Ophidian: an Open Source Library for Physical Design Research and Teaching**. 2019. https://gitlab.com/eclufsc/ophidian. Federal University of Santa Catarina (UFSC).

Embedded Computing Lab. **Ophidian: an Open Source Library for Physical Design Research and Teaching**. 2023. https://gitlab.com/tiagoaugustofontana/ophidian-research. Federal University of Santa Catarina (UFSC).

FONTANA, T. A. 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.

FONTANA, T. A. et al. Ilpgrc: Ilp-based global routing optimization with cell movements. **IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)**, p. 1–1, 2023.

FONTANA, T. A. et al. Towards a reference place and route flow for academic research. **Journal of Integrated Circuits and Systems**, v. 17, n. 3, p. 1–12, 2022.

GAO, J.-R.; WU, P.-C.; WANG, T.-C. A new global router for modern designs. In: IEEE COMPUTER SOCIETY PRESS. **Proceedings of the 2008 Asia and South Pacific Design Automation Conference (DAC)**. [S.l.], 2008. p. 232–237.

GESTER, M. et al. Bonnroute: Algorithms and data structures for fast and good vlsi routing. **ACM Transactions on Design Automation of Electronic Systems (TODAES)**, ACM, v. 18, n. 2, p. 32, 2013.

HART, P. E.; NILSSON, N. J.; RAPHAEL, B. A formal basis for the heuristic determination of minimum cost paths. **IEEE transactions on Systems Science and Cybernetics**, IEEE, v. 4, n. 2, p. 100–107, 1968.

HE, X.; CHOW, W.-K.; YOUNG, E. F. Srp: Simultaneous routing and placement for congestion refinement. In: **ACM International Symposium on Physical Design (ISPD)**. New York, NY, USA: Association for Computing Machinery, 2013. (ISPD '13), p. 108–113. ISBN 9781450319546. Disponível em: https://doi.org/10.1145/2451916.2451943.

HELD, S. et al. Global routing with inherent static timing constraints. In: IEEE PRESS. **IEEE/ACM International Conference on Computer-Aided Design (ICCAD)**. [S.l.], 2015. p. 102–109.

HELD, S. et al. Global routing with timing constraints. **IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)**, IEEE, v. 37, n. 2, p. 406–419, 2017.

HSU, M.-K. et al. Design and manufacturing process co-optimization in nano-technology (designer track paper). In: IEEE. **IEEE/ACM International Conference on Computer-Aided Design (ICCAD)**. [S.l.], 2014. p. 574–581.

HU, J.; ROY, J. A.; MARKOV, I. L. Completing high-quality global routes. In: **International Symposium on Physical Design (ISPD)**. [S.l.: s.n.], 2010. p. 35–41.

HU, K.-S. et al. ICCAD-2020 CAD contest in routing with cell movement. In: **IEEE/ACM International Conference on Computer-Aided Design (ICCAD)**. [S.l.: s.n.], 2020. p. 1–4.

HU, K.-S. et al. **ICCAD-2021 CAD Contest: Routing with Cell Movement Advanced**. Synopsys, Inc., 2021. Disponível em: http://iccad-contest.org/2021/Problems/Problam_B_20210220_1540.pdf.

HUANG, Z. et al. Detailed placement and global routing co-optimization with complex constraints. **Electronics**, MDPI, v. 11, n. 1, p. 51, 2021.

IBM. CPLEX Optimizer 12.8. **URL: https://www. ibm. com/analytics/data-science/prescriptive-analytics/cplex-optimizer**, 2018.

KAHNG, A. B. et al. **VLSI physical design: from graph partitioning to timing closure**. [S.l.]: Springer Science & Business Media, 2011.

KAHNG, A. B.; SPYROU, T. The openroad project: Unleashing hardware innovation. In: **Proc. GOMAC**. [s.n.], 2021. Disponível em: https://vlsicad.ucsd.edu/Publications/Conferences/383/c383.pdf.

KAHNG, A. B.; WANG, L.; XU, B. Tritonroute: An initial detailed router for advanced vlsi technologies. In: **IEEE/ACM International Conference on Computer-Aided Design (ICCAD)**. [S.l.: s.n.], 2018. p. 1–8. ISSN 1933-7760.

KAHNG, A. B.; WANG, L.; XU, B. Tritonroute: The open-source detailed router. **IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)**, IEEE, v. 40, n. 3, p. 547–559, 2020.

KAHNG, A. B.; WANG, L.; XU, B. Tritonroute-wxl: The open-source router with integrated drc engine. **IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)**, IEEE, v. 41, n. 4, p. 1076–1089, 2021.

KIM, M.-C. et al. A simplr method for routability-driven placement. In: IEEE. **IEEE/ACM International Conference on Computer-Aided Design (ICCAD)**. [S.l.], 2011. p. 67–73.

KITCHENHAM, B.; CHARTERS, S. Guidelines for performing systematic literature reviews in software engineering. Citeseer, 2007.

LaPES. **StArt - State of the Art through Systematic Review**. 2019. http://lapes.dc.ufscar.br/tools/start_tool. Laboratory of Research on Software Engineering (LaPES) - Federal University of São Carlos (UFSCar).

LEE, T.-H.; WANG, T.-C. Congestion-constrained layer assignment for via minimization in global routing. **IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)**, IEEE, v. 27, n. 9, p. 1643–1656, 2008.

LIN, T.; CHU, C. Polar 2.0: An effective routability-driven placer. In: **Proceedings of the 51st Annual Design Automation Conference (DAC)**. [S.l.: s.n.], 2014. p. 1–6.

Lin, Y. et al. DREAMPlace: Deep learning toolkit-enabled gpu acceleration for modern vlsi placement. In: **DAC**. [S.l.: s.n.], 2020. p. 1–6.

LIU, D. et al. **Layer assignment and routing optimization for advanced technologies**. Tese (Doutorado) — The University of Texas at Austin, 2018.

LIU, J. et al. CUGR: detailed-routability-driven 3d global routing with probabilistic resource model. In: **DAC**. [S.l.: s.n.], 2020. p. 1–6.

LIU, W.-H. 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.

LIU, W.-H. et al. Ispd 2019 initial detailed routing contest and benchmark with advanced routing rules. In: ACM. **International Symposium on Physical Design (ISPD)**. [S.l.], 2019. p. 147–151.

LIVRAMENTO, V. et al. Incremental layer assignment driven by an external signoff timing engine. **IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)**, v. 36, n. 7, p. 1126–1139, 2017.

MANOLOPOULOS, Y. et al. **"R-Trees: Theory and Applications"**. [S.l.]: Springer London, 2006.

MANTIK, S. et al. Ispd 2018 initial detailed routing contest and benchmarks. In: ACM. **International Symposium on Physical Design (ISPD)**. [S.l.], 2018. p. 140–143.

MOFFITT, M. D. Maizerouter: Engineering an effective global router. **IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)**, IEEE, v. 27, n. 11, p. 2017–2026, 2008.

MOFFITT, M. D.; SZE, C. N. Wire synthesizable global routing for timing closure. In: IEEE PRESS. **Proceedings of the 16th Asia and South Pacific Design Automation Conference (ASP-DAC)**. [S.l.], 2011. p. 545–550.

OZDAL, M. M.; WONG, M. D. Archer: A history-based global routing algorithm. **IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)**, IEEE, v. 28, n. 4, p. 528–540, 2009.

PAN, M.; CHU, C. Fastroute: A step to integrate global routing into placement. In: **IEEE/ACM international conference on Computer-aided design (ICCAD)**. [S.l.: s.n.], 2006. p. 464–471.

PAN, M.; CHU, C. Fastroute 2.0: A high-quality and efficient global router. In: IEEE. **Asia and south pacific design automation conference (ASP-DAC)**. [S.l.], 2007. p. 250–255.

PAN, M.; CHU, C. Fastroute 2.0: A high-quality and efficient global router. In: IEEE. **2007 Asia and south pacific Design Automation Conference (DAC)**. [S.l.], 2007. p. 250–255.

PAN, M.; CHU, C. Ipr: An integrated placement and routing algorithm. In: **Proceedings of the 44th Annual Design Automation Conference (DAC)**. [S.l.: s.n.], 2007. p. 59–62.

PAN, M.; Viswanathan, N.; Chu, C. An efficient and effective detailed placement algorithm. In: **IEEE/ACM International Conference on Computer-Aided Design (ICCAD)**. [S.l.: s.n.], 2005. p. 48–55.

PAN, M. et al. Fastroute: An efficient and high-quality global router. **VLSI Design**, Hindawi, v. 2012, 2012.

PAPA, D. A. Broadening the scope of multi-objective optimizations in physical synthesis of integrated circuits. 2010.

ROY, J. A.; MARKOV, I. L. Seeing the forest and the trees: Steiner wirelength optimization in placement. **IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)**, IEEE, v. 26, n. 4, p. 632–644, 2007.

ROY, J. A.; MARKOV, I. L. High-performance routing at the nanometer scale. **IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)**, IEEE, v. 27, n. 6, p. 1066–1077, 2008.

ROY, J. A. et al. Crisp: Congestion reduction by iterated spreading during placement. In: **IEEE/ACM International Conference on Computer-Aided Design (ICCAD)**. [S.l.: s.n.], 2009. p. 357–362.

SCHALLER, R. R. **Technological innovation in the semiconductor industry: a case study of the international technology roadmap for semiconductors (ITRS)**. Tese (Doutorado) — George Mason University Fairfax, VA, 2004.

SCHEIFELE, R. Rc-aware global routing. In: IEEE. **IEEE/ACM International Conference on Computer-Aided Design (ICCAD)**. [S.l.], 2016. p. 1–8.

SCHLING, B. **The Boost C++ Libraries**. 1.79. ed. [S.l.]: XML Press, 2011.

SPINDLER, P.; SCHLICHTMANN, U.; JOHANNES, F. M. Abacus: Fast legalization of standard cell circuits with minimal movement. In: **International Symposium on Physical Design (ISPD)**. [S.l.: s.n.], 2008. p. 47–53.

TU, P.; CHOW, W.-K.; YOUNG, E. F. Timing driven routing tree construction. In: IEEE. **2017 ACM/IEEE International Workshop on System Level Interconnect Prediction (SLIP)**. [S.l.], 2017. p. 1–8.

VISWANATHAN, N.; PAN, M.; CHU, C. Fastplace 3.0: A fast multilevel quadratic placement algorithm with placement congestion control. In: IEEE. **2007 Asia and South Pacific Design Automation Conference (DAC)**. [S.l.], 2007. p. 135–140.

VOLKOV, A.; DOLGOV, S. ICCAD-2019 CAD contest in lef/def based open-source global router. In: **IEEE/ACM International Conference on Computer-Aided Design (ICCAD)**. [S.l.: s.n.], 2019.

WANG, F. 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.], 2021. p. 1–9.

WU, T.-H.; DAVOODI, A.; LINDEROTH, J. T. Grip: scalable 3d global routing using integer programming. In: ACM. **Proceedings of the 46th Annual Design Automation Conference (DAC)**. [S.l.], 2009. p. 320–325.

XU, Y.; ZHANG, Y.; CHU, C. Fastroute 4.0: global router with efficient via minimization. In: IEEE PRESS. **Proceedings of the 2009 Asia and South Pacific Design Automation Conference (DAC)**. [S.l.], 2009. p. 576–581.

ZANG, X. 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.

ZHU, Z. et al. A robust global routing engine with high-accuracy cell movement under advanced constraints. In: **IEEE/ACM International Conference On Computer Aided Design (ICCAD)**. [S.l.: s.n.], 2022. p. 1–9.

ZOU, P. et al. Incremental 3d global routing considering cell movement and complex routing constraints. **IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)**, IEEE, 2022.

APPENDIX A – LIST OF PUBLICATIONS

A.1 WORKS AS FIRST AUTHOR

Most of the results presented in this text were published as a journal paper in the IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD). The academic flow for place and routing presented in Chapter 3 was published in the Journal of Integrated Circuits and Systems (JICS). Some earlier experiments with the proposed ILP model were published at the IEEE Computer Society Annual Symposium on VLSI (ISVLSI). The details of those publications are as follows:

A.1.1 IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD 2023)

- **Title:** ILPGRC: ILP-Based Global Routing Optimization With Cell Movements
- **Authors:** *Tiago Augusto Fontana*, Erfan Aghaeekiasaraee, Renan Netto, Sheiny Fabre Almeida, Upma Gandhi, Laleh Behjat, José Luís Güntzel
- **DOI: https://doi.org/10.1109/TCAD.2023.3305579**
- **Qualis CC: A2**

A.1.2 Journal of Integrated Circuits and Systems (JICS 2022)

- **Title:** Towards a Reference Place and Route Flow for Academic Research
- **Authors:** *Tiago Augusto Fontana*, Renan Netto, Sheiny Fabre Almeida, Erfan Aghaeekiasaraee, Laleh Behjat, José Luís Güntzel
- **DOI:** https://doi.org/10.29292/jics.v17i3.648
- **Qualis CC: A4**

A.1.3 IEEE Computer Society Annual Symposium on VLSI (ISVLSI 2021)

- **Title:** ILP-Based Global Routing Optimization with Cell Movements
- **Authors:** *Tiago Augusto Fontana*, Erfan Aghaeekiasaraee, Renan Netto, Sheiny Fabre Almeida, Upma Gandhi, Aysa Fakheri Tabrizi, David Westwick, Laleh Behjat, José Luís Güntzel
- **DOI:** https://doi.org/10.1109/ISVLSI51109.2021.00016
- **Qualis CC:** A3

A.2 OTHER WORKS – AS CO-AUTHOR

During the doctorate, I also contributed to other publications, which, although not directly related to this thesis, provided useful tools for developing this thesis. Therefore, their details are provided below:

A.2.1 ACM Transactions on Design Automation of Electronic Systems (TODAES 2023)

- **Title:** CRP2.0: A Fast and Robust Cooperation between Routing and Placement in Advanced Technology Nodes
- **Authors:** Erfan Aghaeekiasaraee, Aysa Fakheri Tabrizi, *Tiago Augusto Fontana*, Renan Netto, Sheiny Fabre Almeida, Upma Gandhi, José Luís Güntzel, David Westwick, Laleh Behjat
- **DOI:** https://doi.org/10.1145/3590962
- **Qualis CC:** A4

A.2.2 Design, Automation & Test in Europe Conference & Exhibition (DATE 2022)

- **Title:** Cr&p: An efficient co-operation between routing and placement
- **Authors:** Erfan Aghaeekiasaraee, Aysa Fakheri Tabrizi, *Tiago Augusto Fontana*, Renan Netto, Sheiny Fabre Almeida, Upma Gandhi, José Luís Güntzel, David Westwick, Laleh Behjat
- **DOI:** https://doi.org/10.23919/DATE54114.2022.9774530
- **Qualis CC:** A1

A.2.3 IEEE International Conference on Electronics, Circuits and Systems (ICECS 2022)

- **Title:** E-RVP: An Initial Design Rule Violation Predictor Using Placement Information
- **Authors:** Sheiny Fabre Almeida, Aysa Fakheri Tabrizi, Erfan Aghaeekiasaraee, Renan Netto, *Tiago Augusto Fontana*, Upma Gandhi, José Luís Güntzel, Laleh Behjat, Cristina Meinhardt
- **DOI:** https://doi.org/10.1109/ICECS202256217.2022.9970846
- **Qualis CC:** B1

A.2.4 IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD 2021)

- **Title:** Algorithm Selection Framework for Legalization Using Deep Convolutional Neural Networks and Transfer Learning
- **Authors:** Renan Netto, Sheiny Fabre, *Tiago Augusto Fontana*, Vinicius Livramento, Laércio L. Pilla, Laleh Behjat, José Luís Güntzel
- **DOI:** https://doi.org/10.1109/TCAD.2021.3079126
- **Qualis CC:** A2

A.2.5 International Symposium on Physical Design (ISPD 2019)

- **Title:** How Deep Learning Can Drive Physical Synthesis Towards More Predictable Legalization
- **Authors:** Renan Netto, Sheiny Fabre, *Tiago Augusto Fontana*, Vinicius Livramento, Laércio Pilla, José Luís Güntzel
- **DOI:** https://doi.org/10.1145/3299902.3309754
- **Qualis CC:** A3

A.2.6 Symposium on Integrated Circuits and Systems Design (SBCCI 2018)

- **Title:** Enhancing Multi-Threaded Legalization Through k-d Tree Circuit Partitioning
- **Authors:** Sheiny Fabre, José Luís Güntzel, Laércio Pilla, Renan Netto, *Tiago Augusto Fontana*, Vinicius Livramento
- **DOI:** https://doi.org/10.1109/SBCCI.2018.8533264
- **Qualis CC:** A4

A.3 AWARDS

During the development of this thesis, I also received the following awards:

A.3.1 Second place in CADathlon at ICCAD 2018

- Team: *Tiago Augusto Fontana*, Renan Netto

A.3.2 Second place in CADathlon at ICCAD 2017

- Team: *Tiago Augusto Fontana*, Renan Netto

A.3.3 Third place in the 2017 ICCAD CAD Contest

- Problem C: Multi-Deck Standard Cell Legalization
- Team: Renan Netto, *Tiago Augusto Fontana*, Sheiny Fabre, Thiago Barbato, Chrystian Guth, José Luís Güntzel, Laércio Lima Pilla

APPENDIX B – SYSTEMATIC LITERATURE REVIEW (SLR)

This appendix contains the Systematic Literature Review (SLR) presented in Section 2.1. This review follows the methodology of Kitchenham & Charters (2007), and consists of three main phases:

1. Planning the review
2. Conducting the review
3. Reporting the review

In the first moment, it is crucial to identify why a Systematic Literature Review (SLR) is needed and then develop the review protocol. This protocol must have the objective of this SLR, the keywords and their synonyms, the source selection criteria, and which languages will be allowed. This protocol should define the search string, which will be applied in all databases.

In the second stage, the researcher should analyze the set of publications resulting from the search in all databases. First, by reading only the title, the unrelated works must be removed. After that, the irrelevant works are identified by reading the abstracts and thus removed.

Finally, in the last phase, all resulting works should be read and their data categorized. With this, the reviewer should be able to define the frontier of knowledge and possible unresolved problems.

For convenience, the present SLR only considers studies written in English and present in Scopus (Elsevier, 2019) database. The StArt (LaPES, 2019) tool was used to follow the Kitchenham & Charters (2007) methodology.

B.0.1 Protocol

This subsection has the primary objective of explaining how these related works were selected and ensuring the reproducibility of this research. The main aim of this SLR is to learn and summarize state-of-the-art global routers. This work uses the following keywords: global routing, routing, physical design, electronic design automation, integrated circuits, and very large-scale integration. With these, it was possible to obtain all the papers influencing global routing.

First, the search utilizes the operator PRE/ (pressed by) between “global” and “routing”. This is necessary to ensure that all results have the first term preceding the second term by a specific number of words. Then, the results must have at least one of the keywords listed in the previous paragraph. Until 26^{th} March 2021, the Scopus platform found 706 documents. When limiting the language to English and the published period from 2010 to 2021, the number of publications was reduced to 549. Then, the result was refined for only the leading conferences and journals, resulting in 281 papers. Finally, reading the title and abstract, 47 articles were accepted for the extraction step. Ap-

pendix B presents the full search string to provide high reproducibility and transparency.

B.0.2 Search String

global PRE/ routing AND ("VLSI" OR "Very Large Scale Integration" OR "IC" OR "Integrated Circuit" OR "Electronic Design Automation" OR "Physical Design" OR "EDA") AND (LIMIT-TO (EXACTSRCTITLE , "IEEE Transactions On Computer Aided Design Of Integrated Circuits And Systems") OR LIMIT-TO (EXACTSRCTITLE , "Proceedings Design Automation Conference") OR LIMIT-TO (EXACTSRCTITLE , "Proceedings Of The International Symposium On Physical Design") OR LIMIT-TO (EXACTSRCTITLE , "Proceedings Of The Asia And South Pacific Design Automation Conference ASP DAC") OR LIMIT-TO (EXACTSRCTITLE , "IEEE ACM International Conference On Computer Aided Design Digest Of Technical Papers Iccad") OR LIMIT-TO (EXACTSRCTITLE , "Proceedings IEEE International Symposium On Circuits And Systems") OR LIMIT-TO (EXACTSRCTITLE , "IEEE Transactions On Very Large Scale Integration VLSI Systems") OR LIMIT-TO (EXACTSRCTITLE , "ACM Transactions On Design Automation Of Electronic Systems")) AND (LIMIT-TO (PUBYEAR , 2019) OR LIMIT-TO (PUBYEAR , 2018) OR LIMIT-TO (PUBYEAR , 2017) OR LIMIT-TO (PUBYEAR , 2016) OR LIMIT-TO (PUBYEAR , 2015) OR LIMIT-TO (PUBYEAR , 2014) OR LIMIT-TO (PUBYEAR , 2013) OR LIMIT-TO (PUBYEAR , 2012) OR LIMIT-TO (PUBYEAR , 2011) OR LIMIT-TO (PUBYEAR , 2010)) AND (LIMIT-TO (DOCTYPE , "cp") OR LIMIT-TO (DOCTYPE , "ar")) AND (LIMIT-TO (LANGUAGE , "English")) AND (LIMIT-TO (SRCTYPE , "p") OR LIMIT-TO (SRCTYPE , "j"))