

UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM MICROELETRÔNICA

JUCEMAR LUIS MONTEIRO

**Algoritmo de Posicionamento Analítico
Detalhado Guiado a Caminhos Críticos**

Dissertação apresentada como requisito parcial
para a obtenção do grau de Mestre em
Microeletrônica

Orientador: Prof. Dr. Marcelo de Oliveira Johann
Co-orientador: Prof. Dr. José Luís Almada Güntzel

Porto Alegre
2014

CIP – CATALOGAÇÃO NA PUBLICAÇÃO

Monteiro, Jucemar Luis

Algoritmo de Posicionamento Analítico Detalhado Guiado a Caminhos Críticos / Jucemar Luis Monteiro. – Porto Alegre: PGMICRO da UFRGS, 2014.

110 f.: il.

Dissertação (mestrado) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Microeletrônica, Porto Alegre, BR-RS, 2014. Orientador: Marcelo de Oliveira Johann; Co-orientador: José Luís Almada Güntzel.

1. Microeletrônica. 2. Ferramentas de EDA. 3. Síntese Física. 4. Algoritmos de Posicionamento. 5. Algoritmo de Posicionamento Analítico Detalhado. I. Johann, Marcelo de Oliveira. II. Güntzel, José Luís Almada. III. Título.

UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

Reitor: Prof. Carlos Alexandre Netto

Vice-Reitor: Prof. Rui Vicente Oppermann

Pró-Reitor de Pós-Graduação: Prof. Vladimir Pinheiro do Nascimento

Diretor do Instituto de Informática: Prof. Luis da Cunha Lamb

Coordenador do PGMICRO: Prof. Gilson Inácio Wirth

Bibliotecária-chefe do Instituto de Informática: Beatriz Regina Bastos Haro

*Dedico este trabalho aos meus pais, Juvencio e Gilvite,
e ao meu avô, Angelo Baldissera.*

AGRADECIMENTOS

Aos meus pais pelo amor e dedicação para que eu pudesse ter as melhores oportunidades profissionais.

Ao meu orientador, professor Marcelo Johann, e ao meu coorientador, professor José Güntzel, pelo tempo e esforço dedicados à orientação neste trabalho de mestrado.

Aos membros da banca pelo tempo dedicado à avaliação rigorosa da dissertação e pelas sugestões.

Aos colegas de laboratório Guilherme Flach, Gracieli Posser e Cristina Meinhardt pela ajuda ao sanar as minhas dúvidas e pelo auxílio técnico.

Aos integrantes da equipe UFRGS/FURG pela esforço em codificar e integrar um fluxo em pouco mais de 3 meses para a Competição do ICCAD de 2014.

À CAPES pela bolsa de mestrado.

Este trabalho foi desenvolvido com a trilha sonora composta pelas discografias de: AC/DC, Back Sabbath, Iron Maiden e Scorpions.

*“If I have seen further than others,
it is because I stood on the shoulders of giants.”*

— SIR ISAAC NEWTON

RESUMO

O posicionamento das portas lógicas tem papel fundamental na qualidade de um circuito digital. A qualidade do posicionamento impacta diretamente no tamanho do circuito, no tempo de propagação dos sinais, consumo de energia, área com problemas de aquecimento, demanda de recursos de roteamento, etc. Desse modo, algoritmos de posicionamento de portas lógicas tem sido investigado por muitas décadas em busca de soluções de posicionamento com melhor qualidade e com o menor tempo de execução possível. Além disso, o posicionamento de portas lógicas é um problema de otimização combinatorial e ele é um dos problemas pertencentes a classe NP-Difícil. Desse modo, obter a solução ótima em tempo computacional razoável é praticamente impossível. Portanto, a investigação de técnicas e algoritmos que provenham melhores soluções do que as obtidas atualmente para o posicionamento de portas lógicas é de fundamental importância para o contínuo avanço da indústria de microeletrônica. Neste trabalho foi proposto um algoritmo de posicionamento analítico detalhado para minimizar as violações no tempo de propagação dos sinais de dados. O algoritmo proposto é uma adaptação de um algoritmo de posicionamento analítico quadrático da etapa de posicionamento global para atuar sobre as portas lógicas combinacionais dos caminhos críticos na etapa de posicionamento detalhado. As portas lógicas movimentadas pela formulação propostas são aquelas combinacionais pertencentes aos caminhos críticos e aquelas que são vizinhas no primeiro nível lógico das pertencentes aos caminhos críticos. O algoritmo proposto opera somente sobre os caminhos com violações no tempo de propagação de *late* dos sinais de dados. A validação experimental do algoritmo proposto mostrou que as violações de *Worst Negative Slack* (WNS) e *Total Negative Slack* (TNS) foram reduzidas, respectivamente, em até 47,9% e 59,6% no tempo de propagação dos sinais de dados. Portanto, a qualidade do posicionamento detalhado incrementa em até 5%. Por outro lado, as violações de *Average Bin Utilization* (ABU) incrementam em até 5,5%. O algoritmo de posicionamento analítico detalhado opera sobre no máximo 1% do total de portas lógicas dos circuitos analisados.

Palavras-chave: Microeletrônica, Ferramentas de EDA, Síntese Física, Algoritmos de Posicionamento, Algoritmo de Posicionamento Analítico Detalhado.

Analytical Detailed Placement Algorithm for Critical Paths

ABSTRACT

The logical gates placement has a fundamental impact on the placement quality of the circuit. The placement quality impacts directly on circuit size, timing propagation, power consumption, hotspot areas, etc. Therefore, placement algorithms have been researched for a long time to improve placement quality with less runtime to find a good solution to the placement problem. In this work was proposed an analytical detailed placement algorithm to minimize timing propagation violations. The proposed algorithm was adapted from a quadratic algorithm of the global placement step to handle critical paths in detailed placement step. Detailed quadratic algorithm moves gates in critical paths and the gates in the first deep logical level of the ones in critical paths that are the immediate neighbors. The detailed analytical algorithm works only in combinational gates that are part of critical paths and for ones in late critical paths. The experimental validation of the proposed detailed analytical algorithm shows a reduction in WNS and TNS violation, respectively, up to 47.9% and 59.6% in timing propagation. Therefore, detailed placement quality had improved up to 5%. Otherwise, ABU penalty also increased up to 5.5%. In our formulation is moved up to 1% of the total number of gates in the benchmarks.

Keywords: Microelectronic, EDA Tools, Physical Synthesis, Placement Algorithms, Analytical Detailed Placement Algorithm.

LISTA DE FIGURAS

1.1	Diagrama do fluxo de projeto digital	23
1.2	Diagrama das etapas da síntese física	23
1.3	Estágios da etapa de posicionamento	24
2.1	Modelo de comprimento de fio <i>Half-Perimeter Wire-Length</i> (HPWL)	31
2.2	Modelo de comprimento de fio Monotônico	32
2.3	Modelo de comprimento de fio Clique	33
2.4	Modelo de comprimento de fio Estrela	34
2.5	Modelo de comprimento de fio <i>Rectilinear Minimum Spanning Tree</i> (RMST)	35
2.6	Modelo de comprimento de fio <i>Rectilinear Steiner Minimum Tree</i> (RSMT)	36
2.7	Modelo de comprimento de fio <i>Rectilinear Steiner Arborescence</i> (RSA)	37
2.8	Modelo de comprimento de fio <i>Single-Trunk Steiner Tree</i> (STST) . .	38
2.9	Sinal de temporização para circuitos sequências (Relógio)	40
2.10	Tempo de <i>skew</i> do Relógio	40
2.11	Incerteza na transição do relógio e tempo requerido de propagação dos sinais de dados do tipo <i>late</i>	41
2.12	<i>Slack</i> dos sinais de dados para violações de <i>late</i>	41
2.13	Incerteza na transição do relógio e tempo requerido de propagação dos sinais de dados para violações de <i>early</i>	42
2.14	<i>Slack</i> dos sinais de dados para violações de <i>early</i>	42
2.15	TNS e WNS para os caminhos de dados com violações no tempo de propagação de <i>early</i>	43
2.16	TNS e WNS para os caminhos de dados com violações de propagação de <i>late</i>	44
2.17	Área de posicionamento global	45
2.18	Área de legalização	46
2.19	Área de posicionamento detalhado	46

2.20	Modelo de conexão clique	47
2.21	Modelo de conexão estrela	48
2.22	Modelo de conexão <i>Bound to Bound</i> (B2B)	49
3.1	Região expandida do SimPL durante otimização aproximada	59
3.2	Região expandida do Polar durante otimização aproximada	60
4.1	Exemplo de caminho crítico	70
4.2	Diagrama do fluxo de execução do UPlace	74
4.3	Diagrama do fluxo de execução do posicionador detalhado	75
5.1	Tempo médio de execução e desvio padrão para o algoritmo de posicionamento analítico detalhado	91
A.1	<i>Layout</i> do circuito vga_lcd	103
A.2	<i>Layout</i> do circuito b19	104
A.3	<i>Layout</i> do circuito leon3mp	104
A.4	<i>Layout</i> do circuito leon2	105
A.5	<i>Layout</i> do circuito netcard	105
B.1	Interface gráfica do UPlace	108

LISTA DE TABELAS

5.1	Configuração dos circuitos de teste (<i>Benchmarks</i>)	79
5.2	Deslocamento máximo permitido e período de relógio	79
5.3	Métrica de qualidade do posicionamento detalhado para os caminhos críticos com violações de <i>late</i> no tempo de propagação do sinais de dados para o deslocamento máximo 1	80
5.4	Razão, em porcentagem, entre a otimização do algoritmo de posicionamento analítico detalhado e a otimização do <i>skew</i> do relógio da métrica de qualidade do posicionamento detalhado dos caminhos críticos de <i>late</i> para o deslocamento 1	80
5.5	Métrica de qualidade do posicionamento detalhado dos caminhos críticos <i>late</i> para o deslocamento 2	81
5.6	Razão, em porcentagem, entre a otimização com o algoritmo analítico dos caminhos críticos e a otimização com o algoritmo de <i>skew</i> do relógio da métrica de qualidade do posicionamento detalhado dos caminhos críticos de <i>late</i> para o deslocamento 2	82
5.7	Penalidade de ABU dos caminhos críticos <i>late</i> ($\times 10^{-2}$) para o deslocamento 1	83
5.8	Razão, em porcentagem, entre a otimização com o algoritmo analítico dos caminhos críticos e após a otimização do <i>skew</i> do relógio das violações de ABU dos caminhos críticos de <i>late</i> para o deslocamento 1	83
5.9	Violações de ABU dos caminhos críticos de <i>late</i> ($\times 10^{-2}$) para o deslocamento 2	84
5.10	Razão, em porcentagem, entre a otimização com o algoritmo de posicionamento analítico detalhado e após a otimização do <i>skew</i> do relógio das violações de ABU dos caminhos críticos de <i>late</i> para o deslocamento 2	84
5.11	WNS de <i>late</i> (<i>ns</i>) para o deslocamento 1	85

5.12	Razão, em porcentagem, entre a otimização com o algoritmo de posicionamento analítico detalhado e a otimização do <i>skew</i> do relógio do WNS <i>late</i> - Deslocamento 1	86
5.13	WNS de <i>late</i> (<i>ns</i>) para o deslocamento 2	86
5.14	Razão, em porcentagem, entre a otimização dos caminhos críticos com o algoritmo analítico detalhado e a otimização do <i>skew</i> do relógio do WNS de <i>late</i> para o deslocamento 2	87
5.15	TNS de <i>late</i> (<i>ns</i>) para o deslocamento 1	88
5.16	Razão, em porcentagem, entre a minimização das violações de TNS com o algoritmo analítico detalhado e a otimização do <i>skew</i> do relógio do TNS para o <i>late</i> e para o deslocamento 1	89
5.17	TNS de <i>late</i> (<i>ns</i>) para o deslocamento 2	89
5.18	Razão, em porcentagem, entre a minimização de violações no tempo de propagação com o algoritmo de posicionamento analítico detalhado e a otimização do <i>skew</i> do relógio de TNS para <i>late</i> para o deslocamento 2	89
5.19	Número de portas lógicas móveis para os caminhos críticos com violações de <i>late</i>	90
5.20	Média e desvio padrão dos tempos de execução dos caminhos críticos	91

LISTA DE ACRÔNIMOS

ABU	<i>Average Bin Utilization</i>
BB	<i>Bounding Box</i>
B2B	<i>Bound to Bound</i>
CI	<i>Circuito Integrado</i>
DEF	<i>Design Exchange Format</i>
DRC	<i>Design Rule Checking</i>
EBB	<i>Explicit Bin-Blocking</i>
EDA	<i>Electronic Design Automation</i>
CTS	<i>Clock-Tree Synthesis</i>
HDL	<i>Linguagem de Descrição de Hardware</i>
HPWL	<i>Half-Perimeter Wire-Length</i>
ILR	<i>Iterative Local Refinement</i>
LEF	<i>Layout Exchange Format</i>
NP	<i>Tempo Polinomial não Determinístico</i>
P	<i>Tempo Polinomial Determinístico</i>
PLL	<i>Phase Lock Loop</i>
ProLR	<i>Progressive Local Refinement</i>
RMST	<i>Rectilinear Minimum Spanning Tree</i>
RSMT	<i>Rectilinear Steiner Minimum Tree</i>
RSA	<i>Rectilinear Steiner Arborescence</i>
RTL	<i>Register-Transfer Level</i>
SA	<i>Simulated-Annealing</i>
SDC	<i>Synopsys Design Constraints</i>
STA	<i>Static Timing Analysis</i>
STL	<i>Standard Template Library</i>
STST	<i>Single-Trunk Steiner Tree</i>
TNS	<i>Total Negative Slack</i>

WNS *Worst Negative Slack*

SUMÁRIO

1	INTRODUÇÃO	21
1.1	Fluxo de Projeto com Células Padrão para Circuitos Integrados Digitais	22
1.2	Etapa de Posicionamento de Portas Lógicas	25
1.3	Objetivos deste Trabalho	26
1.4	Organização desta Dissertação	27
2	BASE CONCEITUAL	29
2.1	Definição Formal do Problema de Posicionamento de Portas Lógicas em Circuitos Integrados	29
2.2	Métricas de Avaliação da Qualidade do Posicionamento	29
2.2.1	Comprimento de Fio	29
2.2.2	Métricas de Estimativa de Densidade de Utilização de Área do Circuito	34
2.2.3	Métrica de Estimativa de Congestionamento no Circuito	37
2.2.4	Análise de Temporização (<i>Timing</i>)	38
2.3	Estágios de Posicionamento	43
2.3.1	Posicionamento Global	43
2.3.2	Legalização	44
2.3.3	Posicionamento Detalhado	45
2.4	Modelos de Conexão para Posicionadores Analíticos	46
2.4.1	Modelo Clique	47
2.4.2	Modelo Estrela	48
2.4.3	Modelo Híbrido	48
2.4.4	Modelo B2B	49
2.5	Algoritmos de Posicionamento Global	49
2.5.1	Técnicas Estocásticas (<i>Simulated-Annealing</i> (SA))	50
2.5.2	Técnicas de Particionamento	51
2.5.3	Técnicas Analíticas	51
2.6	Considerações Finais	55

3	ALGORITMOS DE POSICIONAMENTO E TRABALHOS CORRELATOS	57
3.1	Posicionamento Global	57
3.1.1	SimPL	57
3.1.2	Polar e Polar 2.0	59
3.1.3	MAPLE	61
3.1.4	Ripple	62
3.1.5	ePlace	63
3.2	Legalização	64
3.2.1	Abacus	64
3.2.2	BonnPlace	65
3.2.3	Algoritmo de HU et al.	65
3.3	Posicionamento Detalhado	66
3.3.1	BraveDP	66
3.3.2	Algoritmo de CHOW et al.	67
3.4	Considerações Finais	68
4	ALGORITMO DE POSICIONAMENTO ANALÍTICO DETALHADO	69
4.1	Algoritmo de Posicionamento Analítico Detalhado	69
4.2	Fluxo de Execução Integrado no UPlace	73
4.2.1	Fluxo de Execução no UPlace para a Minimização de Violações no Tempo de Propagação do Sinais	74
4.3	Considerações Finais	76
5	RESULTADOS EXPERIMENTAIS	77
5.1	Configuração Experimental	77
5.2	Circuitos para Teste (<i>Benchmarks</i>)	78
5.3	Resultados e Discussão	79
5.3.1	Métrica de Qualidade do Posicionamento Detalhado	79
5.3.2	Penalidade de ABU	82
5.3.3	WNS para Violações de <i>Late</i>	85
5.3.4	TNS para Violações de <i>Late</i>	87
5.3.5	Número de Portas Lógicas Movimentadas pelo Algoritmo de Posicionamento Analítico Detalhado	90
5.3.6	Tempo Médio de Execução do Algoritmo de Posicionamento Analítico Detalhado	90
5.4	Considerações Finais	92

6	CONCLUSÕES E TRABALHOS FUTUROS	93
6.1	Conclusões	93
6.2	Trabalhos Futuros	94
	REFERÊNCIAS	97
	APÊNDICE A LAYOUT DOS CIRCUITOS POSICIONADOS	103
	APÊNDICE B FRAMEWORK UPLACE PARA POSICIONAMENTO DE CIRCUITOS INTEGRADOS	107
B.1	Estruturas de Dados	109
B.2	Algoritmos de Posicionamento Implementados	109

1 INTRODUÇÃO

Um dos fatores que possibilita a rápida e a constante integração de novas funcionalidades em um Circuito Integrado (CI) é um fluxo de projeto bem definido. Essas novas funcionalidades têm como efeito direto o aumento no número de transistores no CI. Por outro lado, o avanço da tecnologia de fabricação acrescenta restrições e novos desafios ao processo de síntese do fluxo de projeto. Isso deve-se principalmente às novas restrições dos novos processos de fabricação. Portanto, as ferramentas de *Electronic Design Automation* (EDA) devem tratar essas novas restrições oriundas da tecnologia de fabricação, além de suportar o constante aumento no número de transistores integrados nos novos CIs sem comprometer significativamente a escalabilidade. Ainda, elas devem obter uma resposta do problema com a qualidade desejada pelo projetista de CI digital em um tempo de execução razoável.

A formulação da grande maioria dos problemas de otimização das etapas de síntese lógica e física pertencentes ao fluxo de projeto digital tem complexidade de Tempo Polinomial não Determinístico (NP). Se um problema é conhecido por pertencer à classe NP, NP-Completa ou NP-Difícil, então é improvável que um algoritmo de Tempo Polinomial Determinístico (P) exista para essas classes de problemas. Assim, uma solução adequada para problemas dessas classes é obtida com o uso intenso de heurísticas (SHERWANI, 1999). O desenvolvimento de algoritmos e de ferramentas de EDA devem considerar a natureza teórica do problema, o incremento constante no número de transistores e também as novas restrições de *layout* originárias dos novos nodos tecnológicos.

O escopo desta dissertação é o posicionamento de portas lógicas. O posicionamento é uma das etapas realizadas durante a síntese física. Essa etapa é uma das tarefas realizadas no eixo de implementação do CI com o fluxo de projeto digital. Mais especificamente, o algoritmo proposto neste trabalho tem por objetivo reduzir violações no tempo de propagação de *late* dos sinais de dados na etapa de posicionamento detalhado. O algoritmo proposto é a adaptação de um algoritmo de quadrático da etapa de posicionamento global para tratar violações no tempo de propagação dos sinais na etapa de posicionamento detalhado. A premissa do algoritmo proposto é aproximar as portas lógicas pertencentes aos caminhos críticos com o objetivo de reduzir capacitância e resistência de cada rede do

caminho. Conseqüentemente, o tempo de propagação dos sinais é reduzido.

1.1 Fluxo de Projeto com Células Padrão para Circuitos Integrados Digitais

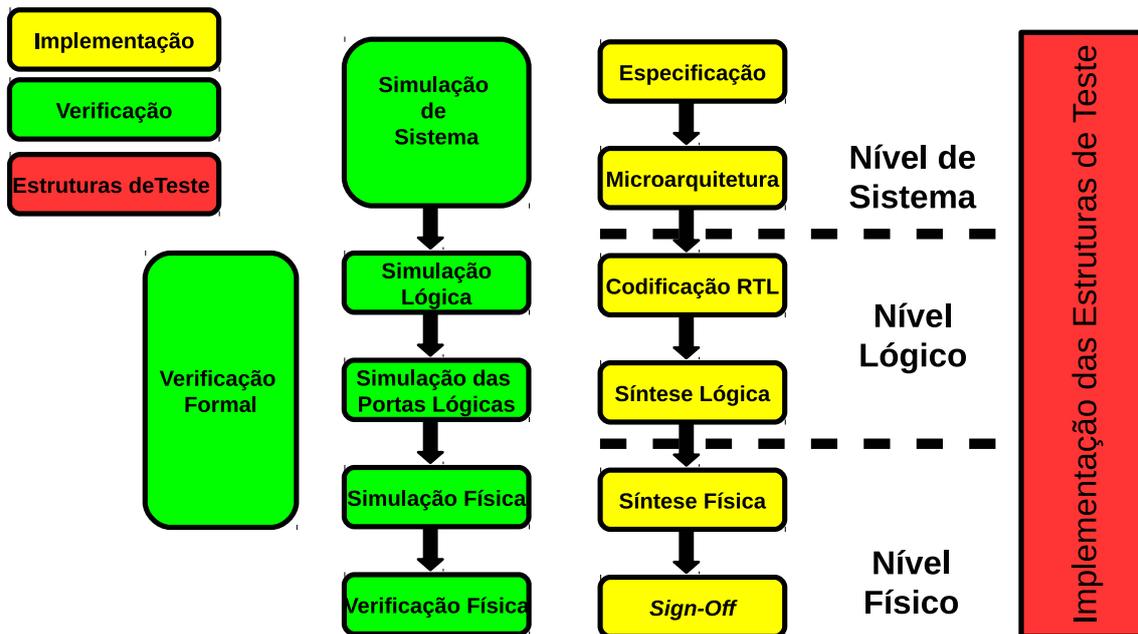
O fluxo de projeto digital é uma seqüência estruturada de passos para o projeto e implementação de CIs. Os principais passos desse fluxo são: a especificação do CI, a descrição dele com uma Linguagem de Descrição de Hardware (HDL), os processos de síntese e validação quanto ao seu funcionamento lógico e elétrico. Ao final desse fluxo, a descrição do *layout* do circuito é gerado e enviado para ser fabricado. No fluxo digital, um CI é sintetizado utilizando as células de um biblioteca de células. As características elétricas dessas células já estão previamente mensuradas. As etapas do fluxo digital tem como objetivo sintetizar os requisitos do CI especificados no primeiro passo em um *layout*.

As ações no fluxo digital são divididas em três eixos, conforme apresentado na Figura 1.1. O primeiro eixo contempla a implementação das estruturas de testes, o segundo a implementação das funcionalidades lógicas e o terceiro a simulação e verificação do circuito. O eixo de implementação é dividido em três níveis: de sistema, lógico e físico. No nível de sistema, os requisitos e as restrições de um CI são documentados em formato textual na etapa de especificação e é projetada a micro arquitetura na etapa seguinte. A codificação em *Register-Transfer Level* (RTL) sintetizável e a síntese lógica são as tarefas realizadas no nível lógico. No nível físico são realizadas as etapas de síntese física e de *sign-off*. Na etapa de síntese física são realizados os processos de definição da planta baixa (*floorplanning*), posicionamento das portas lógicas, roteamento da árvore de relógio e roteamento dos sinais de dados. Na etapa de *sign-off* as características elétricas são extraídas e o seu funcionamento elétrico é validado com simulação elétrica.

A descrição em HDL do circuito é otimizada e mapeada para um conjunto de portas lógicas no nível de síntese lógica. Em (KAHNG, 2011), cada porta lógica é definida como uma instância de uma célula da biblioteca e a célula é definida como a implementação física com transistores de uma função booleana. No nível de síntese física, as dimensões e a forma do circuito são definidas em uma planta baixa (*floorplanning*), a posição final das portas lógicas é determinada na etapa de posicionamento, os sinais de relógio e de dados são roteados, respectivamente, nas etapas de roteamento da árvore de relógio e na etapa de roteamento e as características elétricas do CI são mensuradas na etapa de *sign-off*.

O posicionamento das portas lógicas é a segunda operação realizada na síntese física, como apresentado na Figura 1.2. Essa operação é realizada após o planejamento topológico e antes da síntese da árvore de relógio (*Clock-Tree Synthesis* (CTS)). Na etapa de posicionamento, as portas lógicas são espalhadas pela área reservada para o circuito. A

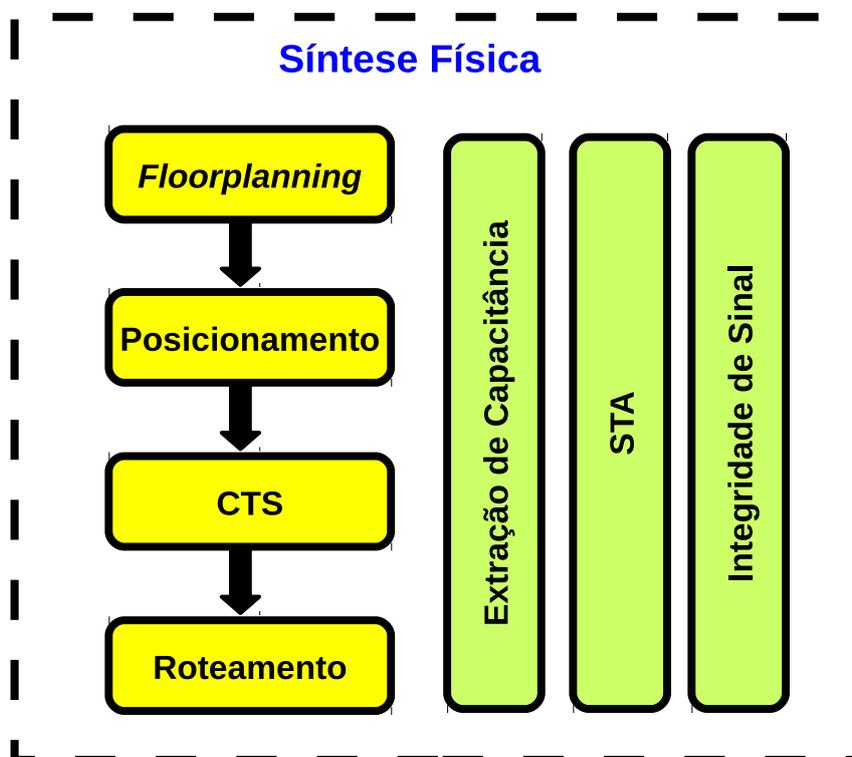
Figura 1.1 - Diagrama do fluxo de projeto digital



Fonte: figura elaborada pelo autor.

meta nessa etapa é determinar a posição mais adequada de cada porta lógica no *layout* do circuito de modo que os objetivos sejam otimizados sem violar as restrições.

Figura 1.2 - Diagrama das etapas da síntese física

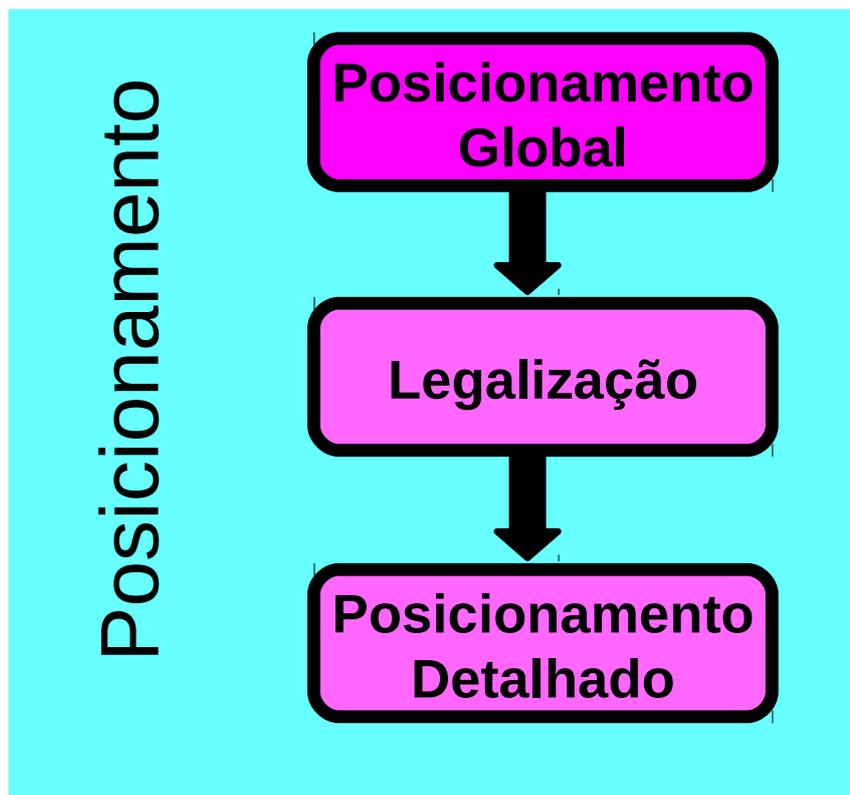


Fonte: figura elaborada pelo autor.

A etapa de posicionamento é frequentemente dividida em três estágios: posiciona-

mento global, legalização e posicionamento detalhado (KAHNG et al., 2011a), como apresentado na Figura 1.3. A divisão dessa etapa em três estágios deve-se a alguns objetivos e restrições específicos para cada um deles. Algumas das restrições podem ser relaxadas em um determinado estágio da etapa de posicionamento. Por exemplo, no estágio de posicionamento global, a restrição de sobreposição entre as portas lógicas é relaxada, então os algoritmos para esse estágio podem focar em otimizar a posição relativa entre as portas lógicas e não precisam dispensar tempo em legalizar o circuito a cada iteração, legalização essa que será violada na próxima iteração do algoritmo posicionamento global.

Figura 1.3 - Estágios da etapa de posicionamento



Fonte: figura elaborada pelo autor.

No estágio de posicionamento global, as portas lógicas são espalhadas dentro da área estabelecida para o circuito. Em seguida, no estágio de legalização, a sobreposição residual entre as portas lógicas do posicionamento global é removida. Na legalização deve-se modificar o mínimo possível a solução de posicionamento. No estágio seguinte, de posicionamento detalhado, as portas lógicas são movimentadas para otimizar os objetivos, mas com restrições mais severas para aceitar troca de posição entre as portas lógicas (CHAN et al., 2007).

1.2 Etapa de Posicionamento de Portas Lógicas

O posicionamento de portas lógicas em um circuito integrado continua sendo um dos principais desafios da síntese física. O problema de posicionamento genérico pertence à classe NP-Completa (SHERWANI, 1999). Portanto, os algoritmos de posicionamento utilizam intensamente heurísticas para obter soluções adequadas em um tempo computacional limitado. Por outro lado, há a necessidade de aumentar a escalabilidade devido ao aumento no número de portas lógicas ao desenvolver novos algoritmos de posicionamento.

A meta na etapa de posicionamento é obter uma posição única para cada porta lógica. Essa posição é aquela em que os objetivos são otimizados sujeitos às restrições. Por exemplo, os objetivos podem ser definidos como a minimização do comprimento de fio e as restrições podem ser definidas com as regras de projeto e elas podem ser mapeadas para densidade de utilização de área do circuito.

O objetivo de minimizar o comprimento de fio pode ser relacionado com a redução do tempo de propagação dos sinais, redução da capacitância e da resistência dos fios, etc. A restrição de densidade de utilização de área do circuito pode ser relacionada com violações das regras de projeto, problemas de congestionamento, regiões com superaquecimento, etc. Desse modo, ao minimizar o comprimento de fio deve-se evitar a violação de densidade de utilização de área. Caso contrário, o circuito pode não funcionar corretamente devido às violações que não foram mitigadas durante o posicionamento

A etapa de posicionamento tem um papel fundamental na qualidade de um circuito (SPINDLER; SCHLICHTMANN; JOHANNES, 2008a). Um posicionamento de baixa qualidade pode inviabilizar que as etapas seguintes obtenham soluções na qualidade estipulada. Portanto, um bom gerenciamento entre objetivos e restrições, quando estes são conflitantes, é primordial para obter um posicionamento de boa qualidade em um circuito.

Os algoritmos de posicionamento recebem como entrada uma lista de portas lógicas, uma lista de conexões e um conjunto de terminais. Uma rede é definida como a conexão lógica em comum de um subconjunto de pinos de portas lógicas. No posicionamento, um circuito é definido como um conjunto de redes (*netlist*) e um conjunto de portas lógicas (CONG; NAM, 2007). Os pinos são as interfaces para ligar as portas lógicas com as redes. Eles são os pontos lógicos para ter acesso à função lógica de uma porta. A rede é o mecanismo para transmitir os dados (propagação dos sinais) entre diferentes portas lógicas.

As primeiras ferramentas de posicionamento surgiram nos anos de 1960 para participar *netlist*. Nos anos de 1970/1980 surgiram os primeiros algoritmos de posicionamento analítico. Porém, os resultados gerados por essa técnica eram inferiores aos obtidos pelos algoritmos estocásticos baseados em *Simulated-Annealing* (SA). Esses algoritmos obtiveram resultados satisfatórios até meados dos anos de 1990, quando o problema de

escalabilidade tornou-se recorrente e de difícil solução (MARKOV; HU; KIM, 2012).

Os posicionadores implementados com os algoritmos de SA utilizam-se da simulação do resfriamento gradual do aço (têmpera). Nesse algoritmo a troca de posição das portas lógicas é inicializada em uma temperatura muito alta, ou seja, as portas lógicas trocam de posição em uma taxa muito elevada. Conforme, gradualmente a temperatura é reduzida, a taxa de troca de posição das portas lógicas é proporcionalmente diminuída. A temperatura inicial e a velocidade de resfriamento influenciam diretamente a qualidade do resultado do posicionamento.

No ano de 1975 QUINN JR. propôs um posicionador baseado em forças da mecânica clássica (SHERWANI, 1999). Contudo, essa técnica de posicionamento ficou relegada a segundo plano até por volta de 2005. Esse ano marca o retorno competitivo dos algoritmos de posicionamento com objetivos quadráticos e baseados em sistemas de forças. A ferramenta FastPlace (VISWANATHAN; CHU, 2004) foi a que marcou a mudança de paradigma para os algoritmos de posicionamento. As técnicas baseadas em sistemas de forças maturaram a ponto de obterem resultados melhores e mais rápidos do que os algoritmos de SA e serem menos propícias a problemas de escalabilidade (MARKOV; HU; KIM, 2012).

A técnica de posicionamento baseada em sistemas de forças explora a similaridade entre o problema de posicionamento e o problema da mecânica clássica de um conjunto de corpos conectados por molas (SHERWANI, 1999). Nessa técnica, as portas lógicas móveis que estão interligadas são atraídas mutuamente pela força aplicada em cada conexão. Portanto, no posicionamento final as portas lógicas estarão em uma posição na qual as forças de atração de cada conexão estejam equilibradas (SHERWANI, 1999). Ao analisar a perspectiva do sistema de molas, a posição final das portas lógicas é aquela em que o sistema de molas possui a mínima energia total.

1.3 Objetivos deste Trabalho

O objetivo desse trabalho é implementar um algoritmo de posicionamento analítico detalhado focado em caminhos com violações no tempo de propagação de *late*. Esse algoritmo é a adaptação de um algoritmo quadrático da etapa de posicionamento global para a etapa de posicionamento detalhado.

Dessa forma, os objetivos em detalhes são:

- Revisão dos algoritmos analíticos estado-do-arte para posicionamento global.
- Revisão dos algoritmos estado-da-arte para posicionamento detalhado.
- Implementação de um algoritmo de posicionamento analítico para a etapa de posicionamento detalhado com o objetivo de minimizar as violações no tempo de propagação.

1.4 Organização desta Dissertação

Esta dissertação está organizada em 6 Capítulos. A seguir é apresentada uma breve descrição de cada um deles.

Os principais conceitos sobre a etapa de posicionamento são apresentados no Capítulo 2. A definição formal de posicionamento, as métricas para estimar o comprimento de fio, a densidade de utilização de área, o congestionamento de roteamento e a temporização dos sinais são abordados nesse capítulo. Também são discutidos os estágios de posicionamento global, legalização e posicionamento detalhado da etapa de posicionamento, os modelos de conexões de redes para posicionadores analíticos e a descrição das principais técnicas de posicionamento.

Os principais algoritmos de posicionamento global, de legalização e de posicionamento detalhado são apresentados no Capítulo 3.

A principal contribuição deste trabalho, o algoritmo de posicionamento analítico-detalhado, é apresentada no Capítulo 4. Esse algoritmo é a adaptação de um algoritmo de posicionamento analítico quadrático do estágio global para mover as portas lógicas de um conjunto de caminhos críticos aplicado ao posicionamento detalhado.

No Capítulo 5 são discutidos os principais resultados dos experimentos realizados com o algoritmo de posicionamento analítico detalhado.

As principais conclusões e os trabalhos futuros são apresentados no Capítulo 6

2 BASE CONCEITUAL

Neste capítulo serão apresentados os principais conceitos sobre a etapa de posicionamento. Esses conceitos são amplamente utilizados no desenvolvimento de algoritmos de posicionamento.

2.1 Definição Formal do Problema de Posicionamento de Portas Lógicas em Circuitos Integrados

O posicionamento de um circuito pode ser expresso como um problema de otimização em um hiper grafo $H = (G, N)$. Seja $G = (g_1, g_2, g_3, \dots, g_p)$ um conjunto de portas lógicas, $N = (n_1, n_2, n_3, \dots, n_q)$ um conjunto de redes e R uma região retangular em que todas as portas lógicas $g_i \in G$, $1 \leq i \leq p$, devem ser posicionadas (CHAN et al., 2007).

2.2 Métricas de Avaliação da Qualidade do Posicionamento

As métricas são um meio de avaliar a qualidade de um posicionamento das portas lógicas em um CI digital. Elas também são utilizadas como um guia para ajustar os parâmetros de convergência nos algoritmos.

2.2.1 Comprimento de Fio

A qualidade de um circuito digital posicionado pode ser medida pela velocidade de processamento, pelo consumo de energia e pela roteabilidade. Contudo, medir esses objetivos não é trivial. Dessa forma, o comprimento de fio é amplamente aceito como uma primeira métrica para avaliar a qualidade de um circuito durante a etapa de posicionamento (CONG; LUO, 2010). A velocidade e a precisão em estimar a métrica de comprimento de fio são dois importantes fatores a serem considerados durante o posicionamento. Assim, diversos modelos de comprimento de fio são utilizados durante o posicionamento das portas lógicas. Estes relacionam o comprimento de fio com as métricas para avaliar a qualidade do posicionamento do circuito.

Determinadas redes podem ser priorizadas em detrimento de outras através do com-

primento de fio. Elas são diferenciadas com o uso de pesos (multiplicadores) atribuídos ao seu comprimento de fio. Os critérios para priorizar uma rede podem ser em função de elas estarem em uma região do circuito em que há congestionamento, violações no tempo de propagação dos sinais, pontos de aquecimento (*hot spot*), etc. Para um determinado posicionamento P , a estimativa de comprimento de fio com pesos diferenciados para determinadas redes é definida como

$$L(P) = \sum_{rede \in P} w(rede) \times L(rede) \quad (2.1)$$

onde $w(rede)$ é o peso da rede, multiplicador, e $L(rede)$ é a estimativa de comprimento de fio da mesma (KAHNG et al., 2011b).

A maioria dos posicionadores utilizam a distância Manhattan, Equação 2.2, para estimar o comprimento de fio em redes com dois pinos. Onde (x_i, y_i) e (x_j, y_j) são as posições opostas de um retângulo. O comprimento de fio em redes com múltiplos pinos é estimado com um modelo de comprimento de fio (KAHNG et al., 2011b). Alguns desses modelos de comprimento de fio são discutidos a seguir.

$$L(rede) = \sum |x_i - x_j| + |y_i - y_j| \quad (2.2)$$

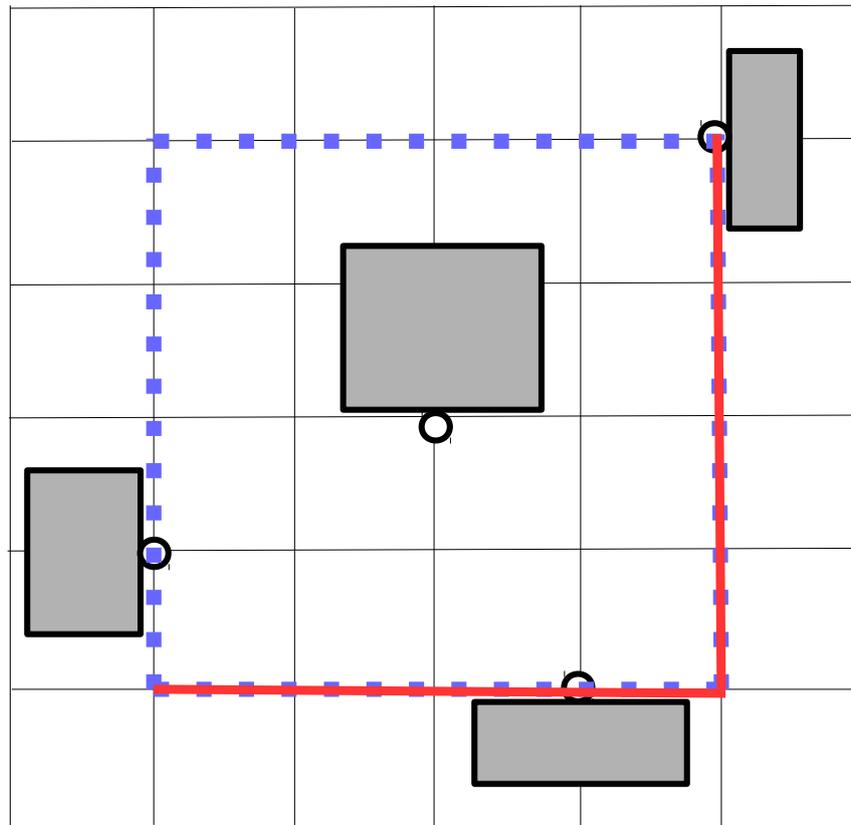
A Equação 2.2 não é derivável. Desse modo, obter um posicionamento adequado com técnicas analíticas utilizando diretamente essa equação como objetivo é praticamente inviável. Os métodos numéricos utilizados para resolver o posicionamento analítico dependem de funções objetivos que sejam continuamente diferenciáveis (CHAN et al., 2007). Contudo, para algumas técnicas de posicionamento, como as estocásticas, essa equação pode ser utilizada, pois para algumas técnicas de posicionamento não é requerido que as funções a serem otimizadas sejam contínuas.

2.2.1.1 Estimativa de Comprimento de Fio com a Métrica de Half-Perimeter Wire-Length

O modelo de *Half-Perimeter Wire-Length* (HPWL), Figura 2.1, para estimar o comprimento de fio é um dos métodos mais utilizado por causa da razoável precisão e o seu comprimento ser calculado de modo rápido e fácil. O comprimento de fio nesse modelo é obtido ao computar o semi-perímetro de um retângulo. Os limites do retângulo, para cada dimensão, são definidos pelos dois pinos com a maior distância entre todos aqueles pertencentes à rede, de modo que todos os pinos da rede sejam internos ou estejam nas bordas do retângulo (KAHNG et al., 2011b).

Dado os vetores \vec{X} e \vec{Y} com as posições dos pinos de uma rede $n \in N$ para, respectivamente, a abscissa e ordenada, o HPWL é obtido ao resolver a Equação 2.3. Onde i, j, k e

Figura 2.1 - Modelo de comprimento de fio HPWL



Fonte: KAHNG et al. (2011b, p. 97).

l são pinos da rede n .

$$\text{HPWL} = \sum_{n \in N} (\max_{i,j} |x_i - x_j| + \max_{k,l} |y_k - y_l|) \quad (2.3)$$

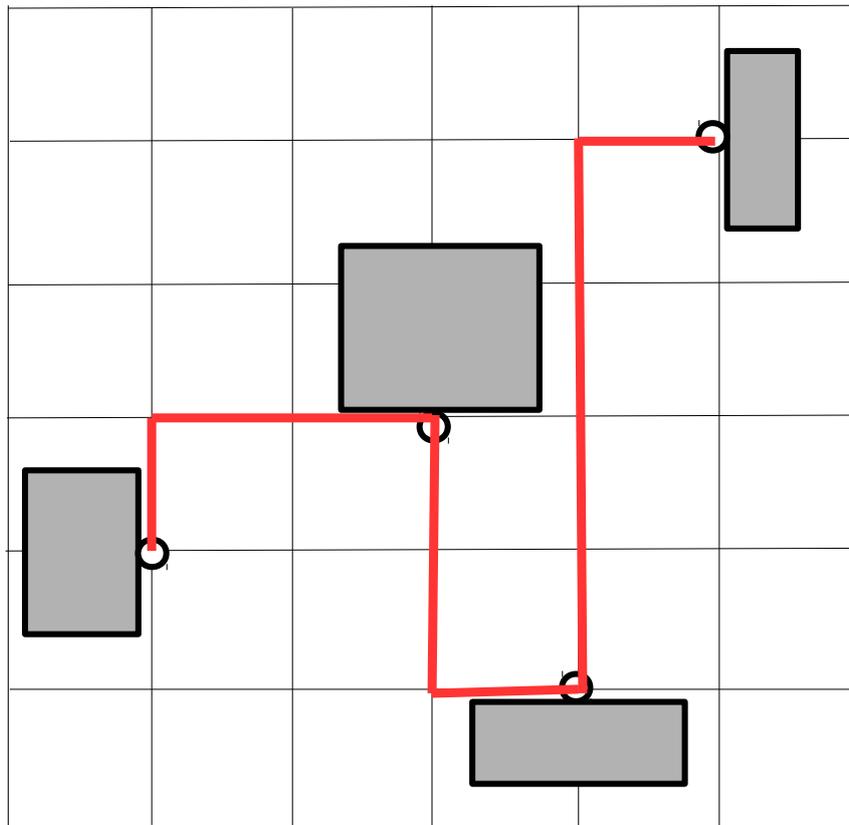
2.2.1.2 Estimativa de Comprimento de Fio com a Métrica de Caminho Monotônico

O modelo de caminho monotônico conecta em sequência todos os pinos de uma rede, conforme apresentado na Figura 2.2. Cada pino intermediário tem grau dois e os pinos nos extremos tem grau um. Encontrar o caminho monotônico com o menor comprimento de fio é um problema de busca de caminho hamiltoniano e este problema é NP-Difícil. Esse método é pessimista em relação ao comprimento de fio real e o caminho é alterado sempre que a posição das portas lógicas é modificada (KAHNG et al., 2011b).

2.2.1.3 Estimativa de Comprimento de Fio com a Métrica Clique

Todos os pinos de uma rede estão conectados entre si no modelo de comprimento de fio clique, conforme apresentado na Figura 2.3. Nesse modelo cada pino tem $p - 1$ conexões. O comprimento total de fio para o modelo clique é obtido ao resolver a Equação

Figura 2.2 - Modelo de comprimento de fio Monotônico



Fonte: KAHNG et al. (2011b, p. 98).

2.4.

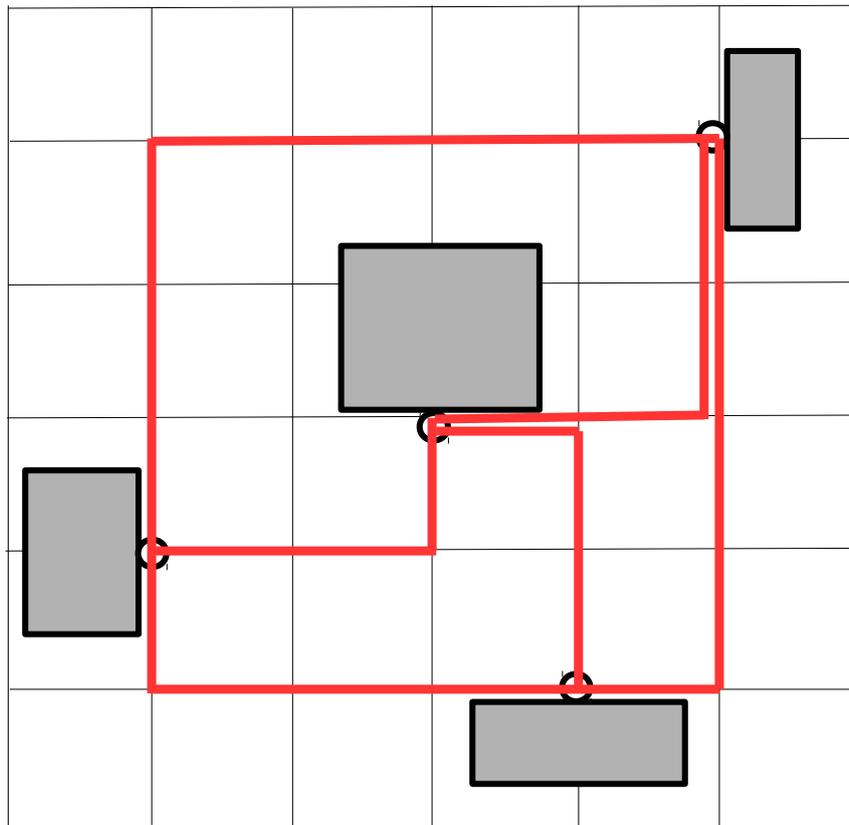
$$L(\text{rede}) = \frac{2}{p} \sum_{e \in \text{clique}} d_m(e) \quad (2.4)$$

Onde e é uma aresta no modelo clique e $d_m(e)$ é a distância Manhattan entre os pinos da aresta e (KAHNG et al., 2011b).

2.2.1.4 Estimativa de Comprimento de Fio com a Métrica Estrela

O modelo de comprimento de fio estrela, Figura 2.4, considera um nodo da rede como fonte e os demais como destino. Portanto, para cada pino destino há uma conexão (aresta) a partir do nodo fonte. Esse modelo é especialmente útil para otimizar a temporização, pois é possível capturar a propagação de um sinal a partir do pino de saída para um ou mais pinos de entrada. Nesse modelo são necessárias somente $p - 1$ arestas. O número reduzido de arestas pode ser vantajoso para modelar redes com elevado número de pinos. Por outro lado, o comprimento de fio é super estimado com esse modelo (KAHNG et al., 2011b).

Figura 2.3 - Modelo de comprimento de fio Clique



Fonte: KAHNG et al. (2011b, p. 98).

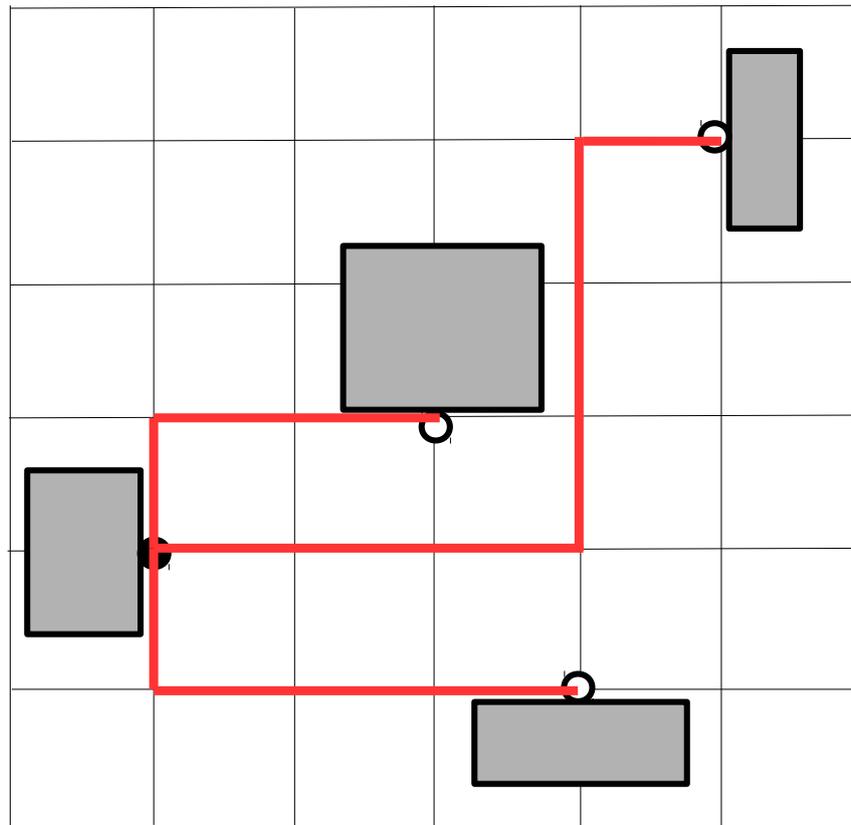
2.2.1.5 Estimativa de Comprimento de Fio com a Métrica de Árvores de Steiner

O modelo de comprimento de fio com árvores de Steiner pode ser utilizado para obter topologias reais de roteamento. Esse modelo é frequentemente utilizado durante o posicionamento para estimar o roteamento do circuito. Nesse caso, as estimativas de congestionamento, de tempo de propagação dos sinais, de comprimento de fio, etc. são mais precisas em comparação com as estimativas obtidas pelos demais modelos de comprimento de fio.

O modelo de comprimento de fio *Rectilinear Minimum Spanning Tree* (RMST), conforme apresentado na Figura 2.5, decompõe uma rede com k -pinos em conexões entre dois pinos. Os k pinos são ligados por $k - 1$ conexões. Os algoritmos de RMST podem explorar a geometria Manhattan para limitar a complexidade em tempo de execução a $O(k \log k)$ (KAHNG et al., 2011b).

No modelo de comprimento de fio *Rectilinear Steiner Minimum Tree* (RSMT) todos os pinos em uma rede estão conectados e eles ainda podem ser ligados com até $k - 2$ pontos de Steiner, conforme apresentado na Figura 2.6. Encontrar um conjunto ótimo de pontos de Steiner para um conjunto arbitrário de pinos é NP-Difícil. Se os pontos de Steiner são conhecidos, uma RSMT pode ser obtida ao construir uma RMST sobre a união dos pontos originais com os pontos de Steiner (KAHNG et al., 2011b).

Figura 2.4 - Modelo de comprimento de fio Estrela



Fonte: KAHNG et al. (2011b, p. 98).

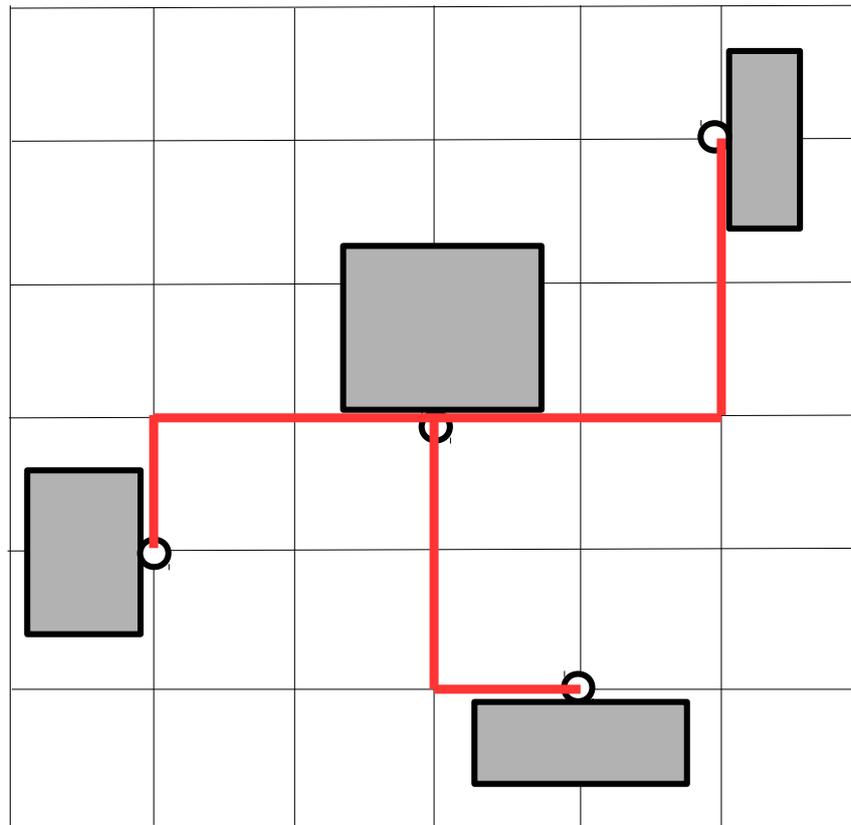
Uma rede de k pinos no modelo de *Rectilinear Steiner Arborescence* (RSA), Figura 2.7, é uma árvore onde um nodo fonte s_0 é conectado aos $k - 1$ nodos alvos s_i , $1 \leq i < p - 1$, deve ter a mesma distância de semi-perímetro. Computar o mínimo comprimento do RSA é NP-Difícil (KAHNG et al., 2011b).

O modelo de *Single-Trunk Steiner Tree* (STST), Figura 2.8, consiste em criar um segmento vertical ou horizontal (*trunk*) e conectar todos os pinos nesse segmento. As STSTs são frequentemente utilizadas com o propósito de estimativa, devido a sua fácil construção (KAHNG et al., 2011b).

2.2.2 Métricas de Estimativa de Densidade de Utilização de Área do Circuito

As métricas de densidade de utilização de área do circuito provêm um modo para medir o espalhamento das portas lógicas durante os passos intermediários do posicionamento. A densidade de utilização de área do circuito também possibilita estimar com melhor precisão o impacto do comprimento de fio após a legalização do circuito (KIM et al., 2012). Por outro lado, a densidade de utilização de área do circuito é uma maneira aproximada de limitar a sobreposição entre as portas lógicas. Muitos algoritmos de posicionamento utilizam um limite máximo de ocupação para cada *bin* (CHAN et al.,

Figura 2.5 - Modelo de comprimento de fio RMST



Fonte: KAHNG et al. (2011b, p. 98).

2007).

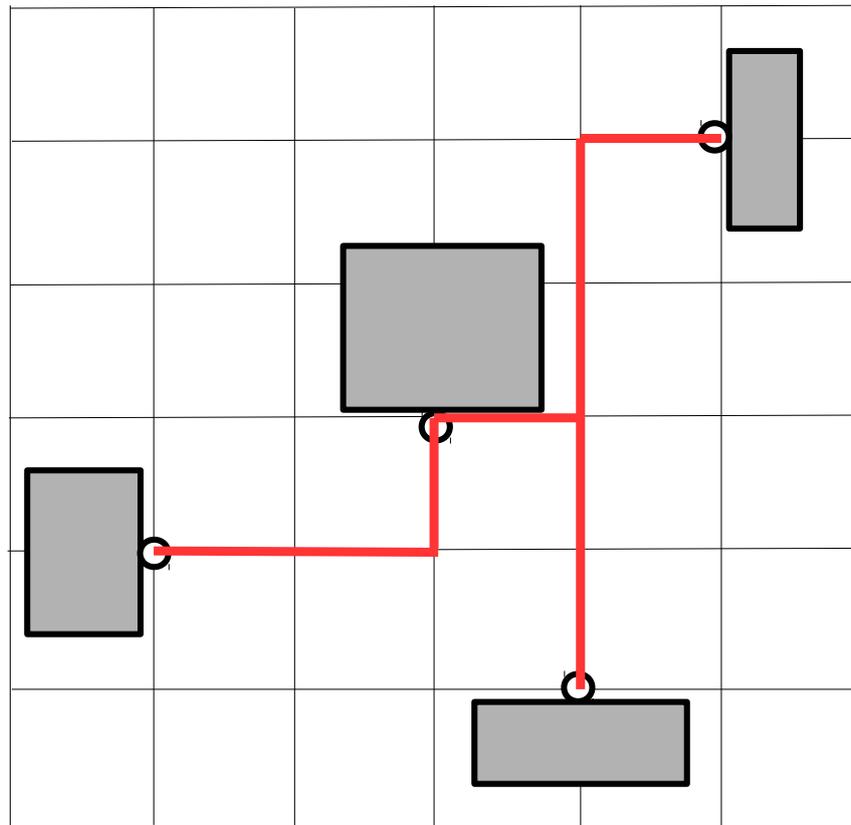
2.2.2.1 Estimativa de Densidade Utilização com a Métrica de Average Bin Utilization (ABU)

No posicionamento, a área para alocar as portas lógicas é decomposta em k ($k = n \times n$) quadrados (*bins*) com a mesma dimensão. A densidade não pode ultrapassar o limite estabelecido para o circuito em cada *bin* (LU et al., 2014). Esse valor é definido experimentalmente.

A métrica de densidade ABU considera os $\gamma\%$ *bins* mais densamente ocupados, exceto aqueles preenchidos totalmente por macro blocos. Essa métrica reflete a distribuição não uniforme das portas lógicas. Um *bin* no ABU é empiricamente definido com um quadrado com lado de tamanho seis vezes a altura padrão das portas lógicas (KIM et al., 2012).

Usualmente o ABU é computado para os *bins* com $\gamma\%$ com a maior taxa de utilização de área. Estes são experimentalmente definidos com 2%, 5%, 10% e 20% dos *bins* com a maior taxa utilização de área do circuito. A penalidade de violação de densidade é computada como a média ponderada da média de densidade para cada $\gamma\%$ *bin*. A média para cada $\gamma\%$ *bin* é considerada para o computo da penalidade de violação de ABU somente quando ela é superior a densidade máxima de utilização estabelecida (POPOVYCH et al.,

Figura 2.6 - Modelo de comprimento de fio RSMT



Fonte: KAHNG et al. (2011b, p. 99).

2014).

Na Equação 2.5 é apresentada a maneira de computar o ABU para cada $\gamma\%$ bin. O ABU_{γ} é a média aritmética simples da utilização de todos os $\gamma\%$ bins.

$$ABU_{\gamma} = \max\left(0, \frac{ABU_{\gamma\%}}{targetDensity} - 1\right) \quad (2.5)$$

A penalidade de violação de densidade para a métrica ABU é computada ao resolver a Equação 2.6. Os pesos, multiplicadores, para cada $ABU_{\gamma\%}$ são definidos experimentalmente.

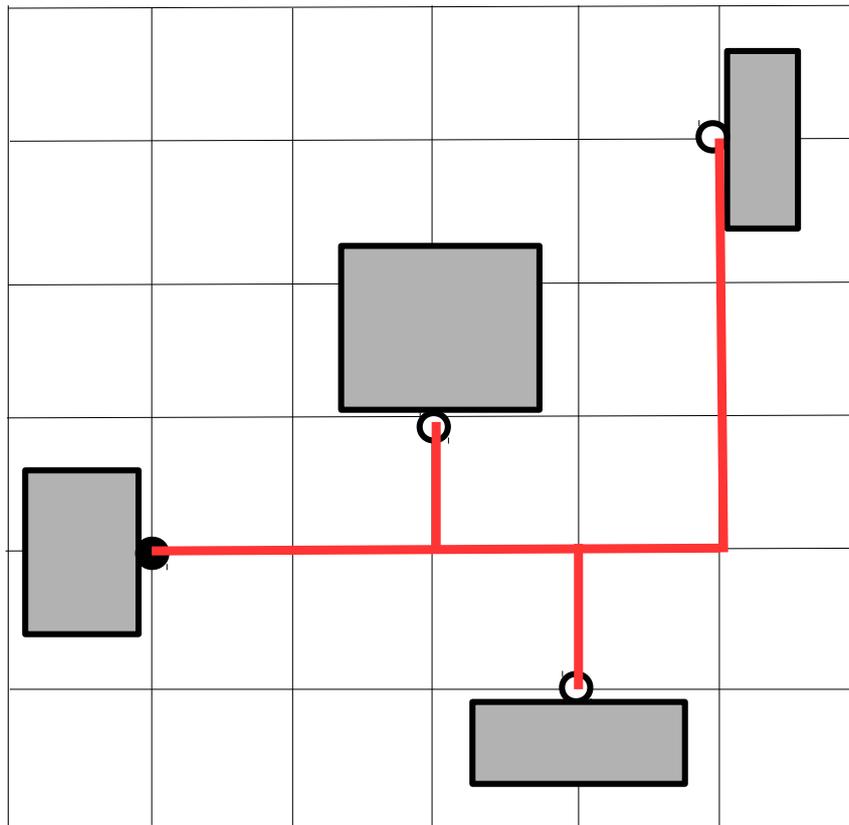
$$ABU_{penalty} = \frac{10 \times ABU_{2\%} + 4 \times ABU_{5\%} + 2 \times ABU_{10\%} + 1 \times ABU_{20\%}}{10 + 4 + 2 + 1} \quad (2.6)$$

2.2.2.2 Estimativa de Densidade de Utilização por Região

A função de densidade por bin não é diferenciável, o que torna difícil de resolvê-la com técnicas matemáticas convencionais. Portanto, é necessário obter uma função contínua para a densidade que seja facilmente resolvida com métodos numéricos (CHAN et al., 2007).

O algoritmo de posicionamento Kraftwerk2 (SPINDLER; SCHLICHTMANN; JOHAN-

Figura 2.7 - Modelo de comprimento de fio RSA



Fonte: KAHNG et al. (2011b, p. 99).

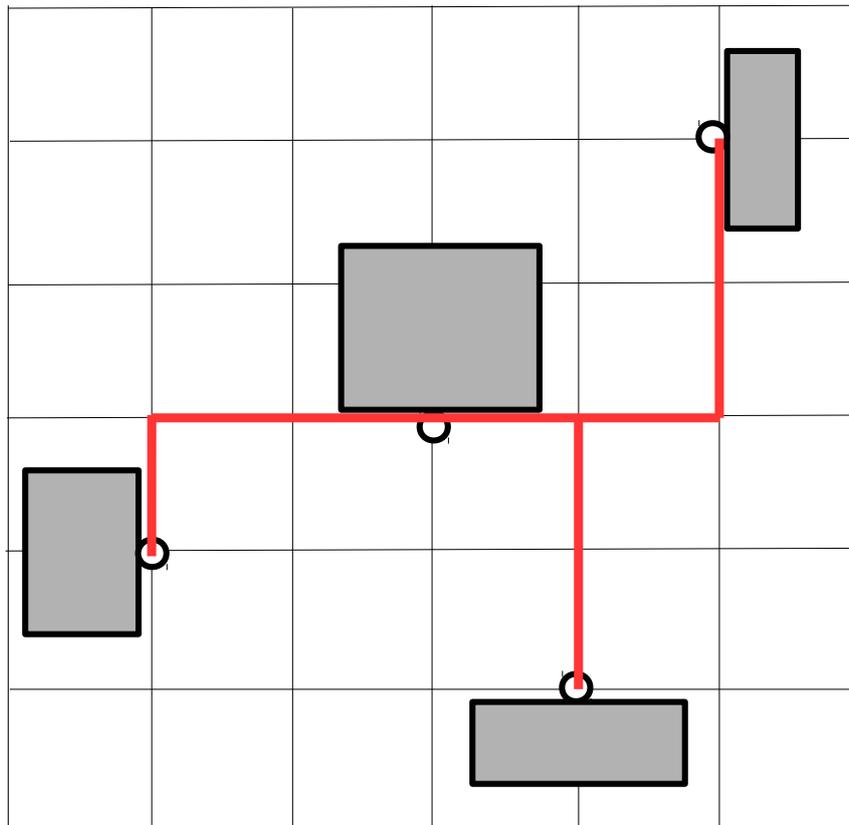
NES, 2008a) utiliza a segunda derivada de uma função potencial $u(x,y)$ em que o gradiente $\vec{F} = \nabla u$ representa a força que aponta em direção oposta às regiões congestionadas. A solução da equação de Poisson $\Delta u = p$ é utilizada para satisfazer a condição da função potencial. A solução da equação de Poisson pode ser interpretada como encontrar um potencial eletrostático u para um campo ∇u rotacional livre gerado pela distribuição p espacial de cargas (MARKOV; HU; KIM, 2012)

2.2.3 Métrica de Estimativa de Congestionamento no Circuito

O congestionamento é uma função da demanda de recursos de roteamento diante da disponibilidade dos mesmos. As características da tecnologia e do circuito, isto é, dimensão do CI, número de camadas de metais, posição dos macro blocos, etc, são fixas para a etapa de roteamento. Portanto, os recursos de roteamento podem ser estimados durante a etapa de posicionamento. A demanda de recursos de roteamento e o congestionamento estão fortemente ligados. Por causa disso, um pode ser convertido no outro (TAGHAVI et al., 2007).

No roteamento global, o congestionamento é um dos principais objetivos a serem otimizados. Uma região altamente congestionada frequentemente requer que o roteador evite-a, assim resultando no aumento no comprimento do fio. As áreas congestionadas

Figura 2.8 - Modelo de comprimento de fio STST



Fonte: KAHNG et al. (2011b, p. 99).

reduzem o desempenho do roteador global (TAGHAVI et al., 2007).

Estimar o congestionamento durante o posicionamento é de fundamental importância para reduzir esse tipo de violação nas etapas seguintes da síntese física. Se essa violação é minimizada durante o posicionamento, o roteador pode dedicar mais tempo ao objetivo de conectar os pinos. Ainda, a qualidade do resultado de roteamento tende a ser melhor. A redução de violações de roteamento durante o posicionamento ajuda a evitar o trabalho de refazer todas as etapas da síntese física, por causa das violações que não são possíveis de serem corrigidas nas etapas finais de síntese.

2.2.4 Análise de Temporização (*Timing*)

A análise estática do tempo de propagação (*Static Timing Analysis (STA)*) é uma técnica para computar o atraso resultante da propagação dos sinais. A STA é uma técnica completa e exaustiva para verificar toda a temporização de um circuito (BHASKER; CHADHA, 2009).

A STA é um método de análise do tempo de propagação que não depende da aplicação de vetores de dados sobre os terminais de entrada (BHASKER; CHADHA, 2009).

O objetivo da STA é validar se o circuito opera sem violações para as restrições temporais estabelecidas para o mesmo (BHASKER; CHADHA, 2009). Portanto, os principais

itens considerados para validar o correto funcionamento de um circuito são as restrições de temporização das interfaces do circuito com o ambiente externo e os tempos de propagação dos sinais pelas redes do circuito.

O controle de propagação dos sinais de dados em circuitos sequências é realizado com um sinal periódico denominado de relógio. Os sinais de dados são propagados através das portas lógicas combinacionais no intervalo entre duas bordas de transição do relógio que sejam consecutivas e de mesma direção. Se uma borda do relógio transicionar do nível lógico 1 (um) para o nível lógico 0 (zero), ela é dita uma borda de transição de descida ou uma borda de transição negativa. Caso contrário, ela é chamada de borda de relógio de subida ou borda de relógio positiva.

A propagação do sinal de relógio pelo circuito sofre interferência de componentes elétricos. Esses componentes usualmente são: a *Phase Lock Loop* (PLL), os *buffers*, os fios da árvore de relógio, os capacitores parasitas, os resistores, etc. Todos esses componentes agregam incerteza no momento em que uma transição de relógio ocorre.

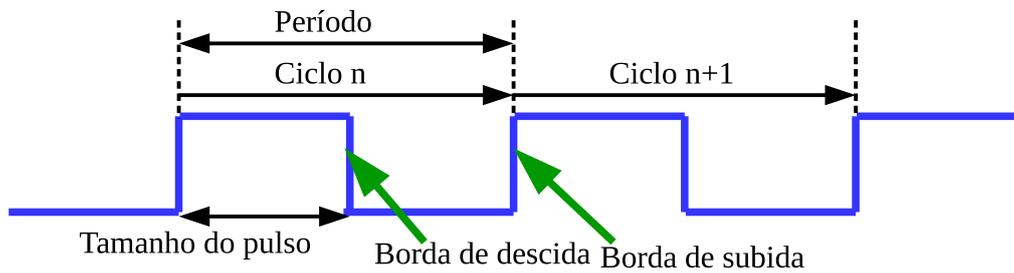
Nas etapas de síntese lógica e física de um circuito busca-se manter o tempo de incerteza para a transição do relógio com o menor intervalo de tempo possível, porém esse intervalo não é uma quantidade de tempo insignificante que possa ser ignorado. Desse modo, o tempo requerido para que os sinais de dados estejam disponíveis nas entradas dos elementos sequenciais é dependente também do tempo de incerteza em que a borda do sinal de relógio transicionar.

Os sinais de dados devem ser propagados pela lógica combinacional entre duas barreiras temporais até o limite de tempo requerido. Esse limite de tempo é definido pelo período de relógio e pela incerteza. O período de relógio é o intervalo de tempo necessário para ocorrer duas bordas de relógio do mesmo tipo. Um pulso de relógio é o intervalo de tempo em que ocorrem duas bordas de relógio em sentido contrário dentro do período de relógio. O ciclo do relógio é a repetição contínua do período de relógio. Na Figura 2.9 são apresentados os pontos iniciais e finais, a partir de uma borda de subida, em que esses parâmetros do sinal de relógio são medidos. Os parâmetros também podem ser medidos a partir da borda de descida do relógio. As transições do relógio apresentadas nessa figura são para um relógio ideal, não há prejuízo semântico quando elas são medidas para o relógio com incerteza e *slew*, apenas variação no valor medido para cada uma delas.

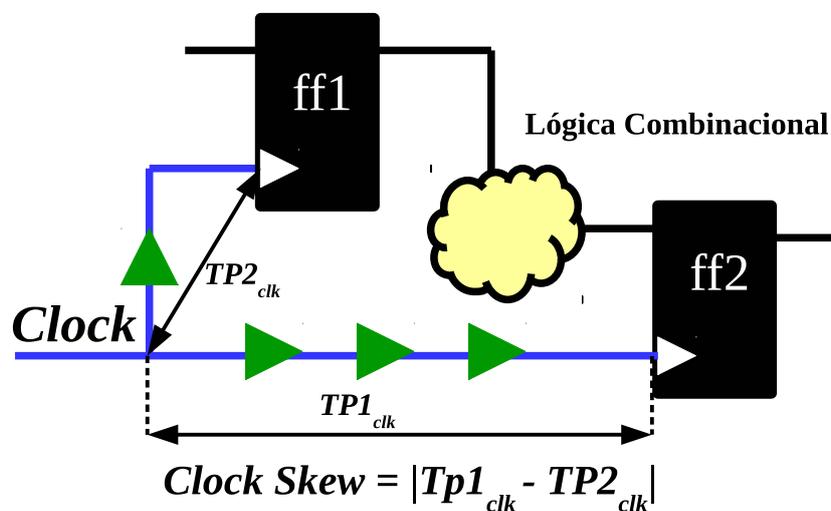
O intervalo de incerteza na transição do sinal de relógio deve-se ao *jitter* e ao *skew* do relógio. O *jitter* é o intervalo de tempo em que uma transição ocorre. Essa imprecisão deve-se à incerteza da transição do sinal de relógio no cristal e na PLL. O *skew* do relógio é a diferença máxima permitida do tempo de propagação do relógio para chegar em registradores distintos a partir do último ponto em comum na árvore de relógio. O tempo de *skew* do relógio para dois registradores é apresentado na Figura 2.10.

Os sinais de dados podem chegar às barreiras temporais fora do intervalo de tempo requerido. Nesse caso ocorrem violações no tempo de propagação dos sinais. Se os

Figura 2.9 - Sinal de temporização para circuitos sequências (Relógio)



Fonte: figura elaborada pelo autor.

Figura 2.10 - Tempo de *skew* do Relógio

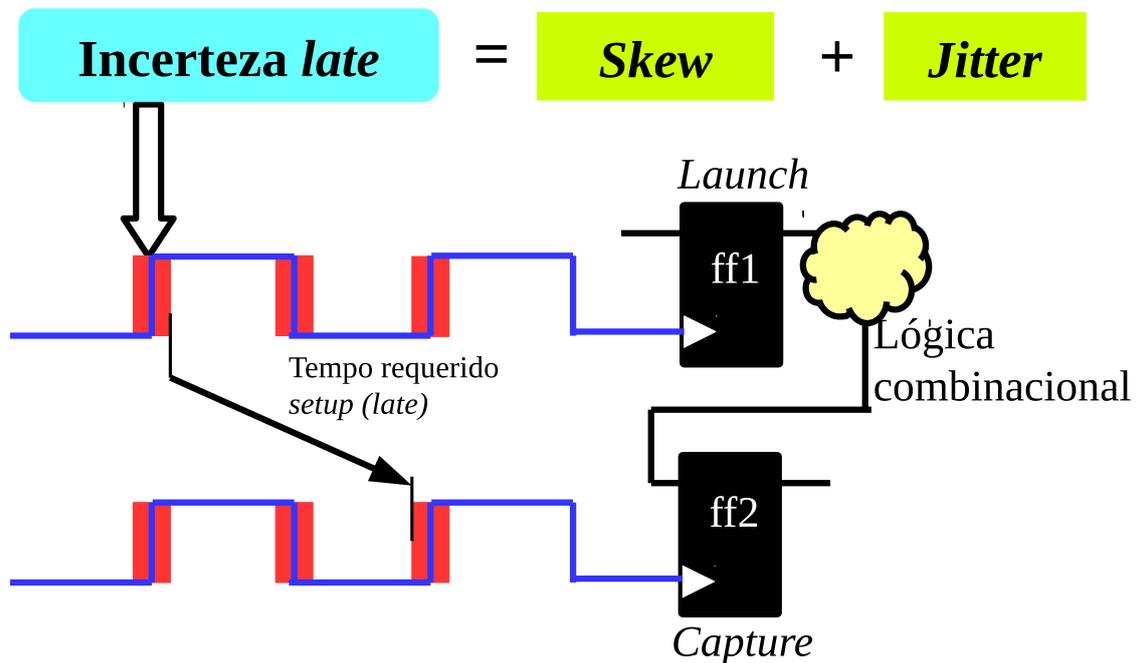
Fonte: figura elaborada pelo autor.

se os sinais de dados chegarem atrasados, eles não são armazenados no momento devido nos registradores. Caso eles cheguem antes do tempo requerido, os sinais de dados anteriores são sobrescritos. Portanto, em todas as etapas da síntese física deve-se garantir que todos os sinais de dados cheguem nas barreiras temporais dentro da janela de tempo requerida.

A incerteza é composta pela imprecisão de *jitter* e pelo *clock skew* é utilizada para obter o tempo requerido ao verificar violações de *late*. Nesse caso, o intervalo de tempo requerido para que os sinais de dados propaguem-se pela lógica combinacional é o período de relógio menos o intervalo de incerteza. Na Figura 2.11 é mostrado o limite de imprecisão para uma transição do relógio e as bordas do relógio em que é medido o tempo requerido para violações de *late*.

Os sinais de dados são lançados na borda de transição de um ciclo de relógio e capturadas na barreira temporal na borda do ciclo de relógio seguinte. Se os sinais de dados chegarem à barreira temporal seguinte antes do tempo máximo requerido não há violação de propagação de *late*. Nesse caso, a diferença entre o tempo requerido e o tempo de chegada é o *slack* positivo. Por outro lado, se os sinais chegarem após o tempo requerido há violação de propagação. Nesse caso o *slack* é negativo, diferença entre o tempo requerido

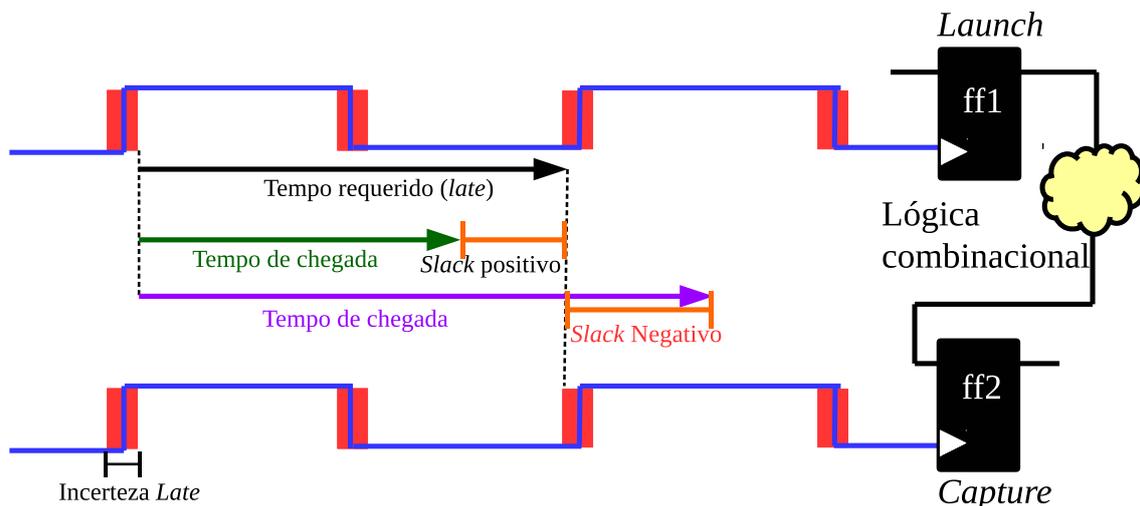
Figura 2.11 - Incerteza na transição do relógio e tempo requerido de propagação dos sinais de dados do tipo *late*



Fonte: figura elaborada pelo autor.

e o tempo de chegada.

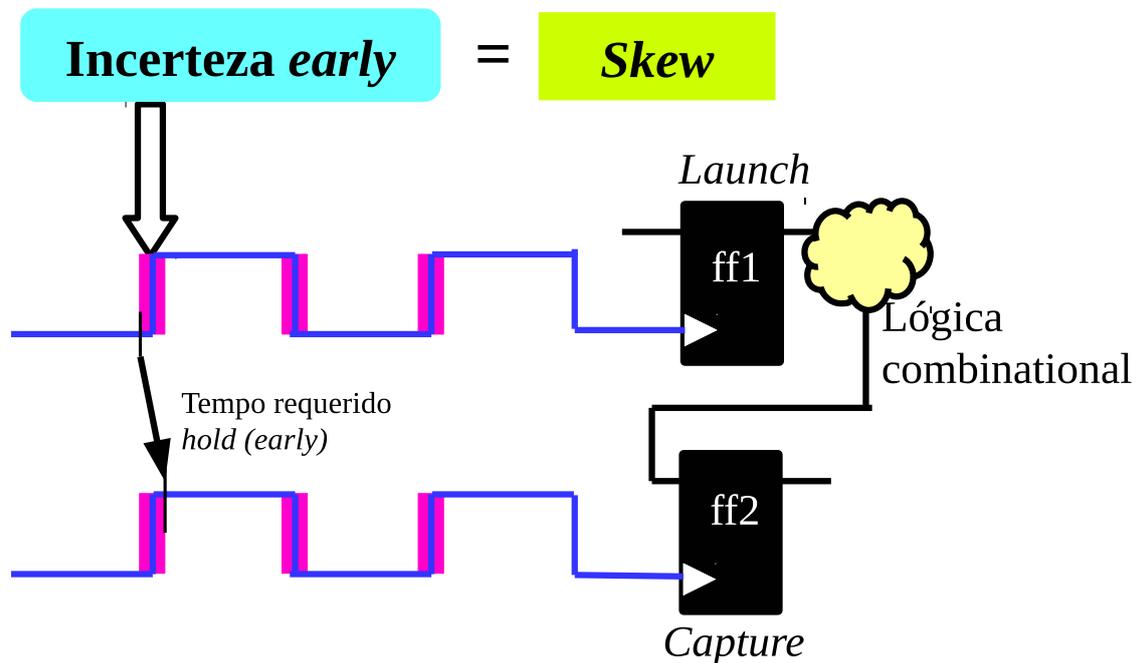
Figura 2.12 - *Slack* dos sinais de dados para violações de *late*



Fonte: figura elaborada pelo autor.

A incerteza para as violações de *early* é composta somente pelo *clock skew*. Nesse caso, não há imprecisão do *jitter*, pois a verificação de propagação para violações de *early* é medida na mesma borda de relógio para barreiras temporais diferentes. O tempo requerido para *early* é o intervalo de tempo da incerteza do mesmo. Na Figura 2.13 são apresentados os pontos de medida do tempo requerido para verificar violações de *early*.

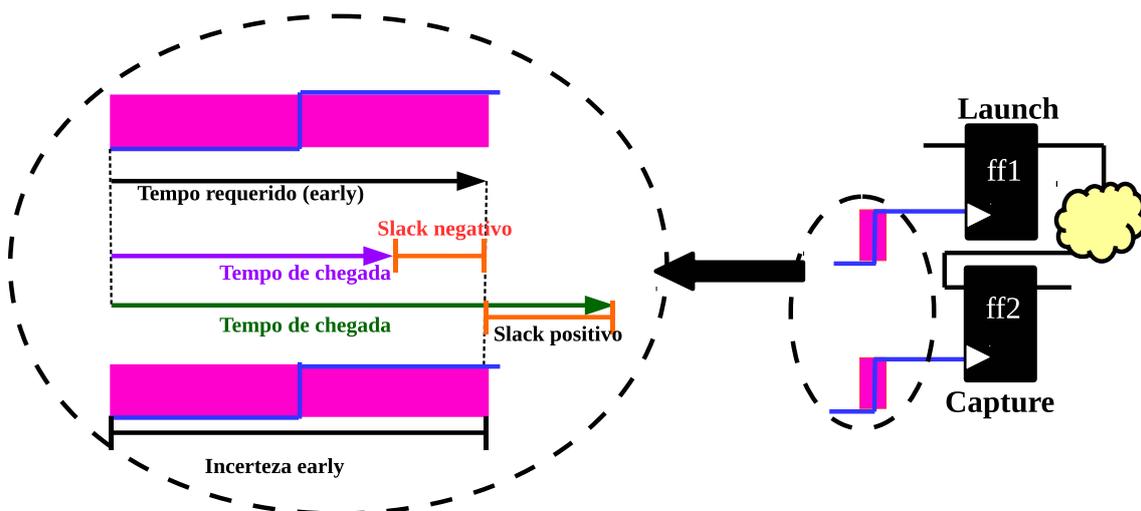
Figura 2.13 - Incerteza na transição do relógio e tempo requerido de propagação dos sinais de dados para violações de *early*



Fonte: figura elaborada pelo autor.

Se os sinais de dados chegarem na barreira temporal seguinte antes do tempo requerido ocorrerá violação de *early*. Nesse caso, os *slacks* dos sinais são negativos. Caso contrário, eles são positivos. Os *slacks* para a verificação de violações de *early* são apresentados na Figura 2.14.

Figura 2.14 - *Slack* dos sinais de dados para violações de *early*



Fonte: figura elaborada pelo autor.

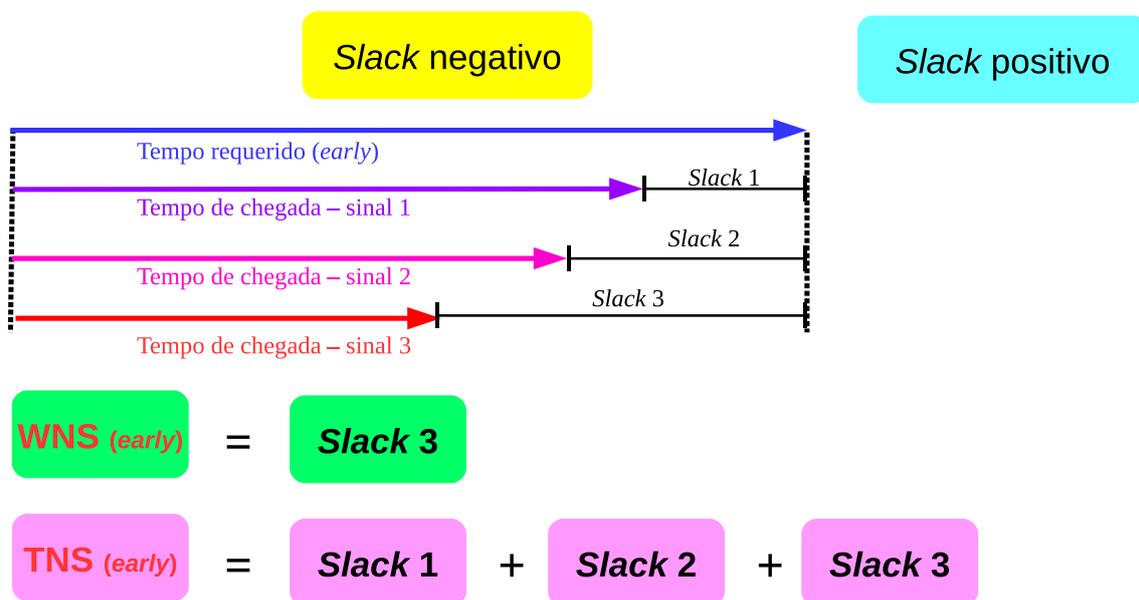
O *Total Negative Slack* (TNS) e o *Worst Negative Slack* (WNS), respectivamente, Equações 2.7 e 2.8, são computados para os caminhos com violações de *early* e *late*. O TNS para *early* é o somatório de todos os *slacks* negativos e o WNS para *early* é o *slack*

do caminho com o menor tempo de propagação em que ocorre violação. Os exemplos de TNS e WNS são apresentados na Figura 2.15 para as violações de propagação dos sinais de dados para *early*.

$$TNS = \sum_{\tau \in T, slack(\tau) < 0} slack(\tau) \quad (2.7)$$

$$WNS = \min_{\tau \in T} (slack(\tau)) \quad (2.8)$$

Figura 2.15 - TNS e WNS para os caminhos de dados com violações no tempo de propagação de *early*



Fonte: figura elaborada pelo autor.

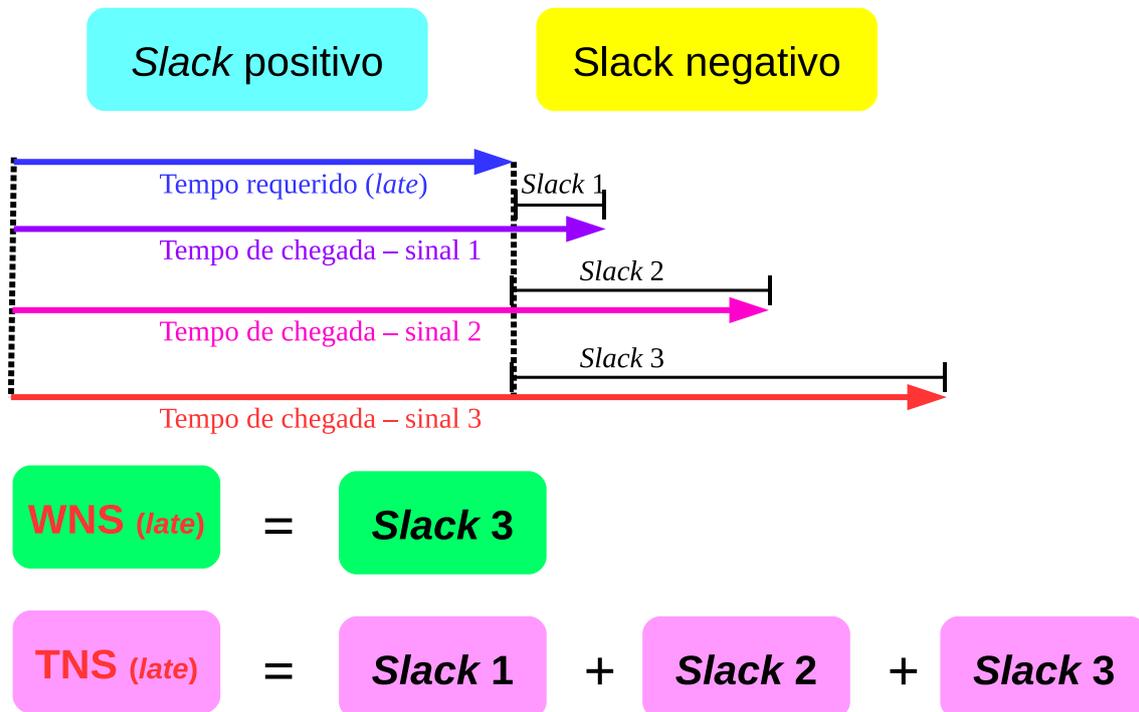
O TNS, Equação 2.7, para *late* é o somatório de todos os *slacks* negativos e o WNS, Equação 2.8, para *late* é o *slack* do caminho com o maior tempo de propagação em que ocorre violação. Os exemplos de TNS e WNS são apresentados na Figura 2.16 para as violações no tempo de propagação dos sinais de dados para *late*.

2.3 Estágios de Posicionamento

2.3.1 Posicionamento Global

A meta principal no estágio de posicionamento global é distribuir bem as portas lógicas de modo a otimizar os objetivos sem violar as restrições. Nesse estágio são consideradas todas as portas lógicas, conexões e terminais. Algumas restrições durante o posicionamento global são relaxadas, como a limitação das portas lógicas permanecerem inteiramente dentro das bandas (*rows*), a proibição de sobreposição entre as portas lógicas, entre outras restrições. No estágio de posicionamento global, a forma e a área

Figura 2.16 - TNS e WNS para os caminhos de dados com violações de propagação de *late*



Fonte: figura elaborada pelo autor.

das portas lógicas móveis não possuem grande relevância, exceto os macros blocos que usualmente são mantidos em posições fixas (KAHNG et al., 2011b; CHEN et al., 2007).

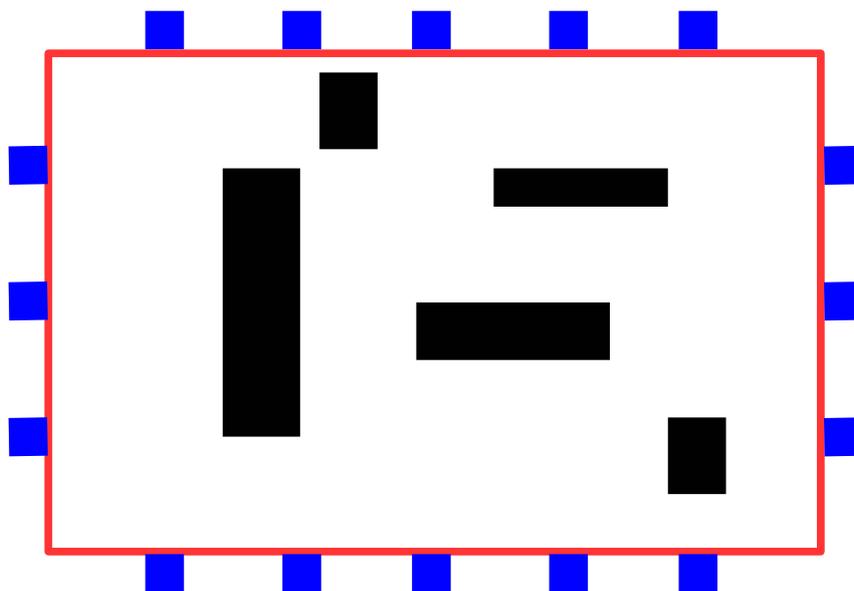
As portas lógicas móveis são aquelas que podem ser posicionadas em qualquer área disponível dentro do espaço estipulado para o circuito. A movimentação delas é permitida dentro dos limites da área do circuito. A posição mais adequada para essas portas são obtidas pelos algoritmos de posicionamento. As portas lógicas fixas são aquelas que a posição está predeterminada antes da etapa de posicionamento ser realizada e os algoritmos de posicionamento estão proibidos de movê-las para uma nova posição. Os pinos são elementos lógicos posicionados nas bordas da área do circuito e são utilizados para passagem dos sinais de dados do circuito para o mundo externo a ele e vice-versa. Os macro blocos são regiões reservadas na área do circuito em que é proibido posicionar portas lógicas dentro delas.

A área de posicionamento do estágio global é apresentada na Figura 2.17. As portas lógicas devem ser posicionadas dentro do maior retângulo (em vermelho). Os retângulos dentro do maior (em preto) são regiões de bloqueio. Essas regiões são reservadas para os macro blocos. Os retângulos nas bordas do maior (em azul) são os pinos ou *PADs*.

2.3.2 Legalização

Após o posicionamento global há uma grande probabilidade do circuito não estar legalizado. Nesse estágio, as portas lógicas são tratadas individualmente e com a limitação

Figura 2.17 - Área de posicionamento global



Fonte: figura elaborada pelo autor.

de não alterar em demasia o resultado do estágio anterior.

Um posicionamento é definido como legal se não há sobreposição entre as portas lógicas e entre elas e os macroblocos e se todas as portas lógicas estão alinhadas dentro das bandas (POPOVYCH et al., 2014). As bandas são formadas pelo espaço entre os pares de linhas de alimentação. Uma das bordas da banda é linha de alimentação de VDD e a outra é a linha de alimentação de GND. A distância entre as linhas de alimentação é definida pela altura padrão das células de uma biblioteca de células.

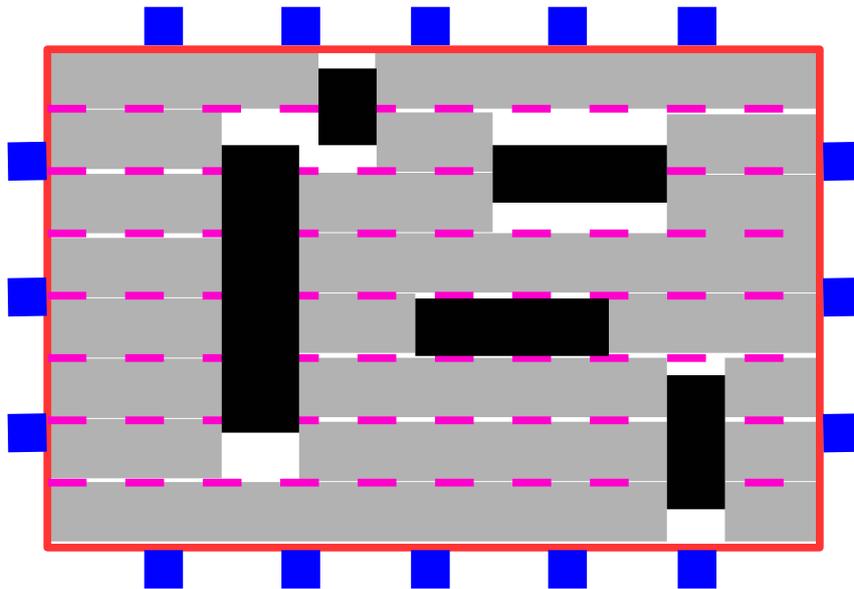
Na Figura 2.18 é apresentada a área de posicionamento dividida em banda e em segmentos. As bandas são formadas pelas linhas tracejadas (em rosa). Os segmentos são os retângulos em cinza. O critério para limitar os tamanhos deles foi as bordas das regiões de bloqueio (retângulos em preto) e as bordas da área de posicionamento (bordas em vermelho do maior retângulo).

2.3.3 Posicionamento Detalhado

No estágio de posicionamento detalhado, os objetivos são otimizados localmente. O escopo de atuação dos algoritmos desse estágio restringe-se a algumas redes, a alguns caminhos de dados ou a algumas portas lógicas.

O problema de posicionamento detalhado é definido como: dado um posicionamento legal com um conjunto de portas lógicas e/ou macro blocos e um conjunto de redes, no posicionamento detalhado os objetivos são otimizados localmente sujeitos a não causar violações no circuito (POPOVYCH et al., 2014). Usualmente o deslocamento máximo das portas lógicas, em relação à posição delas após a legalização, é limitado por uma distância máxima no estágio de posicionamento detalhado.

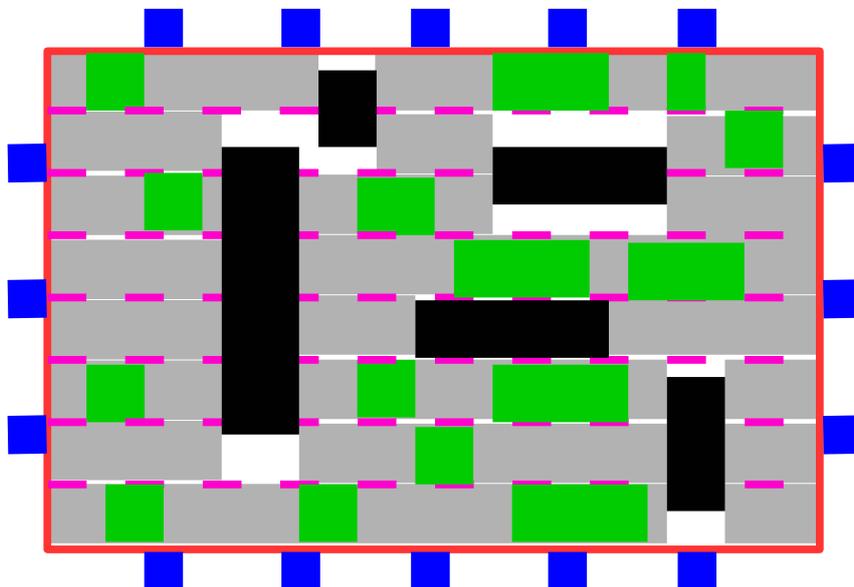
Figura 2.18 - Área de legalização



Fonte: figura elaborada pelo autor.

A área de posicionamento detalhado é apresentada na Figura 2.19. As portas lógicas (retângulos em verde) já estão legalizadas nas respectivas bandas e segmentos.

Figura 2.19 - Área de posicionamento detalhado



Fonte: figura elaborada pelo autor.

2.4 Modelos de Conexão para Posicionadores Analíticos

As redes com dois pinos são modeladas como arestas e as demais redes são modeladas como hiper arestas em um hiper grafo. Nesse caso, o circuito é modelado com um hiper grafo. Nos algoritmos de posicionamento analíticos, as redes com mais de dois pinos são

decompostas em um conjunto de conexões entre dois pinos (SPINDLER; SCHLICHTMANN; JOHANNES, 2008a).

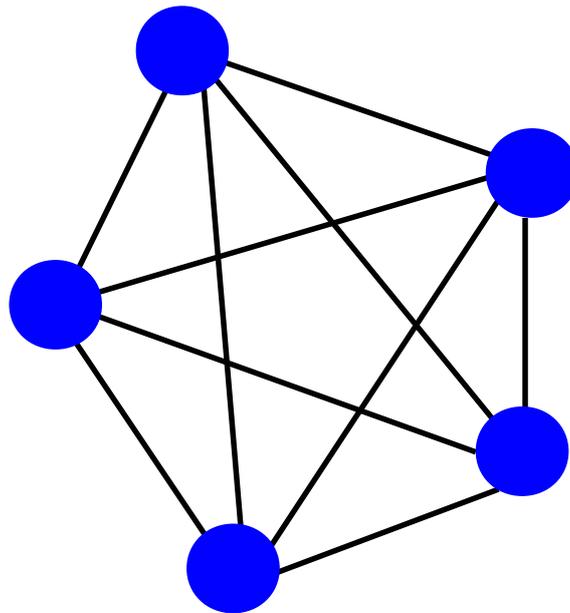
O conjunto de conexões entre dois pinos para a rede n é representado por E_n . A soma do custo quadrático, $\Phi = \sum_{n=1}^N \Phi_n$, para as redes $n = 1, 2, 3, \dots, N$ representa o custo quadrático do circuito (SPINDLER; JOHANNES, 2007).

$$\Phi_n = \sum_{e=(i,j) \in E_n} (\omega_{e,x}(x_i - x_j)^2 + \omega_{e,y}(y_i - y_j)^2) \quad (2.9)$$

2.4.1 Modelo Clique

As hiper arestas são decompostas em conexões de dois pinos em cada rede no modelo Clique, conforme apresentado na Figura 2.20. Esse é tradicionalmente um dos modelos mais utilizados (VISWANATHAN; CHU, 2004).

Figura 2.20 - Modelo de conexão clique



Fonte: figura elaborada pelo autor.

Uma rede no modelo clique (grafo completo) com k pinos possui

$$\binom{k}{2} = \frac{k!}{2!(k-2)!} = \frac{k(k-1)}{2} \quad (2.10)$$

conexões entre dois pinos (KAHNG et al., 2011b).

O peso para uma rede com k pinos é definido por ω em que cada conexão entre dois pinos tem o valor definido por $\frac{\omega}{k-1}$ (VISWANATHAN; CHU, 2004).

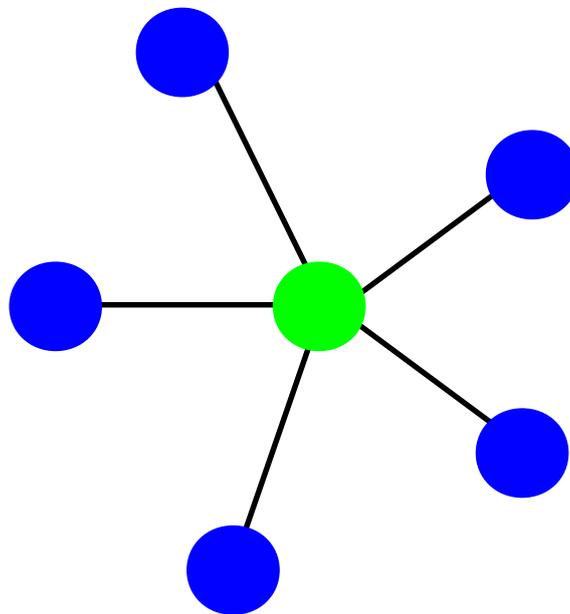
Nos métodos de posicionamento quadrático, esse modelo de conexão tem como principal desvantagem a redução no número de posições não nulas no sistema de equações lineares. Assim, a solução obtida para o sistema precisa de um tempo de processamento

computacional maior em comparação com os demais modelos.

2.4.2 Modelo Estrela

O modelo estrela para decompor hiper grafos é caracterizado pelas conexões dos pinos estarem ligadas a um nodo central (VISWANATHAN; CHU, 2004), conforme apresentado na Figura 2.21. Nesse modelo há k conexões de dois pinos, sendo um dos pinos o nodo central. O nodo central nesse modelo não é uma instância de uma célula da biblioteca, já os demais nodos são instâncias de células da biblioteca. O nodo central é criado somente com a função de ser o ponto de ligação entre as conexões de uma rede para o modelo estrela.

Figura 2.21 - Modelo de conexão estrela



Fonte: figura elaborada pelo autor.

A grande desvantagem desse modelo é a necessidade de um nodo extra para redes com mais de dois pinos. Portanto, na matriz de conectividade há l linhas e colunas a mais do que o número de portas lógicas móveis, l é o número de redes com mais de dois pinos em que deve ser acrescentado o nodo central. Contudo, há uma redução em torno de 30% no número de conexões entre dois pinos comparado ao modelo Clique (VISWANATHAN; CHU, 2004).

2.4.3 Modelo Híbrido

O modelo de conexão Híbrido foi proposto por (VISWANATHAN; CHU, 2004). Nesse modelo as redes com até três pinos são decompostas com o modelo clique e o modelo estrela é utilizado para decompor as demais redes.

Os autores do FastPlace (VISWANATHAN; CHU, 2004) quando propuseram o modelo híbrido, também apresentaram a prova de que o modelo de conexão Clique tem peso

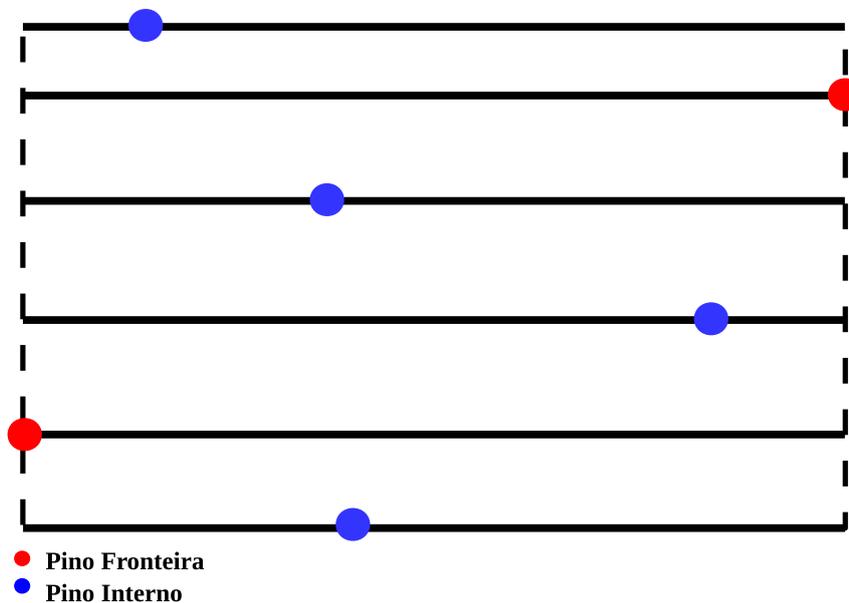
para o circuito equivalente ao modelo de conexão Estrela, se utilizados como modelos de rede para posicionadores analíticos, conforme enunciado no Teorema 1.

Teorema 1 *Para uma rede com k pinos, se o peso para a rede decomposta em conexões entre dois pinos é atribuído ω_c no modelo clique e $k\omega_c$ no modelo estrela, então o modelo clique é equivalente ao modelo estrela no posicionamento quadrático.*

2.4.4 Modelo *Bound to Bound* (B2B)

Uma *Bounding Box* (BB) é um retângulo em que todos os pinos de uma rede são internos ou estão nas bordas desse retângulo (KAHNG et al., 2011b). O modelo de rede B2B utiliza uma BB e define os pinos da rede como de fronteiras ou internos. Os pinos de fronteiras são aqueles que estão nas bordas da BB e os demais pinos são considerados internos. Nesse modelo de rede, todas as conexões entre os pinos internos são removidas, permanecendo somente as conexões dos pinos internos para os de fronteira e as conexões entre os pinos de fronteira (SPINDLER; SCHLICHTMANN; JOHANNES, 2008a; SPINDLER; JOHANNES, 2007). Na Figura 2.22 é apresentada uma rede decomposta com o modelo B2B somente na abscissa. O modelo de rede B2B possui duas BBs para cada rede. Uma delas para as posições dos pinos da rede na abscissa e a outra para as posições dos pinos da rede na ordenada.

Figura 2.22 - Modelo de conexão B2B



Fonte: figura elaborada pelo autor.

2.5 Algoritmos de Posicionamento Global

Os algoritmos de posicionamento global podem ser classificados em três categorias (SPINDLER; JOHANNES, 2007):

- Estocásticos: Os posicionadores baseados em aproximação estocástica que frequentemente utilizam SA. Essa é uma técnica que converge para um ótimo local que não necessariamente é uma solução ótima global. A principal desvantagem dela é o problema de escalabilidade que é proporcional ao aumento no número de portas lógicas.
- Particionamento: Nessa técnica a área de posicionamento e o circuito são particionados recursivamente, onde o objetivo do particionamento é minimizar o número de conexões entre as partições. Assim, em cada partição há apenas uma porta lógica ou não mais que uma dezena de portas lógicas em cada partição.
- Analíticos: O núcleo de todos os posicionadores analíticos é uma função objetivo que é minimizada iterativamente por métodos matemáticos. Dependendo do tipo de função objetivo, os posicionadores podem ser divididos em duas categorias:
 - Posicionadores baseados em otimização não linear: A função objetivo não é linear, por exemplo: Log-Sum-Exp, que é minimizada por técnicas de otimização não lineares como gradiente conjugado.
 - Posicionadores quadráticos: A função objetivo é quadrática e pode ser minimizada eficientemente ao resolver o sistema de equações lineares derivado.

2.5.1 Técnicas Estocásticas (SA)

O SA é um algoritmo de busca iterativo que pertence a classe de busca local aleatória (SHERWANI, 1999). O funcionamento dele é inspirado no processo de têmpera (*annealing*) usado na metalurgia. Nesse processo, um metal é aquecido até a temperatura em que o mesmo chegue ao estado líquido-amorfo. Então, ele é resfriado de forma controlada e gradual. Esse processo faz com que a rede cristalina ajuste-se de maneira a reduzir o número de defeitos, ou seja, para que o padrão da rede cristalina tenha a mínima energia global (SHERWANI, 1999).

A qualidade da solução obtida pelos algoritmos de SA é dependente da temperatura inicial e da velocidade de resfriamento. A função de redução gradual da temperatura αt é uma progressão geométrica em que α é definido tipicamente em 0,95 e a velocidade para reduzir a temperatura é definida pela Equação 2.11. Contudo, os algoritmos de SA são sensíveis à temperatura inicial e à velocidade de resfriamento, os quais são experimentalmente obtidos. A qualidade da solução é melhor quanto maior for a temperatura inicial e menor for a velocidade de resfriamento. Porém, o tempo de processamento é diretamente proporcional à velocidade de resfriamento e à temperatura inicial (SHERWANI, 1999).

$$t = t e^{-0,7t} \quad (2.11)$$

O SA é usado no posicionamento como um algoritmo para melhoria iterativa, isto é, sob um posicionamento inicial, uma mudança no mesmo é realizada ao trocar uma ou

duas portas lógicas de posição. As mudanças de posição das portas lógicas podem piorar ou melhorar a métrica utilizada para avaliar a qualidade do posicionamento. Métrica esta que é reflexo direto da melhora ou piora na qualidade do posicionamento. O SA aceita algumas das mudanças de posicionamento que pioram a métrica com probabilidade inversamente proporcional à temperatura. Assim, ao aceitar um movimento pior, o SA tenta evitar ficar preso em uma solução que seja ótimo local. O SA converge para uma solução que satisfaz às restrições e objetivos. Essa solução pode ser uma ótima global ou local. Entretanto, não é garantido que a solução seja a ótima global com o tempo de processamento limitado (SHERWANI, 1999).

2.5.2 Técnicas de Particionamento

Os métodos de posicionamento baseado em particionamento (Min-Cut) decompõem o circuito em regiões menores de modo descendente. Os posicionadores baseados em particionamento geralmente dividem a área de posicionamento em n partes ao fatorar a *netlist* e a área total das portas lógicas (ROY; PAPA; MARKOV, 2007).

Cada partição no hipergrafo e na área das portas lógicas é baseado em uma região retangular (*bin*) dentro do *layout*. Os *bins* representam uma região do circuito em que é permitido o posicionamento de portas lógicas, também representam um conjunto de portas lógicas para serem posicionadas dentro dessa região e todas as redes com sinais relacionados às portas lógicas dentro do *bin*. O posicionamento baseado em particionamento pode ser visto como uma sequência de passos em que todos os *bins* são examinados e alguns deles são particionados (ROY; PAPA; MARKOV, 2007).

2.5.3 Técnicas Analíticas

Durante o posicionamento, as portas lógicas e as conexões são modeladas usando a analogia da mecânica clássica de um sistema de molas. Cada porta lógica exerce atração sobre as demais conectadas a ela, onde a força de atração é diretamente proporcional ao peso, também denominado de multiplicador, de cada conexão entre as portas lógicas. Se é garantido o movimento livre para todas as portas lógicas móveis pertencentes a um sistema de molas, então todas as portas lógicas móveis vão chegar a uma configuração de equilíbrio em relação às forças de atração. Portanto, o comprimento de fio mínimo é obtido se todas as portas lógicas móveis estiverem em seus pontos de equilíbrio (KAHNG et al., 2011b).

As técnicas de posicionamento analíticas primeiro minimizam a função de comprimento de fio (função objetivo), desconsiderando a sobreposição entre portas lógicas, entre estas e macro blocos, etc. Desse modo, as portas lógicas estão posicionadas em uma região densamente povoada após a solução inicial, tipicamente próximo ao centro da área de posicionamento. Nos passos seguintes, a sobreposição é gradualmente reduzida através de uma série de iterações do algoritmo de posicionamento. Durante esse processo, o

comprimento de fio aumenta lentamente, enquanto o posicionamento converge para uma solução com um resíduo mínimo de sobreposição (KIM; LEE; MARKOV, 2010, 2012).

O espalhamento das portas lógicas pela área do circuito é realizado com o uso de forças de espalhamento. A adição de novas forças interfere no equilíbrio do sistema de molas. Assim, ao resolver novamente o sistema de equações lineares, é obtida uma nova posição para as portas lógicas. A resolução do sistema linear de equações e a adição de novas forças de espalhamento são realizadas alternadamente enquanto há uma significativa melhora nas métricas de avaliação de qualidade do circuito. A diferença entre os algoritmos de posicionamento global que utilizam essa técnica é o modo como a direção das forças de espalhamento é computada e como o valor das forças de espalhamento é adicionado às matrizes do sistema de equações lineares.

Os algoritmos de posicionamento analíticos minimizam uma função objetivo utilizando técnicas matemáticas como análise numérica ou programação linear. Esses métodos geralmente requerem certas suposições, como a função objetivo ser derivável ou os elementos posicionáveis serem tratados como pontos sem dimensão (KAHNG et al., 2011b).

As hiper arestas, redes com mais de dois pinos, são decompostas em um conjunto de conexões entre dois pinos. Cada conexão tem um peso (força) de acordo com o número de pinos na rede. O cálculo do peso da conexão também considera a distância entre as portas lógicas e pinos e/ou macro blocos fixos. Portanto, cada conexão de uma rede exerce força de atração entre as portas lógicas que ela liga. As portas lógicas móveis são posicionadas no circuito de modo a ter a mínima energia total. A posição final de cada porta lógica é obtida ao resolver várias vezes o sistema de equações lineares.

O custo quadrático mínimo, que aproxima o comprimento de fio do circuito, é obtido ao resolver o sistema de equações lineares apresentado na Equação 2.12 (SPINDLER; SCHLICHTMANN; JOHANNES, 2008a). O custo quadrático de um circuito pode ser computado independentemente para as abscissas e ordenadas, pois os sistemas de equações lineares são construídos independentes para elas. Neste trabalho são apresentadas as equações e o modo de construir o sistema de equações lineares somente para as abscissas. O sistema linear e as equações são obtidos da mesma forma para as ordenadas.

$$\Phi_x = \frac{1}{2}X^T C_x X + X^T d_x + const \quad (2.12)$$

A matriz C_x tem dimensão $M \times M$ em que M é o número de portas lógicas móveis. A posição $c_{i,j}$, $1 \leq i, j \leq M$ e $i \neq j$, representa a conexão entre as portas lógicas móveis mapeadas na linha i e na coluna j . Quando $i = j$ significa que na linha i e na coluna j estão mapeadas a mesma porta lógica. A matriz d_x é unidimensional com o tamanho definido por M . A posição i em d_x representa a conexão entre uma porta lógica móvel mapeada na coluna i com um elemento fixo (SPINDLER; SCHLICHTMANN; JOHANNES, 2008a;

SPINDLER; JOHANNES, 2007).

A força de aproximação, peso, $w_{x,i}$ de cada rede é computada ao resolver a Equação 2.13. Assim, para todas as conexões de cada rede, a força $w_{x,i}$ de uma conexão entre dois pinos de uma rede é representada nas matrizes da seguinte forma: Se as duas portas lógicas são elementos móveis, a força $w_{x,i}$ é incluída na matriz C_x adicionando-a nas posições c_{ii} e c_{jj} da diagonal principal e subtraindo o mesmo valor nas posições c_{ij} e c_{ji} . Se um dos módulos é um elemento fixo, o da posição j , a força $w_{x,i}$ é adicionada na posição c_{ii} da diagonal principal da Matriz C_x e a mesma força é multiplicada pela posição do elemento fixo e adicionada na posição d_i da matriz d_x . Caso ambos os elementos sejam fixos, nenhuma alteração é aplicada nas matrizes C_x e d_x (SPINDLER; SCHLICHTMANN; JOHANNES, 2008a; SPINDLER; JOHANNES, 2007).

$$w_{x,i} = \frac{2}{P-1} \frac{1}{l_{x,i}} \quad (2.13)$$

A matriz C_x é positiva semi-definida se não há elementos fixos e ela é positiva definida se há algum elemento fixo no circuito. Em ambos os casos a função Φ_x é convexa e o mínimo é obtido ao igualar a derivada da função a zero (SPINDLER; SCHLICHTMANN; JOHANNES, 2008a). A derivada para as posições referentes à abscissa é descrita pelo operador diferencial apresentado na Equação 2.14. Na Equação 2.15 é apresentada a derivada da Equação 2.12. O valor de x para cada posição das portas lógicas com o comprimento de fio mínimo é obtido ao resolver o sistema de equações lineares construído para a Equação 2.15.

$$\nabla_x = \left(\frac{\partial}{\partial x_1}, \frac{\partial}{\partial x_2}, \frac{\partial}{\partial x_3}, \dots, \frac{\partial}{\partial x_M} \right)^T \quad (2.14)$$

$$\nabla_x \Phi_x = C_x X + d_x = 0 \quad (2.15)$$

Obter uma solução para o sistema de equações lineares apresentado na Equação 2.15 pode ser feito eficientemente devido à matriz C_x ter um significativo número de posições nulas, ou seja, ela é esparsa. Um dos métodos iterativos mais utilizados para resolver um sistema de equações lineares com essa característica é a técnica de gradiente conjugado (SPINDLER; JOHANNES, 2007).

2.5.3.1 Técnicas Analíticas Quadráticas

Os posicionadores analíticos quadráticos otimizam os objetivos como funções quadráticas. Essas funções são contínuas, deriváveis e é bem conhecido o método para obter o ponto de mínimo ou máximo global.

A função de custo quadrática Φ_e de uma conexão entre dois pinos $e = (i, j)$ equivale a energia E_e de uma mola que conecta os objetos i e j . Na Equação 2.16 é apresentada

a relação de equivalência entre a força de atração em uma mola e a força de atração em uma conexão de uma rede (SPINDLER; JOHANNES, 2007).

$$\Phi_e = \frac{\omega_{e,x}}{2}(x_i - x_j)^2 + \frac{\omega_{e,y}}{2}(y_i - y_j)^2 = E_e = \frac{\omega}{2}l^2 \quad (2.16)$$

A adaptação da equação do sistema de molas para o posicionamento de porta lógicas é realizada ao substituir a constante ω por $\omega_{e,x}$ e $\omega_{e,y}$, respectivamente, para os sistemas de equações lineares para os vetores de posições x e y das portas lógicas. A distância quadrática euclidiana entre os corpos ligados às molas é substituída pela distância quadrática euclidiana entre os pinos de cada conexão do seguinte modo $l^2 = (x_i - x_j)^2 + (y_i - y_j)^2$. A função de custo quadrática $\Phi = \sum_{e \in E} \Phi_e$ é a soma das funções de custo quadrático Φ_e de todas as conexões entre dois pinos para todas as redes. Portanto, o custo em Φ representa a soma de todas as energias E_e de todas as molas, isto é, representa a energia total do sistema de molas (SPINDLER; JOHANNES, 2007).

A derivada da energia para as posições em x (ou para as posições em y) é a força na respectiva direção cartesiana. Portanto, a derivada descrita nas Equações 2.14 e 2.15 da função de custo, que representa o comprimento de fio, é a força na respectiva direção cartesiana. Na Equação 2.17, a força é igualada a zero para obter a mínima energia de um sistema elástico de molas que é equivalente ao comprimento de fio mínimo em um circuito (SPINDLER; JOHANNES, 2007).

$$\nabla_x \Phi = F_x^{net} = C_x X + d_x = 0 \quad (2.17)$$

2.5.3.2 Métodos Contínuos Não Lineares

A vantagem de usar um objetivo quadrático para o comprimento de fio é que a minimização pode ser obtida ao resolver um sistema de equações lineares positivo definido. Contudo, o modelo quadrático penaliza em excesso as redes longas e pode produzir um posicionamento com o comprimento de fio sub-ótimo (CHAN et al., 2007).

Diversos posicionadores tratam diretamente as hiper arestas, isto é, elas não são decompostas em um conjunto de conexões. Esses posicionadores tentam simultaneamente minimizar o comprimento de fio linear e distribuir as portas lógicas pela área de posicionamento. A função objetivo utilizada nesses métodos consiste em minimizar uma métrica como o comprimento de fio e medir uma restrição como a sobreposição. Genericamente essa formulação pode ser encapsulada como mostrada na Equação 2.18. Onde β é um parâmetro de ajuste que representa o compromisso entre a qualidade do posicionamento e a restrição durante todo o processamento. Tipicamente o valor da variável β é inicializado próximo de 1 no começo do posicionamento, momento em que o foco é na qualidade do posicionamento, e é reduzido durante o procedimento para encorajar a distribuição das

portas lógicas, que diminui a violação por sobreposição.

$$\Phi = \beta \times f_{\text{objetivo}} + (1 - \beta) \times f_{\text{restrição}} \quad (2.18)$$

2.6 Considerações Finais

Neste capítulo foram apresentados os principais conceitos sobre posicionamento utilizados neste trabalho de mestrado.

Os circuitos são abstraídos com o auxílio de grafo e é definido um conjunto de objetivos a serem otimizados. Assim, as portas lógicas são espalhadas pela área do circuito de modo a otimizar os objetivos sujeito às restrições.

A qualidade do posicionamento é medida através de métricas. As métricas mais utilizadas são: o comprimento de fio, a densidade, o congestionamento e a análise estática do tempo de propagação dos sinais do circuito (*timing*). Essas métricas também podem ser utilizadas para a definir a direção de espalhamento das portas lógicas dentro do circuito.

As redes com mais de dois pinos são modeladas como hiper arestas. Contudo, a maioria dos algoritmos de posicionamento analítico requerem redes somente com conexões entre dois pinos, e não conexões com múltiplos pinos como nas hiper arestas. Assim sendo, as hiper arestas são decompostas em conjuntos de conexões entre dois pinos.

O posicionamento é realizado em um fluxo com três estágios: posicionamento global, legalização e posicionamento detalhado. Ao final do último estágio resulta no *layout* do circuito que ainda a árvore de relógio e o sinais de dados precisam ser roteados.

As principais técnicas de posicionamento são: estocásticas, de particionamento e analíticas. Atualmente, os resultados de posicionamento das duas primeiras são ineficientes em comparação com o resultado obtido pela técnica analítica. O circuito é abstraído como um sistema de molas nos posicionadores analíticos. A solução desse sistema é obtida ao resolver um sistema de equações lineares.

3 ALGORITMOS DE POSICIONAMENTO E TRABALHOS CORRELATOS

A divisão da etapa de posicionamento em três estágios possibilita que sejam projetados algoritmos específicos para cada uma delas. Também é possível que determinadas restrições sejam relaxadas dentro de um determinado estágio. Com isso, o processamento pode ser mais rápido e eficiente.

Os trabalhos recentes em ferramentas de EDA desenvolvidos pelo grupo de pesquisa são citados a seguir. Estes são aqueles com foco em posicionamento: (PINTO, 2011), (SAWICKI, 2009), (HENTSCHEKE, 2007) e (FLACH et al., 2007). Contudo, nenhum desses trabalhos correlatos do grupo tem o objetivo de reduzir as violações nos tempos de propagação dos sinais de dados na etapa de posicionamento detalhado.

3.1 Posicionamento Global

Nesta seção são apresentados os algoritmos estado-da-arte em posicionamento global. Neste estágio o principal objetivo é minimizar o comprimento de fio. As restrições de sobreposição são relaxadas nos algoritmos discutidos a seguir.

3.1.1 SimPL

O SimPL (KIM; LEE; MARKOV, 2010, 2012) é um algoritmo de posicionamento quadrático com o modelo de rede B2B. Essa técnica é auto-contida para posicionamento global sem particionar o circuito. No SimPL são mantidos dois posicionamentos, um superior e outro inferior, que são utilizados para convergir para a solução final. O limite superior do posicionamento é gerado por um algoritmo de legalização aproximada (*rough legalization*). A legalização aproximada é baseada em particionamento geométrico descendente e em escalonamento não linear. O limite inferior para o posicionamento é obtido ao minimizar comprimento de fio a cada iteração em que o sistema de equações lineares é resolvido. Esse sistema de equações é resolvido para todo o circuito em cada iteração do algoritmo. Uma iteração do algoritmo consiste em uma legalização aproximada e em resolver o sistema linear de equações.

O primeiro passo no SimPL é minimizar o comprimento de fio. Após a solução inicial, o grafo de conexão é reconstruído com o modelo de conexão B2B. No SimPL é aplicada a legalização aproximada, a atualização da posição das âncoras, a atualização do modelo de conexão B2B e a resolução do sistema de equações lineares. Esses passos são aplicados até que a redução no comprimento de fio seja insignificante entre as iterações sucessivas do algoritmo.

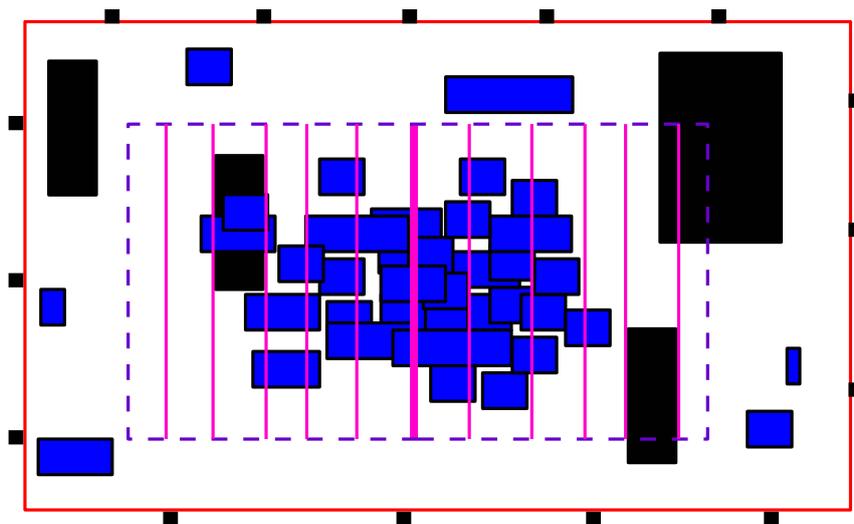
O espalhamento das portas lógicas na legalização rápida é realizado com base na densidade dos *bins*. Desse modo, em cada um deles, a área total disponível A_b e a área total das portas lógicas A_g são computadas. Um *bin* viola o limite superior γ se a densidade dele $\frac{A_g}{A_b}$ excede o limite γ ($0 < \gamma < 1$) definido empiricamente para o circuito. Se um *bin* viola a máxima densidade, então ele é agrupado com os vizinhos até que a densidade deles combinados seja menor do que o limite superior. Os *bins* agrupados são chamados de *clusters*.

O *cluster* é dividido em duas regiões por uma linha de corte C_b . Esta é o limite em que a área das portas lógicas é distribuída igualmente nas duas regiões. A seguir, em cada região, são adicionadas linhas de corte paralelas a C_b nas bordas de cada obstáculo. As duas regiões ainda são divididas por faixas, de modo que cada subdivisão não exceda 10% da largura da região. As portas lógicas, que estão ordenadas pela distância em relação a C_b , são posicionadas dentro das partições formadas pelas faixas. A posição relativa entre elas é mantida. Em cada partição é computada a área A_p disponível. As portas lógicas são posicionadas em uma partição enquanto a densidade de utilização de área da partição for menor do que o limite máximo γ estabelecido para o circuito. Assim, as portas lógicas são linearmente escalonadas. O escalonamento não linear advém do fato de posicionar as portas lógicas em diferentes partições nas diversas iterações do algoritmo. A legalização aproximada é aplicada alternadamente para as abscissas e ordenadas. A Região expandida, bem como, as linhas de corte são apresentadas na Figura 3.1.

A linha de corte baseada na distribuição equilibrada da área das portas lógicas não é suficiente para um espalhamento de boa qualidade. Quando a área dos obstáculos em uma região expandida é menor do que 20% da área total da mesma, calcula-se uma linha de corte C_c em que o corte nas arestas do grafo do circuito é mínimo. Esse corte é sujeito a um particionamento da área total das portas lógicas de no máximo 55% em uma das partições. A razão ρ da densidade de utilização da área das portas lógicas é salva para as duas partições. A linha de corte C_c é utilizada para ajustar a posição da linha de corte C_b de modo que a área de cada divisão tenha a mesma densidade de utilização de área.

Resolver um sistema linear de equações não restrito resulta em uma quantidade significativa de sobreposição. O sistema linear deve ser perturbado gradualmente para remover o excesso de sobreposição. Em cada iteração do algoritmo do SimPL, o particionamento geométrico descendente e o escalonamento geram uma nova legalização aproximada. As posições das portas lógicas, aquelas obtidas após a legalização aproximada, são utilizadas

Figura 3.1 - Região expandida do SimPL durante otimização aproximada



Fonte: figura elaborada pelo autor

como pontos em que são adicionadas âncoras. Assim, uma pseudo conexão é adicionada entre a posição atual de cada porta lógica e a posição da âncora. A adição de pseudo conexões ao circuito não impacta no aumento da dimensão da matriz de conectividade, pois o peso da pseudo conexão é adicionada somente na diagonal principal da matriz de conectividade e na matriz coluna de conexão entre as portas lógicas móveis com elementos fixos.

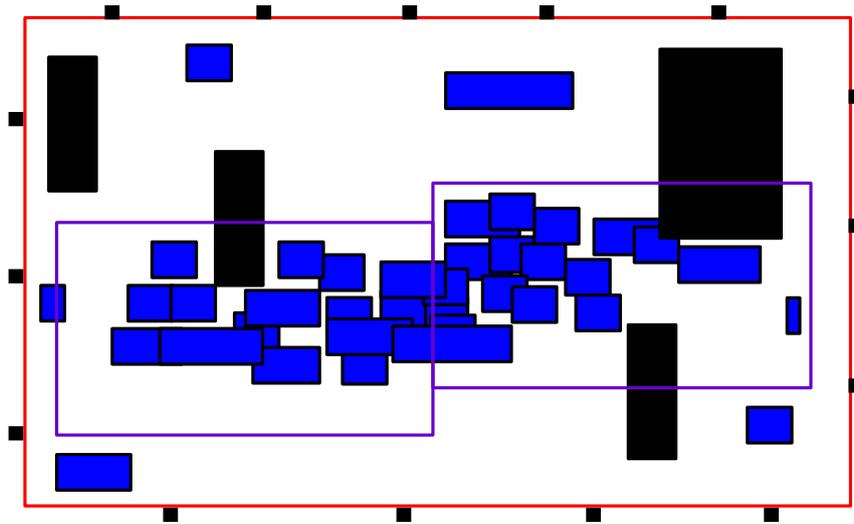
3.1.2 Polar e Polar 2.0

O Polar (LIN et al., 2013) é um algoritmo de posicionamento global analítico quadrático com as conexões modeladas via B2B. O polar utiliza a técnica de legalização aproximada do SimPL (KIM; LEE; MARKOV, 2010, 2012) e o refinamento do comprimento de fio. Durante a legalização aproximada, o objetivo no Polar é manter a posição relativa entre as portas lógicas de modo que o deslocamento delas seja minimizado.

A região expandida de um *bin* pode sobrepor a outros *bins* com violação de densidade. No Polar é considerada a região expandida em forma de retângulo. O ponto esquerdo-inferior do retângulo é a posição esquerda-inferior do *bin* inferior mais a esquerda e ponto superior-direito do retângulo é a posição superior-direita do *bin* superior mais a direita. Portanto, o retângulo da região expandida é formado pela agregação contínua dos *bins*, conforme apresentado na Figura 3.2.

Se considerar somente os *bins* agregados, a forma geométrica pode não ser retangular. Assim, a região expandida em forma de retângulo pode conter *bins* que não estavam originalmente agregados. A razão entre o tamanho das dimensões pode ser muito discrepante para o retângulo formado pela região expandida, o que pode resultar em um longo e desnecessário deslocamento das portas lógicas. O deslocamento máximo é limitado no Polar ao impor um limite superior no número de *bins* em uma região expandida. Caso

Figura 3.2 - Região expandida do Polar durante otimização aproximada



Fonte: figura elaborada pelo autor

esse limite seja violado, a região expandida é dividida em regiões menores com os *bins* contínuos em cada uma delas.

A posição relativa entre as portas lógicas tem uma significativa influência na otimização do comprimento de fio nas primeiras iterações. A solução do posicionamento inicial ignora a sobreposição entre as portas lógicas ao otimizar o comprimento de fio. No Polar foi proposta uma técnica de refinamento global que preserva a densidade ao mover as portas lógicas para uma posição ótima. A região ótima para uma porta lógica é definida pela mediana das posições em cada dimensão. Todas as posições das portas lógicas conectadas a aquela são consideradas para calcular a mediana.

O modelo B2B é atualizado após a posição destino de cada porta lógica ser definida com a legalização aproximada. Então, para cada porta lógica móvel, uma pseudo conexão entre a posição original e a posição alvo é adicionada à matriz de conectividade. O peso da pseudo conexão é calculado conforme a Equação 3.1. Em seguida, o sistema de equações lineares é resolvido para deslocar as portas lógicas em direção às posições obtidas na legalização aproximada.

$$peso = \begin{cases} \varepsilon \times \alpha^{i-1} & se\ i \leq 20 \\ \varepsilon \times \alpha^{20} \times \beta^{i-20} & se\ i > 20 \end{cases} \quad (3.1)$$

A força de deslocamento é modulada por duas condições: o valor de ε ser pequeno e $\beta > \alpha$. Nas iterações iniciais, o peso das pseudo conexões é pequeno para evitar mudanças repentinas no posicionamento, enquanto que nas iterações finais, o peso das pseudo conexões é aumentado para acelerar a velocidade de convergência.

O Polar 2.0 (LIN; CHU, 2014) é uma extensão do Polar (LIN et al., 2013) para tratar a demanda de roteamento como uma das restrições durante o posicionamento global.

A demanda de roteamento é tratada durante a etapa de legalização aproximada. A posição dos pinos, após a legalização aproximada, é utilizada para obter o padrão de congestionamento com a aplicação da técnica FastRoute (XU; ZHANG; CHU, 2009). Após essa etapa, cada porta lógica móvel possui três atributos: a demanda de área, a demanda-H (recursos de roteamento na horizontal) e a demanda-V (recursos de roteamento na vertical). Dessa forma, no estágio de posicionamento global as violações de roteamento são estimadas e tratadas.

A razão entre os recursos disponíveis e a demanda de roteamento não é distribuída uniformemente pelo circuito. A técnica de espalhamento baseada na árvore de bisseção somente distribui as portas lógicas de acordo com a demanda de área. Em algumas regiões há área disponível para posicionar portas lógicas, mas não há recursos de roteamento. Portanto, para evitar problemas de congestionamento local, que não são resolvidos durante a legalização aproximada, a demanda de roteamento de algumas portas lógicas é transformada em inflação da área baseada no histórico de violações de roteamento.

3.1.3 MAPLE

O MAPLE (KIM et al., 2012) é um *framework* multinível para posicionamento de circuitos de larga escala. No MAPLE são tratadas as restrições de utilização e os macro blocos são considerados como elementos móveis durante a etapa de posicionamento. Nesse algoritmo é proposto um guia para a transição suave das métricas para avaliar a qualidade de posicionamento entre as etapas de posicionamento global e detalhado.

Os algoritmos de posicionamento contêm inúmeros passos de otimização com objetivos diferentes. Uma péssima coordenação entre os diversos passos do algoritmo pode causar mudanças radicais nos resultados de posicionamento intermediários. Essas mudanças podem ser destrutivas em relação aos ganhos obtidos em um passo anterior, elas também podem aumentar o tempo de execução e reduzir a qualidade do posicionamento final.

A discrepância entre as soluções dos passos no posicionamento analítico pode ser mitigada ao garantir a transição gradual entre eles. Dessa forma, o fluxo é modificado nos pontos em que há significativa discrepância nos objetivos. Esses pontos estão localizados antes do desagrupamento (*unclustering*) e antes do posicionamento detalhado. No MAPLE foi introduzido um novo estágio intermediário. Nele a combinação linear das funções-objetivos predecessoras e sucessoras são otimizadas, enquanto modificam-se gradualmente os parâmetros para garantir uma transição suave dos objetivos entre os passos do algoritmo. Em cada um deles, o MAPLE busca a melhoria próxima da monotônica para o comprimento de fio e para a densidade de utilização de área do circuito sem comprometer os demais objetivos. Nessa busca é proibida a mudança abrupta de posição das portas lógicas. Também é vedada mudança significativa nos objetivos chaves. Assim, os pequenos movimentos são encorajados para suavizar as mudanças no comprimento de fio

e na densidade de utilização da área do circuito.

O MAPLE utiliza a técnica *Progressive Local Refinement* (ProLR) para reduzir o comprimento de fio e a densidade de utilização de área dos *bins*. No ProLR é adotada uma iteração do *Iterative Local Refinement* (ILR) como base indutora para modificar o posicionamento. O ILR tende a ser destrutivo, enquanto que o ProLR promove a transição gradual ao limitar o redimensionamento do tamanho do *bin* com a técnica *Explicit Bin-Blocking* (EBB). No ProLR é escalonado cuidadosamente a utilização do peso para balancear o comprimento de fio e a densidade de utilização de área do circuito. As mudanças são permitidas em apenas um objetivo por vez, enquanto elas são restritas aos demais. Todos os objetivos são otimizados restritos as modificações abruptas nas posições das portas lógicas e nas métricas para mensurar a qualidade do circuito posicionado.

O tamanho do *bin* na execução do ProLR é limitado a 5 vezes a altura padrão das portas lógicas. O ILR e o ProLR utilizam *bins* regulares enquanto movem as portas lógicas pelos *bins* vizinhos. No ProLR os *bins* são pequenos e o tamanho da dimensão de cada um deles é fixado. No ILR, o tamanho da dimensão de cada *bins* não é mantida fixa.

O objetivo do EBB é tornar o movimento de refinamento local menos inapropriado. Essa técnica consiste de dois componentes. O EBB^+ reduz a entrada de portas lógicas em alguns *bins* quando há expectativa de um movimento prejudicial. Esse procedimento é aplicado a uma certa quantidade de *bins* com o intuito de limitar a densidade de utilização de área do circuito. O EBB^- é utilizado para desencorajar a saída de portas lógicas de alguns *bins* e encorajar a entrada delas nesses *bins*. O EBB^- é aplicado a um grande número de *bins* com o objetivo de atrair as portas lógicas.

No refinamento local, as portas lógicas são movidas individualmente com o propósito de otimizar uma combinação linear entre o comprimento de fio e a densidade de utilização de área do circuito. Nos melhores movimentos de cada porta lógica é procurado aquele que otimize o objetivo em questão sem deteriorar as demais métricas. No ProLR esse refinamento é separado em duas otimizações simples. No ProLR-w é otimizado o comprimento de fio, enquanto que a densidade de utilização de área do circuito é otimizada no ProLR-d.

3.1.4 Ripple

O Ripple (HE et al., 2013) é um posicionador guiado a roteabilidade. O fluxo de posicionamento do Ripple é composto pelo posicionamento global, legalização e posicionamento detalhado. O objetivo dessa técnica é obter o melhor compromisso entre comprimento de fio e a demanda de recursos de roteamento.

O fluxo de posicionamento do Ripple é baseado no ajuste entre os posicionamentos superior e inferior do SimPL (KIM; LEE; MARKOV, 2010, 2012). O primeiro passo é obter o posicionamento inicial. Em seguida é realizada a legalização aproximada (posicionamento superior) e resolvido o sistema de equações lineares (posicionamento inferior).

Esses passos são iterados até que não haja significativa melhora nos objetivos.

No passo de legalização aproximada, a área das portas lógicas que estão em regiões com violações de congestionamento são infladas. A estimativa de congestionamento é realizada após a legalização aproximada. Assim, os dados de congestionamento são mais precisos. As portas lógicas são infladas, alternadamente, na vertical e na horizontal por diversas vezes a cada iteração do algoritmo.

A legalização é realizada ao final do estágio de posicionamento global. O legalizador pode aumentar as violações de roteamento devido à significativa discrepância entre a posição das portas lógicas após o posicionamento global e após a legalização. Para resolver esse problema, o tamanho dos segmentos é limitado no Ripple. Uma banda pode ser fatorada em vários segmentos. Assim, a capacidade é reduzida para cada segmento e indiretamente é estabelecido um limite máximo de deslocamento. Caso um segmento esteja ocupado, o algoritmo de legalização busca um que tenha espaço disponível entre os vizinhos dele.

Os posicionadores detalhados guiados a minimização do comprimento de fio podem gerar problemas de roteamento. Esses problemas podem ser gerados em cada um dos passos realizados pelo posicionador detalhado. No fluxo do Ripple, após cada passo é verificado se ocorre violações de roteamento ou incremento das mesmas. Para tanto, uma versão simplificada do FastRoute (XU; ZHANG; CHU, 2009) é utilizada para verificar congestionamento.

3.1.5 ePlace

O ePlace (LU et al., 2013, 2014) é um algoritmo de posicionamento analítico para tratar circuitos de larga escala com portas lógicas e macro blocos. A função de densidade para remover a sobreposição é baseada em eletrostática e no método de Nesterov para minimizar o custo não linear. O comprimento do passo é estimado como o valor inverso da constante de Lipschitz, a qual é determinada por um método dinâmico com *backtracking*.

As portas lógicas são modeladas como cargas elétricas na função de densidade. Esse modelo representa a energia potencial total. A força elétrica é utilizada como indutor para espalhar os objetos, resultando, no final, na redução da energia total para zero, sendo esse o ponto de equilíbrio eletrostático. Um posicionamento com a mínima taxa de sobreposição é obtido quando a energia total é reduzida para zero.

No ePlace foi proposto utilizar a equação de Poisson para correlacionar a densidade espacial com o potencial de distribuição. As portas lógicas são mantidas dentro da área de posicionamento utilizando a condição de fronteira de Neumann (gradiente zero nas fronteiras). A componente de frequência zero foi removida da densidade espacial e da distribuição de potencial. Isso foi feito para equilibrar a distribuição das cargas (espalhamento das portas lógicas) dentro da área de posicionamento. Caso contrário, as cargas estariam concentradas nas proximidades das fronteiras do circuito.

A otimização dos macro blocos e das portas lógicas pode ser tratada simultaneamente no ePlace. Em cada iteração é computado o gradiente, aproximado por um preconditionador, inferida a constante de Lipschitz e estimado o comprimento do passo via *backtracking*. A solução de posicionamento converge ao resolver iterativamente o sistema não linear com o método de Nesterov.

Dois posicionamentos distintos (u_k e v_k) são obtidos em cada iteração ao otimizar o sistema não linear na formulação do ePlace. A solução final é o posicionamento u resultante da última iteração do algoritmo. O tamanho do passo é computado com base na constante de Lipschitz. O gradiente tem um alto grau de suavidade quando a constante de Lipschitz tem um valor pequeno, o que resulta em uma rápida convergência devido ao comprimento do passo ter um valor grande. Contudo, obter o valor exato para a constante de Lipschitz tem um custo computacional alto. Por outro lado, obter o valor estático dessa constante não é viável devido às mudanças nas funções de custo.

O preconditionador tem uma grande aplicação para posicionadores quadrático, mas nenhuma funcionalidade para posicionadores não lineares, principalmente por causa da característica não convexa da função de densidade. No ePlace, uma matriz Hessiana com a diagonal positiva definida é utilizada como um preconditionador para o vetor de gradiente e para direcionar a otimização.

3.2 Legalização

Nesta seção serão apresentados três algoritmos de legalização. No estágio de legalização, a sobreposição residual da solução do posicionamento global é removida e todas as portas lógicas são alocadas dentro das bandas nos respectivos *sites*. A altura de cada banda é definida pela altura padrão das células da biblioteca.

3.2.1 Abacus

O Abacus (SPINDLER; SCHLICHTMANN; JOHANNES, 2008b) é um algoritmo de legalização rápida com movimentação mínima das portas lógicas. Esse algoritmo é similar ao Tetris (HILL, 2002). As portas lógicas são ordenadas em um vetor pela abscissa e, então, elas são legalizadas em ordem crescente. Assim, a posição relativa das portas lógicas é conhecida e preservada durante o fluxo de legalização. Caso seja necessário, o Abacus também move portas lógicas já legalizadas para reduzir o deslocamento total das mesmas.

O custo para legalizar as portas lógicas é estimado no segmento relativo à posição x em cada banda. Ele é estimado primeiro para as bandas que estão mais próximas da posição y de cada porta lógica. O posicionamento da porta lógica ocorre na banda com o menor custo.

3.2.2 BonnPlace

O BonnPlace (BRENNER, 2013) é um legalizador baseado em fluxo de custo mínimo. O fluxo é incrementado iterativamente conforme o tamanho do caminho aumenta. O fluxo obtido é aquele em que as portas lógicas podem ser legalizadas com movimentação mínima. Assim, ele combina a perspectiva global de minimizar o custo do fluxo com a eficiência da busca local. O BonnPlace é projetado para minimizar o movimento total de cada porta lógica, mas ele é suficientemente flexível para otimizar as demais funções-objetivo que impactam na movimentação das portas lógicas.

O estágio de legalização precisa tratar um grande número de portas lógicas. Portanto, o algoritmo baseado em fluxo pode decidir para qual direção é melhor mover a porta lógica, se é de uma região com capacidade de área esgotada para uma com área livre ou trocar duas portas lógicas de duas regiões distintas. Nem sempre é possível encontrar portas lógicas com o mesmo tamanho. Por causa disso, geralmente é inviável a troca direta de portas lógicas entre regiões diferentes.

Por outro lado, se houver regiões com área livre e a mesma não está bloqueada para movimento de portas lógicas, uma porta lógica pode ser facilmente movida para esse local. Caso as regiões tenham bloqueios, pode-se adicionar *slots* entre os bloqueios. Se o tamanho do *slot* for menor do que o tamanho médio da maioria das portas lógicas, a legalização com o BonnPlace pode ser ineficiente. Diversas heurísticas foram propostas para contornar esse problema. Quando há regiões com pouco espaço disponível, itera-se o método com diferentes parâmetros. Após algumas iterações bloqueiam-se os *slots* com dimensão muito pequena para evitar laço infinito.

A execução do BonnPlace inicia com o particionamento da área de posicionamento em pequenos *bins* e adiciona as portas lógicas para os respectivos *bins*. A redução do número de *bins* com possíveis violações de densidade é utilizada para balancear o assinalamento das portas lógicas entre os *bins*. O algoritmo de Dijkstra (DIJKSTRA, 1959) foi alterado para computar o caminho de menor custo sempre que houver alteração em uma aresta do caminho e também que esse novo caminho obtido seja um conjunto de portas lógicas válido.

Devido a mudança no modo de obter o caminho, no BonnPlace não existe mais a propriedade de encontrar o fluxo ótimo para algumas funções com o custo das arestas pré definido. Contudo, essa técnica ainda mantém uma boa visão global de todo o circuito quando o caminho é aumentado.

3.2.3 Algoritmo de HU et al.

No legalizador proposto por (HU et al., 2013) são considerados três ordenamentos diferentes para legalizar as portas lógicas. As três métricas de ordenamento são: em relação à largura das portas lógicas, ordenamento pelo número de pinos e ordenamento pela posição central. A legalização é dividida em três passos. No primeiro as portas lógicas

são alinhadas aos *sites*, em seguida são realocadas as instâncias com violações e no último passo é removida a sobreposição.

O algoritmo é executado para o primeiro passo da seguinte forma: as portas lógicas são ordenadas de modo decrescente de acordo com o seu tamanho. Em seguida, elas são alinhadas aos *sites* mais próximos de suas posições originais. Caso o *site* já esteja ocupado por outra porta lógica, ambas são consideradas inválidas e a violação será resolvida no próximo passo. As portas lógicas já legalizadas não são movidas. Primeiro são posicionadas as portas lógicas com a maior largura, pois aquelas de largura menor podem fragmentar os segmentos das bandas, o que pode resultar na inutilização de alguns segmentos e consequentemente gerar um aumento no comprimento de fio. A ordenação pelo tamanho permite que as portas lógicas mais largas tenham preferência pelas melhores posições.

No segundo passo, as portas lógicas marcadas como inválidas são ordenadas de modo descendente pelo número de pinos. Assim, as portas lógicas com mais pinos tem preferência na legalização, porque, em geral, as portas lógicas com mais pinos tem mais conexões, o que resulta em um impacto maior no comprimento de fio. Para cada porta lógica com violação é obtida a região retangular (BB) em que ela está inserida. A BB é formada por todas as coordenadas dentro dele de todas as portas lógicas que tem ligação com a porta lógica em que busca-se resolver a violação. A posição com o mínimo de deslocamento é buscada dentro desse retângulo.

No último passo, as portas lógicas em uma banda são ordenadas de modo crescente pelo centro das mesmas. O espaço livre de cada banda ou sub-banda é realocado de modo a remover a sobreposição entre as portas lógicas. Nos trabalhos anteriores, as portas lógicas foram ordenadas pela abscissa nos passos iniciais. Nessa técnica, as portas lógicas são ordenadas pela abscissa no final do fluxo com o propósito de reduzir o deslocamento e minimizar o comprimento de fio.

3.3 Posicionamento Detalhado

A meta principal dos algoritmos de posicionamento detalhado é otimizar os objetivos no escopo local sem gerar violações. Os algoritmos de posicionamento detalhado operam sobre até uma dezena de portas lógicas e de conexões. Nesta seção são apresentados os principais algoritmos de posicionamento detalhado.

3.3.1 BraveDP

O BraveDP (POPOVYCH et al., 2014) é um algoritmo de posicionamento detalhado com o objetivo de minimizar o comprimento de fio e reduzir os picos de densidade. A otimização é restrita ao deslocamento máximo das portas lógicas. A restrição de deslocamento é controlada de modo eficiente com os métodos de legalização instantânea e

reversão de trocas. A técnica de legalização rápida também permite que o BraveDP seja mais agressivo durante as trocas das portas lógicas. A função de atualização incremental da utilização dos *bins* serve como guia para as trocas de portas lógicas. A decisão de mudar de posição uma porta lógica depende da melhoria do comprimento de fio, da sobreposição e da redução da densidade. Portanto, a atualização incremental da densidade permite estimar mais precisamente e rapidamente a variação nas densidades média e de pico.

O método de otimização do BraveDP é baseado na troca de posição entre as portas lógicas. Isto significa que ele tenta sistematicamente trocar a posição delas dentro de uma região. Uma porta lógica pode trocar de posição com um espaço vazio ou trocar de lugar com outra porta lógica. A preferência ao mover uma porta lógica é para uma posição que tenha espaço vazio suficiente para aloca-la. Por causa disso, os *bins* com menor utilização em um região têm prioridade. A atualização da métrica de penalidade da densidade de utilização de área do circuito é computada incrementalmente em tempo constante com um pequeno erro residual.

A legalização instantânea é realizada com uma versão adaptada do algoritmo Abacus (SPINDLER; SCHLICHTMANN; JOHANNES, 2008b). A vantagem de legalizar a cada movimento de porta lógica é conhecer imediatamente as consequências resultantes das mudanças de posições das portas lógicas. O legalizador Abacus foi modificado para buscar o *cluster* e a posição da porta lógica em tempo constante.

O deslocamento das portas lógicas é computado com auxílio de três métricas: região ótima, comprimento de fio e densidade. A região ótima é obtida a partir da mediana de todas as posições das portas lógicas que tem conexões com aquela em que está sendo processada a movimentação. Essa métrica foi proposta por (PAN; VISWANATHAN; CHU, 2005). Ao assumir que uma porta lógica posicionada dentro da região ótima tem o comprimento de fio minimizado, então as portas lógicas que estão fora da região ótima são candidatas a reduzir o comprimento de fio quando movidas para dentro dela. Ainda, mover as portas lógicas de regiões com alta densidade de utilização de área do circuito para aquelas com baixa densidade de utilização de área do circuito ajuda a melhorar a qualidade do posicionamento do circuito.

3.3.2 Algoritmo de CHOW et al.

O algoritmo de (CHOW et al., 2014) reduz o comprimento de fio enquanto procura preservar a solução do posicionamento global. Os movimentos das portas lógicas podem ser restritos por causa da violação de densidade de utilização de área do circuito e da violação do deslocamento máximo permitido. A otimização é realizada em dois passos. No passo global desse algoritmo, uma porta lógica é movida para um *bin* em que é minimizado o comprimento de fio e que não viole a restrição de densidade de utilização de área do circuito. Nesse passo, as portas lógicas não são legalizadas. No passo local do algo-

ritmo é realizado o ajuste da posição da porta lógica dentro do *bin* de modo a legalizá-la e a minimizar ainda mais o comprimento de fio.

No passo de movimento local, são aplicados os algoritmos de movimento vertical, de reordenamento local e de compactação de um segmento. Os dois primeiros algoritmos foram propostos por (PAN; VISWANATHAN; CHU, 2005). No movimento vertical busca-se nas bandas pertencentes ao *bin*, aquela que tem o menor custo para legalizar as portas lógicas. No reordenamento local, as portas lógicas que pertencem a um caminho são trocadas de posições entre elas, limitadas a no máximo três delas em sequência. Em seguida, as portas lógicas são agrupadas em um segmento se as métricas para medir a qualidade do posicionamento do circuito são otimizadas. O terceiro passo, a compactação de um segmento, é uma adaptação do algoritmo de agrupamento em um segmento para suportar as restrições de deslocamento e densidade de utilização de área, uma vez que o algoritmo originalmente proposto por (PAN; VISWANATHAN; CHU, 2005) não suporta tais restrições.

3.4 Considerações Finais

O estágio de posicionamento global tem recebido recentemente uma grande atenção, principalmente na otimização dos objetivos mapeados para o comprimento de fio. Algumas restrições como congestionamento e temporização dos sinais não têm sido foco de pesquisa no âmbito de posicionamento global. Esse é um ponto em aberto para o estágio de posicionamento global.

Por outro lado, os estágios de legalização e posicionamento detalhado não possuem tantos trabalhos recentes quanto os focados em posicionamento global. A legalização, talvez pela simplicidade do objetivo, não desperta muito entusiasmo. Contudo, nesse campo fica em aberto legalizadores que consigam trabalhar eficientemente com restrição de movimentação de portas lógicas, violações de temporização, violações de densidade de utilização de área do circuito, etc. Além disso, as métricas para medir a qualidade do posicionamento têm um grau de integração baixo com os legalizadores atuais.

O posicionamento detalhado teve um bom número de trabalhos até por volta do ano de 2010. Desde então houve um hiato de bons posicionadores detalhados que consigam otimizar os objetivos diante da evolução dos posicionadores globais e da tecnologia de fabricação.

4 ALGORITMO DE POSICIONAMENTO ANALÍTICO DETALHADO

Este capítulo apresenta o algoritmo de posicionamento analítico detalhado para minimizar violações no tempo de propagação nos caminhos com violação de *late*, que é a contribuição dessa dissertação. A ideia desse algoritmo é movimentar somente as portas lógicas combinacionais pertencentes aos caminhos críticos e as vizinhas das combinacionais do caminho crítico no primeiro nível de profundidade lógica. Desse modo, o comprimento de fio de cada rede das portas lógicas no caminho crítico tende a ser reduzido, uma vez que, as portas lógicas com o modelo quadrático são posicionadas no ponto de equilíbrio. No algoritmo proposto neste trabalho também foi integrado um legalizador rápido para garantir a legalidade do circuito após a execução do algoritmo analítico.

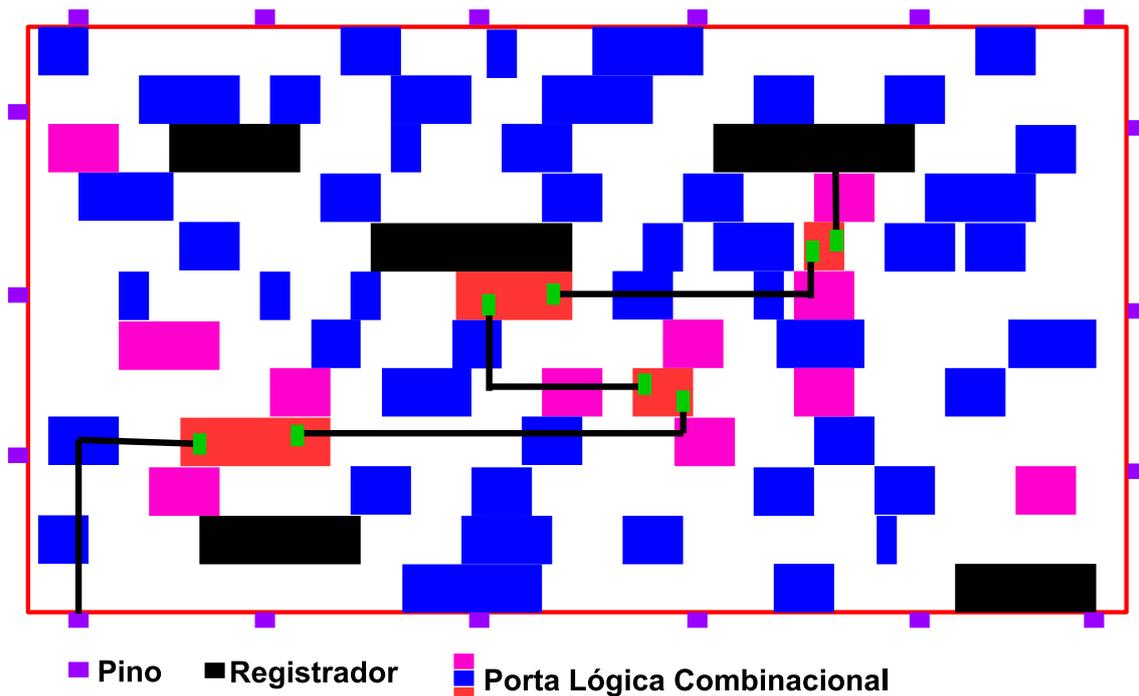
4.1 Algoritmo de Posicionamento Analítico Detalhado

O algoritmo de Posicionamento Analítico Detalhado (PADe) com formulação analítica quadrática tem como foco minimizar as violações de atraso crítico de *late*. Ele atua somente sobre as portas lógicas combinacionais pertencentes aos caminhos críticos e sobre as vizinhas no primeiro nível lógico destas. Na Figura 4.1 é apresentado um caminho crítico (linha em preto) em um pequeno circuito. As portas lógicas em vermelho pertencem ao caminho crítico e as em rosa são vizinhas àquelas no caminho crítico. As portas lógicas vizinhas são consideradas aquelas conectadas diretamente pelo mínimo a uma daquelas que fazem parte do caminho crítico. As demais portas lógicas combinacionais e os registradores são considerados elementos fixos na formulação utilizada nesse trabalho

A movimentação das portas lógicas combinacionais relacionadas aos caminhos críticos tem como efeito a aproximação delas e, por causa disso, há a redução de capacitância e resistência dos caminhos. Essa aproximação indiretamente reduz o tempo de propagação nas redes e conseqüentemente reduz as violações no tempo de propagação de *late*.

As redes com mais de dois pinos foram decompostas com o modelo de conexão clique. Apesar da matriz de conectividade ter menos posições nulas do que os modelos de rede estrela, híbrido e B2B, o número de portas lógicas relacionadas com o caminho

Figura 4.1 - Exemplo de caminho crítico



Fonte: figura elaborada pelo autor.

crítico é uma pequena fração do total de portas lógicas do circuito. Assim, o sistema de equações lineares é pequeno se comparado com um sistema de equações lineares para o posicionamento global, em que praticamente todas as portas lógicas são móveis.

No Algoritmo 1 é apresentado o fluxo de execução do PADE. O algoritmo recebe como entrada os limites da área de posicionamento, uma lista das portas lógicas e uma lista das redes do circuito e o número de caminhos críticos a serem tratados.

O primeiro procedimento no PADE, Linha 1, é inicializar o algoritmo de legalização Jezz. Esse algoritmo é utilizado para fazer a legalização rápida quando as portas lógicas são movimentadas no posicionamento detalhado. Na inicialização é obtida a posição das portas lógicas e a área livre, quando houver, entre elas. Também são identificadas as regiões com bloqueio.

A informação se a porta lógica é fixa ou móvel é salva na Linha 2. Todas as portas lógicas que não estão relacionadas diretamente com o caminho crítico são alteradas para fixas. Essa modificação é realizada para que elas sejam consideradas como portas lógicas fixas quando o sistema de equações lineares é construído. Assim, o PADE não movimenta desnecessariamente as portas lógicas não relacionadas aos caminhos críticos. As portas lógicas que foram alteradas para fixada podem ser movidas pelo legalizador, elas foram alteradas para que ao resolver o sistema de equações lineares a posição delas não fosse alterada.

Os registradores são modificados para fixados nas estruturas de dados do Jezz, conforme mostrado nas Linhas 3 a 5. Essa alteração evitar que eles sejam movidos quando as

Algoritmo 1: Algoritmo de Posicionamento Analítico Detalhado (PADe)

Data: placement area limits, list of circuit elements, number of paths

```

1 jezz = initJezz();
2 backupGatesState();
3 for reg:listRegs do
4   | jezz->setFixed(reg);
5 end
6 criticalPaths = getCriticalPaths(numPaths);
7 for path:criticalPaths do
8   | gates = getGates(path);
9   | criticalGates.add( gates);
10 end
11 for gts:criticalGates do
12   | neigbohrs = getNeigbohr(gts);
13   | listGates.add(gts);
14   | listGates.add(neigbohrs);
15 end
16 for gate:circuit do
17   | if ! listGates.has(gate) then
18     | gate.setState(fixed);
19   end
20   | if isReg(gate) then
21     | gate.setState(fixed);
22   end
23 end
24 linSys = createLinearSystem(listGates);
25 posGates = solveLinearSystem(linSys);
26 for gate:listGates do
27   | originPos = gate.getPos();
28   | fitGateMaxDistance(posGates[gate], gate);
29   | legal = jezz->legalize(gate, posGates[gate]);
30   | if legal then
31     | updateNetBounds(gate);
32     | updateTiming(gate);
33     | updateSteinerTree(gate);
34     | updateABU(gate, originPos);
35     | updateUtilization(gate);
36     | jezz->setFixed(gate);;
37   end
38 end
39 recoverState();
40 updateABU();
41 updateABUPenalty();
42 updateQualityScore();

```

Result: legal placement

portas lógicas combinacionais são legalizadas. Assim, evita-se que o deslocamento dos registradores cause modificações indesejadas nos tempos de propagação dos sinais. Uma vez que, a otimização do *skew* do relógio reduz significativamente as violações no tempo de propagação dos sinais. Isso é ocasionado pelo uso do algoritmo Flute (CHU; WONG, 2008) para caracterizar as redes do circuito. O Flute (CHU; WONG, 2008) considera a rede de relógio como uma rede de dados e não como uma rede especial ao caracterizá-la.

Na Linha 6 é obtida a lista de *endpoints* dos caminhos com os maiores tempos de propagação dos sinais. Um *endpoint* pode ser a entrada de um registrador ou uma saída primária. Uma saída primária é um *PAD* ou um pino. Em seguida, Linhas 7 a 10, são obtidas as portas lógicas pertencentes a cada um dos caminhos críticos (Linha 8) e estas são adicionadas a uma lista (Linha 9).

As portas lógicas que são vizinhas daquelas que fazem parte dos caminhos críticos são armazenadas em uma nova lista juntamente com aquelas que são parte dos caminhos críticos (Linhas 11 a 15). O estado de todas as portas lógicas é alterado para fixado, exceto para aquelas combinacionais que estão relacionadas diretamente aos caminhos críticos (Linhas 16 a 23).

O sistema de equações lineares é construído com o método da Linha 24. Nesse sistema, a grande maioria das portas lógicas são elementos fixos no circuito. As novas posições das portas lógicas móveis são obtidas ao resolver o sistema de equações lineares (Linha 25).

Para todas as portas lógicas móveis é verificado se elas violam a distância máxima (Linha 28). Caso afirmativo, elas são movidas para uma posição intermediária entre a original e a resultante da solução do sistema de equações lineares, uma posição em que não viole o deslocamento máximo. Em seguida (Linha 29) tenta-se legalizar a porta lógica na posição intermediária. Esse procedimento será possível somente se houver espaço livre suficiente para alocá-la na posição alvo. Uma violação ocorre se o custo para mover as portas lógicas presentes na posição destino for proibitivo ou se houver portas lógicas que estão fixadas à esquerda ou à direita da posição destino e impossibilite de alocar espaço suficiente para legalizar a porta lógica em questão. Se não for possível legalizar na posição alvo, ela é mantida na posição original.

Caso seja possível legalizar a porta lógica na posição destino, as métricas (Linhas 30 a 37), são atualizadas de modo incremental para ela. O estado daquelas que foram movidas é alterado para fixado somente no legalizador, Linha 36. Esse procedimento é realizado para evitar que a movimentação delas viole o deslocamento máximo quando uma terceira porta lógica for legalizada nessa mesma região. A métrica de avaliação da qualidade do posicionamento proposta por (KIM; HUI; VISWANATHAN, 2014) é atualizada na Linha 42. A saída do algoritmo de posicionamento analítico detalhado é um novo posicionamento legal para o circuito.

Importante frisar que as portas lógicas fixadas para construir o sistema de equações

lineares podem ser movimentadas no legalizados Jezz. Fixar determinadas portas lógicas atende objetivos e restrições específicas durante a execução dos algoritmos desse fluxo de posicionamento detalhado.

4.2 Fluxo de Execução Integrado no UPlace

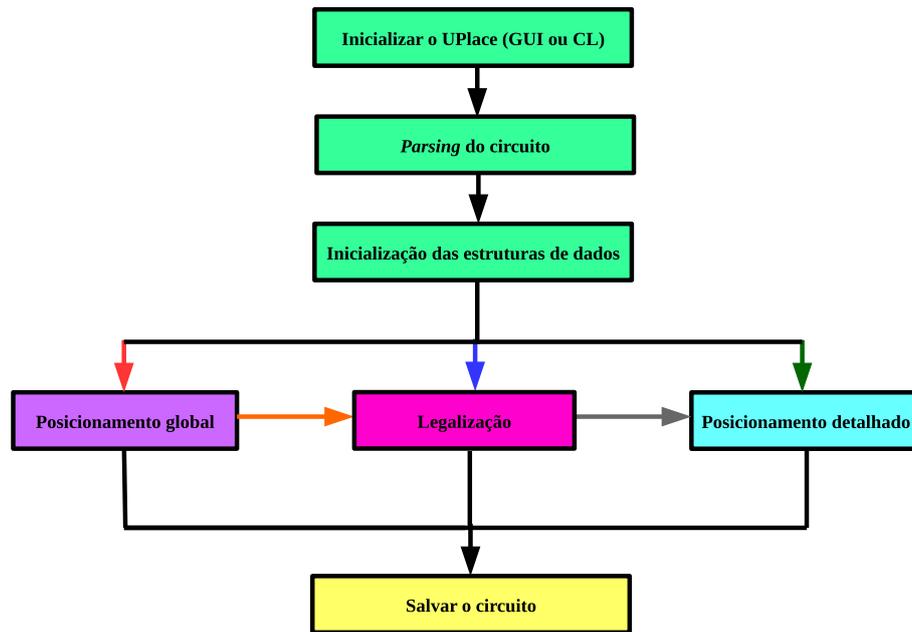
O UPlace é uma ambiente autocontido utilizado para o desenvolvimento do algoritmo de algoritmos de posicionamento global, legalização e posicionamento detalhado. Neste trabalho foi desenvolvido um algoritmo analítico para posicionamento detalhado das portas lógicas pertencentes aos caminhos críticos utilizando o *framework* UPlace. Nesse ambiente diversas funcionalidade já estão implementadas, o que facilita e acelera o desenvolvimento de algoritmos para as etapas de posicionamento. O ambiente UPlace é composto por uma interface gráfica, estruturas de dados para armazenar o circuito e um conjunto de algoritmos. A codificação dos algoritmos do UPlace foi realizada com a linguagem de programação C++. A interface gráfica foi desenvolvida utilizando a versão 2.8 da biblioteca wxWidgets (?). As estruturas de dados integradas no UPlace foram projetadas e implementadas com a biblioteca *Standard Template Library* (STL).

O fluxo de execução pode ser facilmente adaptado e/ou construído de acordo com o tipo e o objetivo do algoritmo em desenvolvimento. Os algoritmos existentes podem ser integrados no fluxo em desenvolvimento sem a necessidade de implementar novamente estruturas de dados para armazenar o circuito, entre outras funcionalidades. Um fluxo de execução simplificado é apresentado na Figura 4.2. Esse fluxo é essencialmente composto pela inicialização do UPlace com a interface gráfica ou com a linha de comando. O *parsing* do circuito é realizado para um dos formatos de descrição de circuitos: *Layout Exchange Format* (LEF)/*Design Exchange Format* (DEF) ou *Bookshelf*. Em seguida são inicializadas as estruturas de dados. Executar pelo menos um dos passos: posicionamento global, legalização ou posicionamento detalhado, ou um encadeamento dos estágios da etapa de posicionamento. O último passo é salvar o resultado do posicionamento do circuito no formato DEF ou *Bookshelf*.

Na inicialização do UPlace é realizado o *parsing* dos argumentos. Nesse passo são separadas as palavras chaves, as quais são utilizadas para controle no *parser*, e os atributos. No passo de *parsing* do circuito é realizada a leitura dos arquivos. Os circuitos podem estar descritos em LEF/DEF ou *bookshelf*. Em seguida é realizada a inicialização das estruturas de dados. Os elementos do circuito são armazenados principalmente em vetores e o relacionamento entre os elementos do circuito também é armazenados em vetores nesse passo.

Os passos de posicionamento global, legalização e posicionamento detalhado podem ser realizados independentes uns dos outros ou encadeados em uma sequência. A ordem de execução deles vai depender do objetivo no fluxo em desenvolvimento. Por outro lado,

Figura 4.2 - Diagrama do fluxo de execução do UPlace



Fonte: figura elaborada pelo autor.

caso seja necessário um fluxo de posicionamento completo, basta fazer as chamadas dos métodos de cada um dos passos na ordem desejada.

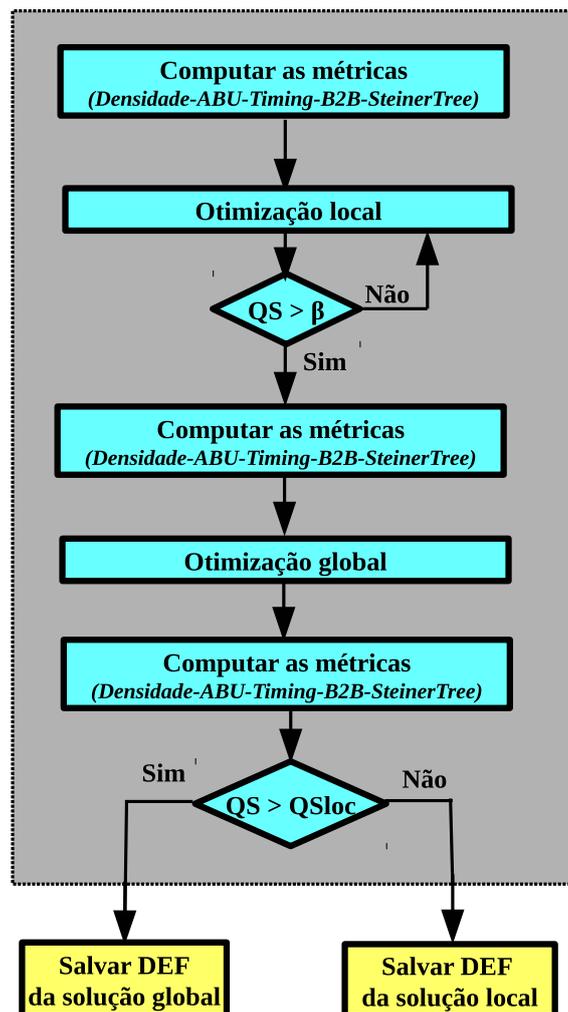
4.2.1 Fluxo de Execução no UPlace para a Minimização de Violações no Tempo de Propagação do Sinais

No fluxo de execução do UPlace para a minimização de violações no tempo de propagação do sinais foi utilizado somente o estágio de posicionamento detalhado, conforme mostrado na Figura 4.3. Nesse fluxo foi incluído um algoritmo de legalização para manter o circuito legal a cada mudança de posição das portas lógicas. Dessa maneira, os estágios de posicionamento global e legalização não foram integrados nesse fluxo de execução implementado, pois os circuitos já estavam posicionados (etapa de posicionamento global) e legalizados. Assim, o objetivo com esse fluxo é de resolver violações de propagação dos sinais somente no estágio de posicionamento detalhado. O fluxo apresentado na Figura 4.3 é a descrição do estágio de posicionamento detalhado mostrado na Figura 4.2.

O primeiro passo, no posicionamento detalhado, é obter os dados das métricas de qualidade do circuito que são utilizadas para avaliar a qualidade do posicionamento com o algoritmo analítico proposto. Para isso são computados a densidade de utilização da área do circuito, obtida a ABU e a penalidade da mesma, calculado o tempo de propagação dos sinais (*timing*), atualizado os limites das B2Bs das portas lógicas e computado as árvores de Steiner para todas as redes.

O passo seguinte é realizar a otimização do *skew* do relógio para o circuito. Os registradores são movidos com o objetivo de minimizar as violações no tempo de propagação

Figura 4.3 - Diagrama do fluxo de execução do posicionador detalhado



Fonte: figura elaborada pelo autor.

dos sinais e também para alterar o tempo *skew* no pino de relógio dele. Aqueles com as violações mais graves são os primeiros a serem movimentados. A decisão sobre se um movimento é benéfico deve considerar todas as métricas utilizadas para avaliar a qualidade da solução (QS) de posicionamento do circuito. O *skew* dos registradores é alterado enquanto a qualidade da solução do posicionamento for menor do que um determinado valor estabelecido. Durante o passo de otimização do *skew* da árvore de relógio é computada dinamicamente a condição de parada para esse algoritmo.

Na etapa de redução das violações no tempo de propagação dos sinais de dados desse fluxo é aplicado um algoritmo de posicionamento analítico quadrático somente para as portas lógicas combinacionais que estão em um determinado número de caminhos críticos e às vizinhas delas. Após esse passo as métricas são novamente computadas para obter o efeito positivo ou negativo em relação a qualidade do circuito posicionado.

Caso na etapa de redução das violações no tempo de propagação dos sinais de dados melhore a qualidade de posicionamento do circuito em relação ao resultado obtido na

etapa de otimização do *skew* do relógio (QSloc), o posicionamento resultante dessa etapa é salvo como a solução final. Senão, o posicionamento obtido após a otimização do *skew* da árvore de relógio é salvo como solução de posicionamento final para o circuito. O circuito está legalizado ao final de cada um desses dois passos nesse fluxo e é uma solução válida de acordo com as restrições definidas para o problema de *Incremental Timing-Driven Placement* (KIM; HUI; VISWANATHAN, 2014). Por causa disso, é possível salvar soluções intermediárias nesse fluxo. Elas são salvas de acordo com um critério de incremento na métrica de qualidade do posicionamento do circuito.

4.3 Considerações Finais

O algoritmo de posicionamento analítico detalhado minimiza as violações de propagação de sinal de *late*. As portas lógicas relacionadas a um conjunto de caminhos críticos são movidas com uma formulação analítica quadrática.

Nesse algoritmo ainda há espaço para adicionar pesos diferentes para as conexões que pertencem ao caminho crítico. A modificação do peso em cada uma dessas conexões modifica a influência da variável dela no sistema linear, ou seja, as portas lógicas que ela conecta podem ser aproximadas ou afastadas dependendo se o peso é aumentado ou reduzido para a conexão em questão. Como consequência, quando as portas lógicas do caminho crítico são aproximadas, a violação de propagação de *late* de um sinal tende a ser resolvida ou minimizada.

Na versão implementada, o algoritmo itera somente uma vez para um conjunto de portas lógicas de um pequeno número de caminhos críticos. Contudo, as iterações podem ser realizadas diversas vezes, adicionando forças às conexões críticas para aproximar ou afastar iterativamente as portas lógicas até não haver mais melhoria significativa nas métricas. As iterações também podem ser realizadas para diferentes conjuntos de portas lógicas pertencentes a diferentes caminhos críticos. Em resumo, as iterações do algoritmo podem ser realizadas diversas vezes sobre um conjunto de caminhos críticos e sobre diferentes conjuntos de caminhos críticos. No posicionamento global, os algoritmos de posicionamento analítico quadrático iteram diversas vezes sobre as portas lógicas do circuito. Entre as iterações há procedimentos para direcionar o deslocamento das mesmas. Portanto, no algoritmo de posicionamento analítico detalhado, há espaço para propor um procedimento que guie o deslocamento das portas lógicas pertencentes aos caminhos críticos.

5 RESULTADOS EXPERIMENTAIS

Neste capítulo serão apresentados e discutidos os principais resultados da validação experimental do algoritmo de posicionamento analítico detalhado proposto. Os resultados foram obtido utilizando a infraestrutura de *benchmarks* de (KIM; HUI; VISWANATHAN, 2014), exceto para os circuitos *mgc_edit_dist* e *mgc_matrix_mult* que foram publicados após o termino da presente dissertação.

5.1 Configuração Experimental

O algoritmo de posicionamento analítico detalhado foi implementado com a linguagem de programação C++ versão 11 e compilado junto com o ambiente UPlace com o g++ (GCC) versão 4.8.3. Os resultados experimentais foram obtidos ao executar o algoritmo junto com a ferramenta UPlace em uma máquina com a seguinte configuração:

- Sistema operacional Ubuntu 14.04.1 LTS
- kernel Linux 3.13.0-37-generic #64-Ubuntu SMP
- 16 GB de memória RAM
- Processador Intel(R) Core(TM) i7-3770 CPU @ 3.40GHz

As métricas para mensurar a qualidade do posicionamento de penalidade ABU, de WNS e de TNS foram medidas após a execução da otimização de *skew* do relógio e novamente após a otimização dos caminhos críticos com o algoritmo analítico detalhado. Na Figura 4.3 é apresentado o fluxo de otimização implementado e os pontos em que as métricas para avaliar a qualidade do posicionamento são medidas. Somente para o algoritmo de posicionamento analítico detalhado foi medido o tempo de execução e número de portas lógicas movimentadas pelo mesmo. Na otimização analítica dos caminhos críticos, o PAdE foi executado para otimizar o caminho com o *slack* mais negativo, 1 (WNS), para otimizar os 5 caminhos com os *slacks* mais negativos, para otimizar os 10 caminhos com os *slacks* mais negativos e também para otimizar os 25 caminhos e o 1% dos caminhos com os *slacks* mais negativos, respectivamente, cada configuração de caminhos críticos otimizados é representada por 1, 5, 10, 25, e 1%. Todos os caminhos analisados nesse

trabalho são somente com violações no tempo de propagação dos sinais de dados de *late*.

A métrica de qualidade proposta por (KIM; HUIJ; VISWANATHAN, 2014) para o posicionamento detalhado é uma média ponderada entre a penalidade de ABU, WNS e TNS. Os tempos de propagação são considerados para violações nos sinais de dados de *late* e *early*. O valor máximo obtido com essa métrica é de 1800, nessa situação todas as violações estão resolvidas. A métrica para avaliar a qualidade da solução de posicionamento detalhado é computada conforme apresentada na Equação 5.1.

$$QS = (w_{tns} \times (w_e \times \Delta tns_{early} + w_l \times \Delta tns_{late}) + w_{wns} \times (w_e \times \Delta wns_{early} + w_l \times \Delta wns_{late})) \times (1 - w_{abu} \times \Delta ABU_{penalty}) \quad (5.1)$$

Nessa equação $w_{tns} = 2$, $w_{wns} = 1$, $w_{abu} = 1$, $w_e = 1$ e $w_l = 5$ são os pesos para a média ponderada da qualidade da solução do posicionamento detalhado e w_e e w_l são os pesos para as violações de propagação de sinal de *early* e *late*, respectivamente. O resultado de cada métrica em Δtns e em Δwns é a diferença entre o obtido após o posicionamento detalhado e resultado dessa métrica para o circuito antes da execução de qualquer redução das violações de tempo de propagação e de penalidade de ABU.

Neste Capítulo serão discutidos os resultados das métricas para medir a qualidade do posicionamento detalhado apresentadas acima, o tempo médio de execução do algoritmo e o número de portas lógicas movimentadas pelo algoritmo de posicionamento analítico detalhado. Esses dados foram obtidos quando são tratados somente os caminhos com violação do tempo de propagação de *late*.

5.2 Circuitos para Teste (*Benchmarks*)

O algoritmo de otimização dos caminhos críticos para o posicionamento incremental detalhado guiado a temporização foi avaliado com cinco circuitos disponibilizados em (?). As principais características de cada circuito são apresentadas na Tabela 5.1. O número de portas lógicas varia entre, aproximadamente, 164 mil e 958 mil. Há circuito com macro blocos e a densidade máxima de utilização de área do circuito pode variar de 70% até 76% dependendo do circuito.

Os circuitos possuem duas configurações para deslocamento máximo, os quais são mostrados na Tabela 5.2. Essa limitação evita que o resultado do posicionamento global seja significativamente modificado. O período de relógio para cada circuito também é mostrado na Tabela 5.2. O período de relógio é o intervalo de tempo máximo que os sinais de dados têm para se propagar entre duas barreiras sequenciais.

Tabela 5.1 - Configuração dos circuitos de teste (*Benchmarks*)

	# Portas Lógicas	# Pinos	Macrobloco	Dimensão (μm)	Densidade
vga_lcd	164 891	184	Não	898,6 x 898	0.70
b16	219 268	47	Não	1187,2 x 1188	0.76
leon3mp	649 191	333	Não	1989,2 x 1990	0.70
leon2	794 286	700	Não	2086,4 x 2086	0.70
netcard	958 792	1846	Sim	2520,6 x 2520	0.72

Fonte: tabela elaborada pelo autor.

Tabela 5.2 - Deslocamento máximo permitido e período de relógio

Circuito	Deslocamento 1 (μm)	Deslocamento 2 (μm)	Período (μs)
vga_lcd	10	200	4
b16	20	200	5
leon3mp	30	300	37
leon2	40	400	65
netcard	50	400	45

Fonte: tabela elaborada pelo autor.

5.3 Resultados e Discussão

5.3.1 Métrica de Qualidade do Posicionamento Detalhado

Na Tabela 5.3 são apresentados os resultados absolutos da métrica de qualidade para o posicionamento detalhado. Esses dados foram obtidos para a distância máxima de movimentação restrita ao deslocamento 1 (Tabela 5.2). Na Tabela 5.3 são apresentados os resultados após a otimização do *skew* do relógio e ao minimizar as violações do tempo de propagação dos caminhos críticos com algoritmo analítico detalhado para os caminhos críticos com o número de 1, 5, 10, 25 e 1% dos caminhos com o *slack* mais negativo para as violações de *late*.

Após a otimização de *skew* do relógio é obtida a pontuação, em média, de 1504 para a métrica de qualidade do posicionamento detalhado. Esse valor é aproximadamente 83% do máximo possível. O resultado da otimização dos caminhos críticos com o algoritmo analítico tem média de 79,56% para 1 caminho crítico e decresce continuamente para 74,47% no 1% de caminhos críticos. Se desconsiderada a pontuação do netcard, a média fica praticamente constante em 85% da pontuação máxima para todas as configurações de caminhos críticos.

Em geral, a otimização dos caminhos críticos com o algoritmo analítico é mais efetiva para tratar um caminho crítico por vez. Conforme o número de caminhos críticos aumenta, a métrica de qualidade do posicionamento detalhado lentamente tende a piorar.

Tabela 5.3 - Métrica de qualidade do posicionamento detalhado para os caminhos críticos com violações de *late* no tempo de propagação do sinais de dados para o deslocamento máximo 1

Circuito	Skew	Número de Caminhos Críticos					Média
		1	5	10	25	1%	
vga_lcd	1454,3	1466,1	1454,5	1450,1	1441,9	1413,9	1445,3
b19	1500,4	1533,1	1546,3	1539,8	1513,8	1492,2	1525,1
leon3mp	1550,5	1545,5	1539,2	1534,7	1518,9	1461,9	1520,0
leon2	1585,6	1583,9	1583,0	1582,2	1579,4	1520,8	1569,9
netcard	1430,5	1032,1	1032,1	1030,1	1026,8	993,4	1022,9
Média	1504,3	1432,2	1431,0	1427,4	1416,2	1376,4	1416,6

Fonte: tabela elaborada pelo autor.

A piora significativa na métrica de qualidade do posicionamento detalhado do circuito netcard é causada pelo repentino aumento na penalidade de ABU.

Na Tabela 5.4 é mostrada, em porcentagem, a razão entre o resultado da métrica de qualidade de posicionamento detalhado obtida com o algoritmo de posicionamento analítico para os caminhos críticos dividido pela métrica obtida com o algoritmo de otimização do *skew* do relógio. Desse modo é possível inferir o impacto na métrica de qualidade do posicionamento detalhado causado pelo algoritmo de posicionamento analítico detalhado para os caminhos críticos.

Tabela 5.4 - Razão, em porcentagem, entre a otimização do algoritmo de posicionamento analítico detalhado e a otimização do *skew* do relógio da métrica de qualidade do posicionamento detalhado dos caminhos críticos de *late* para o deslocamento 1

Circuito	Número de Caminhos Críticos					Média
	1	5	10	25	1%	
vga_lcd	0,81	0,01	-0,29	-0,85	-2,78	-0,62
b19	2,18	3,06	2,63	0,90	-0,55	1,65
leon3mp	-0,32	-0,73	-1,02	-2,04	-5,71	-1,96
leon2	-0,11	-0,16	-0,22	-0,39	-4,09	-0,99
netcard	-27,85	-27,85	-27,99	-28,22	-30,55	-28,49

Fonte: tabela elaborada pelo autor.

O algoritmo de posicionamento analítico detalhado melhora, em média, 1,65% a métrica de qualidade de posicionamento detalhado para o circuito b19. Esse é o circuito com o maior ganho. Nos circuitos vga_lcd, leon3mp e leon2 há uma piora na métrica de qualidade do posicionamento detalhado não muito significativo, com o menor resultado de 5,71% ao tratar o 1% de caminhos críticos com maior tempo de propagação dos sinais de dados no leon3mp. No netcard houve uma degradação em torno de 30% na métrica

de qualidade do posicionamento detalhado. Esse valor mantém-se praticamente constante para todas as configurações de caminhos críticos de *late* desse circuito.

A métrica de qualidade de posicionamento detalhado também foi analisada com o algoritmo de posicionamento analítico para caminhos críticos para a restrição de deslocamento 2 e os resultados são apresentados na Tabela 5.5. Nessa configuração há um ganho maior nessa métrica do que com os resultados do deslocamento 1 para o vga_lcd e o b19. Contudo, a degradação da métrica de qualidade do posicionamento detalhado é mais significativa conforme um número maior de caminhos críticos são tratados simultaneamente. O netcard também tem significativa redução no valor da métrica de qualidade do posicionamento detalhado devido, principalmente, ao aumento da penalidade de ABU. Essa penalidade será analisada mais adiante neste capítulo.

Tabela 5.5 - Métrica de qualidade do posicionamento detalhado dos caminhos críticos *late* para o deslocamento 2

Circuito	Skew	Número de Caminhos Críticos					Média
		1	5	10	25	1%	
vga_lcd	1454,3	1521,1	1473,8	1462,5	1442,7	1328,0	1445,6
b19	1500,4	1612,7	1608,6	1565,5	1571,3	1524,7	1576,5
leon3mp	1550,5	1543,9	1535,2	1521,3	1490,1	1293,9	1476,9
leon2	1585,6	1581,9	1579,6	1576,4	1571,6	1444,1	1550,7
netcard	1430,5	1060,8	1029,8	1026,8	1019,5	996,8	1026,7
Média	1504,2	1464,1	1445,4	1430,5	1419,0	1317,5	1415,3

Fonte: tabela elaborada pelo autor.

A média de pontuação da métrica de qualidade do posicionamento detalhado é mais alta na liberdade de deslocamento 2 do que na liberdade de deslocamento 1. Em média, a pontuação é de 81,33% do máximo possível para o 1 caminho crítico e decresce para 73,19% no 1% de caminhos críticos para a liberdade de deslocamento 2. Quando desconsiderado os resultados do netcard, a média permanece em torno de 86% da máxima pontuação para os caminhos críticos de 1 a 25, com uma queda aproximada de 2% para a liberdade de deslocamento 2. No caso dos caminhos críticos de 1% há uma queda acentuada para 77,6% da pontuação máxima para a liberdade de deslocamento 2.

Na Tabela 5.6 é apresentada a variação, em porcentagem, na métrica de qualidade do posicionamento detalhado. Esse dado é obtido ao dividir o valor da métrica de qualidade do posicionamento detalhado de cada configuração de caminho crítico da otimização com o algoritmo analítico dos caminhos críticos pelo valor resultante da otimização de *skew* do relógio. O algoritmo de otimização analítico dos caminhos críticos é mais efetivo nessa configuração, devido a um grau maior de liberdade de movimento das portas lógicas. Ao tratar o caminho crítico 1, a métrica de qualidade do posicionamento detalhado melhora em 4,59% e 7,48%, respectivamente, para o vga_lcd e b19. A métrica de qualidade do

posicionamento detalhado não sofre alteração significativa com a otimização com o algoritmo analítico dos caminhos críticos para o deslocamento 2, exceto para o netcard e para a configuração de 1% de caminhos críticos.

Tabela 5.6 - Razão, em porcentagem, entre a otimização com o algoritmo analítico dos caminhos críticos e a otimização com o algoritmo de *skew* do relógio da métrica de qualidade do posicionamento detalhado dos caminhos críticos de *late* para o deslocamento 2

Circuito	Número de Caminhos Críticos					Média
	1	5	10	25	1%	
vga_lcd	4,59	1,34	0,56	-0,80	-8,69	-0,60
b19	7,48	7,21	4,34	4,72	1,62	5,08
leon3mp	-0,43	-0,99	-1,88	-3,90	-16,55	-4,75
leon2	-0,23	-0,38	-0,58	-0,89	-8,93	-2,20
netcard	-25,85	-28,01	-28,22	-28,73	-30,32	-28,22

Fonte: tabela elaborada pelo autor.

5.3.2 Penalidade de ABU

A métrica de ABU mede a concentração de portas lógicas em uma região. Um limite máximo de utilização de área do circuito é estabelecido para o circuito, conforme apresentado anteriormente na Tabela 5.1. Caso o limite seja ultrapassado, é aplicada uma penalidade e esta é considerada na métrica de qualidade de posicionamento detalhado proposta por (KIM; HUI; VISWANATHAN, 2014).

Nesse experimento foi executado o algoritmo de otimização analítico dos caminhos críticos, apesar das violações no tempo de propagação já estarem resolvidas após a etapa de otimização local em alguns circuitos. Os caminhos considerados nessa situação são aqueles com o maior tempo de propagação. O objetivo de incluir os circuitos sem violações é ter um corpo de dados maior ao analisar o algoritmo proposto e também avaliar o impacto do algoritmo proposto sobre a métrica de penalidade de ABU.

Na Tabela 5.7 é apresentada a penalidade de ABU ao violar o limite máximo de densidade de utilização da área do circuito para a liberdade de movimento com o deslocamento 1. Os resultados foram obtidos após a execução do algoritmo de otimização do *skew* do relógio e após a execução do algoritmo analítico detalhado para os caminhos críticos de *late*.

A variação da violação de densidade de utilização de área do circuito não é significativa entre a otimização do *skew* do relógio e após a minimização das violações de propagação dos sinais de dados com o algoritmo analítico detalhado para todos os circuitos e configurações de caminhos críticos. Em todas as configurações de caminhos críticos, o netcard tem a penalidade de ABU de aproximadamente 0,30 após a minimização das

Tabela 5.7 - Penalidade de ABU dos caminhos críticos *late* ($\times 10^{-2}$) para o deslocamento 1

Circuito	Skew	Número de Caminhos Críticos					Média
		1	5	10	25	1%	
vga_lcd	1,26	1,36	2,17	2,51	3,08	4,91	2,81
b19	2,59	2,59	2,59	2,59	2,66	4,09	2,91
leon3mp	0,78	1,10	1,51	1,80	2,82	6,50	2,75
leon2	2,46	2,56	2,62	2,67	2,85	6,55	3,45
netcard	1,13	28,57	28,66	28,80	29,03	31,37	29,29

Fonte: tabela elaborada pelo autor.

violações do tempo de propagação dos caminhos críticos com o algoritmo analítico detalhado. Esse circuito tem a peculiaridade das portas lógicas estarem concentradas em uma parte da área de posicionamento. No caso do netcard, o aumento na concentração de portas lógicas pode causar uma grande variação na densidade de utilização de área do circuito. A minimização analítica dos caminhos críticos tende a concentrar as portas lógicas relacionadas a eles. Essas são as principais razões para o incremento significativo na penalidade de ABU.

A porcentagem de variação da penalidade de densidade de utilização de área do circuito é apresentada na Tabela 5.8 para a liberdade de movimentação com o deslocamento 1. A variação de penalidade de ABU foi obtida ao dividir o resultado dela após a otimização com o algoritmo analítico dos caminhos críticos pelo resultado da mesma após a otimização do *skew* do relógio.

Tabela 5.8 - Razão, em porcentagem, entre a otimização com o algoritmo analítico dos caminhos críticos e após a otimização do *skew* do relógio das violações de ABU dos caminhos críticos de *late* para o deslocamento 1

Circuito	Número de Caminhos Críticos					Média
	1	5	10	25	1%	
vga_lcd	1,08	1,73	1,99	2,45	3,90	2,23
b19	1,00	1,00	1,00	1,03	1,58	1,12
leon3mp	1,41	1,93	2,31	3,62	8,33	3,52
leon2	1,04	1,07	1,09	1,16	2,67	1,40
netcard	25,27	25,35	25,48	25,68	27,75	25,91

Fonte: tabela elaborada pelo autor.

A penalidade de ABU cresce à medida que mais caminhos críticos são considerados simultaneamente. Contudo, o crescimento da penalidade de ABU não é significativo, exceto para o leon3mp em 1% dos caminhos críticos e para o circuito netcard. Portanto, aplicar o algoritmo analítico para resolver as violações de propagação não impacta signi-

ficativamente na solução do posicionamento global.

A métrica de penalidade da ABU também foi medida para a movimentação das portas lógicas com o deslocamento 2, conforme os resultados são apresentados na Tabela 5.9. Os resultados absolutos mostram que há uma concentração um pouco maior das portas lógicas quando comparado com o deslocamento 1. Porém, a concentração ainda é significativamente baixa na grande maioria dos circuitos e na maioria das configurações de caminhos críticos. A exceção de baixa penalidade de ABU é o circuito netcard, a concentração dele é muito significativa, especialmente para a configuração de portas lógicas em 1 % de caminhos críticos. Nesse caso pode ser impossível obter um circuito legal na região ou pode inviabilizar que as etapas seguintes ao posicionamento atinjam os objetivos.

Tabela 5.9 - Violações de ABU dos caminhos críticos de *late* ($\times 10^{-2}$) para o deslocamento 2

Circuito	Skew	Número de Caminhos Críticos					Média
		1	5	10	25	1%	
vga_lcd	1,26	1,58	2,99	4,04	5,24	11,98	5,16
b19	2,59	2,66	2,66	2,63	2,72	5,68	3,27
leon3mp	0,78	1,21	1,77	2,67	4,69	17,35	5,54
leon2	2,46	2,69	2,84	3,04	3,34	11,39	4,66
netcard	1,13	28,69	28,98	29,20	29,70	33,25	29,97

Fonte: tabela elaborada pelo autor.

Na Tabela 5.10 é apresentada, em porcentagem, a variação da penalidade ABU. Essa variação é obtida ao dividir a penalidade de ABU após a otimização com o algoritmo analítico detalhado pela obtida após a otimização do *skew* do relógio. As variações de penalidade mais significativas são quando é tratado o 1% de caminhos críticos e para todas as configurações de caminhos críticos para o circuito netcard.

Tabela 5.10 - Razão, em porcentagem, entre a otimização com o algoritmo de posicionamento analítico detalhado e após a otimização do *skew* do relógio das violações de ABU dos caminhos críticos de *late* para o deslocamento 2

Circuito	Número de Caminhos Críticos					Média
	1	5	10	25	1%	
vga_lcd	1,26	2,37	3,21	4,16	9,51	4,10
b19	1,02	1,02	1,01	1,05	2,19	1,26
leon3mp	1,55	2,27	3,42	6,00	22,24	7,10
leon2	1,09	1,15	1,24	1,36	4,64	1,90
netcard	25,38	25,64	25,83	26,28	29,41	26,51

Fonte: tabela elaborada pelo autor.

5.3.3 WNS para Violações de *Late*

A seguir são apresentados e discutidos os resultados para a métrica de WNS ao reduzir as violações de propagação dos caminhos críticos com o algoritmo analítico detalhado. Nesse experimento foi avaliado o impacto de atuar somente sobre o caminho crítico com o maior tempo de propagação e também sobre os *ns* caminhos com as maiores violações de *slack*.

A violação de WNS foi computada ao final da otimização do *skew* do relógio e novamente após a minimização das violações de propagação dos sinais de dados nos caminhos críticos com o algoritmo de posicionamento analítico detalhado. Na Tabela 5.11 é apresentada a variação do WNS, em nanosegundos, para a liberdade de movimento com o deslocamento 1. As violações de WNS para os circuitos *leon3mp* e *leon 2* foram resolvidas ainda na etapa de otimização do *skew* do relógio. No circuito *vga_lcd*, o WNS é, em média, $-0,33$ *ns* para todas as configurações de caminhos críticos. No circuito *b19*, o valor de WNS não é uniforme para as configurações de caminhos críticos, mas em todos os casos há uma pequena redução nas violações. O algoritmo de posicionamento analítico detalhado causou no circuito *netcard* um aumento na violação de WNS em relação à otimização do *skew* do relógio.

Tabela 5.11 - WNS de *late* (*ns*) para o deslocamento 1

Circuito	<i>Skew</i>	Número de Caminhos Críticos					Média
		1	5	10	25	1%	
vga_lcd	-0,365	-0,330	-0,330	-0,329	-0,329	-0,332	-0,330
b19	-0,279	-0,235	-0,209	-0,220	-0,261	-0,261	-0,237
leon3mp	0	0	0	0	0	0	0
leon2	0	0	0	0	0	0	0
netcard	-0,787	-0,857	-0,847	-0,847	-0,847	-0,847	-0,849

Fonte: tabela elaborada pelo autor.

Na Tabela 5.12 é apresentada, em porcentagem, a razão entre a violação de WNS do algoritmo de posicionamento analítico detalhado para a minimização das violações no tempo de propagação dos sinais de dados nos caminhos críticos pelo resultado da otimização do *skew* do relógio para o deslocamento 1. Nos circuitos *vga_lcd* e *b19* houve uma redução na violação de propagação dos sinais de dados de WNS em relação ao algoritmo de otimização do *skew* do relógio de até 25,33%. Contudo, a violação de propagação dos sinais de dados de WNS piorou pelo mínimo 7,66% para o circuito *netcard*.

A violação de WNS foi solucionada ainda durante a execução do algoritmo de otimização do *skew* do relógio nos circuitos *leon3mp* e *leon2*. A otimização com o algoritmo de posicionamento analítico detalhado resultou na redução de violação de WNS para os circuitos *vga_lcd* e *b19*. A redução na violação de tempo de propagação no circuito *vga_lcd*

Tabela 5.12 - Razão, em porcentagem, entre a otimização com o algoritmo de posicionamento analítico detalhado e a otimização do *skew* do relógio do WNS *late* - Deslocamento 1

Circuito	Número de Caminhos Críticos					Média
	1	5	10	25	1%	
vga_lcd	9,55	9,62	9,89	9,93	9,17	9,63
b19	16,00	25,33	21,23	6,50	6,50	15,11
leon3mp	-	-	-	-	-	-
leon2	-	-	-	-	-	-
netcard	-8,93	-7,73	-7,73	-7,73	-7,66	-7,96

Fonte: tabela elaborada pelo autor.

é praticamente constante para todas as configurações de caminhos críticos. Em contrapartida, para o circuito b19 há uma oscilação dependendo da configuração do número de caminhos críticos tratados na redução da violação no tempo de propagação. Para o circuito netcard houve uma piora na violação do tempo de propagação de WNS para todas as configurações de caminhos críticos.

Os resultados absolutos de WNS são apresentados na Tabela 5.13 para o deslocamento 2. Nos circuitos vga_lcd e b19 houve a redução da violação para todas as configurações do número de caminhos críticos com a otimização com o algoritmo de posicionamento analítico detalhado. No circuito netcard, a violação do tempo de propagação de WNS foi reduzida ao tratar os caminhos críticos 1 e 1%. Contudo, para as demais configurações de caminhos críticos nesse circuito houve o aumento da violação do tempo de propagação de WNS.

Tabela 5.13 - WNS de *late* (*ns*) para o deslocamento 2

Circuito	<i>Skew</i>	Número de Caminhos Críticos					Média
		1	5	10	25	1%	
vga_lcd	-0,365	-0,177	-0,249	-0,239	-0,247	-0,289	-0,240
b19	-0,279	-0,101	-0,109	-0,182	-0,168	-0,168	-0,146
leon3mp	0	0	0	0	0	0	0
leon2	0	0	0	0	0	0	0
netcard	-0,787	-0,499	-0,822	-0,822	-0,823	-0,471	-0,687

Fonte: tabela elaborada pelo autor.

A porcentagem de variação sobre o WNS é obtida após a execução da otimização do *skew* do relógio e após a execução da otimização com o algoritmo de posicionamento analítico detalhado para o deslocamento 2. O WNS resultante após execução do algoritmo de posicionamento analítico detalhado é dividido pelo resultado da otimização do *skew* do

relógio. A exceção é para os circuitos leon3mp e leon2 em que as violações foram corrigidas somente com a otimização do *skew* do relógio.

Na Tabela 5.14 é apresentada a razão do WNS de *late* da otimização analítica dos caminhos críticos pela otimização do *skew* da árvore de relógio. Nos circuitos leon3mp e leon2 somente a otimização de *skew* na árvore de relógio foi suficiente para resolver as violações no tempo de propagação de WNS. Em todas as configurações de números de caminhos críticos nos circuitos vga_lcd e b19 há redução nas violações de tempo de propagação de WNS para *late* após a otimização com a técnica analítica. No circuito netcard houve redução nas violações de tempo de propagação de *late* somente para as configurações de números de caminhos críticos de 1 e 1%, nos demais casos a técnica analítica aumentou as violações de tempo de propagação de *late*.

Tabela 5.14 - Razão, em porcentagem, entre a otimização dos caminhos críticos com o algoritmo analítico detalhado e a otimização do *skew* do relógio do WNS de *late* para o deslocamento 2

Circuito	Número de Caminhos Críticos					Média
	1	5	10	25	1%	
vga_lcd	51,41	31,75	34,66	32,49	20,73	34,21
b19	63,73	61,06	34,98	39,87	39,87	47,90
leon3mp	-	-	-	-	-	-
leon2	-	-	-	-	-	-
netcard	36,53	-4,52	-4,52	-4,57	40,12	12,61

Fonte: tabela elaborada pelo autor.

Os ganhos mais significativos ao tentar resolver as violações de WNS são obtidos quando é tratado somente o pior caminho crítico. Nos demais casos, não há um ganho tão significativo quanto ao do pior caminho, pois esse caminho com o pior atraso crítico não é necessariamente o mesmo obtido na otimização do *skew* do relógio e na otimização dos caminhos críticos com o algoritmo analítico. O caminho de WNS pode ser um terceiro caminho obtido ao analisar o circuito com STA após a otimização com o algoritmo analítico, que não era parte da configuração de caminhos críticos ao executar o algoritmo de posicionamento analítico detalhado.

5.3.4 TNS para Violações de *Late*

A métrica de TNS foi avaliada após a otimização do *skew* do relógio e após a minimização das violações dos caminhos críticos com o algoritmo analítico detalhado. A movimentação das portas lógicas foi limitada com os deslocamentos máximos 1 e 2 apresentados na Tabela 5.2.

Na Tabela 5.15 é apresentado as violações de TNS dos circuitos com deslocamento 1 para todas as configurações de caminhos críticos. Nos circuitos leon3mp e leon2, somente

a otimização do *skew* na árvore de relógio foi suficiente para resolver todas as violações de tempo de propagação dos sinais de dados. Nesse caso, ao executar o algoritmo de posicionamento analítico, o circuito está sem nenhuma violação no tempo de propagação dos sinais de dados.

Tabela 5.15 - TNS de *late* (*ns*) para o deslocamento 1

Circuito	<i>Skew</i>	Número de Caminhos Críticos					Média
		1	5	10	25	1%	
vga_lcd	-2,405	-2,288	-2,173	-2,125	-1,970	-2,081	-2,127
b19	-0,776	-0,562	-0,531	-0,556	-0,671	-0,671	-0,598
leon3mp	0	0	0	0	0	0	0
leon2	0	0	0	0	0	0	0
netcard	-1,773	-1,895	-1,667	-1,667	-1,667	-1,666	-1,712

Fonte: tabela elaborada pelo autor.

Nos circuitos *vga_lcd*, *b19* e *netcard* houve redução na violação de TNS para todas as configurações de caminhos críticos, exceto no *netcard* com 1 caminho. A redução na violação é maior na configuração com os 5 caminhos com maior atraso no tempo de propagação.

O resultado das violações de TNS após a minimização com o algoritmo de posicionamento analítico detalhado foi dividido pelo resultado obtido após a otimização do *skew* do relógio. Na Tabela 5.16 é apresentada a variação, em porcentagem, ao movimentar as portas lógicas com o deslocamento 1. O ganho em reduzir a violação é praticamente constante até 10 caminhos críticos no circuito *b19*. Nas outras duas configurações há uma significativa redução no ganho ao minimizar as violações nesse circuito. No circuito *vga_lcd* há uma significativa melhora na redução de violação do tempo de propagação de TNS até 25 caminhos críticos. O ganho na redução de violação de TNS no circuito *netcard* é praticamente constante, com exceção do pior caminho em que houve aumento significativo nessa violação.

A variação de violação do tempo de propagação de TNS foi medida para todos os circuitos e todas as configurações de caminho crítico para a liberdade de deslocamento 2. Os resultados absolutos são apresentados na Tabela 5.17. Eles foram medidos após a execução da otimização do *skew* do relógio e após a minimização das violações do tempo de propagação dos caminhos críticos com o algoritmo analítico detalhado. Com a liberdade de movimento maior, reduziu-se mais significativamente a violação de TNS em comparação com o deslocamento 1. O valor discrepante de TNS do pior caminho crítico entre o deslocamento 1 e 2 deve-se a alguns movimentos de portas lógicas proibidos no primeiro deslocamento e permitido no segundo.

A razão de variação de TNS, em porcentagem, é apresentada na Tabela 5.18. Os valores são obtidos somente para os circuitos com violação. O resultado de TNS após a

Tabela 5.16 - Razão, em porcentagem, entre a minimização das violações de TNS com o algoritmo analítico detalhado e a otimização do *skew* do relógio do TNS para o *late* e para o deslocamento 1

Circuito	Número de Caminhos Críticos					Média
	1	5	10	25	1%	
vga_lcd	4,87	9,66	11,64	18,09	13,47	11,55
b19	27,58	31,62	28,43	13,59	13,59	22,97
leon3mp	-	-	-	-	-	-
leon2	-	-	-	-	-	-
netcard	-6,88	5,96	5,96	5,96	6,03	3,41

Fonte: tabela elaborada pelo autor.

Tabela 5.17 - TNS de *late* (*ns*) para o deslocamento 2

Circuito	<i>Skew</i>	Número de Caminhos Críticos					Média
		1	5	10	25	1%	
vga_lcd	-2,405	-1,906	-1,763	-1,443	-0,811	-0,998	-1,384
b19	-0,776	-0,195	-0,208	-0,402	-0,382	-0,382	-0,314
leon3mp	0	0	0	0	0	0	0
leon2	0	0	0	0	0	0	0
netcard	-1,773	-1,069	-1,499	-1,499	-1,500	-0,726	-1,258

Fonte: tabela elaborada pelo autor.

otimização com o algoritmo de posicionamento analítico detalhado é dividido pelo resultado obtido após a otimização do *skew* do relógio. Nos circuitos b19 e netcard, o maior ganho ao reduzir a violação de TNS ocorreu ao tratar a configuração de 1 caminho crítico. No caso do circuito vga_lcd o maior ganho foi com 25 caminhos críticos.

Tabela 5.18 - Razão, em porcentagem, entre a minimização de violações no tempo de propagação com o algoritmo de posicionamento analítico detalhado e a otimização do *skew* do relógio de TNS para *late* para o deslocamento 2

Circuito	Número de Caminhos Críticos					Média
	1	5	10	25	1%	
vga_lcd	20,75	26,69	40,00	66,27	58,49	42,44
b19	74,91	73,16	48,21	50,86	50,86	59,60
leon3mp	-	-	-	-	-	-
leon2	-	-	-	-	-	-
netcard	39,70	15,46	15,46	15,40	59,07	29,02

Fonte: tabela elaborada pelo autor.

5.3.5 Número de Portas Lógicas Movimentadas pelo Algoritmo de Posicionamento Analítico Detalhado

O número de portas lógicas móveis foi medido ao executar o algoritmo de minimização das violações dos caminhos críticos com a técnica analítica detalhada. Esse dado é apresentado na Tabela 5.19. O algoritmo de posicionamento analítico detalhado considera como portas lógicas móveis somente aquelas combinacionais que estão entre o ponto de início e ponto final de um caminho crítico. No algoritmo proposto também são consideradas portas lógicas móveis as combinacionais que são vizinhas imediatas daquelas combinacionais que estão entre o ponto de início e ponto de fim do caminho crítico. Nas configurações de 1 a 25 caminhos críticos são movimentadas no máximo até pouco mais de duas centenas de portas lógicas. Esse número de portas lógicas móveis é bem pequeno em comparação ao número total delas nos circuito que varia de 164 mil a 958 mil. Na configuração de 1% de caminhos críticos há um acréscimo de até algumas unidades de milhar de portas lógicas móveis.

Tabela 5.19 - Número de portas lógicas móveis para os caminhos críticos com violações de *late*

Circuito	# Caminhos Críticos				
	1	5	10	25	1%
vga_lcd	14	50	96	204	1.112
b19	45	77	119	236	613
leon3mp	22	49	64	223	3.189
leon2	18	41	77	162	6.897
netcard	21	38	86	163	5.223

Fonte: tabela elaborada pelo autor.

Apesar das violações de propagação de *late* serem resolvidas durante a otimização do *skew* do relógio para os circuitos leon3mp e leon2, optou-se por executar o algoritmo de posicionamento analítico detalhado para as configurações de caminhos críticos com os maiores tempos de propagação dos sinais de dados. Desse modo, é possível medir o número de portas lógicas que são movimentadas com esse algoritmo. Esse número não é tão discrepante daquele obtido nos circuitos com violações.

5.3.6 Tempo Médio de Execução do Algoritmo de Posicionamento Analítico Detalhado

O tempo de execução e o desvio padrão foram mensurados para o algoritmo de posicionamento analítico detalhado. Os dados são apresentados na Tabela 5.20 e na Figura 5.1. O tempo de execução é praticamente constante para todas as configurações de caminhos críticos. Esse fato decorre da necessidade de percorrer diversas vezes todas as portas

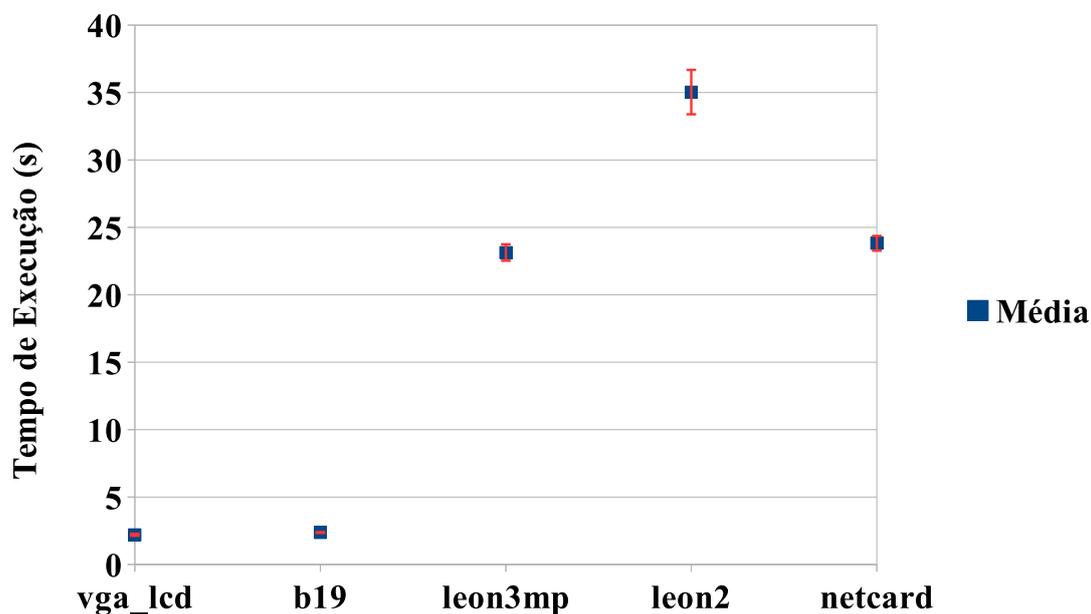
lógicas do circuito. Essa é uma a operação que demanda praticamente todo o tempo de execução. Nesse ponto há espaço para otimizar o tempo de execução do algoritmo de posicionamento analítico detalhado.

Tabela 5.20 - Média e desvio padrão dos tempos de execução dos caminhos críticos

Circuito	Média (s)	σ
vga_lcd	2,23	0,070
b19	2,40	0,044
leon3mp	23,09	0,545
leon2	35,06	1,492
netcard	23,58	0,284

Fonte: tabela elaborada pelo autor.

Figura 5.1 - Tempo médio de execução e desvio padrão para o algoritmo de posicionamento analítico detalhado



Fonte: figura elaborada pelo autor

A execução do circuito leon2 é o que tem o maior desvio padrão, com aproximadamente 1,5 segundo. Por outro lado, o circuito b19 é o que possui o menor desvio padrão, com 0,044 segundo. A execução do algoritmo de posicionamento analítico detalhado para os circuitos vga_lcd e b19 precisa de menos de 2,5 segundos. A execução do algoritmo proposto para o leon2 é o que demanda mais tempo, com 35,06 segundos.

5.4 Considerações Finais

Nesse capítulo foram apresentados os resultados experimentais para avaliar o algoritmo de posicionamento analítico detalhado. O escopo do algoritmo é buscar a solução para as violações de propagação dos sinais de *late*. Esse algoritmo trata simultaneamente um conjunto de portas lógicas relacionadas a um determinado número de caminhos críticos. As conexões entre as portas lógicas dos caminhos críticos são tratadas simultaneamente com o algoritmo proposto. Ao contrário de um algoritmo de otimização mais local, que busca individualmente o melhor posicionamento para cada porta lógica, no algoritmo proposto busca-se a melhor posição de cada porta lógica do caminhos crítico considerando as conexões locais, entre portas lógicas vizinhas, e o caminho crítico (conexão de sucessivas portas lógicas).

Os dados foram obtidos ao executar o algoritmo de minimização das violações no tempo de propagação com o algoritmo proposto após as mesmas violações já terem sido minimizadas com a otimização do *skew* do relógio. Os 5 circuitos utilizados possuem entre 164 mil e 958 mil portas lógicas. Eles são parte da infraestrutura disponibilizada por (?).

Em geral, o algoritmo de posicionamento analítico detalhado conseguiu reduzir as violações na maioria dos casos, apesar de que, em cada, a posição das portas lógicas foi movida somente com uma iteração. Esse algoritmo pode ter resultados promissores ao completá-lo com um fluxo de execução para que seja possível reduzir as violações de propagação de *late* com mais de uma iteração do algoritmo. Há também a possibilidade de adicionar pesos diferenciados para as conexões de caminhos críticos.

O algoritmo de posicionamento analítico detalhado tem o potencial de reduzir as violações de WNS e o TNS mesmo depois da execução do algoritmo de otimização do *skew* do relógio, que já minimiza esse tipo de violações, e depois de ter o posicionamento com um bom resultado de comprimento de fio (resultado do estágio de posicionamento global). As pontuações ruins na métrica para avaliar a qualidade do posicionamento detalhado são causadas principalmente pelas violações de ABU. Um algoritmo dedicado e consciente da criticalidade das portas lógicas pode ser utilizado para resolver as violações de ABU.

As soluções de posicionamento obtidas com o algoritmo de posicionamento analítico detalhado são muito rápidas. Assim é possível obter soluções de posicionamento especulativamente para diversas configurações de números de caminhos críticos (por exemplo: 1, 5, 10, 25, 1%, etc.) e selecionar a solução com melhor métrica que avalia a qualidade do posicionamento detalhado.

6 CONCLUSÕES E TRABALHOS FUTUROS

6.1 Conclusões

Neste trabalho foi apresentada a proposta de um algoritmo analítico para tratar violações no tempo de propagação dos sinais para a etapa de posicionamento detalhado. O algoritmo proposto é a adaptação de um algoritmo de posicionamento analítico quadrático da etapa de posicionamento global para posicionar as portas lógicas na etapa de posicionamento detalhado. O algoritmo de posicionamento analítico detalhado proposto movimenta somente as portas lógicas combinacionais pertencentes aos caminhos críticos e aquelas que estão conectadas no primeiro nível lógico, vizinhas imediatas, às portas lógicas combinacionais pertencentes aos caminhos críticos.

A aplicação de um algoritmo analítico quadrático nas portas lógicas relacionadas com os caminhos críticos move-as para uma posição de equilíbrio. Essa posição, em muitos casos, reduz o comprimento de fio de redes. Isso reduzirá o tempo de propagação dos sinais uma vez que o tempo de propagação está também diretamente relacionado com a resistência do fio e com a capacitância do mesmo. As quais são maiores quanto maior for o comprimento de fio.

O algoritmo de posicionamento analítico detalhado foi implementado com a linguagem de programação C++ versão 11. Esse algoritmo foi integrado ao *framework* UPlace. No fluxo de execução implementado, antes da execução do algoritmo de posicionamento analítico detalhado é otimizado o *skew* do relógio. Na versão desse algoritmo apresentada nesse trabalho é executada somente uma iteração do algoritmo. Ou seja, o sistema de equações lineares é resolvido somente uma vez. O algoritmo proposto opera sobre um circuito já otimizado e em alguns casos somente a otimização do *skew* do relógio já é suficiente para eliminar as violações no tempo de propagação dos sinais de dados. Junto com o algoritmo de posicionamento analítico detalhado foi integrado o legalizador Jezz. Uma vez que a solução do sistema de equações lineares resulta em novas posições para as portas lógicas que não são posições legalizadas. Desse modo, a legalização após a aplicação do algoritmo analítico detalhado é necessário para garantir que o circuito não tenha violações de *Design Rule Checking* (DRC).

Os resultados experimentais foram obtidos ao executar o algoritmo de posicionamento analítico detalhado com os *benchmarks* disponibilizados por (?). Foram utilizados somente 5 circuitos para validar experimentalmente o algoritmo proposto, uma vez que dois circuitos ainda não haviam sido publicados quando o algoritmo proposto neste trabalho foi avaliado. Nos experimentos realizados sobre o algoritmo de posicionamento analítico detalhado foram analisados: a métrica de qualidade do posicionamento detalhado proposta por (KIM; HUI; VISWANATHAN, 2014), a penalidade de ABU, as violações de WNS e TNS, o número de portas lógicas movidas e o tempo médio de execução do algoritmo. Os resultados experimentais foram obtidos após a otimização do *skew* do relógio. O algoritmo analítico detalhado foi executado para minimizar as violações de propagação dos sinais de dados para *late*. Os dados analisados nesse trabalho foram obtido com a execução do algoritmo proposto para dois limites de deslocamento máximo das portas lógicas, um deles bem restrito e o segundo com mais liberdade de movimentação das portas lógicas. As limitações de deslocamento máximo das portas lógicas foram apresentadas na Tabela 5.2.

Os resultados experimentais mostraram que o algoritmo de posicionamento analítico detalhado incrementa em até 5% na métrica de qualidade do posicionamento detalhado. Por outro lado, há um incremento não muito significativo de até 5,5% na penalidade de ABU. Isso significa que há um pequeno aumento na concentração das portas lógicas dos circuitos analisados. Quanto às violações de WNS e TNS há uma redução de até, respectivamente, 47,9% e 59,6% no tempo de propagação dos sinais de dados. O número total de portas lógicas movimentadas pelo algoritmo de posicionamento analítico detalhado são inferiores a 1% do total de portas lógicas de cada circuito analisado. Uma vez que são pouquíssimas as portas lógicas movimentadas com o algoritmo de posicionamento analítico detalhado, o sistema de equações lineares a ser resolvido também é bem pequeno, o que resulta em um baixo tempo de execução. O maior tempo de execução obtido experimentalmente foi de 35,06 segundos com o desvio padrão de 1,49 segundo.

6.2 Trabalhos Futuros

O fluxo de execução do algoritmo de posicionamento analítico detalhado ainda precisa ser completado. Ele pode iterar sobre diversos caminhos críticos e também realizar várias iterações sobre o mesmo conjunto de caminhos críticos. Nesse caso, também pode ser alterado o peso sobre as conexões críticas com o objetivo de minimizar as violações de tempo de propagação.

As demais métricas para avaliar a qualidade do posicionamento do circuito, como densidade de utilização de área, penalidade de ABU, etc, precisam ser consideradas quando uma porta lógica é movimentada com o algoritmo proposto neste trabalho. Desse modo, um movimento prejudicial pode ser evitado. Nesse caso, com mais de uma iteração do

algoritmo por configuração de caminho crítico, pode ser obtido um resultado de posicionamento em que seja possível um ganho maior do que o atual nas métricas avaliadas para medir a qualidade do posicionamento.

A topologia do caminho crítico deve ter significativa influência sobre os resultados das métricas analisadas com o algoritmo de posicionamento analítico detalhado. Então, é necessário investigar quais são as topologias de caminhos em que esse algoritmo obtém os melhores resultados. Para as demais topologias, analisar se é possível utilizar a metodologia analítica quadrática para minimizar as violações nos tempos de propagação dos sinais.

Os caminhos com violação no tempo de propagação dos sinais de *early* também são possíveis de serem tratados com o algoritmo de posicionamento analítico detalhado. Para esse caso é necessário reduzir a influência das conexões do caminho com violação. Portanto, essa técnica deve ser adaptada para não concentrar as portas lógicas em torno daquelas dos caminhos críticos. Mas sim, afastá-las do caminho crítico.

No estágio de legalização há a necessidade de integrar um legalizador consciente de deslocamento máximo permitido para as portas lógicas. Desse modo, o legalizador buscaria a melhor posição dentro de uma região do circuito limitada pelo deslocamento máximo permitido, evitando, assim, que portas lógicas sem violações de deslocamento máximo, após a execução do algoritmo de posicionamento analítico detalhado, venham a violar essa restrição por causa do legalizador.

Um algoritmo que minimize as violações de densidade de ABU consciente da criticalidade de algumas portas lógicas pode ser extremamente útil para integrar junto do algoritmo de posicionamento analítico detalhado. Esse algoritmo moveria as portas lógicas que tem uma boa folga de *slack* para regiões na qual há disponibilidade de espaço. Isso facilitaria o posicionamento das portas lógicas dos caminhos críticos em regiões em que há redução na violação do tempo de propagação dos sinais de dados dessas portas lógicas críticas.

REFERÊNCIAS

- BHASKER, J.; CHADHA, R. **Static Timing Analysis for Nanometer Designs**: a practical approach. [S.l.]: Springer US, 2009.
- BRENNER, U. BonnPlace Legalization: minimizing movement by iterative augmentation. **Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on**, [S.l.], v.32, n.8, p.1215–1227, Aug 2013.
- CHAN, T. et al. mPL6: enhanced multilevel mixed-size placement with congestion control. In: NAM, G.-J.; CONG, J. (Ed.). **Modern Circuit Placement**: best practices and results. 1.ed. [S.l.]: Springer US, 2007. p.247–288. (Series on Integrated Circuits and Systems).
- CHEN, T.-C. et al. NTUplace3: an analytical placer for large-scale mixed-size designs. In: NAM, G.-J.; CONG, J. (Ed.). **Modern Circuit Placement**: best practices and results. 1.ed. [S.l.]: Springer US, 2007. p.289–309. (Series on Integrated Circuits and Systems).
- CHOW, W.-K. et al. Cell Density-driven Detailed Placement with Displacement Constraint. In: INTERNATIONAL SYMPOSIUM ON PHYSICAL DESIGN, 2014., 2014, New York, NY, USA. **Proceedings...** ACM, 2014. p.3–10. (ISPD '14).
- CHU, C.; WONG, Y.-C. FLUTE: fast lookup table based rectilinear steiner minimal tree algorithm for vlsi design. **Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on**, [S.l.], v.27, n.1, p.70–83, Jan 2008.
- CONG, J.; LUO, G. Thermal-Aware 3D Placement. In: XIE, Y.; CONG, J.; SAPATNEKAR, S. (Ed.). **Three Dimensional Integrated Circuit Design**: eda, design and microarchitectures. 1.ed. [S.l.]: Springer US, 2010. p.103–144. (Integrated Circuits and Systems).
- CONG, J.; NAM, G. **Modern Circuit Placement**: best practices and results. 1.ed. [S.l.]: Springer London, 2007. (Series on integrated circuits and systems).
- DIJKSTRA, E. W. A Note on Two Problems in Connexion with Graphs. **NUMERISCHE MATHEMATIK**, [S.l.], v.1, n.1, p.269–271, 1959.

FLACH, G. et al. Cell Placement on Graphics Processing Units. In: ANNUAL CONFERENCE ON INTEGRATED CIRCUITS AND SYSTEMS DESIGN, 20., 2007, New York, NY, USA. **Proceedings...** ACM, 2007. p.87–92. (SBCCI '07).

FLACH, G. et al. **UPlace**. 2008–2014.

HE, X. et al. Ripple: a robust and effective routability-driven placer. **Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on**, [S.l.], v.32, n.10, p.1546–1556, Oct 2013.

HENTSCHKE, R. F. **Algorithms for Wire Length Improvement of VLSI Circuits with Concern to Critical Paths**. 2007. 175p. Tese (Doutorado em Computação) — Instituto de Informática, Universidade Federal do Rio Grande do Sul, Porto Alegre, 2007.

HILL, D. **Method and system for high speed detailed placement of cells within an integrated circuit design**. 2002. n.US 6370673 B1.

HU, J. et al. An effective legalization approach based on multiple ordering. In: COMMUNICATIONS, CIRCUITS AND SYSTEMS (ICCCAS), 2013 INTERNATIONAL CONFERENCE ON, 2013. **Anais...** [S.l.: s.n.], 2013. v.2, p.514–518.

ISPD2012. **ISPD 2012 Constest**. Disponível em: http://www.ispd.cc/contests/12/ispd2012_contest.html Acessado em: 22 de setembro de 2014.

KAHNG, A. **VLSI Physical Design: from graph partitioning to timing closure**. 1.ed. [S.l.]: Springer London, 2011.

KAHNG, A. B. et al. Introduction. In: **VLSI Physical Design: from graph partitioning to timing closure**. 1.ed. [S.l.]: Springer Netherlands, 2011. p.1–30.

KAHNG, A. B. et al. Global and Detailed Placement. In: **VLSI Physical Design: from graph partitioning to timing closure**. 1.ed. [S.l.]: Springer Netherlands, 2011. p.93–128.

KIM, M.-C. et al. MAPLE: multilevel adaptive placement for mixed-size designs. In: ACM INTERNATIONAL SYMPOSIUM ON INTERNATIONAL SYMPOSIUM ON PHYSICAL DESIGN, 2012., 2012, New York, NY, USA. **Proceedings...** ACM, 2012. p.193–200. (ISPD '12).

KIM, M.-C.; HUJ, J.; VISWANATHAN, N. ICCAD-2014 CAD contest in incremental timing-driven placement and benchmark suite: special session paper: cad contest. In: COMPUTER-AIDED DESIGN (ICCAD), 2014 IEEE/ACM INTERNATIONAL CONFERENCE ON, 2014. **Anais...** [S.l.: s.n.], 2014. p.361–366.

- KIM, M.-C.; LEE, D.-J.; MARKOV, I. SimPL: an effective placement algorithm. In: COMPUTER-AIDED DESIGN (ICCAD), 2010 IEEE/ACM INTERNATIONAL CONFERENCE ON, 2010. **Anais...** [S.l.: s.n.], 2010. p.649–656.
- KIM, M.-C.; LEE, D.-J.; MARKOV, I. SimPL: an effective placement algorithm. **Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on**, [S.l.], v.31, n.1, p.50–60, Jan 2012.
- LIN, T.; CHU, C. POLAR 2.0: an effective routability-driven placer. In: THE 51ST ANNUAL DESIGN AUTOMATION CONFERENCE ON DESIGN AUTOMATION CONFERENCE, 2014, New York, NY, USA. **Proceedings...** ACM, 2014. p.123:1–123:6. (DAC '14).
- LIN, T. et al. POLAR: placement based on novel rough legalization and refinement. In: INTERNATIONAL CONFERENCE ON COMPUTER-AIDED DESIGN, 2013, Piscataway, NJ, USA. **Proceedings...** IEEE Press, 2013. p.357–362. (ICCAD '13).
- LU, J. et al. FFTPL: an analytic placement algorithm using fast fourier transform for density equalization. **ArXiv e-prints**, [S.l.], Dec. 2013.
- LU, J. et al. ePlace: electrostatics based placement using nesterov's method. In: THE 51ST ANNUAL DESIGN AUTOMATION CONFERENCE ON DESIGN AUTOMATION CONFERENCE, 2014, New York, NY, USA. **Proceedings...** ACM, 2014. p.121:1–121:6. (DAC '14).
- MARKOV, I.; HU, J.; KIM, M.-C. Progress and challenges in VLSI placement research. In: COMPUTER-AIDED DESIGN (ICCAD), 2012 IEEE/ACM INTERNATIONAL CONFERENCE ON, 2012. **Anais...** [S.l.: s.n.], 2012. p.275–282.
- PAN, M.; VISWANATHAN, N.; CHU, C. An Efficient and Effective Detailed Placement Algorithm. In: IEEE/ACM INTERNATIONAL CONFERENCE ON COMPUTER-AIDED DESIGN, 2005., 2005, Washington, DC, USA. **Proceedings...** IEEE Computer Society, 2005. p.48–55. (ICCAD '05).
- PINTO, F. d. A. **Posicionamento Visando Redução do Comprimento das Conexões**. 2011. 93p. Dissertação (Mestrado em Computação) — Instituto de Informática, Universidade Federal do Rio Grande do Sul, Porto Alegre, 2011.
- POPOVYCH, S. et al. Density-aware Detailed Placement with Instant Legalization. In: THE 51ST ANNUAL DESIGN AUTOMATION CONFERENCE ON DESIGN AUTOMATION CONFERENCE, 2014, New York, NY, USA. **Proceedings...** ACM, 2014. p.122:1–122:6. (DAC '14).

- QUINN JR., N. R. The Placement Problem As Viewed from the Physics of Classical Mechanics. In: DESIGN AUTOMATION CONFERENCE, 12., 1975, Piscataway, NJ, USA. **Proceedings...** IEEE Press, 1975. p.173–178. (DAC '75).
- ROY, J.; PAPA, D.; MARKOV, I. Capo: congestion-driven placement for standard-cell and rtl netlists with incremental capability. In: NAM, G.-J.; CONG, J. (Ed.). **Modern Circuit Placement: best practices and results**. 1.ed. [S.l.]: Springer US, 2007. p.97–133. (Series on Integrated Circuits and Systems).
- SAWICKI, S. **Particionamento de Células e PADs de I/O em Circuitos VLSI 3D**. 2009. 153p. Tese (Doutorado em Computação) — Instituto de Informática, Universidade Federal do Rio Grande do Sul, Porto Alegre, 2009.
- SHERWANI, N. **Algorithms for VLSI physical design automation**. 3rd.ed. [S.l.]: Kluwer Academic Publishers, 1999.
- SI2. **ISPD Constest**. Disponível em: http://www.si2.org/si2_home.php. Acessado em: 19 de setembro de 2014.
- SPINDLER, P.; JOHANNES, F. Kraftwerk: a fast and robust quadratic placer using an exact linear net model. In: NAM, G.-J.; CONG, J. (Ed.). **Modern Circuit Placement: best practices and results**. 1.ed. [S.l.]: Springer US, 2007. p.59–93. (Series on Integrated Circuits and Systems).
- SPINDLER, P.; SCHLICHTMANN, U.; JOHANNES, F. Kraftwerk2 - A Fast Force-Directed Quadratic Placement Approach Using an Accurate Net Model. **Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on**, [S.l.], v.27, n.8, p.1398–1411, Aug 2008.
- SPINDLER, P.; SCHLICHTMANN, U.; JOHANNES, F. M. Abacus: fast legalization of standard cell circuits with minimal movement. In: INTERNATIONAL SYMPOSIUM ON PHYSICAL DESIGN, 2008., 2008, New York, NY, USA. **Proceedings...** ACM, 2008. p.47–53. (ISPD '08).
- TAGHAVI, T. et al. Congestion Minimization in Modern Placement Circuits. In: NAM, G.-J.; CONG, J. (Ed.). **Modern Circuit Placement: best practices and results**. 1.ed. [S.l.]: Springer US, 2007. p.135–163. (Series on Integrated Circuits and Systems).
- VISWANATHAN, N.; CHU, C. C.-N. FastPlace: efficient analytical placement using cell shifting, iterative local refinement and a hybrid net model. **Proceedings of the 2004 International Symposium on Physical Design**, New York, NY, USA, p.26–33, 2004.

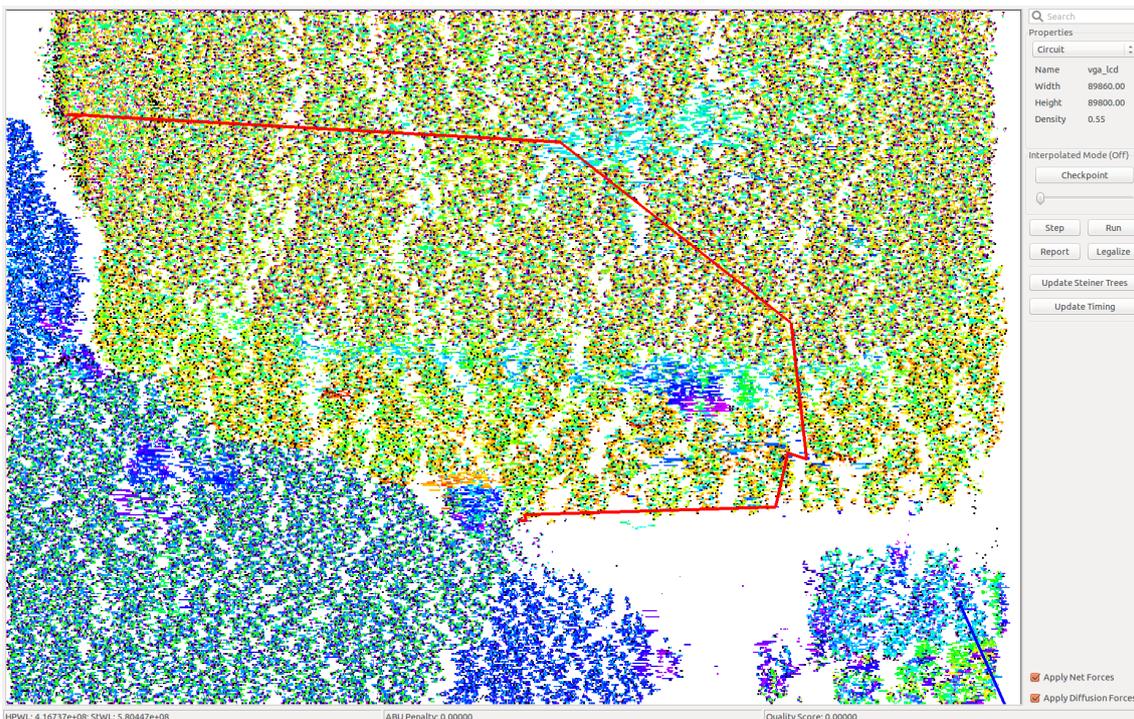
XU, Y.; ZHANG, Y.; CHU, C. FastRoute 4.0: global router with efficient via minimization. In: DESIGN AUTOMATION CONFERENCE, 2009. ASP-DAC 2009. ASIA AND SOUTH PACIFIC, 2009. **Anais...** [S.l.: s.n.], 2009. p.576–581.

APÊNDICE A LAYOUT DOS CIRCUITOS POSICIONADOS

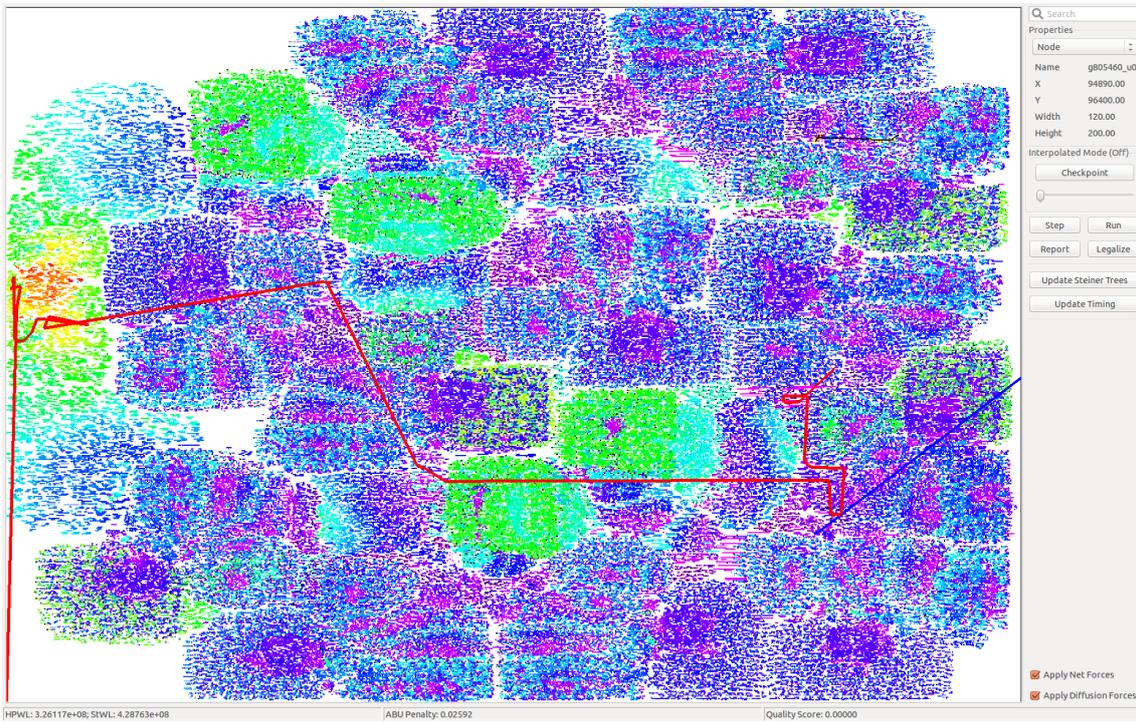
Neste apêndice são apresentados os *layouts* dos cinco circuitos utilizados durante o desenvolvimento dos algoritmos para otimização *skew* do relógio e para minimizar as violações no tempo de propagação dos sinais para o *late*. Esses circuitos são: *vga_lcd*, *b19*, *leon3mp*, *leon2* e *netcard*.

No *layout* dos circuitos abaixo, a linha em vermelho é o caminho crítico (WNS) para as violações de *late* e a linha em azul é o caminho crítico (WNS) para as violações de *early*. As portas lógicas estão coloridas com o critério de profundidade lógica e os registradores estão coloridos em preto. Em preto também estão coloridos os macro blocos no circuito *netcard*. Porém, estes tem as dimensões algumas ordens de grandeza maior do que a dimensão das portas lógicas.

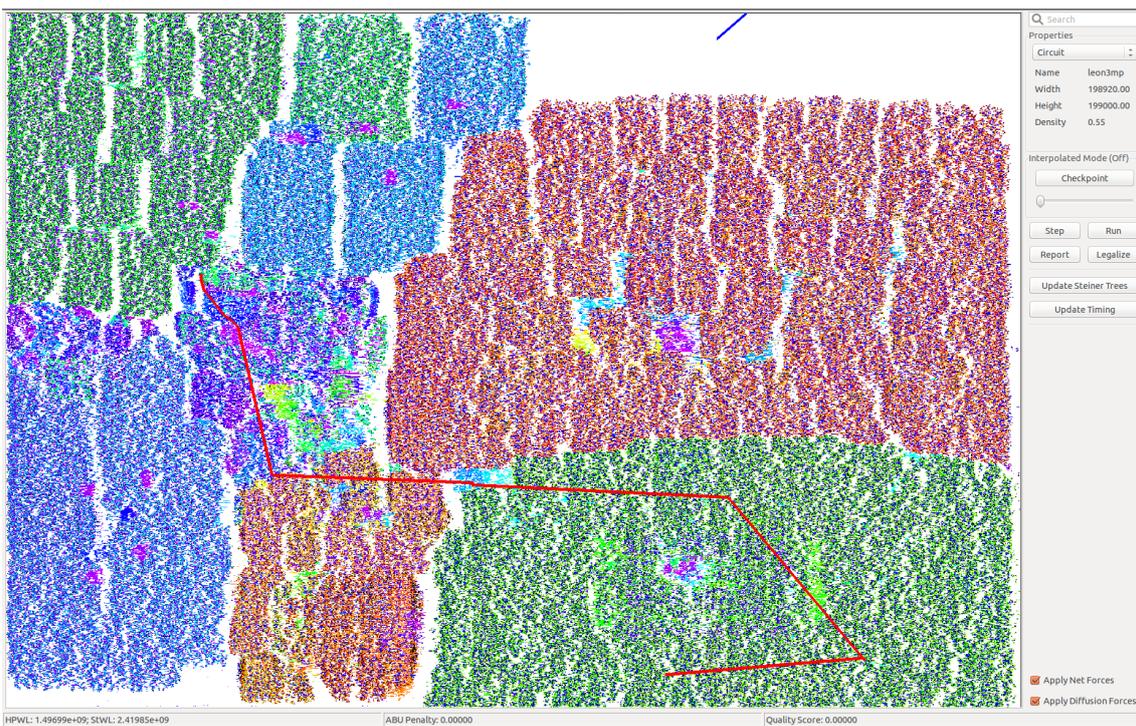
Figura A.1 - *Layout* do circuito *vga_lcd*



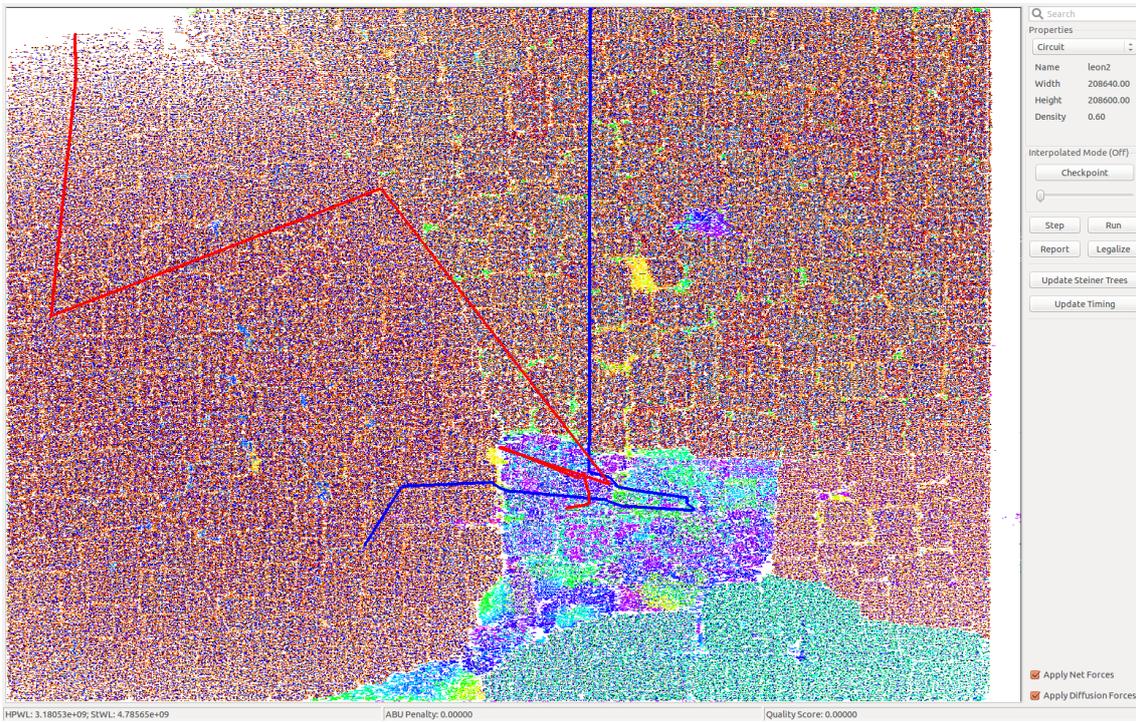
Fonte: figura elaborada pelo autor.

Figura A.2 - *Layout* do circuito b19

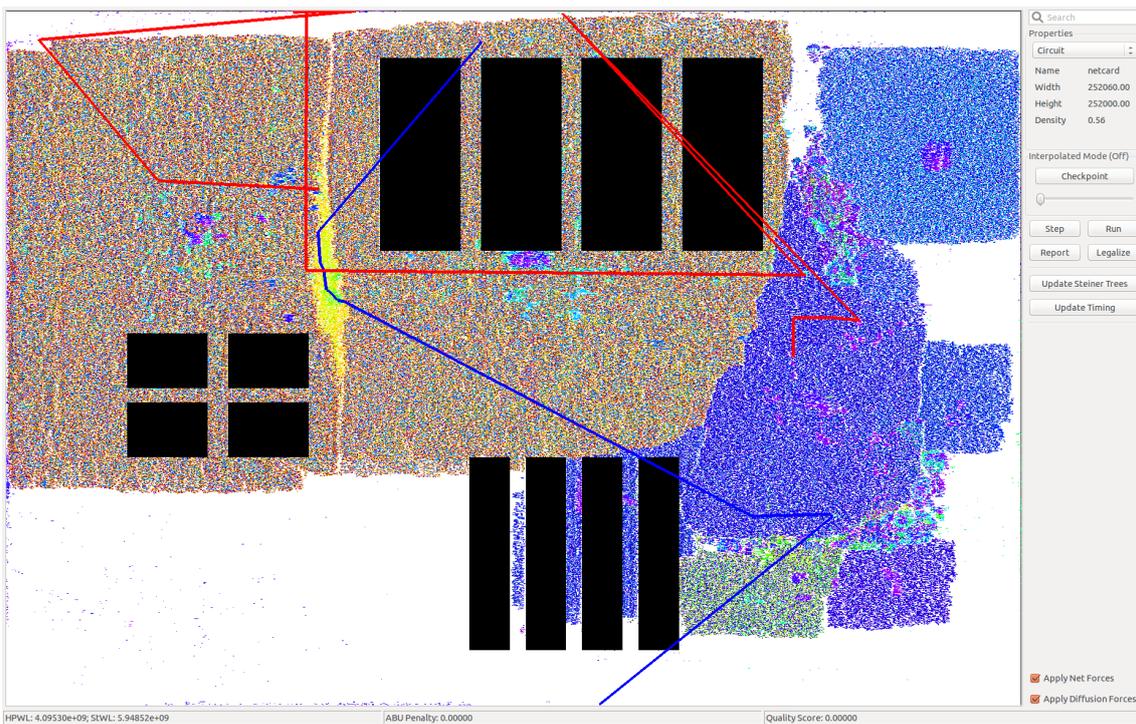
Fonte: figura elaborada pelo autor.

Figura A.3 - *Layout* do circuito leon3mp

Fonte: figura elaborada pelo autor.

Figura A.4 - *Layout* do circuito leon2

Fonte: figura elaborada pelo autor.

Figura A.5 - *Layout* do circuito netcard

Fonte: figura elaborada pelo autor.

APÊNDICE B *FRAMEWORK* UPLACE PARA POSICIONAMENTO DE CIRCUITOS INTEGRADOS

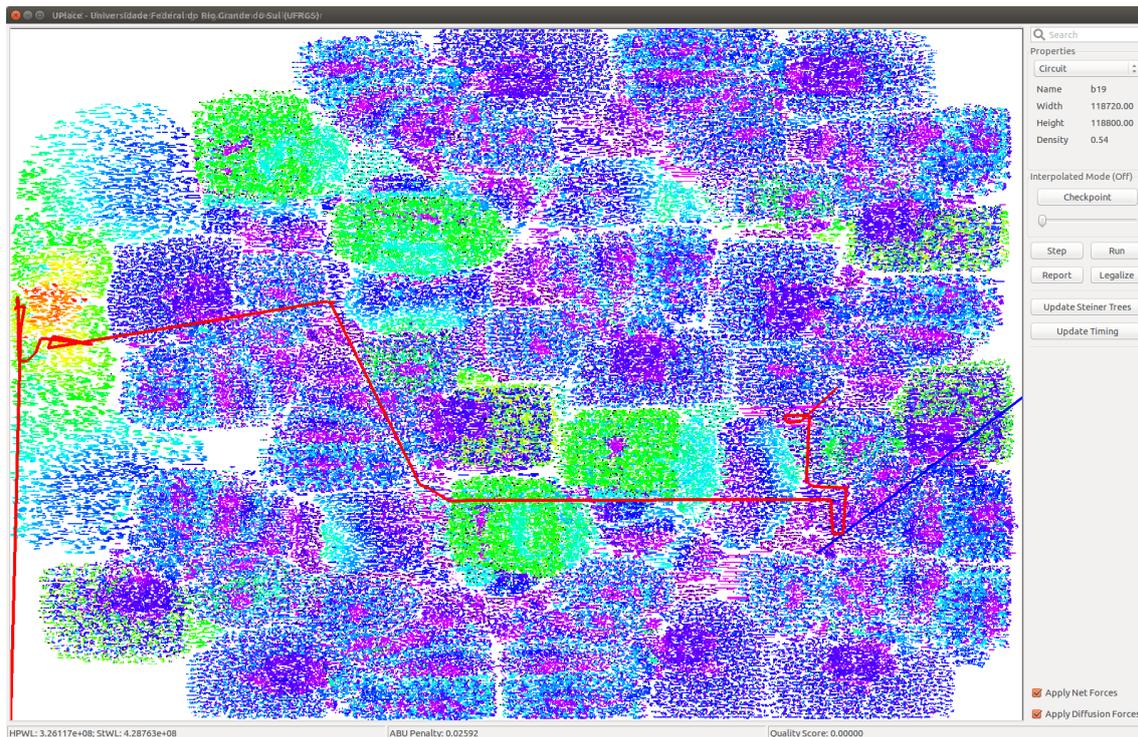
O UPlace é uma ambiente autocontido utilizado para o desenvolvimento de algoritmos de posicionamento global, legalização e posicionamento detalhado. Neste trabalho foi desenvolvido um algoritmo analítico para posicionamento detalhado das portas lógicas pertencentes aos caminhos críticos utilizando o *framework* UPlace. Nesse ambiente diversas funcionalidade já estão implementadas, o que facilita e acelera o desenvolvimento de algoritmos para as etapas de posicionamento. O ambiente UPlace é composto por uma interface gráfica, estruturas de dados para armazenar o circuito e um conjunto de algoritmos. A codificação dos algoritmos do UPlace foi realizada com a linguagem de programação C++. A interface gráfica foi desenvolvida utilizando a versão 2.8 da biblioteca wxWidgets. As estruturas de dados integradas no UPlace foram projetadas e implementadas com o auxílio da biblioteca STL.

No UPlace, os circuitos podem ser processados via linha de comando ou com auxílio da interface gráfica. O suporte a linha de comando permite que os circuitos sejam rapidamente processados com o auxílio de *scripts*. A interface gráfica, apresentada na Figura B.1, é extremamente útil enquanto são corrigidos os erros lógicos nos algoritmos e também para observar a movimentação das portas lógicas durante os passos do algoritmo em desenvolvimento.

No UPlace também estão integrados *parsers* para dois formatos de descrição dos circuitos, algoritmos de posicionamento global, de legalização e de posicionamento detalhado, um STA, o algoritmo FLUTE (CHU; WONG, 2008) para estimar o roteamento, resolvidor de sistema de equações lineares e algoritmos de suporte para o processamento do circuito. Esses algoritmos de suporte foram implementados para realizar rotinas auxiliares nos fluxos de execução, como garantir que todas as portas lógicas sejam posicionadas dentro dos limites do circuito, verificar sobreposição entre as portas lógicas, etc.

Os formatos suportados de arquivos para descrição de circuitos são DEF/LEF e *Bookshelf*. Também foram integrado os *parsers* para um subconjunto do Verilog sintetizado e do *Synopsys Design Constraints* (SDC). No UPlace foi integrada a versão 5.7 do *parser* para LEF/DEF disponibilizado pela (SI2, 2014). Com relação ao *parser* para o formato

Figura B.1 - Interface gráfica do UPlace



Fonte: FLACH et al. (2008–2014, p. S.I.).

Bookshelf, foi implementada uma versão na própria ferramenta. Os *parsers* para Verilog sintetizado e SDC foram integrados aqueles disponibilizados pela organização da competição do (ISPD2012, 2014).

No modo gráfico há funcionalidades que permitem observar as modificações no circuito originárias do algoritmo em implementação. As principais funcionalidades disponíveis nesse modo são:

- abrir os arquivos dos circuitos
- salvar em arquivo o resultado do posicionamento
- chamar os algoritmos de posicionamento ou a passos deles
- colorir as portas lógicas
- visualizar as propriedades do circuito e das portas lógicas
- interpolar as posições das portas lógicas:
 - entre etapas/passos de algum algoritmo
 - entre posicionamentos com algoritmos diferentes
- *zoom in e out*
- movimentar as portas lógicas na interface gráfica

Os principais critérios utilizados para colorir as portas lógicas são:

- registradores

- profundidade lógica
- portas lógicas no caminho crítico
- portas lógicas em diferentes partições

No modo gráfico também é possível visualizar as árvores de Steiner, a região ótima, as B2Bs, etc. referentes a uma determinada porta lógica.

B.1 Estruturas de Dados

Uma das metas ao projetar e implementar algoritmos para o UPlace é explorar ao máximo a localidade espacial das memória *cache*. As estruturas de dados utilizadas na ferramenta e a modelagem do circuito foram implementadas explorando ao máximo essa propriedade. Para tanto, é dada preferência em utilizar as estruturas de dados que sejam contínuas na memória. Essa prática limita o número de estruturas de dados candidatas e também limita o modo como o circuito pode ser modelado.

Vetores lineares (*arrays*) são as estruturas de dados mais utilizadas para armazenar os elementos do circuito no UPlace. Ao transferi-los para as estruturas de dados, é dada preferência que o tamanho de cada vetor seja conhecido no momento da inicialização. O acesso aos elementos é realizado com complexidade constante quando o índice é conhecido. Ao programar os algoritmos é dada preferência para passar o índice ou um vetor de índices dos elementos do circuito como parâmetro. Por outro lado, a referência da relação entre elementos do circuito (por exemplo, entre portas lógicas e redes) precisa ser armazenada em um terceiro vetor. Portanto, essa abordagem em armazenar o circuito contém vantagens e desvantagens que devem ser consideradas durante a fase de desenvolvimento dos algoritmos de posicionamento.

B.2 Algoritmos de Posicionamento Implementados

Os algoritmos implementados no ambiente Uplace foram aqueles para tratar o estágio de posicionamento global. Devido às restrições para o algoritmo implementado para tratar o problema de *Incremental Timing-Driven Placement*, no UPlace foi implementado um algoritmo de legalização e um algoritmo de posicionamento detalhado para tratar os caminhos com violações no tempo de propagação dos sinais de dados.

No estágio de posicionamento global, os algoritmos foram implementados com a técnica analítica quadrática. Contudo, os algoritmos de posicionamento global ainda necessitam extensão para tratar adequadamente circuitos com macro blocos.

Os algoritmos Jezz e Abacus (SPINDLER; SCHLICHTMANN; JOHANNES, 2008b) foram implementados para o estágio de legalização no UPlace. A versão do Abacus não tem suporte para tratar bloqueios nas bandas causados por macro blocos. O Jezz é baseado no Abacus, porém ele considera os bloqueios nas bandas durante a legalização.

Os macro blocos são os principais causadores de bloqueios nas bandas. O Jezz também suporta o bloqueio do movimento de portas lógicas durante a legalização. Essa funcionalidade pode ser extremamente útil para posicionadores detalhados guiados a atraso crítico. Quando uma determinada porta lógica é movimentada para liberar espaço para legalizar uma terceira, este movimento pode causar mudanças drásticas indesejadas nas métricas de temporização.

Dois algoritmos para o estágio de posicionamento detalhado foram implementados no UPlace. O primeiro deles é o BraveDP (POPOVYCH et al., 2014). O principal foco desse algoritmo é reduzir a densidade dos *bins* com violação. As portas lógicas são movidas preferencialmente para as regiões ótimas. Esse objetivo tem como meta reduzir o comprimento de fio. Contudo, a otimização de posicionamento detalhado do BraveDP não tem como objetivo reduzir ou resolver as violações no tempo de propagação dos sinais. O BraveDP implementado ainda não está totalmente em conformidade com o proposto por POPOVYCH et al..

O foco do segundo algoritmo de posicionamento detalhado é resolver as violações de propagação de sinal. Esse algoritmo foi desenvolvido para integrar o fluxo que trata os caminhos com violações de propagação de sinal. O algoritmo tem dois passos distintos, um para otimizar o *clock skew* e outro para tratar as violações de tempo de propagação dos sinais de dados. No primeiro passo tenta-se reduzir as violações de propagação dos sinais de dados tentando modificar a posição dos registradores. Esse passo afeta principalmente a árvore de relógio. No segundo passo busca-se aproximar as portas lógicas com violação no tempo de propagação *late* dos sinais de dados. Para reduzir a violação nesse tipo de caminho, somente as portas lógicas combinacionais relacionadas aos caminhos críticos e as vizinhas diretas delas são movidas. Elas são movimentadas com um algoritmo analítico quadrático. A posição das demais portas lógicas não é alterada, e elas são elementos fixos ao montar o sistema de equações lineares.