Universidade Federal de Ouro Preto - UFOP
Instituto de Ciências Exatas e Biológicas - ICEB
Departamento de Computação - DECOM

Pesquisa Operacional Aplicada a Produção de Semicondutores em Minas Gerais

Bolsista: Júnior Rhis Lima
Orientador: Marco Antonio Moreira de Carvalho – DECOM/UFOP

Nota: Relatório Parcial referente ao período de 01/03/2016 a 01/07/2016, apresentado à Universidade Federal de Ouro Preto, como parte das exigências do programa de iniciação científica / Fundação de Amparo à Pesquisa do Estado de Minas Gerais (FAPEMIG).

Local: Ouro Preto - Minas Gerais - Brasil
Data: 31 de agosto de 2016

Pesquisa Operacional Aplicada a Produção de Semicondutores em Minas Gerais

O presente projeto de pesquisa propõe o desenvolvimento de um método heurístico para a otimização de leiautes de matrizes de portas, um problema combinatório relacionado à dobradura de matrizes lógicas programáveis no projeto de circuitos eletrônicos integrados em larga escala, como por exemplo, processadores de computadores e celulares. Nestes circuitos eletrônicos, portas são conectadas entre si por trilhas, e no intuito de minimizar os custos destas conexões e a área destes circuitos, busca-se um leiaute otimizado das portas e das respectivas trilhas. Desta forma, é possível produzir componentes mais baratos, mais rápidos e mais compactos. O problema objeto de estudos do presente projeto possui aplicação prática direta na indústria microeletrônica, um ramo da eletrônica voltado à integração de circuitos eletrônicos, estando presente na informática, nas telecomunicações, nos controles de processos industriais, na automação dos serviços industriais e comerciais e nos bens de consumo. Atualmente, a primeira fábrica de semicondutores (elemento primordial na indústria eletrônica e confecção de seus componentes) do hemisfério sul do globo está sediada no estado de Minas Gerais e entrou em operação no segundo semestre do ano de 2015. Este projeto de pesquisa possui o potencial para gerar a inovação aplicável a este novo nicho industrial, em consonância com as políticas governamentais de investimento em pesquisa e desenvolvimento, e também reforçar os elos de cooperação da Universidade Federal de Ouro Preto com a indústria. Este projeto de iniciação científica é parte integrante de um projeto aprovado no edital universal do Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), aprovado pelo proponente a área Engenharias III.

Palavras-chave: Matriz de Portas, Engenharia de Produção, Pesquisa Operacional

Assinatura do orientador(a): _____

Nome Completo do orientador(a)

Assinatura do co-orientador(a): _____

Nome Completo do co-orientador(a)

Assinatura do bolsista: _____

Nome Completo do bolsista

Sumário

Lista de Acrônimos

GMLP *Gate Matrix Layout Problem*

IGAP *Interval Graph Augmentation Problem*

SCOOP *Sheet Cutting and Process Optimization for Enterprises*

OPTSICOM *Optimization of Complex Systems*

CMOS *Complementary metal-oxide-semiconductor*

BRKGA *Biased Random-Key Genetic Algorithm*

MOSP *Minimization of Open Stacks Problem*

Capítulo 1

Introdução

A microeletrônica é um ramo da eletrônica voltado à integração de circuitos eletrônicos, promovendo uma miniaturização dos componentes em escala microscópica. A área engloba tanto os processos físico-químicos de fabricação dos circuitos integrados como o projeto do circuito em si. São considerados ramos desta área, igualmente, o desenvolvimento de métodos de apoio ao projeto de circuitos, modelagem de componentes e técnicas de teste, entre outras.

Os componentes utilizados na microeletrônica são construídos na escala de mícrons ou mesmo nanômetros, tornando-se parte do ramo de nanotecnologia. O conjunto de componentes usados para um mesmo projeto é tipicamente chamado de circuito integrado, ou ainda, chip. Alguns exemplos de circuitos integrados são memórias de computadores, processadores, modems e conversores analógicos digitais.

Os circuitos integrados são produzidos em *wafers*, discos de silício, normalmente de 300mm de diâmetro, sobre os quais são fabricadas estruturas laminares, em um processo denominado fotolitografia. Tipicamente, diversos circuitos eletrônicos (idênticos ou não) são fabricados em um mesmo *wafer* por vez, de modo a ocupar toda a área disponível. O custo de produção do circuito será o custo de fabricação do *wafer*, dividido pelo número de circuitos nele presentes (excluindo-se os defeituosos). Assim, um circuito é mais caro pelo fato de usar uma maior área do *wafer* – e, consequentemente, tendo um menor número de circuitos por *wafer* – e não necessariamente devido ao número de componentes presentes. Circuitos de leiaute otimizado são menores, e por consequência, mais baratos, uma vez que ocupam uma menor área do *wafer*.

Uma Matriz Lógica Programável (MLP) ou Matriz de Portas é um tipo de dispositivo lógico programável usado para implementar circuitos lógicos combinatoriais. Uma MLP possui uma dimensão com portas AND programáveis que podem ser conectadas a uma dimensão de portas OR igualmente programáveis. Desta forma, conexões entre portas podem ser estabelecidas para produzir um resultado específico. A Figura 1 apresenta a disposição das dimensões (ou planos) de uma MLP, o que permite que um grande número de funções lógicas sejam sintetizadas.

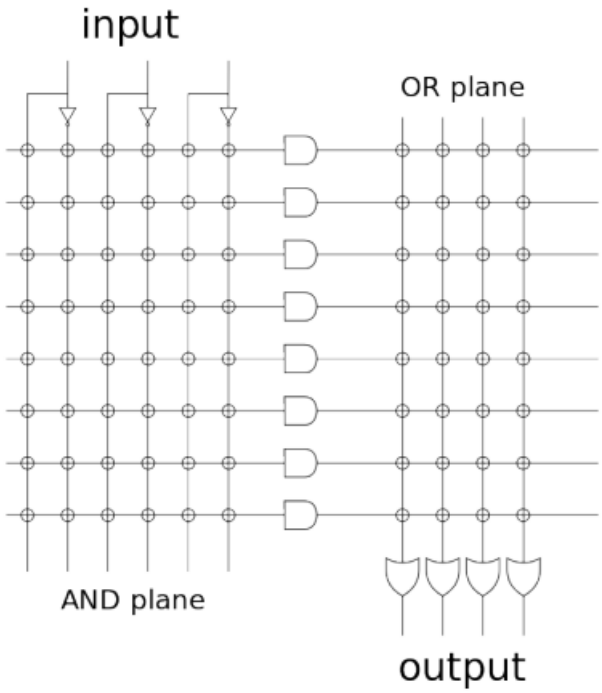

Figura 1.1: Exemplo de Matriz Lógica Programável, ou Matriz de Portas

1.1 Definição do Problema

O Problema de Leiaute de Matrizes de Portas (PLMP), conforme descrito originalmente em [Wing et al., 1985] é definido como o problema de determinar uma permutação de portas de maneira tal que o número necessário de trilhas é minimizado e consequentemente, a área e o custo de produção do circuito correspondente sejam minimizados.

Em um circuito eletrônico impresso implementado por uma matriz lógica programável, as portas correspondem a nós, e diferentes conexões são exigidas entre eles. Cada conexão é realizada por um fio, envolve um subconjunto de nós e é chamada de rede. A Figura 2(a) apresenta um exemplo de matriz de portas em que sete portas (linhas verticais) precisam ser conectadas de acordo com cinco redes diferentes, representadas por linhas em um mesmo nível: a rede A conecta as portas 1, 3 e 5; a rede B conecta as redes 1, 4, 5 e 6, e assim por diante. Fios condutores são utilizados para criar as conexões, um para cada rede, conforme mostrado nas linhas horizontais na Figura 2(b).

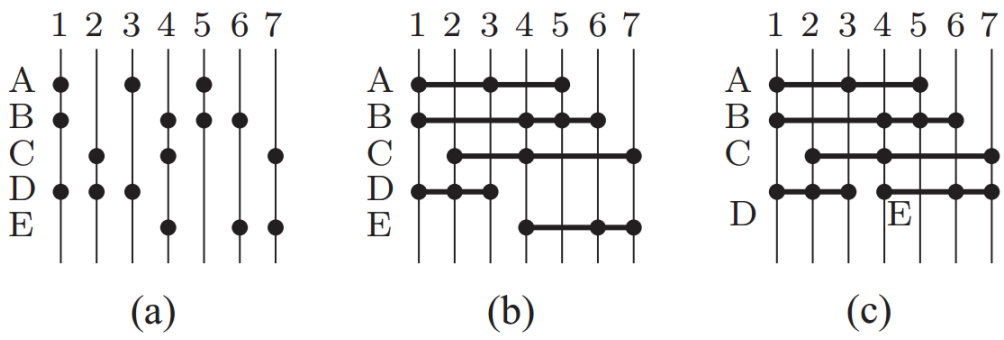

Figura 1.2: Exemplo de matriz de portas: (a) conexões exigidas (b) redes já com fios e (c) trilhas de conexão

É importante notar que para conectar as portas de uma rede, pode ser necessário cruzar outras portas não incluídas na rede, dependendo da disposição das portas. Também, uma única trilha de conexão pode ser utilizada para posicionar fios sem sobreposição, conforme exibido na Figura 2(c) para as redes D e E. O comprimento total dos fios utilizados determina o custo das conexões, ao passo que o número de trilhas determina a área total do circuito eletrônico, o que pode ser limitado de acordo com restrições de projeto. Ambos indicadores fornecem uma estimativa da eficiência do leiaute do circuito e dependem de como as portas são sequenciadas fisicamente.

O leiaute exibido na Figura 2 exige 19 unidades de fios e 4 trilhas, correspondente ao número máximo de trilhas utilizadas. Um leiaute otimizado é apresentado na Figura 3, usando 15 unidades de fios e 3 trilhas. Note que a diferença entre o leiaute da mesma matriz de portas original é a disposição das portas. Na Figura 2 temos a sequência 1, 2, 3, 4, 5, 6 e 7, ao passo que na Figura 3 temos a sequência 1, 3, 5, 2, 4, 6 e 7.

Este é um problema NP-Difícil [Linhares and Yanasse, 2002] e possui aplicações em diferentes áreas, bem como diferentes problemas de formulação equivalente [Möhring, 1990]: Problema de Corte Modificado (Modified Cutwidth), Problema de Minimização de Pilhas Abertas (Minimization of Open Stacks Problem), Dobradura de Arranjos Lógicos Programáveis (Programmable Logic Array Folding, ou PLA Folding), Interval Thickness, Node Search Game, Edge Search Game, Narrowness, Split Bandwidth, Graph Pathwidth, Edge Separation e Vertex Separation.

Atualmente, a primeira fábrica de semicondutores (elemento primordial na indústria eletrônica e confecção de seus componentes) do hemisfério sul do globo está sediada no estado de Minas Gerais e entrou em operação no segundo semestre do ano de 2015. Este projeto de pesquisa possui o potencial para gerar a inovação aplicável a este novo nicho industrial, em consonância com as políticas governamentais de investimento em pesquisa e desenvolvimento, e também reforçar os elos de cooperação da Universidade Federal de Ouro Preto com a indústria.

Capítulo 2

Objetivos

O trabalho de pesquisa abordado neste projeto se propõe a estudar do Problema de Leiaute de Matrizes de Portas e a desenvolver abordagens heurísticas consistentes, robustas e que constituam uma contribuição para o estado da arte relacionado ao problema tratado e correlatos. Será buscada ainda a aplicação prática dos métodos desenvolvidos em contextos reais, a fim de que também seja constituído um avanço para a indústria nacional recentemente instalada no estado de Minas Gerais.

2.1 Objetivos Específicos

1. Realizar pesquisa para geração de embasamento teórico e revisão bibliográfica sobre o tema tratado;
2. Propor um método heurístico que contemple apropriadamente as especificidades do problema tratado;
3. Avaliar o método implementado considerando dados reais e também com problemas teste publicamente disponíveis, realizando uma análise crítica considerando outros métodos da literatura;
4. Buscar a aplicação prática dos métodos desenvolvidos em contextos reais, a fim de que também seja constituído um avanço para as indústrias nacionais.

Capítulo 3

Revisão da Literatura

Wing et al. [1985] foi o primeiro trabalho a utilizar métodos heurísticos para resolver o *Gate Matrix Layout Problem* (GMLP). O GMLP foi modelado usando grafos de intervalos, onde as portas em comum de diferentes trilhas são consideradas interseções. Essas trilhas com portas em comum, para diminuir a área do circuito, podem ser implementadas como uma única trilha. Como testes, foi possível determinar leiautes para circuitos *Complementary metal-oxide-semiconductor* (CMOS) de 180 e 306 transistores em 9 e 28 segundos

Kashiwabara and Fujisawa [1979] (*apud* Linhares e Yanasse, 2002) provaram que o GMLP é NP-difícil reduzindo-o ao problema de aumento de grafos de intervalo (*Interval Graph Augmentation Problem* (IGAP)). Linhares and Yanasse [2002] mostraram diferentes áreas de aplicação do problema: Corte Modificado (*Modified Cutwidth*), Problema de Minimização de Pilhas Abertas, *Minimization of Open Stacks Problem* (MOSP), Dobradura de Arranjos Lógicos Programáveis (*Programmable Logic Array Folding, ou PLA Folding*), *Interval Thickness, Node Search Game, Edge Search Game, Narrowness, Split Bandwidth, Graph Pathwidth, Edge Separation e Vertex Separation.*

Chen and Hu [1988] utilizaram o algoritmo *min-net-cut* com uma representação dinâmica das conexões do circuito e criaram um algoritmo para o GMLP que analisa a dualidade dos CMOS. O algoritmo tem complexidade $O(n \log n)$ onde n é o número de portas. Nos experimentos realizados, o *min-net-cut* obteve os melhores resultados nos 5 circuitos testados retirados da literatura [Hwang et al., 1987]. Chen and Hu [1990a] apresentaram dois algoritmos, *GM-PLAN* e *GM-LEARN*, ambos baseados em paradigmas de inteligência artificial. *GM-LEARN* usa resultados gerados anteriormente obtidos a partir do *GM-PLAN* para melhorar a busca e utiliza uma função "aprender" para modificar as heurísticas utilizadas no *GM-PLAN* com base nos resultados anteriores. O *GM-PLAN* utiliza duas estratégias de inteligência artificial: planejamento hierárquico e políticas de meta-planejamento. Os resultados do *GM-LEARN* foram comparados com 16 circuitos retirados da literatura e, em 8 deles, o melhor resultado foi superado. Em outros 6 circuitos, os melhores resultados foram igualados e em apenas 2 circuitos, piores resultados foram obtidos. Já o *GM-PLAN* foi testado com 12 circuitos retirados da literatura e foram obtidos resultados para apenas 7 deles. O algoritmo conseguiu melhor resultado para um circuito, e, nos outros 6 circuitos foram igualados os melhores resultados já conhecidos.

Shahookar et al. [1994] descrevem uma nova implementação de algoritmo genético para problemas de permutação utilizando o *beam search* para reduzir o espaço de busca. O algoritmo foi testado em 3 conjuntos de instâncias *benchmark* e o melhor resultado obtido foi em relação ao comprimento das redes.

Linhares et al. [1999] utilizaram uma heurística derivada da estatística mecânica: otimização microcanônica (*microcanonical optimization algorithm*). Os autores reportam que o algoritmo

de otimização microcanônica foi capaz de atingir os resultados relatados por outras abordagens e superou, em 7 trilhas, as melhores soluções encontradas até então. Os resultados foram comparados com 5 heurísticas (recozimento simulado de Hong et al. [1989], métodos de construção unidirecional e bidirecional de Hong et al. [1989] e as heurísticas de inteligência artificial *GM-PLAN* e *GM-LEARN* de Chen and Hu [1990a]). Para a solução do problema de atribuição de redes para trilhas foi utilizado um algoritmo de tempo polinomial conhecido como *left-edge*.

Linhares [1999] utilizou uma estratégia de busca predatória, onde a área de busca é restringida e a cada iteração é determinada uma solução. Esse algoritmo é baseado no comportamento dos predadores na natureza, onde, ao encontrar uma presa, o predador procura em volta novas presas. Das 25 instâncias testadas, o método conseguiu igualar os resultados em 12 delas e em 4 dessas instâncias conseguiu melhorar os resultados da literatura, reportados em Heinbuch [1988]. No restante das instâncias, os resultados obtidos por *GM-PLAN* e *GM-LEARN* (Chen and Hu [1990b]), foram atingidos.

Mendes and Linhares [2004] utilizaram algoritmos genético e memético e uma população hierarquicamente estruturada com base em uma árvore ternária completa. Nesta árvore foram colocados grupos de subpopulações sobrepostas, constituídas por um líder e três suportes. Os lideres contêm a melhor solução dentre todos os elementos do grupo, considerando o número de trilhas. Além disso, o algoritmo proposto apresenta uma busca local especialmente adaptada, que depende de uma vizinhança definida. Este trabalho apresentou resultados iguais aos da otimização microcanônica e superou todas as abordagens metaheurísticas e heurísticas anteriores da literatura.

Oliveira and Lorena [2002] utilizaram um algoritmo genético construtivo. Este algoritmo tem características novas em comparação com um algoritmo genético tradicional. Este inclui uma população de tamanho dinâmico composto por esquemas e estruturas, nas quais há a possibilidade de usar heurísticas para representá-las e definir funções de aptidão. Os autores modelaram o GMLP como o problema de otimização bi-objetivo(reduzir o numero de trilhas e diminuir o tamanhos dos fios). Os problemas utilizados foram os maiores encontrados na literatura até então. O algoritmo alcançou os melhores resultados da literatura, obtidos pela otimização microcanônica e busca predatória, sendo considerado pelos autores mais robusto que esta.

Giovanni et al. [2013] considerou não só o GMLP, mas também o problema de minimização de custo de conexões do GMLP, levando em consideração o comprimento dos fios. Foram utilizados dois algoritmos, o primeiro uma heurística baseada em uma abordagem genética em conjunto com uma definição composta e dinâmica da função de *fitness*. O segundo, um algoritmo *branch-and-cut* baseado em uma nova formulação de programação linear inteira. Em casos pequenos, o algoritmo *branch-and-cut* comprova os melhores resultados gerados pela abordagem genética. Os autores utilizaram 330 instâncias de diferentes conjuntos de dados. Em 11 instâncias, os métodos propostos foram melhores que os resultados já conhecidos.

Gonçalves et al. [2014] apresentaram um algoritmo genético de chaves aleatórias viciadas (*Biased Random-Key Genetic Algorithm* (BRKGA)) para resolver o MOSP. Esta abordagem é divida em quatro fases: Representação e Decodificação do Cromossomo; Construção da Solução; Ajuste do Cromossomo e Avaliação. Na primeira fase, a solução do problema possui duas representações, direta e indireta: a representação direta é a padrão do problema, enquanto a indireta é um vetor de chaves randômicas que na fase de decodificação é ordenado de maneira crescente. A fase de construção da solução é composta por duas etapas que são repetidas ate que todos os padrões sejam sequenciados: A primeira etapa seleciona o padrão a ser inserido e a segunda seleciona a posição na solução parcial para inserir o padrão. A busca local gera soluções que não obedecem às regras impostas pelo cromossomo, e para adequar a solução é necessário utilizar uma fase de ajuste. A heurística ajusta a ordem dos genes de acordo com a posição de

cada padrão na solução final, de modo a reduzir o número de gerações necessárias para obtenção de melhores valores. A fase final utiliza uma função *fitness* específica do MOSP. Os autores utilizaram 6141 instâncias disponíveis na literatura, divididas nas classes: *Harvey*; *Simonis*; *Shaw*; *Miller e Wilson* [Smith and Gent, 2005]; *Faggioli e Bentivoglio* [Faggioli and Bentivoglio, 1998]; *VLSI* [Chen and Hu, 1990a]; *SCOOP* (*Disponível em http://www.scoopproject.net*); *Harder150* [De Giovanni et al., 2013] e *Becceneri* [Becceneri, 1999a]. Os experimentos realizados mostraram que esta nova proposta igualou ou superou todas as outras abordagens da literatura, obtendo os melhores resultados conhecidos.

Capítulo 4

Fundamentação Teórica

Este capítulo apresenta a fundamentação teórica do GMLP. Nas próximas seções são apresentadas uma definição formal do problema GMLP e os métodos disponíveis na literatura que serviram como base para os métodos propostos. A fundamentação apresentada servirá de apoio para compreensão dos métodos desenvolvidos no decorrer deste trabalho.

4.1 O Problema de Determinação de Leiaute de Matrizes de Portas

Formalmente, uma instância do GMLP contendo m redes e n portas pode ser descrita por uma matriz P binária $m \times n$, em que p_{ij}=1 se a rede i incluir a porta j e p_{ij}=0, caso contrário. A Figura 4.1 ilustra este formato, em que 5 redes (nomeadas de A a E) e 6 portas (enumeradas de 1 a 6) são representadas.

	1	2	3	4	5	6
A	1	0	0	0	0	1
B	0	1	0	1	0	0
C	0	0	1	0	1	1
D	1	0	0	1	0	0
E	0	0	1	0	1	0

Figura 4.1: : Exemplo de matriz de binária original

Uma solução para o GMLP consiste em uma permutação π das n colunas da matriz P. Ao alterarmos a posição relativa das portas, é necessário reservar espaço para os fios, porque os mesmos não podem se cruzar em uma mesma trilha (vide Figura 4.2).

Desta forma, uma solução π para este problema induz uma matriz permutação Q^{π} que contém as colunas de P na ordem determinada por π. A matriz Q possui a propriedade dos 1s consecutivos, de maneira que q^{π}_{ij}=1 se e somente se o fio da rede i incluir ou cruzar a porta j. Desta forma, todos os elementos da matriz Q que estão entre entradas de valor 1 originalmente também são considerados como tendo o valor 1. A Equação 4.1 descreve formalmente preenchimento da matriz Q.

$$q^{\pi}_{ij} = \begin{cases} 1 & \text{se } \exists x, \exists y \mid \pi(x) \leqslant j \leqslant \pi(y) \text{ e } p_{ix} = p_{iy} = 1, \\ 0 & \text{caso contrário} \end{cases} \tag{4.1}$$

	1	2	3	4	5	6
A	1	**1**	**1**	**1**	**1**	1
B	0	1	**1**	1	0	0
C	0	0	1	**1**	1	1
D	1	**1**	**1**	1	0	0
E	0	0	1	**1**	1	0

Figura 4.2: Exemplo de matriz de binária com 1s consecutivos

Uma solução para o GMLP é avaliada maximizando a soma dos 1s das colunas da matriz Q de acordo com a Equação 4.2. Neste problema as colunas de maior soma são consideradas o gargalo do problema, tendo em vista que o número da soma define o numero de trilhas necessárias.

$$Z^{\pi}_{GMLP}(Q^{\pi}) = \max_{j\in\{1,\ldots,n\}} \sum_{i=1}^{m} q^{\pi}_{ij} \tag{4.2}$$

A Figura 4.3 mostra uma segunda permutação determinada como solução. O valor da soma das colunas mostra o número mínimo de trilhas que podemos usar, já que não podemos ter duas redes usando a mesma porta na mesma trilha, pois isso mudaria a lógica do circuito. A soma das colunas da Figura 4.3 é igual a 2, mostrando que o circuito pode ser implementado usando apenas 2 trilhas. Uma outra permutação possível para as portas do exemplo seria π =[3, 5, 1, 6, 4, 2]. Esta permutação, gera uma matriz compactada com três trilhas. Daí surge a necessidade de utilizar métodos computacionais para determinar uma permutação que leve a um leiaute otimizado, tendo em vista que o número de possíveis permutações é $n!$.

	3	5	6	1	4	2
A	0	0	1	1	0	0
B	0	0	0	0	1	1
C	1	1	1	0	0	0
D	0	0	0	1	1	0
E	1	1	0	0	0	0

Figura 4.3: Exemplo de matriz de binária permutada.

Por fim, a função objetivo do GMLP é determinar a permutação π das colunas de P tal que o número máximo de 1s consecutivos em uma coluna seja minimizado, vide Equação 4.3.

$$\min_{\pi\in\Pi} Z^{\pi}_{GMLP}(P) \tag{4.3}$$

4.2 Pré-processamento dos Dados de Entrada

Métodos de pré-processamento dos dados de entrada de problemas combinatórios são utilizados comumente na tentativa de reduzir as dimensões dos problemas pela identificação de informações redundantes ou identificação de estruturas especiais.

Adaptando o conceito introduzido por Yanasse and Senne [2010], uma porta dominada é aquela que compartilha todas as suas redes com outra(s) porta(s) do problema. Em outras palavras, seja $R(i)$ uma função que retorna o conjunto de redes que utilizam a porta i, temos que

uma porta i é dominada por uma porta j se $R(i) \subseteq R(j)$. Portas dominadas podem ser retiradas do problema, uma vez que posteriormente podem ser sequenciadas imediatamente após a sua porta dominante, sem o aumento do número de trilhas. Para identificarmos a dominância entre portas, é necessário comparar as redes de todos os pares de portas, exigindo complexidade de $O(n^2m)$, em que n denota o número de portas e m o número de redes.

A Figura 4.4 apresenta a matriz original utilizada como exemplo, onde $R(1) = \{A,B,D\}$, $R(2) = \{B,D\}$ e $R(3) = \{D\}$. Desta forma, a porta 1 domina as portas 2 e 3, que podem então ser removidas do processo de solução do problema e adicionadas posteriormente na solução após a porta 1.

	1	2	3	4	5
A	1	0	0	1	0
B	1	1	0	1	0
C	0	0	0	1	1
D	1	1	1	0	1
E	0	0	0	0	1

Figura 4.4: Matriz Original.

A Figura 4.5 apresenta a matriz resultante após o pré-processamento, sem as colunas que representam as portas dominadas.

	1	4	5
A	1	1	0
B	1	1	0
C	0	1	1
D	1	0	1
E	0	0	1

Figura 4.5: Matriz resultante do pré-processamento.

4.3 Representação por Grafos

Yanasse [1997] mostrou que qualquer instância MOSP (um problema de formulação equivalente ao GMLP) pode ser modelada como um problema em grafos. Nesta modelagem, cada vértice corresponde a uma rede, havendo uma aresta entre dois vértices se e somente se existir pelo menos uma porta compartilhada entre as duas redes correspondentes.

Esta representação permite que durante a solução do problema seja levada em consideração a relação entre portas e redes e também das redes entre si.

A Figura 4.6 apresenta o grafo associado ao exemplo anterior.

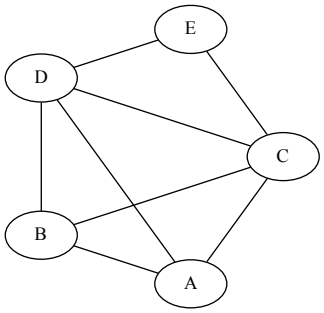

Figura 4.6: Exemplo de grafo gerado a partir do exemplo anterior.

No exemplo anterior temos que as redes A, B e D compartilham a porta 1, portanto, há uma aresta entre cada par de vértices associados a estas portas induzindo um clique de tamanho 3 no grafo ({A, B, C}). Analogamente, dois outros cliques de tamanho 3 são induzidos pelo compartilhamento da porta 4 pelas redes A, B e C e pelo compartilhamento da porta 5 entre as redes C, D e E. Outros cliques podem ser induzidos pela união dos cliques anteriores, como é o caso do clique {A, B, C, D}. Os grafos associados a instâncias do GMLP são, portanto, formados pela união dos cliques induzidos pelas portas. Arestas paralelas e laços não são considerados nesta representação.

É interessante notar que diferentes instâncias GMLP podem possuir grafos associados idênticos. A remoção de colunas no pré-processamento apresentado não implica na alteração do grafo resultante.

4.4 Busca em Largura

Cormen et al. [2001] define a *Busca em Largura* (ou BFS, do inglês *Breadth-First Search*) do seguinte modo: dado um grafo e um vértice de origem, a busca em largura explora sistematicamente as arestas do grafo para "descobrir" cada vértice que pode ser alcançado a partir da origem. A busca explora toda a vizinhança de um vértice por vez e utiliza a estrutura de fila como guia: cada novo vértice explorado é adicionado ao final da fila. Ao final, a BFS retorna uma lista que contém a sequência dos vértices na ordem em que foram explorados. A complexidade computacional da BFS é $O(V+E)$ onde V é o número de vértices e E o número de arestas.

4.5 Descida em Vizinhança Variável

Métodos projetados para aprimoramento de soluções de problemas combinatórios frequentemente empregam a utilização de algoritmos de *busca local*: dada uma solução inicial gerada por outro método, um *movimento* consiste em um tipo de operação que altera pontualmente a solução original, transformando-a em uma solução diferente. A nova solução obtida é dita *vizinha* da solução original. Desta forma, a aplicação de um movimento particular define uma estrutura de *vizinhança*, ou seja, um conjunto de soluções que podem ser geradas a partir de uma solução original pela aplicação do referido movimento. Os algoritmos de busca local, ao explorarem uma estrutura de vizinhança, exploram uma região específica do espaço de busca.

A solução obtida pela busca local em uma estrutura de vizinhança (dependendo da qualidade da mesma) pode ser um ótimo local, ou mesmo, um ótimo global.

Hansen and Mladenović [2001] propõem dois métodos metaheurísticos simples (e outras variações) de aplicação geral, baseadas em busca local em vizinhanças variáveis. A primeira delas é a *Descida em Vizinhança Variável* (ou *Variable Neighborhood Descent* – VND). Segundo os autores, o VND se baseia em três princípios:

1. Um ótimo local com relação a uma estrutura de vizinhança não é necessariamente um ótimo local relativo à outra estrutura de vizinhança;

2. Um ótimo global é um ótimo local com relação a todas as estruturas de vizinhança; e

3. Para muitos problemas, ótimos locais com relação a uma ou mais estruturas de vizinhança são relativamente próximos.

Este método consiste em definir um conjunto N_k de k_{max} diferentes estruturas de vizinhança (definidas por movimentos específicos) e determinar o ótimo local em cada uma destas vizinhanças, iteradamente. A cada iteração, uma vizinhança k específica é explorada pela aplicação sucessiva do movimento associado à solução π e, a cada ótimo local π' encontrado, a solução atual π é atualizada. Quando não há melhoria possível, a próxima vizinhança passa a ser explorada e o método se repete até que todas as k_{max} vizinhanças sejam analisadas. O Algoritmo 1 apresenta um pseudocódigo genérico para o VND.

Algoritmo 1: Pseudocódigo do método de Descida em Vizinhança Variável.

```
1  Inicialização: Selecione o conjunto de estruturas de vizinhança N_k = 1,...,k_max.
2  Determine uma solução inicial π;
3  k ← 1;
4  repita
5  |  Encontre um melhor vizinho π' de π (π' ∈ N_k(π));
6  |  se π' melhor que π então
7  |  |  π ← π';
8  |  fim
9  |  senão
10 |  |  k ← k + 1;
11 |  fim
12 até k = k_max;
```

Alguns dos componentes do VND que serão utilizados neste trabalho, bem como o método de geração de uma solução inicial, foram retirados da literatura. As seções a seguir descrevem brevemente estes componentes que farão parte das propostas de implementação na próxima etapa deste trabalho.

4.6 Uma Heurística Baseada em Busca em Grafos

Antes de determinar a sequência π de portas, determina-se a sequência ϕ de redes, de maneira a refletir o relacionamento entre as mesmas e tentar identificar quais as trilhas usadas simultaneamente. Para este fim, aplica-se uma *Busca em Largura* no grafo MOSP sobre o grafo

que representa a instância. A BFS gera uma sequência de vértices, referentes à ordem de exploração dos mesmos, utilizando-se como critério inicialmente escolher o vértice com menor grau dentre todos, e posteriormente, todos os demais em ordem crescente do grau. Caso o grafo seja desconexo, o processo se repete com o sequenciamento de um outro vértice com o menor grau dentre os vértices que ainda não tenham sido explorados.

O sequenciamento das portas é obtido por meio do método elaborado por Becceneri [1999b]. Este método simula a criação sucessiva das redes presentes em ϕ e verifica a composição das portas do problema, de modo que, caso a composição de uma porta seja um subconjunto das das redes criadas em determinado instante, ou seja, caso exista um subconjunto das trilhas criadas que utilizam a mesma porta, esta porta é imediatamente inserida ao final da solução π. O Algoritmo 2 apresenta o pseudocódigo para o sequenciamento das portas. A entrada consiste na lista ϕ de redes obtida a partir do sequenciamento das trilhas. A função $c(p_i)$ retorna quais redes compõem a porta p_i, e, a cada instante, o conjunto A de redes, representa as redes que vão sendo criadas a cada iteração do algoritmo.

Algoritmo 2: Pseudocódigo do sequenciamento de padrões [Becceneri, 1999b].

```
1 Entrada: Lista de peças φ.
2 A ← ∅;
3 para cada peça i ∈ φ faça
4 |   A ← A ∪ i;
5 |   se c(p_i) ⊆ A então
6 |   |   Insira p_i ao final de π;
7 |   fim
8 fim
```

4.7 Método *k-swap*

O *k-swap* é considerado uma generalização do método original *swap* – em que dois elementos são trocados de posição – projetado para o mesmo fim. Este método também é aplicado com frequência como método de busca local ou método de perturbação de soluções, incorporado em metaheurísticas.

O princípio do *k-swap* consiste na realização de trocas de posição entre k elementos pertencentes à solução de um problema combinatório, gerando soluções que diferem em exatamente k elementos da configuração inicial. Por exemplo, considere um problema em que são dispostos 4 elementos. A Figura 4.7 ilustra as transições decorrentes da aplicação do *swap* entre todos os pares de elementos dada uma configuração inicial $[1,2,3,4]$. A cada instante, dois elementos trocam de posição – as setas indicam os elementos envolvidos na troca de posições. A aplicação do método produz uma estrutura de vizinhança de seis elementos, referentes as combinações dos elementos tomados 2 a 2.

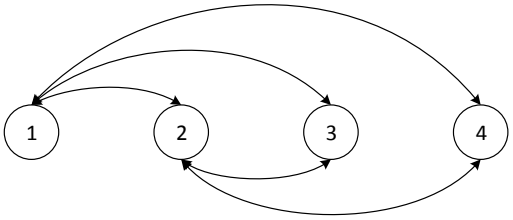

Figura 4.7: Exemplo de transições no método *k-swap*, no caso em que *k*=2 (*swap* clássico).

4.8 Descida Mais Rápida

O método de *Descida Mais Rápida* trata-se de um método de busca local baseado no conceito de melhoria iterativa de uma solução inicial. Um método de busca local é aplicado a uma solução iterativamente, atualizando-a sempre que uma melhoria for atingida. Quando um ótimo local for atingido (ou seja, quando não houver melhoria possível), o método se encerra. Este é um método de minimização, frequentemente utilizado para encontrar mínimos locais.

O Algoritmo 3 apresenta o pseudo-código genérico para o método de descida mais rápida em que $N(\pi)$ denota a vizinhança da solução corrente sob análise.

Algoritmo 3: Pseudocódigo do método Descida Mais Rápida.

```
1 Inicialização: Determine uma solução inicial π.
2 melhoria ← falso;
3 repita
4 |   Determine um vizinho π' de π (π' ∈ N(π));
5 |   se π' melhor que π então
6 |   |   π ← π';
7 |   |   melhoria ← verdadeiro;
8 |   fim
9 até melhoria = falso;
```

No Algoritmo 3, em um primeiro momento é encontrada uma solução inicial. Neste momento a solução corrente passa a ser a solução inicial e a análise sobre sua vizinhança tem início. Caso alguma solução vizinha seja melhor que a solução corrente o processo se repete para a nova solução corrente enquanto houver melhoria. Desse modo, o algoritmo só termina quando após a análise de toda uma vizinhança não for encontrada sequer uma solução melhor que a solução corrente.

Capítulo 5

Resultados e Discussão

O cronograma do projeto foi cumprido rigorosamente. Conforme apresentado no Capítulo 3, uma revisão sistemática da literatura foi realizada, sendo gerada toda a base conceitual sobre o problema abordado.

Foram analisadas técnicas e estruturas promissoras que ainda não foram consideradas na literatura para resolução do GMLP.

Na próxima etapa deste trabalho serão propostos novos métodos baseados nas técnicas e estruturas definidas no Capítulo 4 que serão aplicados à resolução do GMLP.

Capítulo 6

Conclusões

Este relatório parcial de iniciação científica relatou um trabalho de pesquisa realizado durante quatro meses referentes à primeira parte de um projeto. O cronograma do projeto foi cumprido rigorosamente. Foi gerada uma base conceitual a ser utilizada na etapa seguinte. Um relatório técnico foi gerado e será aproveitado em futuras publicações relacionadas ao tema e ao longo da execução deste projeto.

Referências Bibliográficas

JC Becceneri. O problema de sequenciamento de padrões para minimização do número máximo de pilhas abertas em ambientes de corte industriais. *São José dos Campos. Tese (Doutorado em Ciências no Curso de Engenharia Eletrônica e Computação na Área de Informática)– ITA, Centro Tecnológico Aeroespacial*, 1999a.

José Carlos Becceneri. O problema de sequenciamento de padrões para a minimização do número máximo de pilhas abertas em ambientes de cortes industriais. *European Journal of Operational Research*, page 145, 1999b.

Sao-Jie Chen and Yu Hen Hu. A new algorithm for cmos gate matrix layout. *IEEE International Conference*, 21(8):969–974, August 1988.

Sao-Jie Chen and Yu Hen Hu. A heuristic and an exact method for the gate matrix connection cost minimization problem. *Computers and Digital Techniques, IEE Proceed*, 1990a.

Sao-Jie Chen and Yu Hen Hu. Gm-plan: A gate matrix layout algorithm based on artificial intelligence planning techniques. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on, 9(8):836–845*, 1990b.

T. H. Cormen, C. Stein, R. L. Rivest, and C. E. Leiserson. *Introduction to Algorithms.* 2nd edn, mcgraw-hill higher education: Cambridge, ma. edition, 2001.

L De Giovanni, G Massi, and F Pezzella. An adaptive genetic algorithm for large-size open stack problems. *International Journal of Production Research*, 51(3):682–697, 2013.

Enrico Faggioli and Carlo Alberto Bentivoglio. Heuristic and exact methods for the cutting sequencing problem. *European Journal of Operational Research*, 110(3):564–575, 1998.

L. De Giovanni, G. Massib, F. Pezzella, M.E. Pfetschc, G. Rinaldid, and P. Venturad. Gm-learn : an iterative learning algorithm for cmos gate matrix layout. *International Transactions in Operational Research, 20(5):627–643*, 2013.

José Fernando Gonçalves, Mauricio G. C. Resende, and Miguel Dias Costa. A biased random-key genetic algorithm for the minimization of open stacks problem. *International Transactions in Operational Research*, 2014.

Pierre Hansen and Nenad Mladenović. Variable neighborhood search: Principles and applications. *European Journal of Operational Research*, 130(3):449 – 467, 2001. ISSN 0377-2217. doi: http://dx.doi.org/10.1016/S0377-2217(00)00100-4.

D. V. Heinbuch. Cmos3 cell library. *Reading, MA: AddisonWesley*, 1988.

Y. S. Hong, K. H. Park, and M. Kim. A heuristic for ordering the columns in one-dimensional logic array. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on, 9(8):836–845IEEE Trans. Computer-Aided Design, vol. 8, pp. 547–562*, 1989.

D. K. Hwang, W. K. Fuchs, and S. M. Kang. An efficient approach to gate matrix layout. *IEEE Trans. On Computer-Aided Design, Vol. CAD-6, No. 5, pp. 802-809, September*, 1987.

T. Kashiwabara and T. Fujisawa. Np-completeness of the problem of finding a minimum clique number interval graph containing a given graph as a subgraph. *Proceedings of the 1979 IEEE International Symposium on Circuits e Systems, Tokyo, Japan, July. p. 657–60.*, 1979.

Alexandre Linhares. Synthesizing a predatory search strategy for vlsi layouts. *IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION*, 1999.

Alexandre Linhares and Horacio Hideki Yanasse. Connections between cutting-pattern sequencing, vlsi design, and exible machines. *Computers and Operations Research 29*, 2002.

Alexandre Linhares, Horacio H. Yanasse, and José R. A. Torreão. Linear gate assignment: A fast statistical mechanics approach. *IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS*, 1999.

A. Mendes and A. Linhares. A multiple-population evolutionary approach to gate matrix layout. *International Journal of Systems Science.*, August 2004.

Rolf H Möhring. Graph problems related to gate matrix layout and pla folding. In *Computational graph theory*, pages 17–51. Springer, 1990.

Alexandre C. M. Oliveira and Luiz A. N. Lorena. A constructive genetic algorithm for gate matrix layout problems. *IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS, VOL. 21, NO. 8, AUGUST*, 2002.

K. Shahookar, W. Khamisani, P. Mazurnder, and S.M. Reddy. Genetic beam search for gate matrix layout. *Genetic beam search for gate matrix layout*, 1994.

B Smith and I Gent. Constraint modelling challenge 2005. In *IJCAI 2005 Fifth Workshop on Modelling and Solving Problems with Constraints*, pages 1–8, 2005.

Omar Wing, Shuo Huang, and Rui Wang. A constructive genetic algorithm for gate matrix layout problems. *IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN, VOL. CAD-4, NO. 3*, 1985.

H. H. Yanasse. A transformation for solving a pattern sequencing problem in the wood cut industry. *Pesquisa Operacional 17 (1), 57–70.*, 1997.

Horacio Hideki Yanasse and Edson Luiz França Senne. The minimization of open stacks problem: A review of some properties and their use in pre-processing operations. *European Journal of Operational Research*, 203(3):559 – 567, 2010. ISSN 0377-2217. doi: http://dx.doi.org/10.1016/j.ejor.2009.09.017. URL `http://www.sciencedirect.com/science/article/pii/S0377221709006328`.