GILBERTO KOJI NISHIOKA

**Estudo da programação de
atividades de desenvolvimento
de poços de petróleo marítimos**

**Dissertação apresentada à Escola
Politécnica da Universidade de São
Paulo para obtenção do título de
Mestre em Ciências**

**São Paulo
2014**

GILBERTO KOJI NISHIOKA

**Estudo da programação de
atividades de desenvolvimento
de poços de petróleo marítimos**

**Dissertação apresentada à Escola
Politécnica da Universidade de São
Paulo para obtenção do título de
Mestre em Ciências**

**Área de Concentração:
Engenharia Química**

**Orientador: Prof. Dr.
Galo Antonio Carrillo Le Roux**

**São Paulo
2014**

Este exemplar foi revisado e corrigido em relação à versão original, sob responsabilidade única do autor e com a anuência de seu orientador.

São Paulo, 06 de janeiro de 2014.

Assinatura do autor _____

Assinatura do orientador _____

FICHA CATALOGRÁFICA

Nishioka, Gilberto Koji

Estudo da programação de atividades de desenvolvimento de poços de petróleo marítimos / G.K. Nishioka. – versão corr. -- São Paulo, 2014.

114 p.

Dissertação (Mestrado) - Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia Química.

1. Pesquisa operacional 2. Programação matemática 3. Petróleo I. Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia Química II. t.

Dedico este trabalho a meus pais.

AGRADECIMENTOS

Agradeço ao Prof. Dr. Galo Antonio Carrillo Le Roux e ao Dr. Marcel Joly pela orientação e pelo apoio em todo o decorrer do trabalho.

Agradeço ao Prof. Dr. Marco Antonio Brinatti e ao Prof. Dr. Roger Rocha por participarem da minha banca de defesa, e por apontarem diversos pontos de melhoria que enriqueceram a versão revisada deste trabalho.

Agradeço à Petrobras S.A., em especial a Maria Janice Quevedo Ferro Cafasso e a Jacir Massayuki Murakami, por possibilitarem o início deste trabalho.

Agradeço a todos os amigos, e em especial a Fernando José de Moura Marcellino, Thiago Serra Azevedo Silva e Daniel Gustavo Exposti Trovó, pelas valiosas sugestões e discussões que enriqueceram este trabalho.

E agradeço à minha família e à minha noiva pelo incondicional apoio e incentivo durante todo o período deste trabalho.

RESUMO

Este trabalho detalha o *problema de programação de atividades de desenvolvimento de poços marítimos* (PPADPM), cujo objetivo é definir o uso de sondas e navios, de forma a otimizar a produção de óleo dos poços marítimos. São apresentados cinco modelos baseados em programação linear-inteira mista (MILP) e um modelo gerado via programação por restrições (PR) para este problema. Concluiu-se que, dentre os modelos MILP, aquele baseado em variáveis que definem a ordem de execução das atividades apresenta os melhores resultados, tanto em termos de qualidade de soluções quanto em termos de tamanho do modelo. O aumento do tamanho da instância (em número de poços e/ou em número de recursos) traz como consequência um menor número de nós analisados dentro de um tempo limite de execução do modelo, ao mesmo tempo em que se observa que as estratégias de escolha de variáveis, de escolha de nós e de uso de solução inicial nos modelos MILP tornam-se ineficientes. Na comparação entre modelos MILP e o modelo baseado em PR, constatou-se que este último obteve as melhores soluções em mais de 90% dos casos analisados e, consistentemente, obteve mais soluções viáveis para cada instância. Entretanto, por não fornecer informações sobre a qualidade da solução, o modelo baseado em PR somente obtém a solução ótima ao analisar todo o espaço de busca do modelo; desta forma, este apenas chegou à solução ótima em instâncias com até quatro poços, enquanto alguns dos modelos MILP obtiveram a solução ótima para instâncias com até treze poços.

Palavras-chave: Pesquisa operacional. Programação matemática. Petróleo.

ABSTRACT

This work details the offshore wells activities on specialized resources scheduling problem, focusing on the optimization of the total oil production. A Constraint Programming (CP) model and five Mixed-Integer Linear Programming (MILP) models were studied. Among the linear models, those based on activities order variables are smaller, and led to the better solutions for almost all of the instances analyzed. As far as the size of the instances (number of wells and/or number of resources) increases, the number of branched nodes in a limited time decreases and, consequently, the influence of B&B nodes and variable selection strategies and of the initial guesses on the MILP model solution gets less efficient. Comparing best solutions for each instance, the CP model outperformed MILP models in 90% of the studied cases. In addition, the CP based model consistently found more solutions per instance than the MILP based models did. However, because the CP model does not give information on the quality of a solution (gap), it was only able to obtain optimal solutions for instances with up to four wells, while MILP based models found optimal solutions for instances up to thirteen wells.

Keywords: Operational Research. Mathematical optimization. Petroleum.

LISTA DE FIGURAS

Figura 1.1 - Atividades de desenvolvimento típicas em um poço marítimo	18 -
Figura 1.2 - Profundidade e lâmina d'água de um poço marítimo	19 -
Figura 1.3 – Exemplo de uma programação de recursos e cálculo de perda de óleo acumulada. Atividades de deslocamento dos recursos são representadas pela letra “D”	22 -
Figura 2.1 – Representação matemática de um modelo MILP.....	23 -
Figura 2.2 – Exemplo de partição do espaço de busca no algoritmo B&B para variáveis binárias	28 -
Figura 2.3 – Diagrama esquemático do algoritmo B&B para um problema de minimização	30 -
Figura 2.4 – Representação matemática do JSP	35 -
Figura 2.5 – Representação matemática do FJSP	36 -
Figura 4.1 – Representação da variável de decisão do modelo VJS	40 -
Figura 4.2 – Representação da variável de decisão do modelo VOA	41 -
Figura 4.3 – Representação da variável de decisão do modelo VTD.....	42 -
Figura 4.4 – Exemplo de uso da relaxação Big-M.....	43 -
Figura 4.5 – Exemplo de uso da programação disjuntiva.....	43 -
Figura 4.6 - Cálculo de melhor data de conclusão de uma atividade	50 -
Figura 4.7 - Cálculo de ganho em um poço	51 -
Figura 4.8 – Representação da modelagem com variáveis de janela sequencial	53 -
Figura 4.9 – Representação da modelagem com variáveis de ordem entre atividades	60 -
Figura 4.10 – Representação da modelagem com variáveis de tempo discreto.....	68 -
Figura 5.1 – Tamanho em memória do modelo VJS-BM, conforme o número de poços e de recursos da instância considerada.	88 -
Figura 5.2 – Tamanho em memória do modelo VTD-BM, conforme o número de poços e de recursos da instância considerada.	89 -
Figura 5.3 – Tamanho em memória do modelo VTD-RL, conforme o número de poços e de recursos da instância considerada.	89 -
Figura 5.4 – Tamanho em memória do modelo VOA-BM, conforme o número de poços e de recursos da instância considerada.	90 -
Figura 5.5 – Tamanho em memória do modelo VOA-RL, conforme o número de poços e de recursos da instância considerada.	90 -
Figura 5.6 – Nós analisados conforme tipo de processamento paralelo, para um mesmo tempo de execução	91 -
Figura 5.7 – Comparação dos resultados obtidos com o processamento paralelo	92 -

Figura 5.8 – Gap obtido por cada modelo conforme o número de poços da instância (para três recursos)	93 -
Figura 5.9 – Melhor solução, melhor limite inferior e porcentagem de soluções viáveis obtidos.....	94 -
Figura 5.10 – Comparação do parâmetro de seleção de nós no B&B para o modelo VOA-RL, em relação ao número de poços (instâncias com três recursos).....	96 -
Figura 5.11 - Comparação do parâmetro de seleção de nós no B&B para o modelo VOA-RL, em relação ao número de recursos.....	97 -
Figura 5.12 - Comparação do parâmetro de seleção de nós no B&B para o modelo VOA-BM, em relação ao número de poços (instâncias com três recursos).....	97 -
Figura 5.13 - Comparação do parâmetro de seleção de nós no B&B para o modelo VOA-BM, em relação ao número de recursos	98 -
Figura 5.14 - Comparação do parâmetro de seleção de variáveis no B&B para o modelo VOA-RL, em relação ao número de poços (instâncias com três recursos)	99 -
Figura 5.15 - Comparação do parâmetro de seleção de variáveis no B&B para o modelo VOA-RL, em relação ao número de recursos.....	100 -
Figura 5.16 - Comparação do parâmetro de seleção de variáveis no B&B para o modelo VOA-BM, em relação ao número de poços (instâncias com três recursos)	100 -
Figura 5.17 - Comparação do parâmetro de seleção de variáveis no B&B para o modelo VOA-BM, em relação ao número de recursos.....	101 -
Figura 5.18 – Gap obtido com o uso de uma solução inicial no modelo VOA-BM (instâncias com três recursos)	102 -
Figura 5.19 – Gap obtido com o uso de uma solução inicial no modelo VOA-BM (instâncias com vinte poços)	103 -
Figura 5.20 – Gap obtido com o uso de uma solução inicial no modelo VOA-RL (instâncias com três recursos)	103 -
Figura 5.21 – Gap obtido com o uso de uma solução inicial no modelo VOA-RL (instâncias com vinte poços).....	104 -
Figura 5.22 – Gap final obtido conforme o número de poços da instância.....	105 -
Figura 5.23 – Tamanho em memória do modelo MPR, conforme o número de poços e de recursos da instância considerada.	106 -
Figura 5.24 – Soluções viáveis obtidas para uma instância com 50 poços e 3 recursos	107 -
Figura 5.25 – Soluções viáveis obtidas para uma instância com 50 poços e 19 recursos	107 -
Figura 5.26 – Comparação entre melhores soluções viáveis obtidas pelos modelos, conforme o número de poços da instância (referência: 3 recursos)	108 -

Figura 5.27 – Comparação entre melhores soluções viáveis obtidas pelos
modelos, conforme o número de recursos da instância
(referência: 20 poços)- 109 -

LISTA DE TABELAS

Tabela 2.1 – Comparação entre características da modelagem MILP e da programação por restrições (PR)	- 37 -
Tabela 4.1 – Parâmetros utilizados pelos modelos estudados.....	- 47 -
Tabela 4.2 – Equações do modelo VJS-BM.....	- 55 -
Tabela 4.3 – Equações do modelo VOA-BM.....	- 62 -
Tabela 4.4 – Equações do modelo VOA-RL	- 65 -
Tabela 4.5 – Equações do modelo VTD-BM	- 70 -
Tabela 4.6 – Equações do modelo VTD-RL.....	- 74 -
Tabela 4.7 – Equações do modelo MPR.....	- 79 -
Tabela 4.8 – Instâncias de dados utilizadas para os testes dos modelos	- 82 -
Tabela 5.1 – Estimativa de variáveis e de restrições dos modelos MILP (considerando-se o pior caso).....	- 87 -

LISTA DE ABREVIATURAS E SIGLAS

B&B	Algoritmo <i>Branch & Bound</i>
BBS	<i>Best Bound Search</i>
BES	<i>Best Estimated Search</i>
boed	Barril equivalente de óleo por dia
DFS	<i>Depth First Search</i>
FJSP	<i>Flexible Job-Shop Scheduling Problem</i>
JSP	<i>Job-Shop Scheduling Problem</i>
MILP	<i>Mixed Integer Linear Programming</i> (em português, Programação Linear Inteira Mista)
MPR	Modelo baseado em Programação por Restrições
PM	Programação Matemática (do inglês, <i>Mathematical Programming</i>)
PPADPM	Problema de Programação de Atividades de Desenvolvimento de Poços Marítimos
PR	Programação por Restrições (do inglês, <i>Constraint Programming</i>)
VOA-BM	Modelo baseado em variáveis de ordem entre atividades com relaxação tipo Big-M
VOA-RL	Modelo baseado em variáveis de ordem entre atividades e restrições lógicas
VJS-BM	Modelo baseado em variáveis de janela sequencial nos recursos com relaxação tipo Big-M
VTD-BM	Modelo baseado em variáveis de tempo discreto com relaxação tipo Big-M
VTD-RL	Modelo baseado em variáveis de tempo discreto e restrições lógicas

SUMÁRIO

1 INTRODUÇÃO	- 16 -
1.1 MOTIVAÇÃO.....	- 16 -
1.2 OBJETIVOS	- 17 -
1.3 DESCRIÇÃO DO PROBLEMA.....	- 17 -
2 INTRODUÇÃO TEÓRICA	- 23 -
2.1 MODELAGEM BASEADA EM MILP	- 23 -
2.1.1 Domínio de uma variável de decisão	- 24 -
2.1.2 Função objetivo	- 24 -
2.1.3 Variáveis de decisão.....	- 24 -
2.1.4 Restrições.....	- 24 -
2.1.5 Parâmetros	- 25 -
2.1.6 Espaço de busca de solução	- 25 -
2.1.7 Solução viável.....	- 25 -
2.1.8 Solução ótima	- 25 -
2.1.9 Relaxação linear de um MILP	- 26 -
2.2 ALGORITMO BRANCH & BOUND (B&B).....	- 26 -
2.2.1 Definições iniciais do Algoritmo B&B	- 26 -
2.2.1.1 Nó e árvore	- 26 -
2.2.1.2 Limite superior, limite inferior e gap	- 27 -
2.2.2 Partição do espaço de busca de soluções.....	- 27 -
2.2.3 Corte de nós	- 28 -
2.2.4 Funcionamento do algoritmo B&B	- 29 -
2.2.5 Estratégias de seleção de nós	- 31 -
2.2.6 Estratégia de seleção de variáveis	- 31 -
2.3 CONCEITOS DE COMPLEXIDADE DE ALGORITMOS.....	- 32 -
2.3.1 Problema de decisão	- 32 -
2.3.2 Classe P	- 32 -
2.3.3 Classe NP.....	- 33 -
2.3.4 Redução polinomial	- 33 -
2.3.5 Classe NP-completo	- 33 -

2.3.6 Classe NP-difícil	- 33 -
2.4 PROGRAMAÇÃO (SCHEDULING).....	- 34 -
2.4.1 Job-Shop Problem (JSP)	- 34 -
2.4.2 Flexible Job-Shop Problem (FJSP).....	- 35 -
2.5 PROGRAMAÇÃO POR RESTRIÇÕES (PR)	- 36 -
3 REVISÃO BIBLIOGRÁFICA	- 38 -
4 MATERIAIS E MÉTODOS.....	- 40 -
4.1 ABORDAGEM MATEMÁTICA DO PPADPM	- 40 -
4.2 CARACTERÍSTICAS ESPECÍFICAS DO PROBLEMA.....	- 44 -
4.3 MODELAGEM MATEMÁTICA	- 45 -
4.3.1 Considerações iniciais	- 45 -
4.3.2 Simplificações em relação ao problema real	- 45 -
4.3.3 Restrições.....	- 46 -
4.3.4 Parâmetros	- 47 -
4.3.5 Função objetivo	- 52 -
4.3.6 Modelagem MILP.....	- 52 -
4.3.6.1 Modelo com Variáveis de Janela Sequencial - Big-M (VJS-BM)	- 52 -
4.3.6.1.1 Variáveis de decisão	- 54 -
4.3.6.1.2 Função objetivo	- 54 -
4.3.6.1.3 Restrições	- 54 -
4.3.6.2 Modelo com Variáveis de Ordem entre Atividades – Big-M (VOA-BM).....	- 59 -
4.3.6.2.1 Variáveis de decisão	- 60 -
4.3.6.2.2 Função objetivo	- 61 -
4.3.6.2.3 Restrições	- 61 -
4.3.6.3 Modelo com Variáveis de Ordem entre Atividades – Restrições Lógicas (VOA-RL)	- 64 -
4.3.6.3.1 Restrições	- 64 -
4.3.6.4 Modelo com variáveis de tempo discreto – Big-M (VTD-BM)	- 67 -
4.3.6.4.1 Variáveis de decisão	- 68 -
4.3.6.4.2 Função objetivo	- 69 -
4.3.6.4.3 Restrições	- 69 -
4.3.6.5 Modelo com variáveis de tempo discreto – Restrições Lógicas (VTD-RL).....	- 73 -

4.3.6.5.1 Restrições	- 73 -
4.3.7 Modelagem MPR	- 77 -
4.3.7.1 Variáveis de decisão.....	- 78 -
4.3.7.2 Função objetivo	- 78 -
4.3.7.3 Restrições.....	- 79 -
4.4 EXECUÇÃO DOS TESTES	- 80 -
4.4.1 Instâncias de dados utilizadas	- 80 -
4.4.2 Condições para a realização dos testes	- 83 -
4.4.3 Processamento paralelo no pacote de otimização CPLEX®	- 83 -
4.4.4 Parâmetros do pacote de otimização CPLEX® utilizados	- 84 -
4.4.5 Resultados gerados para cada execução de modelo	- 85 -
4.4.6 Comparação e análise de resultados	- 86 -
4.4.6.1 Tuplas de resultados	- 86 -
4.4.6.2 Critério de comparação de resultados	- 86 -
5 DISCUSSÃO DOS RESULTADOS	- 87 -
5.1 TAMANHO DOS MODELOS MILP	- 87 -
5.2 PROCESSAMENTO PARALELO	- 91 -
5.3 COMPARAÇÃO ENTRE SOLUÇÕES OBTIDAS.....	- 93 -
5.4 PARÂMETRO DE SELEÇÃO DE NÓS NO B&B	- 95 -
5.5 PARÂMETRO DE SELEÇÃO DE VARIÁVEIS NO B&B	- 98 -
5.6 USO DE SOLUÇÃO INICIAL NO B&B.....	- 102 -
5.7 COMPARAÇÃO COM O MODELO DE PROGRAMAÇÃO POR RESTRIÇÕES (MPR)	- 105 -
6 CONCLUSÕES	- 110 -
7 TRABALHOS FUTUROS	- 111 -
8 REFERÊNCIAS.....	- 112 -

1 INTRODUÇÃO

1.1 MOTIVAÇÃO

O consumo de energia em todo o mundo cresce significativamente a cada ano, e alguns estudos estimam que esse consumo terá um acréscimo de 53% no período entre 2008-2035 (U. S. ENERGY INFORMATION ADMINISTRATION, 2011). E um dos grandes desafios atuais é garantir o atendimento desta demanda energética.

Dentre as diversas fontes de energia disponíveis, os derivados de petróleo compõem a fonte energética mais importante, respondendo por mais de 33% de toda a energia mundial atual (INTERNATIONAL ENERGY AGENCY (IEA), 2011). E projeções para 2035 mostram que os combustíveis fósseis permanecerão como a principal fonte de energia em todo o mundo (U. S. ENERGY INFORMATION ADMINISTRATION, 2011).

A Petrobras, uma das maiores empresas de energia do mundo, produz em torno de dois milhões de boed¹, dos quais aproximadamente 80% são provenientes de águas profundas (PETROBRAS S.A., 2013a). Com um plano de negócios agressivo, a empresa busca duplicar sua produção diária até 2020 (PETROBRAS S.A., 2013b).

Como aproximadamente 80% das reservas comprovadas de petróleo no Brasil localizam-se em áreas de águas profundas (inclusive as áreas de Pré-Sal) (PETROBRAS S.A., 2013a), a empresa naturalmente direciona grande parte dos seus investimentos na exploração dessas áreas.

Para que a Petrobras atinja suas metas de produção, é imprescindível que novos poços sejam perfurados. Uma vez confirmada a viabilidade econômica de um poço, engenheiros especializados da empresa definem quais atividades devem ser executadas para que, efetivamente, seja iniciada a produção de óleo deste poço. E estas atividades são executadas por recursos dedicados para esse fim.

Na prática, como o número de poços a serem perfurados é elevado (o que, conseqüentemente, resulta em um grande número de atividades a serem realizadas),

¹ barril equivalente de óleo por dia

e o número de recursos capazes de atender estas demandas é escasso, surge a necessidade de se obter programações otimizadas para esses recursos.

Ao considerar-se que esses recursos têm um alto custo de operação, e que alguns dos poços têm uma produção diária elevada, o resultado dessa programação, além de afetar o retorno sobre investimento da empresa, influi diretamente nas metas de produção de óleo estabelecidas em seu plano de negócio.

Com este cenário, o uso de técnicas de pesquisa operacional torna-se uma ferramenta essencial para a obtenção de programações de qualidade, garantindo um melhor retorno sobre o investimento da empresa.

1.2 OBJETIVOS

Este trabalho tem como contribuição principal apresentar uma análise comparativa entre cinco modelos matemáticos e um modelo de programação por restrições para o *Problema de Programação de Atividades de Desenvolvimento de Poços Marítimos* (PPADPM). As diferenças entre os modelos testados devem-se às variáveis de decisão empregadas e às técnicas de modelagem utilizadas, que se aproveitam de características intrínsecas do problema.

1.3 DESCRIÇÃO DO PROBLEMA

Após a comprovação da viabilidade econômica de um poço marítimo, é necessária a execução de atividades específicas – denominadas *atividades de desenvolvimento* - para que seja possível extrair o óleo desse poço. A figura 1.1 apresenta as atividades de desenvolvimento típicas realizadas em um poço marítimo.

Figura 1.1 - Atividades de desenvolvimento típicas em um poço marítimo

As atividades de desenvolvimento tipicamente executadas em um poço são as seguintes:

- Perfuração: conjunto de operações relacionadas à perfuração do poço, com o objetivo de alcançar reservatórios de petróleo e/ou comunica-los à superfície;
- Completação: conjunto de operações necessárias para deixar um poço pronto para produzir óleo. Esta atividade é executada por uma sonda;
- Avaliação: conjunto de operações destinadas a avaliar a produtividade do poço. Esta atividade é executada por uma sonda;
- Interligação: executada por um navio lançador de linhas, tem por objetivo conectar um duto do poço até uma plataforma localizada ao nível do mar. É através deste duto que todos os fluidos obtidos do poço são escoados.

Cada atividade de desenvolvimento possui como características:

- Duração: tempo necessário para a realização da atividade, em dias;
- Categoria da atividade: define qual o tipo de recurso pode atender a atividade.

- Lâmina d'água do poço associado: distância da cabeça do poço até o nível do mar, em metros (figura 1.2);
- Profundidade do poço associado: esta informação, dada em metros, é importante apenas para as atividades que são executadas por sondas (figura 1.2);

Figura 1.2 - Profundidade e lâmina d'água de um poço marítimo

Os recursos disponíveis para a execução das atividades de desenvolvimento são divididos em dois tipos:

- Sondas de perfuração: navios com equipamentos específicos para perfuração de poços marítimos. As sondas têm como características a capacidade de perfuração e a máxima lâmina d'água em que podem operar;
- Navios lançadores de linhas - PLSV²: navios com equipamentos específicos para a conexão de um poço marítimo a uma plataforma

² PLSV: Sigla em inglês de *Pipe-Laying Support Vessel*.

petrolífera. Estes navios têm como característica a máxima lâmina d'água em que podem operar.

Cada atividade de desenvolvimento deve ser atendida por apenas um dos recursos cujas características atendam as especificações da atividade (denominados *recursos compatíveis*).

Além disso, quando é iniciada a execução de uma atividade de desenvolvimento, esta não pode ser interrompida até o seu término.

Outra característica a considerar é que apenas um recurso pode operar por vez em um poço, de forma a minimizar a possibilidade de acidentes operacionais, como colisões entre recursos.

Por fim, outra característica importante do problema é a existência de relações de precedência entre todas as atividades de um poço, devido a restrições tecnológicas (por exemplo, uma atividade de completação só deve ser executada após o término da atividade de perfuração).

A figura 1.3 apresenta um exemplo de programação de três recursos (duas sondas e um navio) para a execução de atividades de seis poços e um gráfico com a perda acumulada de óleo de cada um dos poços da instância. Nesta figura, as atividades de um mesmo poço possuem a mesma coloração, e cada um dos poços tem informações de produção de óleo e de melhor data possível de conclusão. As atividades de deslocamento dos recursos são representadas na figura com a letra "D". Caso as atividades de um poço não sejam concluídas até a data ideal, a partir desta inicia-se a contabilização de perda de produção do poço, que cessa no momento em que todas as atividades do poço são concluídas. No gráfico de perda de produção de óleo apresentado na figura, a contribuição de cada poço tem a mesma coloração das atividades desse poço.

Em relação à dinâmica de aprovação das programações, são realizadas reuniões nas quais os interessados verificam os resultados obtidos, e aprovam a programação que efetivamente será aplicada aos recursos. Usualmente, os participantes dessas reuniões propõem alterações na programação, que podem ou não gerar melhores resultados em termos de produção de óleo. Como a programação final deve ser aprovada ao final dessas reuniões, é necessário que novas programações sejam apresentadas no decorrer da reunião, para avaliar se as alterações propostas são aceitáveis. Devido a essa dinâmica de aprovação das

programações, é interessante que uma programação dos recursos seja obtida dentro de um tempo limite.

O *Problema de Programação de Atividades de Desenvolvimento de Poços Marítimos* (PPADPM) consiste em definir quando cada atividade será executada e qual recurso atenderá cada atividade, maximizando a produção de óleo dos poços considerados e obedecendo às restrições acima descritas. Com esta definição, observa-se que o PPADPM é um problema de programação de recursos³.

³ Termo em inglês: *Scheduling*. Outro termo em português comumente utilizado é *escalonamento*.

Figura 1.3 – Exemplo de uma programação de recursos e cálculo de perda de óleo acumulada. Atividades de deslocamento dos recursos são representadas pela letra “D”.

2 INTRODUÇÃO TEÓRICA

Este capítulo apresenta uma breve introdução teórica dos tópicos e dos termos que serão utilizados no decorrer da dissertação.

2.1 MODELAGEM BASEADA EM MILP

A representação matemática de um MILP é dada a seguir (NEMHAUSER e WOLSEY, 1998):

$$\begin{aligned} \max \quad & cx + hy \\ \text{s.t.} \quad & Ax + Gy \leq b \\ & x \in R^+, y \in Z^+ \end{aligned}$$

Onde:

- c é um vetor linha n -dimensional
- h é um vetor linha p -dimensional
- A é uma matriz $m \times n$
- x é um vetor coluna n -dimensional
- G é uma matriz $m \times p$
- y é um vetor coluna p -dimensional
- b é um vetor coluna m -dimensional

Figura 2.1 – Representação matemática de um modelo MILP

Esta representação possui as seguintes características:

- Utiliza tanto variáveis de decisão inteiras quanto variáveis de decisão contínuas e/ou binárias;
- As restrições do problema são representadas por equações e inequações lineares;
- Possui como função objetivo uma equação linear, que deve ser maximizada ou minimizada;

- Cada variável possui um domínio de valores, que pode ou não ser limitado.

2.1.1 Domínio de uma variável de decisão

Na representação matemática do MILP, $x \in R^+$ e $y \in Z^+$ definem os possíveis valores que as variáveis de decisão podem assumir. Esse conjunto de valores possíveis é denominado *domínio* da variável de decisão.

2.1.2 Função objetivo

A função $cx + hy$ é denominada *função objetivo* do MILP, e é através do seu valor que é possível otimizar o problema, uma vez que tem-se um escalar que permite a comparação entre soluções viáveis.

2.1.3 Variáveis de decisão

As variáveis x e y do MILP são denominadas *variáveis de decisão*, e seus valores devem estar dentro do respectivo domínio estabelecido no problema.

2.1.4 Restrições

As inequações $Ax + Gy \leq b$ representam as *restrições* do modelo, e são a partir delas que se garante que as variáveis de decisão atendem a todas as características do problema.

2.1.5 Parâmetros

As constantes c , h , A , G e b são denominadas *parâmetros*. Os valores dessas constantes são obtidos através de informações obtidas do problema real.

2.1.6 Espaço de busca de solução

Denomina-se *espaço de busca de solução* o conjunto formado pelas possíveis combinações de valores das variáveis do problema. Embora cada uma dessas combinações não necessariamente atenda a todas as restrições modeladas, obrigatoriamente as variáveis de decisão devem assumir valores dentro de seu respectivo domínio.

2.1.7 Solução viável

Uma *solução viável* de um MILP corresponde a uma combinação das variáveis de decisão que atende a todas as restrições do modelo. E cada solução viável tem um valor de função objetivo associado.

2.1.8 Solução ótima

Uma solução é denominada *ótima* se for uma solução viável que resulta no melhor valor possível de função objetivo do modelo.

2.1.9 Relaxação linear de um MILP

A relaxação linear equivale à exclusão da condição de integralidade das variáveis de decisão de um MILP, permitindo que as variáveis de decisão inteiras e binárias assumam valores reais. Assim, tem-se que em um modelo linear, o valor da função objetivo de sua solução ótima pode ser utilizado como um limite inferior do MILP original em um problema de minimização (propriedade demonstrada em NEMHAUSER e WOLSEY (1998), pag. 24 e 25).

2.2 ALGORITMO BRANCH & BOUND (B&B)

O algoritmo Branch & Bound baseia-se na ideia de enumerar de forma inteligente todas as possíveis soluções de um problema combinatório, permitindo provar que uma solução é ótima (PAPADIMITRIOU e STEIGLITZ, 1998). Este algoritmo B&B é composto por duas etapas, a partição do espaço de busca e a eliminação de nós.

2.2.1 Definições iniciais do Algoritmo B&B

2.2.1.1 Nó e árvore

Neste algoritmo, um *nó*⁴ representa uma combinação de valores das variáveis de decisão do problema.

Se em um nó, o valor atribuído para cada uma das variáveis de decisão estiver dentro do seu respectivo domínio, este nó ou será inviável (se alguma das restrições do problema for violada), ou resultará numa solução viável do problema.

⁴ Na teoria de grafos, um nó do algoritmo B&B seria equivalente a um *vértice*.

2.2.1.2 Limite superior, limite inferior e gap

Em um problema de *minimização*, o menor valor dentre as funções objetivo das soluções viáveis já encontradas é denominado *limite superior (LS)*⁵.

E o menor valor de função objetivo obtida da relaxação dos nós restantes do algoritmo B&B é denominado *limite inferior (LI)*⁶.

O valor dado por:

$$gap = \frac{(LS - LI)}{LS} \quad (2.1)$$

É denominado *gap* da solução, e representa a diferença entre a melhor solução encontrada e o melhor limite inferior obtido pelo algoritmo B&B.

2.2.2 Partição do espaço de busca de soluções⁷

A partir de um nó, criam-se nós filhos idênticos ao nó original, exceto pela inclusão de uma restrição que limita os valores que podem ser atribuídos a uma das variáveis de decisão (especialmente no caso de uma variável binária, essa restrição força essa variável a ter um valor fixo). Este processo se repete enquanto houver nós que contenham variáveis de decisão com domínio violado.

Com esse processo, monta-se uma *árvore*⁸ de nós do problema.

⁵ Termo em inglês: *Upper Bound*

⁶ Termo em inglês: *Lower Bound*

⁷ No algoritmo B&B, esta etapa equivale ao *branching*.

⁸ Na teoria de grafos, uma *árvore* equivale a um grafo onde existe um único caminho para cada par de nós.

Por exemplo:

Dado um problema onde hajam duas variáveis binárias de decisão $a_1, a_2 \in B$. Se o nó pai tem definido que a variável $a_1 = 1$, serão criados os seguintes nós filhos:

- Nó filho 1: $a_1 = 1$ e restrição $a_2 \leq 0$ (isto é, $a_2 = 0$);
- Nó filho 2: $a_1 = 1$ e restrição $a_2 \geq 1$ (isto é, $a_2 = 1$);

Neste caso, como cada um dos nós filhos tem valores atribuídos para todas as variáveis de decisão, não há necessidade de repetir o processo para os nós recém-criados.

Figura 2.2 – Exemplo de partição do espaço de busca no algoritmo B&B para variáveis binárias

2.2.3 Corte de nós⁹

No algoritmo B&B, um nó é eliminado por:

- Infactibilidade: a relaxação linear do nó é inviável;
- Otimalidade: a relaxação linear do nó gera uma solução na qual todas as variáveis de decisão possuem valores que respeitem os respectivos domínios do problema original;
- Limitante: a relaxação linear do nó gera um valor de função objetivo maior que o limite superior encontrado pelo algoritmo B&B (considerando um problema de minimização).

Em contrapartida aos cortes por infactibilidade e por otimalidade, o corte de nós por limitante permite a eliminação de nós sem a necessidade de analisá-los.

⁹ No algoritmo B&B, esta etapa equivale ao *bound*.

2.2.4 Funcionamento do algoritmo B&B

A figura 2.3 apresenta um diagrama representando o funcionamento de um algoritmo B&B padrão para o caso de minimização (PAPADIMITRIOU e STEIGLITZ, 1998).

Neste algoritmo, uma vez definido o nó a analisar, é calculada sua relaxação linear, resultando em um valor de limite inferior. Após a seleção de uma das variáveis com domínio violado em relação ao problema original, são criados os nós filhos e são calculadas suas respectivas relaxações. Se na relaxação de um nó, todas as variáveis inteiras do problema assumirem valores inteiros, a solução gerada será adotada como limite superior do problema, caso seu valor seja o melhor já obtido até o momento. E o nó em questão é retirado da lista de nós a analisar.

Se a relaxação linear de um nó for inviável, ou se seu valor for superior ao limite superior atual, este nó é cortado.

O limite inferior global sempre é atualizado para que seja igual ao menor valor de relaxação linear dentre os nós que ainda não foram analisados.

O processo se repete até que não existam mais nós a analisar.

Figura 2.3 – Diagrama esquemático do algoritmo B&B para um problema de minimização

2.2.5 Estratégias de seleção de nós

Como observado no diagrama esquemático do B&B da figura 2.3, é possível a escolha do nó a particionar. As estratégias de seleção de nós mais utilizadas para o algoritmo B&B são as seguintes:

- Depth-First Search (DFS):

Seleciona-se o último nó criado no algoritmo B&B;

- Best-Bound Search (BBS):

Seleciona-se o nó com a melhor função objetivo para a relaxação linear associada;

- Best-Estimated Search (BES):

Seleção do nó com melhor objetivo estimado (baseado na estimativa de pseudo-custos de uma variável)

2.2.6 Estratégia de seleção de variáveis

Pelo diagrama esquemático do B&B (figura 2.3), observa-se que é possível escolher a variável utilizada para a partição do nó. As estratégias de seleção de variáveis mais utilizadas são as seguintes:

- Pseudo-custos:

Dentre as variáveis fracionárias, seleciona-se a que mais degrada a função objetivo;

- Máxima infactibilidade:

Seleciona-se a variável com o maior termo fracionário (isto é, com a parte fracionária mais próxima de 0.5);

- Escolha agressiva:

Seleciona-se a variável mais promissora após a resolução parcial de alguns subproblemas gerados a partir do ramo analisado.

2.3 CONCEITOS DE COMPLEXIDADE DE ALGORITMOS

Esta seção traz algumas definições da teoria de complexidade de algoritmos, baseadas em WOLSEY (1998).

2.3.1 Problema de decisão

Dada uma informação de entrada, o *problema de decisão* tem como resposta ou um “sim” ou um “não”.

Como os problemas de programação geralmente são de otimização, alternativamente, é possível formular um problema de decisão equivalente, da seguinte forma: “*Dada uma solução S, existe outra solução com função objetivo melhor do que S?*”.

2.3.2 Classe P

Se, para qualquer entrada de dados, existe um algoritmo que, em tempo polinomial, traz a resposta do problema de decisão, então o problema em questão pertence à *classe P*.

Com esta definição, tem-se que a classe P está contida na classe NP.

2.3.3 Classe NP

Se, para qualquer entrada de dados, é possível verificar em tempo polinomial que a resposta de um problema de decisão está correta, então o problema em questão pertence à *classe NP*.

2.3.4 Redução polinomial

Dados dois problemas Q e R , ambos pertencentes à classe NP, se uma instância de R pode ser convertida em uma instância de Q em tempo polinomial, então R é *polinomialmente redutível* a Q .

2.3.5 Classe NP-completo

Um problema de decisão Q pertence à classe *NP-completo* se todos os problemas pertencentes à classe NP podem ser reduzidos polinomialmente para Q .

Uma consequência desta definição é que se existir um algoritmo polinomial que resolva o problema Q , todos os problemas da classe NP também serão resolvidos em tempo polinomial.

2.3.6 Classe NP-difícil

Um problema de otimização é chamado de *NP-difícil* se o problema de decisão equivalente pertence à classe NP-completo.

2.4 PROGRAMAÇÃO (SCHEDULING)

A programação é um processo de apoio à decisão comumente encontrado em indústrias de manufatura e em empresas de serviços, e trata da alocação de tarefas a um conjunto de máquinas no decorrer do tempo, buscando a otimização de um ou mais objetivos.

A solução de um problema de programação tem a forma de uma sequência de tarefas no tempo, para cada uma das máquinas.

A seguir, serão detalhados dois problemas de programação utilizados como base para o objeto de estudo deste trabalho.

Para um aprofundamento em problemas de programação, sugere-se a leitura de PINEDO (2011).

2.4.1 Job-Shop Problem (JSP)

O problema denominado *Job-Shop Problem* clássico (JSP) tem a seguinte definição, baseada em BRUCKER, JURISCH e SIEVERS (1994):

Dado:

- Um conjunto de máquinas $M = \{M_1, M_2, \dots, M_k\}$;
- Um conjunto de tarefas $J = \{J_1, J_2, \dots, J_n\}$;
- Cada tarefa J_j é composta por um conjunto de operações $O_n = \{o_{1j}, o_{2j}, \dots, o_{nj}\}$;
- Cada operação o_{ij} tem uma duração pré-definida;
- Há uma ordem definida de execução das operações o_{ij} de uma tarefa J_j ;
- Uma operação, quando iniciada, não pode ser interrompida;
- Cada operação é executada por uma única máquina, já definida;
- Uma máquina atende apenas uma operação por vez.

O objetivo é definir a sequência de execução das operações para cada máquina, otimizando um determinado critério.

Figura 2.4 – Representação matemática do JSP

Em relação à complexidade, GAREY, JOHNSON e SETHI (1976) demonstram que o JSP é um problema da classe NP-difícil.

2.4.2 Flexible Job-Shop Problem (FJSP)

O problema denominado FJSP é definido da seguinte forma (BRUCKER e SCHLIE, 1990):

Dado:

- Um conjunto de máquinas $M = \{M_1, M_2, \dots, M_k\}$;
- Um conjunto de tarefas $J = \{J_1, J_2, \dots, J_n\}$;
- Cada tarefa J_j é composta por um conjunto de operações $O_n = \{o_{1j}, o_{2j}, \dots, o_{nj}\}$;
- Cada operação o_{ij} tem uma duração pré-definida;
- Para cada operação o_{ij} há um conjunto de máquinas $M_{ij} \subseteq M$ que podem executá-la;
- Há uma ordem definida de execução das operações o_{ij} de uma tarefa J_j ;
- Uma operação, quando iniciada, não pode ser interrompida;
- Cada operação é executada por uma única máquina do conjunto M_{ij} ;
- Uma máquina atende apenas uma operação por vez.

O objetivo é definir a sequência de execução das operações para cada máquina, otimizando um determinado critério.

Figura 2.5 – Representação matemática do FJSP

O FJSP é uma generalização do JSP clássico, ao considerar que há um conjunto de recursos que podem executar uma determinada operação. E, da mesma forma como o JSP clássico, o FJSP é um problema NP-difícil.

2.5 PROGRAMAÇÃO POR RESTRIÇÕES (PR)

Enquanto a Programação Matemática (PM) teve sua origem na comunidade de pesquisa operacional, a Programação por Restrições (PR) originou-se das comunidades de Inteligência Artificial e de Ciências da Computação. A PR foi inicialmente concebida para obter soluções de problemas de satisfação¹⁰, mas nas duas últimas décadas evoluiu de forma a ser aplicada também em problemas de otimização.

¹⁰ Problema de decisão cuja definição é a seguinte: “Dado um conjunto de variáveis binárias e um conjunto de expressões baseadas nessas variáveis, existe uma combinação de valores das variáveis que tornam todas as expressões verdadeiras?”

A PR utiliza a propagação de restrições, que tem por objetivo diminuir o domínio de cada variável de decisão, de acordo com as inferências sobre restrições do problema considerado. Também faz uso de restrições globais, que podem ser propagadas de modo eficiente por se beneficiarem da estrutura do problema (um exemplo de restrição global é o *allDifferent*, que garante que cada uma das variáveis consideradas terá um valor diferente entre si).

Para problemas de otimização, o algoritmo utilizado para a busca de soluções viáveis da PR foi adaptado de forma que, a cada solução viável encontrada, inclui-se uma nova restrição relacionada ao valor da função objetivo do problema.

Para um aprofundamento na PR, sugere-se a consulta à dissertação de SERRA (2012), que descreve detalhadamente as características e os mecanismos de busca utilizados pela técnica.

A tabela a seguir apresenta uma comparação entre as características da PR e da modelagem MILP:

Tabela 2.1 – Comparação entre características da modelagem MILP e da programação por restrições (PR)

	PR	MILP
Foco	Soluções viáveis	Solução ótima
Representação de restrições e da função objetivo	Flexível	Via inequações lineares
Informações de qualidade da solução	Nenhuma	Diferença entre a melhor solução encontrada e a estimativa da relaxação linear (gap)

Observa-se que a PR é uma modelagem mais flexível que a modelagem MILP, pois não está limitada ao uso de uma representação linear; por outro lado, a PR não fornece nenhuma informação sobre a qualidade das soluções encontradas.

3 REVISÃO BIBLIOGRÁFICA

O Problema de Programação de Atividades de Desenvolvimento de Poços Marítimos (PPADPM) pode ser comparado ao JSP clássico, pois é possível criar uma correspondência direta entre os dois problemas, da seguinte forma:

- Cada máquina do JSP representa um recurso do PPADPM;
- Cada tarefa do JSP representa um poço do PPADPM;
- Cada operação do JSP representa uma atividade de poço do PPADPM;
- O objetivo é maximizar a produção de óleo calculada a partir do término de execução de cada atividade do JSP.

Através da correspondência acima descrita, NASCIMENTO (2002) demonstrou que o PPADPM pertence à classe NP-difícil.

Uma diferença do PPADPM em relação ao JSP clássico é o fato de não existir uma predefinição de qual recurso atenderá cada atividade. Com isto, o problema considerado torna-se um FJSP, em que cada operação é executada por um recurso dentre um conjunto de recursos compatíveis.

Como ainda não é conhecido um algoritmo para problemas classe NP-difícil que obtenha, em tempo polinomial, a solução ótima para qualquer instância de dados (NEMHAUSER e WOLSEY, 1998), justifica-se o uso de heurísticas para a obtenção de soluções para problemas dessa classe na resolução de instâncias maiores.

Partindo desta justificativa, os artigos científicos que tratam de problemas semelhantes ao PPADPM adotam diversas técnicas baseadas em heurísticas, com o objetivo de obter soluções de qualidade no menor tempo possível de processamento computacional. HASLE, HAUT, *et al.* (1996) aplicam o CR¹¹ para obter soluções para o problema de alocação de atividades de poços. DA COSTA e FERREIRA FILHO (2005) propõem uma heurística de montagem dinâmica em atividades de

¹¹ CR: Constraint Reasoning

manutenção de poços, problema similar ao PPADPM, exceto por não considerar precedências entre as atividades. MOURA, PEREIRA e DE SOUZA (2008) e PEREIRA, MOURA e DE SOUZA (2005) aplicam a meta-heurística GRASP¹² em uma versão do PPADPM que considera períodos de indisponibilidade dos recursos, restrições geográficas no uso dos recursos e janelas de tempo de execução das atividades. DARIO, ALOISEA, *et al.* (2006) utilizam a heurística VNS¹³ em um problema que trata a locação de sondas em poços terrestres. SERRA, NISHIOKA e MARCELLINO (2011) aplicam a programação por restrições em uma versão mais completa do PPADPM, que considera períodos de indisponibilidade dos recursos, restrições geográficas no uso dos recursos, janelas de tempo de execução das atividades e capacidade de carga dos navios.

Embora a modelagem baseada em MILP não seja a técnica mais eficiente para gerar soluções para um problema de classe NP-difícil (BLAZEWICZ, DROR e WEGLARZ, 1991; NEMHAUSER e WOLSEY, 1998), o uso desta traz os seguintes benefícios:

- Obtém informações da qualidade da solução (diferença entre a solução encontrada e o limite inferior do problema) e
- Auxilia na identificação e análise de estruturas intrínsecas do problema, permitindo, por exemplo, a inclusão de restrições de corte.

Aplicando técnicas utilizadas na modelagem MILP, este trabalho visa identificar características específicas do PPADPM que possam ser exploradas para gerar soluções viáveis de melhor qualidade, exigindo um menor tempo de processamento.

¹² A meta-heurística GRASP (*Greedy Randomized Adaptive Search Procedure*) constrói soluções de forma semi-gulosa, utilizando-se de uma lista de soluções parciais. A seleção dos elementos utilizados dessa lista é feita de forma aleatória, e para cada nova solução completa encontrada, é aplicado um processo de busca local.

¹³ A meta-heurística VNS (*Variable Neighborhood Search*) explora dinamicamente vizinhanças crescentes de uma solução, utilizando-se de perturbações da vizinhança atual e de um processo de busca local.

4 MATERIAIS E MÉTODOS

4.1 ABORDAGEM MATEMÁTICA DO PPADPM

Como é possível modelar matematicamente um problema de diversas formas, as diversas modelagens encontradas na literatura foram categorizadas conforme o tipo de variável de decisão utilizada:

1. Variáveis de Janela Sequencial (VJS)

Abordagem proposta inicialmente por WAGNER (1959), define o conceito de janelas de execução em cada recurso, atribuindo variáveis de decisão binárias que representam a alocação de uma atividade em cada janela (figura 4.1).

Nesta modelagem, o início e o término de execução associados a uma janela do recurso dependem tanto da atividade nela alocada quanto da janela imediatamente anterior.

Figura 4.1 – Representação da variável de decisão do modelo VJS

II. Variáveis de Ordem entre Atividades (VOA)

Modelagem inicialmente proposta por MANNE (1960) utiliza variáveis binárias de decisão que definem a ordem de execução entre pares de atividades. A figura 4.2 apresenta um exemplo de como é definido o valor da variável de ordem entre atividades.

Figura 4.2 – Representação da variável de decisão do modelo VOA

III. Variáveis de Tempo Discreto (VTD)

Modelagem inicialmente proposta por BOWMAN (1959), adota um passo de tempo que é aplicado em cada recurso, de forma a criar janelas de tempo com duração fixa. Estas janelas já possuem seu instante de início e de fim definidos, e cada atividade deve ocupar uma quantidade de janelas consecutivas (de um mesmo recurso) equivalente à sua duração (figura 4.3).

Figura 4.3 – Representação da variável de decisão do modelo VTD

Além da classificação por tipo de variável de decisão, é importante considerar que o valor de limite inferior (obtido com relaxações das variáveis de decisão) afeta substancialmente o desempenho computacional do modelo (RAMAN e GROSSMANN, 1994).

Assim, para cada modelo, aplica-se uma ou mais das seguintes abordagens relativa à relaxação das variáveis:

(a). Big-M (BM)

Esta abordagem utiliza uma constante de valor elevado (Big-M) para a modelagem de restrições disjuntivas existentes nos modelos. Com isso, é possível garantir a ativação e a desativação das restrições conforme o valor de uma variável binária de decisão.

A figura a seguir apresenta um exemplo de restrição disjuntiva modelada com a técnica Big-M:

Exemplo:

Dadas duas atividades: **Ativ1** ($DUR_1; fim_1$) e **Ativ2** ($DUR_2; fim_2$), a restrição disjuntiva que relaciona o fato de uma atividade ocorrer antes da outra pode ser modelada com o uso de uma variável M de valor elevado:

$$fim_1 + DUR_2 \leq fim_2 + M * (1 - o_{1,2})$$

$$fim_2 + DUR_1 \leq fim_1 + M * o_{1,2}$$

Onde $o_{1,2}$ é a uma variável de decisão que é igual a 1 se a **Ativ1** é executada antes de **Ativ2**.

Figura 4.4 – Exemplo de uso da relaxação Big-M

Observa-se que nesta modelagem, considerando-se os domínios das variáveis envolvidas e o fato do valor de M ser elevado, uma das restrições sempre estará inativa.

(b). Programação disjuntiva (PD)

Abordagem inicialmente proposta por BALAS (1985), tem por característica decompor as variáveis de cada restrição disjuntiva do problema, diminuindo o espaço de busca de soluções do problema. A figura seguinte ilustra o resultado da aplicação desta técnica no exemplo apresentado pela figura 4.4:

Exemplo:

Novas variáveis:

$$fim_1 = fim_1^a + fim_1^b$$

$$fim_2 = fim_2^a + fim_2^b$$

Restrições:

$$fim_2^a - fim_1^a \geq DUR_2 * o_{1,2}$$

$$fim_1^b - fim_2^b \geq DUR_1 * (1 - o_{1,2})$$

Limites das variáveis:

$$fim_1^a \leq M * o_{1,2}$$

$$fim_2^a \leq M * o_{1,2}$$

$$fim_1^b \leq M * (1 - o_{1,2})$$

$$fim_2^b \leq M * (1 - o_{1,2})$$

Figura 4.5 – Exemplo de uso da programação disjuntiva

Da mesma forma que a técnica Big-M, a programação disjuntiva sempre apresenta uma restrição inativa. Entretanto, esta técnica tem uma relaxação que é, na pior das hipóteses, tão restritiva quanto a da abordagem Big-M, como demonstrado por VECCHIETTI, LEE e GROSSMANN (2003). Como consequência da utilização de mais variáveis e restrições se comparado à abordagem Big-M, o desempenho computacional do modelo pode ser comprometido.

(c). Restrições Lógicas (RL)

Esta abordagem tem por objetivo adicionar restrições que envolvam as variáveis binárias do modelo, diminuindo o espaço de busca de soluções (RAMAN e GROSSMANN, 1991; BIEGLER, GROSSMANN e WESTERBERG, 1997).

Considerando-se, por exemplo, o modelo que utiliza variáveis de ordem entre atividades, dadas as variáveis de decisão $o_{1,2}$ e $o_{2,1}$, que representam a ordem de execução entre a atividade 1 e a atividade 2 de um mesmo poço, uma vez que estas atividades não podem ser executadas simultaneamente, obrigatoriamente tem-se que $o_{1,2} + o_{2,1} = 1$.

4.2 CARACTERÍSTICAS ESPECÍFICAS DO PROBLEMA

Como as atividades de desenvolvimento possuem como característica o tipo de recurso que pode atendê-las, é possível separar as atividades em dois conjuntos independentes, de tal forma que um recurso só pode atender as atividades de apenas um desses conjuntos. Aplicando esta característica nos modelos criados, elimina-se uma parte das restrições e das variáveis de decisão do modelo.

Outra característica, intrínseca a problemas modelados com a abordagem MILP, é a simetria. Para o PPADPM, por exemplo, dada uma determinada solução, é possível permutar todas as atividades alocadas de um recurso para outro do mesmo tipo, e vice-versa, sem modificar o valor obtido da função objetivo (no caso, a produção total de óleo dos poços). Esse tipo de simetria prejudica o desempenho do

algoritmo, uma vez que diversas combinações simétricas são analisadas sem necessidade (MARGOT, 2010; SHERALI e SMITH, 2001).

4.3 MODELAGEM MATEMÁTICA

4.3.1 Considerações iniciais

O Problema de Programação de Atividades de Desenvolvimento de Poços Marítimos (PPADPM) consiste em alocar recursos (sondas e navios) para o atendimento de atividades de desenvolvimento de poços ao longo de um horizonte de tempo.

Uma programação viável define:

- Quando cada atividade será realizada e
- Qual recurso atenderá cada atividade.

4.3.2 Simplificações em relação ao problema real

Os modelos analisados no decorrer deste trabalho possuem algumas simplificações em relação ao problema observado na prática:

- Todas as atividades e todos os recursos estão disponíveis desde o início da programação, isto é, no instante 0 (zero) é possível alocar qualquer recurso para atender qualquer atividade considerada, desde que sejam obedecidas as demais restrições do problema;

- Para os navios não é considerado um limite de capacidade de carga de linhas, o que significa que os navios podem transportar um peso ilimitado de linhas para o atendimento das atividades de interligação;
- O tempo de deslocamento considerado entre dois poços é simétrico e é independente do recurso considerado.

4.3.3 Restrições

As restrições que uma programação válida deve atender são as seguintes:

- (1).Todas as atividades devem ser executadas;
- (2).Uma vez iniciada uma atividade, esta não pode ser interrompida até o seu término;
- (3).Um recurso atende apenas uma atividade por vez;
- (4).Uma atividade é executada por um único recurso;
- (5).Atividades de um mesmo poço só podem ser executadas uma de cada vez;
- (6).O recurso deve ser compatível com as características da atividade;
- (7).Devem ser obedecidas as precedências entre as atividades;
- (8).Deve ser considerado um tempo de deslocamento dos recursos entre duas atividades de poços distintos.

4.3.4 Parâmetros

A tabela 4.1 apresenta os parâmetros utilizados pelos modelos estudados, e o detalhamento de cada parâmetro é apresentado a seguir.

Tabela 4.1 – Parâmetros utilizados pelos modelos estudados

Parâmetro	Descrição
$i \in NA$	Índice que representa cada atividade, com NA representando o conjunto de atividades
$k \in NR$	Índice que representa cada recurso, com NR representando o conjunto de recursos
$POSICAO_k$	Número máximo de janelas de execução de um recurso (modelo VJS-BM)
$j \leq POSICAO_k$	Índice que representa cada janela de execução em um recurso k (modelo VJS-BM)
max_t	Número de janelas de tempo considerado (modelos VTD-BM e VTD-RL)
$t \leq max_t$	Índice que representa cada janela de tempo (modelos VTD-BM e VTD-RL)
$POCO_i$	Poço ao qual a atividade i está associada
$COMPAT_{i,k}$	Compatibilidade entre recurso k e atividade i
$DESLOC_{i,ii}$	Tempo de deslocamento de qualquer recurso entre a atividade i e a atividade ii
DUR_i	Duração de cada atividade i
$PRECED_{i,ii}$	Precedência entre as atividades i e ii
$DATA_{IDEAL}_i$	Melhor data possível para concluir a atividade i
$GANHO_i$	Produção de óleo associada à atividade i
M	Constante de valor elevado utilizado na modelagem Big-M

- (a). *Conjunto de atividades: NA*
 - (b). *Índice da atividade: $i \in NA$*
 - (c). *Conjunto de sondas de perfuração: NS*
 - (d). *Conjunto de navios: NB*
 - (e). *Conjunto de recursos: $NR = NS + NB$*
 - (f). *Índice do recurso: $k \in NR$*
 - (g). *Número máximo de janelas de execução de um recurso k ($POSICAO_k$)*
- Define o número máximo de janelas de execução que um recurso k possui. Este valor equivale ao total de atividades que possuem compatibilidade com o recurso k .
- (h). *Índice da janela de execução em um recurso: $j \leq POSICAO_k$*
 - (i). *Número de janelas de tempo do modelo (max_t)*

Cada janela de tempo representa a menor unidade de tempo utilizada no modelo (no caso dos modelos analisados, cada janela tem a duração de um dia). O número de janelas de tempo max_t é utilizado exclusivamente pelos modelos de tempo discreto. Se o seu valor for muito pequeno, soluções viáveis (inclusive a solução ótima) podem ser cortadas do modelo. Por outro lado, se o seu valor for muito grande, o número de variáveis consideradas pelo modelo também torna-se elevado, comprometendo o desempenho do modelo.

Por isso, o valor ideal de max_t é tal que comporte exatamente o número de janelas utilizadas pela solução ótima do problema. Todavia, como não é conhecida a solução ótima do problema, também não é conhecido de antemão o valor ótimo para o total de janelas.

Assim, para evitar a eliminação de soluções viáveis do problema, o valor de $\max_t$ considerado equivale à soma da duração de todas as atividades da instância de dados, que representa a situação onde todas as atividades estão alocadas em um mesmo recurso.

(j). *Índice da janela de tempo em um recurso: $t \leq \max_t$*

(k). *Poço ao qual a atividade i está associada ($POCO_i$)*

(l). *Compatibilidade entre recurso e atividade ($COMPAT_{i,k}$)*

Se um recurso k pode executar uma atividade i , tem-se $COMPAT_{i,k} = 1$; caso contrário, $COMPAT_{i,k} = 0$.

(m). *Tempo de deslocamento de recurso entre dois poços ($DESLOC_{i,ii}$)*

$DESLOC_{i,ii}$ representa o tempo de deslocamento (em dias) entre dois poços, e depende exclusivamente da atividade de origem i , que está associada ao poço $POCO_i$, e da atividade de destino ii , que está associada ao poço $POCO_{ii}$. Como citado anteriormente, os deslocamentos não dependem do recurso utilizado, e há simetria nos deslocamentos, isto é, $DESLOC_{i,ii} = DESLOC_{ii,i}$.

Caso as atividades pertençam ao mesmo poço, tem-se $DESLOC_{i,ii} = 0$.

(n). *Duração de cada atividade (DUR_i)*

Toda atividade tem uma duração em dias, não nula e de valor inteiro positivo.

(o). *Precedências entre as atividades de um poço ($PRECED_{i,ii}$)*

Este parâmetro relaciona as relações de precedência entre as atividades de um mesmo poço. Se a atividade i deve ser concluída antes do início da atividade ii , tem-se que $PRECED_{i,ii} = 1$; caso contrário, $PRECED_{i,ii} = 0$.

(p). *Melhor data possível para concluir uma atividade ($DATA_IDEAL_i$)*

Com as informações de duração e de relação de precedência entre as atividades, é possível obter a melhor data possível para que uma determinada atividade seja concluída (figura 4.6).

Figura 4.6 - Cálculo de melhor data de conclusão de uma atividade

(q). *Produção de óleo de um poço ($GANHO_i$)*

Como o ganho de produção de um poço só é contabilizado após a execução de todas suas atividades e como todas as atividades do poço possuem relações de precedência, associa-se o ganho de óleo apenas ao término de execução da última atividade da rede de precedências desse poço (figura 4.7). Para as demais atividades, associa-se um ganho nulo.

Figura 4.7 - Cálculo de ganho em um poço

(r). Número grande (M)

Em todos os modelos utilizados, o número M representa o maior valor de término de execução dentre as atividades de uma solução. Seu valor deve ser grande o suficiente para garantir que as restrições disjuntivas sejam desativadas mas, por outro lado, não pode ter um valor excessivamente elevado, pois pode provocar problemas de precisão computacional que podem prejudicar o processo de busca de solução.

Como não é possível determinar o valor ideal de M sem conhecer a solução ótima do problema, faz-se uma estimativa de forma a evitar a eliminação de soluções viáveis do problema.

Assim, o valor de M adotado é a soma da duração de todas as atividades da instância de dados, que equivale à situação onde todas as atividades são alocadas em um único recurso.

4.3.5 Função objetivo

O objetivo dos modelos é maximizar o valor acumulado de produção de todos os poços, garantindo o melhor retorno sobre investimento para a empresa.

Entretanto, com a informação de melhor data possível para execução de cada atividade, ao invés de calcular a produção total de óleo dos poços, pode-se calcular a perda de óleo dos poços, através da seguinte fórmula:

$$PERDA_i = \sum_i [GANHO_i \cdot (fim_i - DATA_{IDEAL_i})] , i \in NA \quad (4.1)$$

Onde fim_i é a variável de decisão que representa o instante de término de execução da atividade i .

4.3.6 Modelagem MILP

Esta seção descreve cada um dos modelos MILP utilizados para representar o PPADPM.

4.3.6.1 Modelo com Variáveis de Janela Sequencial - Big-M (VJS-BM)

A figura 4.8 apresenta um diagrama esquemático da modelagem com variáveis de janela sequencial.

Figura 4.8 – Representação da modelagem com variáveis de janela sequencial

Nesta modelagem, cada recurso k possui um número de janelas de execução $POSICAO_k$, e para cada uma das janelas de execução, é possível associar no máximo uma atividade.

Este modelo inclui restrições que garantem que a duração da janela de execução será a mesma da atividade associada. Além disso, em cada um dos recursos, as atividades são associadas às janelas de execução de forma sequencial, isto é, a janela $j+1$ só terá uma atividade associada se a janela j já tiver alguma outra atividade associada.

Por fim, são consideradas restrições que garantem que, dentre as atividades alocadas em um dado recurso, o término de execução de uma atividade associada à janela j será anterior ao início da atividade associada à janela $j+1$.

Para representar as restrições disjuntivas, este modelo utiliza a relaxação Big-M.

4.3.6.1.1 Variáveis de decisão

- $ta_{i,j,k} \in R^+$: indica o instante de término da atividade i na janela j do recurso k ;
- $tp_{j,k} \in R^+$: indica o instante de término da atividade alocada na janela j do recurso k ;
- $fim_i \in R^+$: indica o instante de término da atividade i ;
- $x_{i,j,k} \in B$: caso seja igual a 1, indica que a atividade i é alocada na janela j do recurso k .

4.3.6.1.2 Função objetivo

O objetivo é minimizar o montante de óleo não produzido, baseando-se na melhor data possível para a realização de cada atividade i :

$$\min Z = \sum_i [GANHO_i \cdot (fim_i - DATA_{IDEAL_i})] \quad (4.2)$$

4.3.6.1.3 Restrições

A tabela a seguir apresenta todas as restrições utilizadas no modelo VJS-BM, e o detalhamento de cada equação é apresentada a seguir.

Tabela 4.2 – Equações do modelo VJS-BM

Equação	Descrição
$fim_i = \sum_{j,k} ta_{i,j,k}$ $\forall i$	Término da atividade i
$tp_{j,k} = \sum_i ta_{i,j,k}$ $j \leq POSICAO_k, \forall k$	Término da janela j no recurso k
$ta_{i,1,k} \geq x_{i,1,k} \cdot DUR_i$ $\forall i, \forall k$	Término da primeira janela no recurso k
$ta_{i,j,k} \leq M \cdot x_{i,j,k}$ $\forall i, j \leq POSICAO_k, \forall k$	Limita o término da atividade i na janela j do recurso k
$\sum_{j,k} x_{i,j,k} = 1$ $\forall i$	Atividade i é alocada a uma única janela e em um único recurso
$fim_{ii} \geq fim_i + DUR_{ii}$ $\forall i, \forall ii, PRECED_{i,ii} = 1$	Precedências devem ser consideradas
$\sum_i x_{i,j,k} \leq 1$ $j \leq POSICAO_k, \forall k$	Uma janela j de um recurso k atende no máximo uma atividade
$tp_{j+1,k} \geq tp_{j,k} + DESLOC_{i,ii} + DUR_{ii}$ $- M \cdot (2 - x_{i,j,k} - x_{ii,j+1,k})$ $\forall k, j \leq POSICAO_k, i \neq ii$	Inclui a informação de deslocamento do recurso entre as atividades
$\sum_j x_{i,j,k} \leq COMPAT_{i,k}$ $\forall i, \forall k$	Compatibilidade entre recurso k e atividade i
$\sum_i x_{i,j-1,k} \geq \sum_i x_{i,j,k}$ $\forall k, j > 1, j \leq POSICAO_k$	Preenchimento ordenado das janelas j de um recurso k
$ta_{i,j,k} \geq DATA_IDEAL_i \cdot x_{i,j,k}$ $\forall i, \forall j, \forall k$	Limitação do valor mínimo de término de uma atividade i

- *Término da atividade i*

Define o valor da variável que armazena o término da atividade i :

$$fim_i = \sum_{j,k} ta_{ij,k} \quad \forall i \quad (4.3)$$

- *Término relacionado a uma janela j de um recurso k :*

Armazena o término da atividade alocada na janela j do recurso k . Esta variável não armazena a informação de qual atividade está alocada na janela j do recurso k .

$$tp_{j,k} = \sum_i ta_{ij,k} \\ j \leq POSICA0_k, \forall k \quad (4.4)$$

- *Término relacionado à primeira janela de um recurso k :*

Se a atividade i está na primeira janela do recurso k ($x_{i,1,k} = 1$), o instante de término da atividade ($ta_{i,1,k}$) deve ser maior ou igual à duração da atividade i .

$$ta_{i,1,k} \geq x_{i,1,k} \cdot DUR_i \\ \forall i, \forall k \quad (4.5)$$

- *Relação entre a variável real de término e a variável binária de alocação:*

$$ta_{ij,k} \leq M \cdot x_{ij,k} \\ \forall i, j \leq POSICA0_k, \forall k \quad (4.6)$$

Se a atividade i não for alocada na janela j do recurso k ($x_{ij,k} = 0$), tem-se que o término associado será nulo.

- *Restrição de execução das atividades:*

$$\sum_{j,k} x_{i,j,k} = 1$$
$$\forall i \quad (4.7)$$

Cada atividade i está associada a uma única janela em apenas um dos recursos. Além disso, esta restrição garante que toda atividade é alocada a uma janela.

- *Limite de alocação em uma janela de execução:*

$$\sum_i x_{i,j,k} \leq 1$$
$$j \leq POSICAOK, \forall k \quad (4.8)$$

Uma janela j do recurso k é alocada a no máximo uma atividade.

- *Relação de precedência entre atividades:*

$$fim_{ii} \geq fim_i + DUR_{ii}$$
$$\forall i, \forall ii, PRECED_{i,ii} = 1 \quad (4.9)$$

Se existe uma relação de precedência entre as atividades ($PRECED_{i,ii} = 1$), a atividade ii é executada após a atividade i .

- *Deslocamentos entre atividades:*

$$tp_{j+1,k} \geq tp_{j,k} + DESLOC_{i,ii} + DUR_{ii} - M \cdot (2 - x_{i,j,k} - x_{ii,j+1,k})$$
$$\forall k, j \leq POSICAOK, i \neq ii \quad (4.10)$$

Caso as atividades **i** e **ii** sejam consecutivas no mesmo recurso **k**, isto é, a atividade **i** está alocada na janela **j**, e a atividade **ii** está alocada na janela **j+1**, tem-se que $x_{i,j,k} + x_{ii,j+1,k} = 2$. Assim, o término na janela **j+1** ($tp_{j+1,k}$) deve considerar o término da janela **j** ($tp_{j,k}$), o deslocamento entre a atividade **i** e a atividade **ii** ($DESLOC_{i,ii}$) e a duração da atividade **ii** (DUR_{ii}).

Se as atividades não são consecutivas em um recurso **k**, significa que $x_{i,j,k} = 0$ ou $x_{ii,j+1,k} = 0$, resultando em $tp_{j+1,k} \geq -M$, o que torna a restrição redundante.

- *Compatibilidade entre recurso **k** e atividade **i**:*

$$\sum_j x_{i,j,k} \leq COMPAT_{i,k} \quad \forall i, \forall k \quad (4.11)$$

Se não houver compatibilidade entre atividade **i** e recurso **k** ($COMPAT_{i,k} = 0$), a atividade **i** não será alocada em nenhuma janela do recurso **k**.

- *Preenchimento ordenado das janelas de execução **j** de um recurso **k**:*

$$\sum_i x_{i,j-1,k} \geq \sum_i x_{i,j,k} \quad \forall k, j > 1, j \leq POSICA0_k \quad (4.12)$$

Esta restrição garante que as janelas de execução em um recurso sejam preenchidas sequencialmente. Se em um recurso **k** a janela **j** atende uma atividade, obrigatoriamente a janela **j-1** desse mesmo recurso deve também atender alguma outra atividade.

- *Limitação do valor mínimo de término de uma atividade i:*

$$ta_{i,j,k} \geq DATA_IDEAL_i \cdot x_{i,j,k} \quad \forall i, \forall j, \forall k \quad (4.13)$$

Se $x_{i,j,k} = 1$, o menor valor possível para o término de execução da atividade i equivale à data ideal da atividade¹⁴. Se $x_{i,j,k} = 0$, considerando-se o resultado da equação 4.6 tem-se que $ta_{i,j,k} = 0$.

- *Limitação das variáveis de decisão do modelo:*

$$\begin{aligned} ta_{i,j,k} &= 0 & \forall i, \forall k, j > POSICAO_k \\ x_{i,j,k} &= 0 & \forall i, \forall k, j > POSICAO_k \\ tp_{j,k} &= 0 & \forall i, \forall k, j > POSICAO_k \end{aligned} \quad (4.14)$$

Em um recurso, todas as variáveis relacionadas a uma janela j de valor maior que $POSICAO_k$ são nulas.

4.3.6.2 Modelo com Variáveis de Ordem entre Atividades – Big-M (VOA-BM)

A figura 4.9 apresenta a representação gráfica deste modelo.

¹⁴ Seção 4.3.4

Figura 4.9 – Representação da modelagem com variáveis de ordem entre atividades

Nesta modelagem, são utilizadas variáveis que definem a ordem de execução entre duas atividades quaisquer do modelo. As restrições que completam o modelo garantem a ordem de execução das atividades nos poços e evitam a execução simultânea de duas atividades em um mesmo recurso.

Para representar as restrições ligadas à execução das atividades nos recursos, utiliza-se a relaxação Big-M.

4.3.6.2.1 Variáveis de decisão

- $o_{i,ii} \in B$: caso seja igual a 1, indica que a atividade i é concluída antes da atividade ii . Esta variável de decisão relaciona, inclusive, atividades de poços distintos (em contrapartida ao parâmetro $PRECED_{i,ii}$, que relaciona apenas as relações de precedência entre atividades no mesmo poço.)

- $fim_i \in R^+$: indica o instante de término de execução da atividade i
- $y_{i,k} \in B$: caso seja igual a 1, indica que a atividade i está alocada no recurso k

4.3.6.2.2 Função objetivo

O objetivo é minimizar o montante de óleo não produzido, baseando-se na melhor data possível para a realização de cada uma das atividades i :

$$\min Z = \sum_i [GANHO_i \cdot (fim_i - DATA_IDEAL_i)] \quad (4.15)$$

4.3.6.2.3 Restrições

A tabela a seguir apresenta todas as restrições utilizadas no modelo VOA-BM, e o detalhamento de cada equação é apresentada a seguir.

Tabela 4.3 – Equações do modelo VOA-BM

Equação	Descrição
$\sum_k y_{i,k} = 1$ $\forall i$	Cada atividade i é executada por um único recurso k
$y_{i,k} \leq COMPAT_{i,k}$ $\forall i, \forall k$	Uma atividade i é executada apenas por um recurso compatível
$fim_{ii} \geq fim_i + DUR_{ii}$ $\forall i, \forall ii, PRECED_{i,ii} = 1$	Precedência entre atividade i e ii deve ser considerada
$fim_i + DESLOC_{i,ii} + DUR_{ii}$ $\leq fim_{ii} + M \cdot (2 - y_{i,k} - y_{ii,k})$ $+ M \cdot (1 - o_{i,ii})$ $fim_{ii} + DESLOC_{i,ii} + DUR_i$ $\leq fim_i + M \cdot (2 - y_{i,k} - y_{ii,k})$ $+ M \cdot o_{i,ii}$ $\forall i, \forall ii, \forall k, i < ii$	Garante a sequência de atividades executadas por um recurso k

- A atividade é associada a um único recurso:

$$\sum_k y_{i,k} = 1$$

$$\forall i$$
(4.16)

A atividade **i** deve ser alocada a um único recurso **k**.

- Compatibilidade entre recurso e atividade:

$$y_{i,k} \leq COMPAT_{i,k} \quad \forall i, \forall k$$
(4.17)

Se não houver compatibilidade entre atividade **i** e recurso **k** ($COMPAT_{i,k} = 0$), a atividade **i** não pode ser alocada no recurso **k**.

- *Precedência entre atividades:*

$$\begin{aligned} fim_{ii} &\geq fim_i + DUR_{ii} \\ \forall i, \forall ii, PRECED_{i,ii} &= 1 \end{aligned} \quad (4.18)$$

Caso haja relação de precedência entre as atividades **i** e **ii**, a atividade **ii** deve ocorrer após o término da atividade **i**.

- *Sequência de atividades em um recurso:*

$$\begin{aligned} fim_i + DESLOC_{i,ii} + DUR_{ii} &\leq fim_{ii} + M \cdot (2 - y_{i,k} - y_{ii,k}) + M \cdot (1 - o_{i,ii}) \\ fim_{ii} + DESLOC_{i,ii} + DUR_i &\leq fim_i + M \cdot (2 - y_{i,k} - y_{ii,k}) + M \cdot o_{i,ii} \\ \forall i, \forall ii, \forall k, i &< ii \end{aligned} \quad (4.19)$$

Caso as atividades **i** e **ii** não estejam alocadas no mesmo recurso **k** ($y_{i,k} = 0$ ou $y_{ii,k} = 0$), as restrições tornam-se inativas.

Para o caso onde as atividades são alocadas no mesmo recurso **k** ($y_{i,k} = y_{ii,k} = 1$):

- Se a atividade **i** é executada antes da atividade **ii** ($o_{i,ii} = 1$), o término da atividade **ii** (fim_{ii}) deve considerar o término da atividade **i** (fim_i), o deslocamento entre a atividade **i** e a atividade **ii** ($DESLOC_{i,ii}$) e a duração da atividade **ii** (DUR_{ii});

- Se a atividade **i** é executada após a atividade **ii** ($o_{i,ii} = 0$), o término da atividade **i** (fim_i) deve considerar o término da atividade **ii** (fim_{ii}), o deslocamento entre a atividade **i** e a atividade **ii** ($DESLOC_{i,ii}$) e a duração da atividade **i** (DUR_i).

- *Limitação das variáveis de decisão do modelo:*

$$o_{i,ii} = 0 \quad i \geq ii \quad (4.20)$$

Devido à simetria das variáveis de ordem ($o_{i,ii} + o_{ii,i} = 1$) e à forma como são criadas as restrições disjuntivas para cada valor de $o_{i,ii}$, só é necessário considerar as variáveis de ordem que atendam a condição $i < ii$.

$$o_{i,ii} = 1 \quad PRECED_{i,ii} = 1 \quad (4.21)$$

Se existe precedência entre duas atividades, a variável que define a ordem entre essas atividades já está automaticamente definida.

4.3.6.3 *Modelo com Variáveis de Ordem entre Atividades – Restrições Lógicas (VOA-RL)*

O modelo VOA-RL utiliza as mesmas variáveis e função objetivo do modelo VOA-BM. A diferença entre estes modelos está no acréscimo de restrições lógicas no modelo VOA-RL.

As restrições utilizadas neste modelo, baseadas no artigo de JAIN e GROSSMANN (2001), modificam o comportamento da variável binária de ordem $o_{i,ii}$, que terá valor não-nulo somente nas situações onde as atividades i e ii são executadas pelo mesmo recurso. Conforme demonstrado por CASTRO e GROSSMANN (2012), estas restrições podem ser obtidas através do uso da técnica de programação disjuntiva¹⁵

4.3.6.3.1 Restrições

A tabela a seguir apresenta todas as restrições utilizadas no modelo VOA-RL, e o detalhamento de cada equação é apresentada a seguir.

¹⁵ Detalhes na seção 4.1 .

Tabela 4.4 – Equações do modelo VOA-RL

Equação	Descrição
$\sum_k y_{i,k} = 1$ $\forall i$	Cada atividade i é executada por um único recurso k
$y_{i,k} \leq COMPAT_{i,k}$ $\forall i, \forall k$	Uma atividade i é executada por um recurso compatível
$fim_{ii} \geq fim_i + DUR_{ii}$ $\forall i, \forall ii, PRECED_{i,ii} = 1$	Precedências devem ser consideradas
$fim_i + DESLOC_{i,ii} + DUR_{ii} \leq fim_{ii} + M \cdot (1 - o_{i,ii})$ $i \neq ii$	Sequência de atividades executadas por um recurso
$o_{i,ii} + o_{ii,i} \geq y_{i,k} + y_{ii,k} - 1$ $i \neq ii$	$o_{i,ii}$ está ativo se as atividades i e ii são alocadas para o mesmo recurso k
$o_{i,ii} + o_{ii,i} \leq 1$ $i \neq ii$	Restrição nas variáveis de ordem $o_{i,ii}$
$o_{i,ii} + o_{ii,i} + y_{i,k} + y_{ii,kk} \leq 2$ $i \neq ii, k \neq kk$	$o_{i,ii}$ será zerado se as atividades i e ii são alocadas em diferentes recursos

- A atividade é associada a um único recurso:

$$\sum_k y_{i,k} = 1 \quad \forall i \quad (4.22)$$

A atividade **i** deve ser alocada a um único recurso **k**.

- Compatibilidade entre recurso e atividade:

$$y_{i,k} \leq COMPAT_{i,k} \quad \forall i, \forall k \quad (4.23)$$

Se não houver compatibilidade entre atividade **i** e recurso **k** ($COMPAT_{i,k} = 0$), a atividade **i** não pode ser alocada no recurso **k**.

- *Precedência entre atividades:*

$$\begin{aligned} fim_{ii} &\geq fim_i + DUR_{ii} \\ \forall i, \forall ii, PRECED_{i,ii} &= 1 \end{aligned} \quad (4.24)$$

Casos haja relação de precedência entre as atividades **i** e **ii**, a atividade **ii** deve ocorrer após o término da atividade **i**.

- *Sequência de atividades em um recurso:*

$$\begin{aligned} fim_i + DESLOC_{i,ii} + DUR_{ii} &\leq fim_{ii} + M \cdot (1 - o_{i,ii}) \\ \forall i, \forall ii, i &\neq ii \end{aligned} \quad (4.25)$$

Se a atividade **i** é executada antes da atividade **ii** ($o_{i,ii} = 1$), o fim da atividade **ii** (fim_{ii}) deve considerar o fim da atividade **i** (fim_i), o deslocamento entre a atividade **i** e a atividade **ii** ($DESLOC_{i,ii}$) e a duração da atividade **ii** (DUR_{ii}).

- *Valor da variável de ordem para atividades executadas no mesmo recurso:*

$$\begin{aligned} o_{i,ii} + o_{ii,i} &\geq y_{i,k} + y_{ii,k} - 1 \\ \forall i, \forall ii, i &\neq ii \end{aligned} \quad (4.26)$$

$$\begin{aligned} o_{i,ii} + o_{ii,i} &\leq 1 \\ \forall i, \forall ii, i &\neq ii \end{aligned} \quad (4.27)$$

Se duas atividades **i** e **ii** são alocadas a um mesmo recurso ($y_{i,k} + y_{ii,k} = 2$), estas equações garantem que uma das variáveis de ordem entre atividades será ativada (isto é, $o_{i,ii} + o_{ii,i} = 1$).

- *Valor da variável de ordem para atividades executadas em recursos diferentes:*

$$\begin{aligned} o_{i,ii} + o_{ii,i} + y_{i,k} + y_{ii,kk} &\leq 2 \\ \forall i, \forall ii, i \neq ii, \forall k, \forall kk, k \neq kk \end{aligned} \quad (4.28)$$

Se duas atividades *i* e *ii* são alocadas em recursos distintos, tem-se que $y_{i,k} + y_{ii,kk} = 2$, forçando a desativação das variáveis de ordem que relacionam estas atividades ($o_{i,ii} + o_{ii,i} = 0$).

- *Limitação das variáveis de decisão do modelo:*

$$o_{i,ii} = 0 \quad i = ii \quad (4.29)$$

$$o_{i,ii} = 1 \quad PRECED_{i,ii} = 1 \quad (4.30)$$

Se existe precedência entre duas atividades, a variável que define a ordem entre essas atividades já está automaticamente definida.

4.3.6.4 Modelo com variáveis de tempo discreto – Big-M (VTD-BM)

A figura 4.10 apresenta um diagrama esquemático da modelagem com variáveis que representam intervalos de tempo em um recurso.

Figura 4.10 – Representação da modelagem com variáveis de tempo discreto

Nesta modelagem, cada recurso k possui um número de janelas de tempo $\max_t$. Se uma atividade é alocada, esta utiliza um número de janelas de tempo do recurso proporcional à sua duração. Além disso, as janelas de tempo alocadas à atividade devem ser consecutivas.

Para representar as restrições disjuntivas, este modelo utiliza-se da relaxação Big-M.

4.3.6.4.1 Variáveis de decisão

- $o_{i,ii} \in B$: caso seja igual a 1, indica que a atividade i é concluída antes da atividade ii . Neste modelo, esta variável de decisão relaciona, inclusive, atividades de poços distintos (em contrapartida ao parâmetro $PRECED_{i,ii}$, que relaciona apenas as relações de precedência entre atividades no mesmo poço.)

- $fim_i \in R^+$: indica o instante de término de execução da atividade i
- $y_{i,k} \in B$: caso seja igual a 1, indica que a atividade i está alocada no recurso k
- $z_{i,t} \in B$: caso seja igual a 1, indica que a atividade i termina sua execução no tempo t .

4.3.6.4.2 Função objetivo

O objetivo é minimizar o valor não produzido, baseando-se na melhor data possível para a realização da atividade, considerando suas precedências.

$$\min Z = \sum_i [GANHO_i \cdot (fim_i - DATA_{IDEAL_i})] \quad (4.31)$$

4.3.6.4.3 Restrições

A tabela seguinte apresenta todas as restrições utilizadas no modelo VTD-BM, e o detalhamento de cada equação é apresentado a seguir.

Tabela 4.5 – Equações do modelo VTD-BM

Equação	Descrição
$\sum_k y_{i,k} = 1$ $\forall i$	Cada atividade i é executada por um único recurso
$y_{i,k} \leq \text{COMPAT}_{i,k}$ $\forall i, \forall k$	Uma atividade i é executada por um recurso compatível
$fim_{ii} \geq fim_i + DUR_{ii}$ $\forall i, \forall ii, PRECED_{i,ii} = 1$	Precedências devem ser consideradas
$fim_i + DESLOC_{i,ii} + DUR_{ii}$ $\leq fim_{ii} + M \cdot (2 - y_{i,k} - y_{ii,k})$ $+ M \cdot (1 - o_{i,ii})$ $fim_{ii} + DESLOC_{i,ii} + DUR_i$ $\leq fim_i + M \cdot (2 - y_{i,k} - y_{ii,k})$ $+ M \cdot o_{i,ii}$ $\forall i, \forall ii, \forall k, i < ii$	Garante a sequência de atividades executadas por um recurso k
$\sum_t z_{i,t} = 1$ $\forall i$	Cada atividade tem seu término em uma única janela de tempo t
$fim_i = \sum_{t=DATA_IDEAL_i}^{max_t} t \cdot z_{i,t}$ $\forall i$	Término de uma atividade i é definida pela janela de tempo ativada

- *Término de execução da atividade:*

$$\sum_t z_{i,t} = 1$$
$$fim_i = \sum_{t=DATA_IDEAL_i}^{max_t} t \cdot z_{i,t}$$
$$\forall i \quad (4.32)$$

Estas restrições garantem que a variável ***fim_i*** armazene o valor de um único instante ***t*** para uma atividade ***i***. Este valor deve ser igual ou maior a ***DATA_IDEAL_i***.

- *A atividade é associada a um único recurso:*

$$\sum_k y_{i,k} = 1$$
$$\forall i \quad (4.33)$$

A atividade ***i*** deve ser alocada a um único recurso ***k***.

- *Compatibilidade entre recurso e atividade:*

$$y_{i,k} \leq COMPAT_{i,k}$$
$$\forall i, \forall k \quad (4.34)$$

Se não houver compatibilidade entre atividade ***i*** e recurso ***k*** (***COMPAT_{i,k}*** = 0), a atividade ***i*** não pode ser alocada no recurso ***k***.

- *Precedência entre atividades:*

$$\begin{aligned} fim_{ii} &\geq fim_i + DUR_i \\ \forall i, \forall ii, PRECED_{i,ii} &= 1 \end{aligned} \quad (4.35)$$

Caso haja relação de precedência entre as atividades **i** e **ii**, a atividade **ii** deve ser executada após o término da atividade **i**.

- *Sequência de atividades em um recurso:*

$$\begin{aligned} fim_i + DESLOC_{i,ii} + DUR_{ii} &\leq fim_{ii} + M \cdot (2 - y_{i,k} - y_{ii,k}) + M \cdot (1 - o_{i,ii}) \\ fim_{ii} + DESLOC_{i,ii} + DUR_i &\leq fim_i + M \cdot (2 - y_{i,k} - y_{ii,k}) + M \cdot o_{i,ii} \\ \forall i, \forall ii, \forall k, i &< ii \end{aligned} \quad (4.36)$$

Caso as atividades **i** e **ii** não estejam alocadas no mesmo recurso **k** ($y_{i,k} = 0$ ou $y_{ii,k} = 0$), as restrições tornam-se inativas.

Para o caso onde as atividades são alocadas no mesmo recurso **k** ($y_{i,k} = y_{ii,k} = 1$):

- Se a atividade **i** é executada antes da atividade **ii** ($o_{i,ii} = 1$), o término da atividade **ii** (fim_{ii}) deve considerar o término da atividade **i** (fim_i), o deslocamento entre a atividade **i** e a atividade **ii** ($DESLOC_{i,ii}$) e a duração da atividade **ii** (DUR_{ii}).

- Se a atividade **i** é executada após a atividade **ii** ($o_{i,ii} = 0$), o término da atividade **i** (fim_i) deve considerar o término da atividade **ii** (fim_{ii}), o deslocamento entre a atividade **i** e a atividade **ii** ($DESLOC_{i,ii}$) e a duração da atividade **i** (DUR_i).

- *Limitação das variáveis de decisão do modelo:*

$$o_{i,ii} = 0 \quad i \geq ii \quad (4.37)$$

Devido à simetria das variáveis de ordem ($o_{i,ii} + o_{ii,i} = 1$) e à forma como são criadas as restrições disjuntivas para cada valor de $o_{i,ii}$, só é necessário considerar as variáveis de ordem que atendam a condição $i < ii$.

$$o_{i,ii} = 1 \quad PRECED_{i,ii} = 1 \quad (4.38)$$

Se existe precedência entre duas atividades, a variável que define a ordem entre essas atividades já está automaticamente definida.

$$z_{i,t} = 0 \quad \forall i, t < DATA_{IDEAL_i}, t > max_t \quad (4.39)$$

Esta equação garante que a atividade, na melhor das hipóteses, finaliza-se em $DATA_{IDEAL_i}$, e não ultrapassa o valor máximo de tempo definido no modelo (max_t).

4.3.6.5 Modelo com variáveis de tempo discreto – Restrições Lógicas (VTD-RL)

O modelo VTD-RL utiliza as mesmas variáveis e função objetivo do modelo VTD-BM. A diferença entre estes modelos é a substituição de restrições da modelagem Big-M por restrições lógicas no modelo VTD-RL. Estas restrições modificam o comportamento da variável binária de ordem $o_{i,ii}$, que terá valor não-nulo se, e somente se, as atividades i e ii são executadas pelo mesmo recurso.

4.3.6.5.1 Restrições

A tabela a seguir apresenta todas as restrições utilizadas no modelo VTD-RL, e o detalhamento de cada equação é apresentada a seguir.

Tabela 4.6 – Equações do modelo VTD-RL

Equação	Descrição
$\sum_k y_{i,k} = 1$ $\forall i$	Cada atividade i é executada por um único recurso
$y_{i,k} \leq COMPAT_{i,k}$ $\forall i, \forall k$	Uma atividade i é executada por um recurso compatível
$fim_{ii} \geq fim_i + DUR_{ii}$ $\forall i, \forall ii, PRECED_{i,ii} = 1$	Precedências devem ser consideradas
$fim_i + DESLOC_{i,ii} + DUR_{ii}$ $\leq fim_{ii} + M \cdot (1 - o_{i,ii})$ $i \neq ii$	Sequência de atividades executadas por um recurso
$\sum_t z_{i,t} = 1$ $\forall i$	Cada atividade tem seu término em uma única janela de tempo t
$fim_i = \sum_{t=DATA_IDEAL_i}^{\max_t} t \cdot z_{i,t}$ $\forall i$	Término de uma atividade i
$o_{i,ii} + o_{ii,i} \geq y_{i,k} + y_{ii,k} - 1$ $i \neq ii$	$o_{i,ii}$ está ativo se as atividades i e ii são alocadas para o mesmo recurso k
$o_{i,ii} + o_{ii,i} \leq 1$ $i \neq ii$	Restrição nas variáveis de ordem $o_{i,ii}$
$o_{i,ii} + o_{ii,i} + y_{i,k} + y_{ii,kk} \leq 2$ $i \neq ii, k \neq kk$	$o_{i,ii}$ será zerado se as atividades i e ii são alocadas em diferentes recursos

- *Início de alocação da atividade:*

$$\sum_t z_{i,t} = 1$$
$$fim_i = \sum_{t=DATA_IDEAL_i}^{max_t} t \cdot z_{i,t}$$
$$\forall i \quad (4.40)$$

Estas restrições garantem que a variável ***fim_i*** armazene o valor de um único instante ***t*** para uma atividade ***i***. Este valor deve ser igual ou maior que a ***DATA_IDEAL_i***.

- *A atividade é associada a um único recurso:*

$$\sum_k y_{i,k} = 1 \quad \forall i \quad (4.41)$$

A atividade ***i*** deve ser alocada a um único recurso ***k***.

- *Compatibilidade entre recurso e atividade:*

$$y_{i,k} \leq COMPAT_{i,k}$$
$$\forall i, \forall k \quad (4.42)$$

Se não houver compatibilidade entre atividade ***i*** e recurso ***k*** (***COMPAT_{i,k}*** = 0), a atividade ***i*** não pode ser alocada no recurso ***k***.

- *Precedência entre atividades:*

$$fim_{ii} \geq fim_i + DUR_i$$
$$\forall i, \forall ii, PRECED_{i,ii} = 1 \quad (4.43)$$

Caso haja relação de precedência entre as atividades **i** e **ii**, a atividade **ii** deve ocorrer após o término da atividade **i**.

- Sequência de atividades em um recurso:

$$\begin{aligned} fim_i + DESLOC_{i,ii} + DUR_{ii} &\leq fim_{ii} + M \cdot (1 - o_{i,ii}) \\ \forall i, \forall ii, i &\neq ii \end{aligned} \quad (4.44)$$

Se a atividade **i** é executada antes da atividade **ii** ($o_{i,ii} = 1$), o término da atividade **ii** (fim_{ii}) deve considerar o término da atividade **i** (fim_i), o deslocamento entre a atividade **i** e a atividade **ii** ($DESLOC_{i,ii}$) e a duração da atividade **ii** (DUR_{ii}).

Se a atividade **i** é executada após a atividade **ii** ($o_{i,ii} = 0$), o término da atividade **i** (fim_i) deve considerar o término da atividade **ii** (fim_{ii}), o deslocamento entre a atividade **i** e a atividade **ii** ($DESLOC_{i,ii}$) e a duração da atividade **i** (DUR_i).

- Valor da variável de ordem para atividades executadas no mesmo recurso:

$$\begin{aligned} o_{i,ii} + o_{ii,i} &\geq y_{i,k} + y_{ii,k} - 1 \\ \forall i, \forall ii, i &\neq ii \end{aligned} \quad (4.45)$$

$$\begin{aligned} o_{i,ii} + o_{ii,i} &\leq 1 \\ \forall i, \forall ii, i &\neq ii \end{aligned} \quad (4.46)$$

Se duas atividades **i** e **ii** são alocadas a um mesmo recurso ($y_{i,k} + y_{ii,k} = 2$), estas equações garantem que uma das variáveis de ordem entre atividades será ativada (isto é, $o_{i,ii} + o_{ii,i} = 1$).

- Valor da variável de ordem para atividades executadas em recursos diferentes:

$$\begin{aligned} o_{i,ii} + o_{ii,i} + y_{i,k} + y_{ii,kk} &\leq 2 \\ \forall i, \forall ii, i \neq ii, \forall k, \forall kk, k \neq kk \end{aligned} \quad (4.47)$$

Se duas atividades i e ii são alocadas em recursos distintos, tem-se que $y_{i,k} + y_{ii,kk} = 2$, forçando a desativação das variáveis de ordem que relacionam estas atividades ($o_{i,ii} + o_{ii,i} = 0$).

- Limitação das variáveis de decisão do modelo:

$$o_{i,ii} = 0 \quad i = ii \quad (4.48)$$

$$o_{i,ii} = 1 \quad PRECED_{i,ii} = 1 \quad (4.49)$$

Se existe precedência entre duas atividades, a variável que define a ordem entre essas atividades já está automaticamente definida.

$$\begin{aligned} z_{i,t} &= 0 \\ \forall i, t < DATA_{IDEAL_i}, t > max_t \end{aligned} \quad (4.50)$$

Esta equação garante que a atividade, na melhor das hipóteses, finaliza-se em $DATA_{IDEAL_i}$, e não ultrapassa o valor máximo de tempo definido no modelo (max_t).

4.3.7 Modelagem MPR

O modelo baseado em Programação por Restrições para o problema PPADPM está descrito a seguir.

4.3.7.1 Variáveis de decisão

Na PR, as atividades e os recursos são representados por intervalos de tempo, e no processo de busca de uma solução, a PR busca encaixar cada intervalo que representa uma atividade dentro do intervalo que representa um recurso. O início de um intervalo da PR é obtido através da função **startOf**, enquanto o término do intervalo é obtido via função **endOf**.

As variáveis de decisão utilizadas no modelo MPR são as seguintes:

- **intAtiv_i** : intervalo de tempo que representa uma atividade **i**. A duração do intervalo é dada por **DUR_i**, e o início e o término do intervalo são obtidos com a execução do modelo;
- **intCompat_{i,k}** : intervalo de tempo que representa a compatibilidade entre a atividade **i** e o recurso **k**, é criado apenas se **COMPAT_{i,k} = 1**. A duração do intervalo é igual a **DUR_i**;
- **seqRec_k** : representa a sequência dos intervalos de tempo executados pelo recurso **k**, e é composta por todos os intervalos de compatibilidade **intCompat_{i,k}** que estão associados a intervalos **intAtiv_i**.

4.3.7.2 Função objetivo

A função objetivo do modelo MPR considera o instante de conclusão de cada intervalo que representa uma atividade:

$$\min Z = \sum_i [GANHO_i \cdot (endOf(intAtiv_i) - DATA_{IDEAL_i})] \quad (4.51)$$

4.3.7.3 Restrições

A tabela seguinte apresenta todas as restrições utilizadas no modelo MPR, e o detalhamento de cada equação é apresentada a seguir.

Tabela 4.7 – Equações do modelo MPR

Equação	Descrição
$endBeforeStart[intAtiv_i, intAtiv_{ii}]$ $i, ii \in NA$ $POCO_i = POCO_{ii}$ $PRECED_{i,ii} = 1$	Estrutura que garante que as atividades i e ii de um mesmo poço não se sobreponham
$alternative[intAtiv_i, \langle intCompat_{i,k} \rangle]$ $\forall i, \forall k$	Estrutura que garante que a atividade i é executada por um único recurso compatível
$noOverlap[seqRec(k), \langle DESLOC_{i,ii} \rangle]$ $\forall k, \forall i, \forall ii$	Estrutura que garante que atividades que são executadas por um recurso k não se sobreponham, considerando os deslocamentos do recurso entre as atividades i e ii

- Execução das atividades em um poço:

$$\begin{aligned}
 &endBeforeStart[intAtiv_i, intAtiv_{ii}] \\
 & i, ii \in NA \\
 & POCO_i = POCO_{ii} \\
 & PRECED_{i,ii} = 1
 \end{aligned} \tag{4.52}$$

Se duas atividades **i** e **ii** são do mesmo poço, e a atividade **i** deve ser executada antes de **ii**, esta restrição garante que a atividade **ii** só será executada após a conclusão da atividade **i**.

- *Compatibilidade entre recurso e atividade:*

$$\begin{aligned} & \mathbf{alternative}[intAtiv_i, \langle intCompat_{i,k} \rangle] \\ & \forall i, \forall k \end{aligned} \quad (4.53)$$

A restrição global da PR, **alternative**, garante que o intervalo **intAtiv_i**, será associado a apenas um dos intervalos do conjunto que representa as compatibilidades da atividade **i**, **⟨intCompat_{i,k}⟩**. Dessa forma, garante-se que a atividade **i** será atendida por apenas um dos recursos compatíveis.

- *Execução das atividades por um recurso:*

$$\begin{aligned} & \mathbf{noOverlap}[seqRec(k), \langle DESLOC_{i,ii} \rangle] \\ & \forall k, \forall i, \forall ii \end{aligned} \quad (4.54)$$

Através da restrição global da PR, **noOverlap**, garante-se que não ocorra sobreposição das atividades executadas por um recurso **k**. Esta restrição global também considera todos os deslocamentos entre as atividades (representados pelo conjunto de deslocamentos, **⟨DESLOC_{i,ii}⟩**).

4.4 EXECUÇÃO DOS TESTES

4.4.1 Instâncias de dados utilizadas

Através de dados obtidos de situações reais de alocação dos recursos (conjunto com mais de 500 poços e 2000 atividades), foram extraídas instâncias de teste para os modelos descritos. Cada instância de teste gerada mantém as seguintes características observadas nos dados reais:

- Distribuição <Duração atividades PLSV>/<Duração atividades sondas>;
- Distribuição <Ganho de produção do poço>/<Duração das atividades do poço>;
- Relação <número de sondas>/<número de navios>.

A tabela 4.8 apresenta as características de todas as instâncias de dados utilizadas nos testes dos modelos.

Tabela 4.8 – Instâncias de dados utilizadas para os testes dos modelos

Número da instância	Número de poços	Número total de atividades	Número de PLSVs	Número de sondas	Total de recursos
1 a 5	20	40	1	2	3
6 a 10	30	60	1	2	3
11 a 15	40	80	1	2	3
16 a 20	50	100	1	2	3
21 a 25	20	40	2	5	7
26 a 30	20	40	3	9	12
31	20	40	4	15	19
32 a 34	30	60	2	5	7
35	30	60	3	9	12
36	30	60	4	15	19
37	40	80	2	5	7
38	40	80	3	9	12
39	40	80	4	15	19
40	50	100	4	15	19
41	4	8	1	2	3
42	5	10	1	2	3
43	6	12	1	2	3
44	7	14	1	2	3
45	8	16	1	2	3
46	9	18	1	2	3
47	10	20	1	2	3
48	11	22	1	2	3
49	12	24	1	2	3
50	13	26	1	2	3
51	14	28	1	2	3
52	15	30	1	2	3
53	16	32	1	2	3
54	17	34	1	2	3
55	18	36	1	2	3
56	19	38	1	2	3

4.4.2 Condições para a realização dos testes

Para obter comparações válidas entre os modelos, foram obedecidas as seguintes condições:

- Todos os testes foram realizados em uma mesma plataforma computacional dedicada exclusivamente para esse fim, garantindo que toda a capacidade de processamento fosse direcionada para a execução dos testes;
- Cada execução foi limitada em 1200 segundos de tempo de CPU¹⁶, considerando-se a necessidade de obtenção de uma programação em um curto prazo de tempo¹⁷.

O computador utilizado possui um processador Intel® Core i3®, com frequência de operação de 1.4GHz, com suporte para execução de até quatro processos simultâneos. A memória total do sistema é de 8GB, e o sistema operacional utilizado foi o Microsoft® Windows® 8 Pro.

O pacote de otimização utilizado foi o IBM® CPLEX® Optimization Studio versão 12.4, de 32 bits.

4.4.3 Processamento paralelo no pacote de otimização CPLEX®

A biblioteca CPLEX® permite o uso de processamento paralelo na execução de modelos MILP. Através de um parâmetro fornecido antes da execução do modelo, define-se qual o modo de processamento paralelo será utilizado pela biblioteca. O número de processadores utilizados também pode ser definido via um parâmetro de inicialização.

¹⁶ Tempo de CPU (*Central Processing Unit*) equivale ao tempo efetivamente utilizado pelo processador para a resolução do modelo. O tempo de relógio engloba o tempo de CPU e o tempo utilizado para a troca de informações entre processos que são executados em paralelo.

¹⁷ Detalhes na seção 1.3

Como cada um dos processadores trabalha de forma independente dos demais, é importante que exista uma sincronização entre eles, de forma a evitar análises de uma mesma porção do espaço de busca do problema. Como a sincronização demanda parte do tempo de processamento, quanto maior esta for, menor será o tempo utilizado pelos processadores na busca de soluções.

Deve-se notar que os ganhos com o processamento paralelo dependem fortemente da estrutura do modelo analisado (IBM, 2012).

Os tipos de processamento paralelo disponibilizados pela biblioteca são os seguintes:

- Processo único: um único processador é utilizado durante a execução do modelo;
- Processamento paralelo determinístico: são utilizados diversos processadores em paralelo, com forte sincronização entre eles. Como consequência da forte sincronização, geralmente é possível conseguir a repetitividade de solução, isto é, dadas as mesmas condições para o modelo, a mesma solução final será obtida;
- Processamento paralelo oportunista: são utilizados diversos processadores em paralelo, com fraca sincronização entre eles. Como uma maior fatia de tempo dos processadores é dedicada exclusivamente à busca de soluções, geralmente esse modo de operação traz resultados melhores que o modo determinístico. Entretanto, não é garantida a repetitividade dos modelos (IBM, 2012).

4.4.4 Parâmetros do pacote de otimização CPLEX® utilizados

Para cada combinação [instância de dados - modelo], foram realizados testes para determinar a influência dos seguintes parâmetros de ajuste do pacote CPLEX:

- Processamento paralelo

- Processo único;
- Processamento paralelo determinístico;
- Processamento paralelo oportunista.

- Seleção de nó no algoritmo B&B¹⁸
 - DFS;
 - BBS;
 - BES.

- Seleção de variável no algoritmo B&B¹⁹
 - Pseudo-custos;
 - Máxima infactibilidade;
 - Seleção agressiva.

4.4.5 Resultados gerados para cada execução de modelo

Ao término de cada execução de uma combinação [instância de dados - modelo], os seguintes resultados foram obtidos:

- a) Melhor valor de perda de produção encontrado (função objetivo);
- b) Melhor valor obtido de limite inferior;
- c) Diferença entre limite inferior e função objetivo;
- d) *Gap* final;
- e) Número de soluções viáveis encontradas;
- f) Número de nós analisados;
- g) Estatísticas do modelo (número de equações, número de variáveis, número de elementos não nulos)
- h) Memória utilizada pelo modelo.

¹⁸ Detalhes na seção 2.2.5

¹⁹ Detalhes na seção 2.2.6

4.4.6 Comparação e análise de resultados

Para a análise dos resultados, as seguintes convenções foram consideradas:

4.4.6.1 Tuplas de resultados

Na análise dos resultados de um determinado parâmetro, considerou-se como uma *tupla de resultados* os experimentos cujos demais parâmetros analisados são idênticos.

Por exemplo, para o caso de análise do parâmetro de processamento paralelo, cada tupla de resultados é composta por três experimentos:

- a) Uma para processo único;
- b) Uma para processamento paralelo determinístico;
- c) Uma para processamento paralelo oportunista.

Todos os demais parâmetros (estratégia de seleção de nós, estratégia de seleção de variáveis, modelo, instância de dados) são idênticos para esses três experimentos.

4.4.6.2 Critério de comparação de resultados

Para definir qual foi o melhor resultado entre experimentos, o critério de comparação adotado foi o seguinte (observando-se que o problema considerado é de minimização):

- Menor gap (diferença percentual entre a melhor solução obtida e o melhor limite inferior obtido);
- Em caso de empate, valor de função objetivo da melhor solução obtida;
- Em caso de novo empate, maior limite inferior obtido.

5 DISCUSSÃO DOS RESULTADOS

5.1 TAMANHO DOS MODELOS MILP

O fato de o PPADPM ser um problema combinatório da classe NP-difícil traz como consequência um aumento expressivo dos modelos, mesmo para pequenas alterações no número de dados de entrada. Assim, a análise do tamanho de um modelo auxilia no dimensionamento do esforço computacional necessário para a resolução de uma instância.

Através de cada modelo gerado, é possível estimar o número de variáveis e o número de restrições utilizado, chegando-se à tabela a seguir.

Tabela 5.1 – Estimativa de variáveis e de restrições dos modelos MILP (considerando-se o pior caso)

Modelo	Variáveis de decisão inteiras	Variáveis de decisão reais	Restrições (termo dominante)
VOA-BM	$N_{atv}(N_{atv} + N_{rec})$	N_{atv}	$2 \cdot N_{atv}^2 \cdot N_{rec}$
VOA-RL	$N_{atv}(N_{atv} + N_{rec})$	N_{atv}	$N_{atv}^2 \cdot N_{rec}^2$
VJS-BM	$N_{atv}^2 \cdot N_{rec}$	$N_{atv}(N_{atv} + N_{rec} + 1)$	$N_{atv}^4 \cdot N_{rec}$
VTD-BM	$N_{atv}(N_{atv} + N_{rec} + N_{tempo})$	N_{atv}	$2 \cdot N_{atv}^2 \cdot N_{rec}$
VTD-RL	$N_{atv}(N_{atv} + N_{rec} + N_{tempo})$	N_{atv}	$N_{atv}^2 \cdot N_{rec}^2$

A tabela 5.1 apresenta uma estimativa de pior caso para cada um dos modelos MILP, considerando que:

- O número de atividades (N_{atv}) é maior que o número de recursos (N_{rec}) da instância;
- Um recurso pode, no máximo, atender a todas as atividades da instância (o que significa que o número de janelas de execução de um recurso não ultrapassa o número de atividades da instância);
- O valor de N_{tempo} (número de janelas de tempo de uma instância) sempre será superior ao número de atividades.

Através da tabela 5.1, observa-se que os modelos VOA utilizam menos variáveis de decisão inteiras que os modelos VTD, e que os modelos que utilizam a relaxação Big-M utilizam menos restrições que os modelos com restrições lógicas.

As figuras 5.1 a 5.5 apresentam o uso de memória obtido na execução dos modelos MILP conforme o tamanho da instância.

Figura 5.1 – Tamanho em memória do modelo VJS-BM, conforme o número de poços e de recursos da instância considerada.

Figura 5.2 – Tamanho em memória do modelo VTD-BM, conforme o número de poços e de recursos da instância considerada.

Figura 5.3 – Tamanho em memória do modelo VTD-RL, conforme o número de poços e de recursos da instância considerada.

Figura 5.4 – Tamanho em memória do modelo VOA-BM, conforme o número de poços e de recursos da instância considerada.

Figura 5.5 – Tamanho em memória do modelo VOA-RL, conforme o número de poços e de recursos da instância considerada.

Como o tamanho do modelo depende diretamente do número de restrições e do número de variáveis consideradas, observa-se através das figuras 5.1 a 5.5 que o

modelo que menos utiliza memória é o VOA-BM (variáveis de ordem entre atividades, com relaxação Big-M).

Em relação aos modelos VTD-BM e VTD-RL, o tamanho do modelo também depende do número de janelas de tempo. Entretanto, analisando-se a tabela 5.1, conclui-se que os modelos VTD sempre são maiores que os modelos VOA.

Constata-se também que, comparada à relaxação Big-M, o uso de restrições lógicas provoca um aumento de uma ordem de grandeza no uso de memória.

Por fim, observa-se que o acréscimo de recursos na instância aumenta de forma mais acentuada o tamanho do modelo, se comparado com o acréscimo de poços na instância.

5.2 PROCESSAMENTO PARALELO

Foram realizados diversos testes para verificar se o uso do processamento paralelo traz benefícios nos modelos MILP gerados para o PPADPM.

A figura 5.6 apresenta o número de nós analisados dentro do tempo limite estabelecido (1200 segundos), para os tipos de processamento paralelo disponibilizados pela biblioteca CPLEX.

Figura 5.6 – Nós analisados conforme tipo de processamento paralelo, para um mesmo tempo de execução

Através da figura 5.6, observa-se que tanto o processamento paralelo oportunista quanto o paralelo determinístico sistematicamente analisam mais nós que o processo único.

Em relação à qualidade das soluções geradas, foram analisados os melhores resultados obtidos em cada tupla, e categorizados conforme o tipo de processamento utilizado (figura 5.7).

Figura 5.7 – Comparação dos resultados obtidos com o processamento paralelo

Além da maior quantidade de nós analisados, através da figura 5.7 observa-se que aproximadamente 90% das instâncias analisadas têm seus melhores resultados (gap, melhor solução, melhor limite inferior) obtidos pelo processamento paralelo.

Assim, para o PPADPM, o uso de processamento nos modelos MILP analisados traz benefícios em termos de qualidade de soluções geradas.

5.3 COMPARAÇÃO ENTRE SOLUÇÕES OBTIDAS

Para verificar a qualidade das soluções obtidas analisou-se o gap final obtido após a execução do modelo (gap nulo indica que a solução obtida pelo modelo é ótima). A figura 5.8 apresenta o gap final obtido por cada modelo conforme o número de poços da instância considerada.

Figura 5.8 – Gap obtido por cada modelo conforme o número de poços da instância (para três recursos)

A figura 5.8 mostra que, em termos de gap final, o modelo VOA-RL apresenta os melhores resultados dentre os modelos MILP analisados pois, sistematicamente, obtém gaps menores em relação aos demais modelos (obtendo a solução ótima em uma instância de 13 poços).

Além disso, embora os modelos que utilizam apenas a relaxação Big-M sejam menores que os modelos que utilizam restrições lógicas, constata-se que estes últimos apresentam um melhor desempenho. Este resultado está relacionado com o corte do espaço de busca de soluções e à estrutura do modelo resultante ao incluir as restrições lógicas.

A figura 5.9 apresenta os seguintes gráficos: (a) melhor solução obtida por instância; (b) melhor limite inferior por instância; (c) porcentagem de instâncias com ao menos uma solução viável obtida por modelo.

Figura 5.9 – Melhor solução, melhor limite inferior e porcentagem de soluções viáveis obtidos

Analisando o gráfico (a) da figura 5.9, observa-se que os modelos baseados em variáveis de ordem de atividades obtiveram a melhor solução para todas as instâncias testadas. Embora os demais modelos sejam corretos, no sentido de representarem o problema analisado, as variáveis de decisão e as restrições desses modelos formam uma estrutura que não é eficiente durante o processo de busca. Soma-se a isto o fato de utilizarem mais restrições e mais variáveis que os modelos baseados em variáveis de ordem entre atividades, o que aumenta o custo computacional do processo de busca, prejudicando ainda mais o desempenho durante a busca de soluções.

Através do gráfico (b), constata-se que praticamente todos os melhores limites inferiores foram obtidos pelo modelo que utiliza restrições lógicas, o que evidencia a fraca relaxação obtida pelos modelos que utilizam a abordagem Big-M.

Por fim, analisando-se o gráfico (c), os modelos baseados em variáveis de ordem de atividades (VOA-BM e VOA-RL) obtiveram soluções viáveis em mais de 90% das instâncias testadas.

Uma vez que os melhores resultados obtidos se concentram basicamente nos modelos VOA-BM e VOA-RL, decidiu-se direcionar as análises subsequentes apenas nesses dois modelos.

5.4 PARÂMETRO DE SELEÇÃO DE NÓS NO B&B

Para verificar a influência das estratégias de seleção de nós no algoritmo B&B²⁰, foram analisados os gaps finais obtidos em cada execução de um modelo. Para isso, inicialmente foram montadas tuplas, cujos membros têm como único parâmetro de distinção a estratégia de seleção de nós. Com os resultados obtidos, foram analisados:

- a. A diferença média entre os gaps obtidos pelas demais estratégias em relação ao melhor gap obtido, dentro de uma tupla;
- b. A maior diferença entre os gaps obtidos pelas demais estratégias em relação ao melhor gap obtido, dentro de uma tupla;
- c. A estratégia de seleção de nós que gerou o melhor gap;
- d. Número médio de nós analisados por cada uma das estratégias de seleção de nós.

Para cada um dos itens acima, foi gerado um gráfico e, exceto para o item a, os dados também foram classificados conforme o número de poços ou o número de recursos da instância.

²⁰ Seção 2.2.5

As figuras 5.10 e 5.11 apresentam os gráficos obtidos para o modelo VOA-RL, em relação ao número de poços da instância e ao número de recursos da instância, respectivamente. E as figuras 5.12 e 5.13 apresentam os gráficos obtidos para o modelo VOA-BM, respectivamente em relação ao número de poços e ao número de recursos das instâncias.

Figura 5.10 – Comparação do parâmetro de seleção de nós no B&B para o modelo VOA-RL, em relação ao número de poços (instâncias com três recursos)

Figura 5.11 - Comparação do parâmetro de seleção de nós no B&B para o modelo VOA-RL, em relação ao número de recursos

Figura 5.12 - Comparação do parâmetro de seleção de nós no B&B para o modelo VOA-BM, em relação ao número de poços (instâncias com três recursos)

Figura 5.13 - Comparação do parâmetro de seleção de nós no B&B para o modelo VOA-BM, em relação ao número de recursos

Observa-se que, dentre as estratégias de seleção de nós do B&B, a estratégia BBS traz melhores resultados em pelo menos 50% das tuplas analisadas. Entretanto, observa-se que a diferença entre os resultados diminui rapidamente tanto para instâncias com mais poços, quanto para instâncias que consideram mais recursos.

Esse comportamento deve-se ao fato do modelo crescer rapidamente com o número de poços e de recursos da instância, o que aumenta o custo computacional para se analisar cada nó do B&B e, consequentemente, diminui o número de nós analisados durante o tempo de busca. E com a diminuição dos nós analisados, a influência da escolha da estratégia de seleção de nós também diminui.

5.5 PARÂMETRO DE SELEÇÃO DE VARIÁVEIS NO B&B

Para analisar a influência das estratégias de seleção de variáveis no B&B²¹, o procedimento de análise foi semelhante ao da seleção de nós no B&B. A única

²¹ Seção 2.2.6

diferença é que as tuplas foram montadas de forma que a única distinção entre seus membros é a seleção de variáveis no B&B.

As figuras 5.14 e 5.15 apresentam os gráficos obtidos para o modelo VOA-RL, em relação ao número de poços da instância e ao número de recursos da instância, respectivamente. E as figuras 5.16 e 5.17 apresentam os gráficos obtidos para o modelo VOA-BM, respectivamente em relação ao número de poços e ao número de recursos das instâncias.

Figura 5.14 - Comparação do parâmetro de seleção de variáveis no B&B para o modelo VOA-RL, em relação ao número de poços (instâncias com três recursos)

Figura 5.15 - Comparação do parâmetro de seleção de variáveis no B&B para o modelo VOA-RL, em relação ao número de recursos

Figura 5.16 - Comparação do parâmetro de seleção de variáveis no B&B para o modelo VOA-BM, em relação ao número de poços (instâncias com três recursos)

Figura 5.17 - Comparação do parâmetro de seleção de variáveis no B&B para o modelo VOA-BM, em relação ao número de recursos

Observa-se que dentre as estratégias de seleção de variáveis do B&B, a seleção da variável com a máxima infactibilidade traz melhores resultados para o modelo VOA-BM. Para o modelo VOA-RL, observa-se que a estratégia agressiva traz melhores resultados na comparação por poços, e a estratégia de custo reduzido traz melhores resultados na comparação por recursos.

Entretanto, observa-se nos gráficos que a diferença entre os resultados diminui sensivelmente tanto com o aumento do número de poços quanto do número de recursos da instância.

O comportamento verificado é consequência do rápido crescimento do modelo com o aumento do número de poços e de recursos das instâncias, o que aumenta o custo computacional para se analisar cada nó do B&B e, consequentemente, diminui o número de nós analisados durante o tempo de busca especificado. Com isso, a influência da estratégia de seleção de variáveis diminui com o aumento de poços e de recursos da instância.

5.6 USO DE SOLUÇÃO INICIAL NO B&B

Como a biblioteca de otimização utilizada (CPLEX) permite que seja utilizada uma solução viável da instância como ponto inicial de busca do algoritmo B&B, testou-se essa possibilidade em todos os modelos MILP propostos, analisando-se o gap final obtido com e sem o uso de uma solução viável inicial. Nos testes realizados, foi utilizada como solução inicial aquela com o melhor valor de função objetivo já obtido para a instância.

As figuras 5.18 e 5.19 apresentam os gráficos de gap final obtido para o modelo VOA-BM, considerando-se o número de poços e o número de recursos da instância, respectivamente. E as figuras 5.20 e 5.21 apresentam os gráficos de gap final para o modelo VOA-RL, respectivamente para o número de poços e para o número de recursos da instância.

Figura 5.18 – Gap obtido com o uso de uma solução inicial no modelo VOA-BM (instâncias com três recursos)

Figura 5.19 – Gap obtido com o uso de uma solução inicial no modelo VOA-BM (instâncias com vinte poços)

Figura 5.20 – Gap obtido com o uso de uma solução inicial no modelo VOA-RL (instâncias com três recursos)

Figura 5.21 – Gap obtido com o uso de uma solução inicial no modelo VOA-RL (instâncias com vinte poços)

Observa-se que, para instâncias menores (tanto em número de poços quanto em número de recursos), o uso de uma solução inicial não traz melhorias nos gaps finais obtidos. Para instâncias com mais poços (40 e 50 poços), tem-se um pequeno ganho no valor de gap, da ordem de 0.5%. E para instâncias com mais recursos, não é observada uma tendência bem definida. Assim, em todos os testes realizados, o uso de uma solução inicial não trouxe ganhos significativos.

Um ponto importante a ressaltar é que não houve melhora na função objetivo nas execuções que fizeram uso de uma solução inicial, isto é, não foi encontrada uma solução melhor que a solução viável no tempo de busca especificado.

Semelhante às análises anteriores, o resultado observado deve-se ao aumento do tamanho do modelo (aumento de poços e de recursos das instâncias), que resulta na diminuição do número de nós analisados no tempo limite especificado e, por consequência, minimiza a influência do uso de uma solução inicial nos modelos.

5.7 COMPARAÇÃO COM O MODELO DE PROGRAMAÇÃO POR RESTRIÇÕES (MPR)

No modelo MPR, que utiliza a programação por restrições (PR), foram executadas as mesmas instâncias utilizadas para os testes dos modelos MILP. Como a PR privilegia a busca de soluções viáveis, e não a otimalidade, este só garante uma solução ótima se todas as combinações possíveis da instância foram analisadas.

Através da figura 5.22, observa-se que o modelo MPR obteve a solução ótima apenas na instância com quatro poços, enquanto todos os modelos MILP testados chegaram à solução ótima na instância de seis poços.

Figura 5.22 – Gap final obtido conforme o número de poços da instância (para três recursos)

Em relação ao tamanho do modelo MPR, o gráfico da figura 5.23 mostra que este é menor que o modelo VOA-BM (menor dos modelos MILP testados). Na maior instância testada (50 poços e 19 recursos), o modelo MPR é aproximadamente 30% menor que o modelo VOA-BM.

Figura 5.23 – Tamanho em memória do modelo MPR, conforme o número de poços e de recursos da instância considerada.

Em relação ao desempenho do modelo MPR, analisou-se o número de soluções viáveis encontradas pelo modelo dentro do tempo limite estabelecido. As figuras 5.24 e 5.25 apresentam os gráficos obtidos, respectivamente para uma instância de 50 poços e uma instância de 19 recursos, e consideram a diferença entre o valor da função objetivo da solução viável encontrada e o da melhor solução obtida para a instância.

Figura 5.24 – Soluções viáveis obtidas para uma instância com 50 poços e 3 recursos

Figura 5.25 – Soluções viáveis obtidas para uma instância com 50 poços e 19 recursos. Os modelos VOA não obtiveram nenhuma solução viável para esta instância.

Através dos gráficos das figuras 5.24 e 5.25, observa-se claramente que o modelo MPR é mais eficiente na obtenção de soluções viáveis. Esse comportamento era previsível, considerando-se que a programação por restrições tem como foco a busca de soluções viáveis, e não a busca pela otimalidade.

Para analisar a qualidade das soluções viáveis encontradas, foram comparadas as melhores soluções obtidas pelos modelos VOA-BM e VOA-RL

(modelos que obtiveram as melhores soluções viáveis, dentre os modelos MILP) e as melhores soluções obtidas pelo modelo MPR. Os resultados obtidos em relação ao número de poços e ao número de recursos da instância são apresentados nas figuras 5.26 e 5.27, respectivamente.

Nesses gráficos, como referência de comparação foi utilizada a melhor solução viável do modelo MPR. Dessa forma, se a diferença entre as soluções for positiva, indica que a solução obtida pelo modelo MPR é melhor que a do outro modelo.

Figura 5.26 – Comparação entre melhores soluções viáveis obtidas pelos modelos, conforme o número de poços da instância (referência: 3 recursos)

Figura 5.27 – Comparação entre melhores soluções viáveis obtidas pelos modelos, conforme o número de recursos da instância (referência: 20 poços)

Observa-se que o modelo MPR, além de obter mais soluções viáveis que os modelos MILP, gera soluções com melhor valor de função objetivo. Além disso, com o aumento de número de poços, observa-se que a diferença entre as soluções geradas pelo MPR e as geradas pelos modelos MILP torna-se cada vez mais significativa.

Através das comparações realizadas, conclui-se que, para o problema PPADPM, o modelo MPR é mais eficiente que os modelos MILP propostos, tanto em termos de tamanho do modelo, quanto em termos de qualidade de soluções geradas. Entretanto, devido às características intrínsecas da programação por restrições, não é possível determinar a qualidade das soluções obtidas em relação à solução ótima.

6 CONCLUSÕES

Dos modelos MILP testados, verificou-se que aqueles baseados em variáveis de ordem de atividades apresentaram os melhores resultados, por utilizarem variáveis de decisão e restrições que formam uma estrutura que favorece o processo de busca, o que evidencia a importância da modelagem do problema na obtenção de soluções para as instâncias.

Além disso, foi demonstrado que o uso de restrições lógicas melhora consideravelmente a performance do modelo, se comparado com a relaxação via abordagem Big-M. Este comportamento se deve ao corte do espaço de busca de soluções e à estrutura do modelo resultante, que melhoram as estimativas de limite inferior no algoritmo B&B para o problema de minimização. Dessa forma, torna-se interessante a busca de novas restrições lógicas que melhorem as estimativas de limite inferior.

Observou-se também que todos os modelos MILP estudados apresentam baixa eficiência quando instâncias maiores do problema são consideradas, mesmo aplicando estratégias de seleção de nós, de seleção de variáveis e o uso de uma solução inicial. Este resultado deve-se ao aumento do número de poços e de recursos da instância, que gera um aumento expressivo do modelo e, portanto, aumenta o custo computacional de análise para cada nó. Com isso, o número de nós analisados diminui, eliminando os benefícios que seriam obtidos com o uso dessas estratégias.

Comparando-se os resultados obtidos com os de um modelo gerado via programação por restrições, comprovou-se que este tem desempenho superior em relação aos modelos MILP, tanto em termos de memória utilizada quanto na qualidade de soluções obtidas. Entretanto, ao contrário dos modelos MILP, o modelo de programação por restrições não fornece informações relativas à qualidade da solução gerada (gap). Com isso, torna-se interessante combinar essas duas abordagens, de forma a aproveitar a eficiência da busca de soluções da PR e as informações de qualidade de solução obtidas pela modelagem MILP.

7 TRABALHOS FUTUROS

Pelo exposto no item anterior, a criação de um modelo híbrido para o PPADPM, combinando a modelagem MILP com a modelagem PR seria uma extensão natural para o estudo apresentado neste trabalho. Tal abordagem, inicialmente proposta por HOOKER (2000), foi explorada por JAIN e GROSSMANN (2001) para um problema de programação semelhante ao PPADPM, exceto por não considerar o tempo de deslocamento entre recursos e a precedência entre as atividades. Considerando instâncias com solução ótima conhecida, com esta abordagem híbrida os autores conseguiram tempos de busca de uma a duas ordens de magnitude menores que o tempo utilizado pelos modelos puramente MILP ou puramente PR. LI e WOMER (2009), aplicando essa abordagem híbrida em um problema de alocação de recursos, sem considerar o tempo de deslocamento dos recursos, também obtiveram resultados melhores que os obtidos com modelos puramente PR ou puramente MILP.

Outra linha de investigação futura seria a análise de novas estratégias que cortem o espaço de busca do PPADPM, como a procura de restrições lógicas como as aplicadas no modelo VOA-RL, ou a inclusão de restrições que cortem simetrias do problema (MARGOT, 2010; SHERALI e SMITH, 2001).

Além disso, como o PPADPM considerado neste trabalho é uma simplificação do problema real, torna-se interessante a inclusão de outras restrições operacionais como, por exemplo, as indisponibilidades de recursos devido a manutenções programadas, o uso de datas limite tanto para o início quanto para o término de execução das atividades, a situação operacional dos recursos no início da programação, a dependência do tempo de deslocamento com o recurso utilizado e a capacidade de carga de linhas dos navios.

8 REFERÊNCIAS

- BALAS, E. **Disjunctive programming and a hierarchy of relaxations for discrete optimization problems**. Siam Journal on Algebraic Discrete Methods v.6(3), 1985. p. 466-486.
- BIEGLER, L. T.; GROSSMANN, I. E.; WESTERBERG, A. W. **Systematic methods of chemical processing design**, NJ, EUA, 1997.
- BLAZEWICZ, J.; DROR, M.; WEGLARZ, J. **Mathematical programming formulations for machine scheduling: a survey**. European Journal of Operational Research, 51(3), abril 1991. p. 283-300. invited review.
- BOWMAN, E. H. **The scheduling sequence problem**. Operations Research, v.7, 1959. p. 621–624.
- BRUCKER, P.; JURISCH, B.; SIEVERS, B. **A branch and bound algorithm for the job-shop scheduling problem**. Discrete Applied Mathematics, v.49, 1994. p. 107-127.
- BRUCKER, P.; SCHLIE, R. **Job-shop scheduling with multi-purpose machines**. Computing, v.45, 1990. p. 369-375.
- CASTRO, P. M.; GROSSMANN, I. E. **Generalized Disjunctive Programming as a Systematic Modeling Framework to Derive Scheduling Formulations**. Industrial & Engineering Chemistry Research, 2012. v. 51, p. 5781-5792.
- DA COSTA, L. R.; FERREIRA FILHO, V. J. M. **Uma heurística de montagem dinâmica para o problema de otimização de itinerários de sondas**. Proceedings of the 37rd Brazilian Symposium on Operations Research. [S.l.]: [s.n.]. 2005.
- DARIO, J. et al. **Scheduling workover rigs for onshore oil production**. Discrete Applied Mathematics, 154, 2006. p. 695–702.
- GAREY, M. R.; JOHNSON, D. S.; SETHI, R. **The complexity of flowshop and jobshop scheduling scheduling**. Mathematics of Operation Research v.1(2), 1976. p. 117–129.
- H.D., S.; J.C., S. **Improving discrete model representations via symmetry considerations**. Management Science Informs v. 47(10), 2001. p. 1396-1407.
- HASLE, G. et al. **Well activity scheduling - an application of constraint reasoning**. Artificial Intelligence in the Petroleum Industry: Symbolic and Computational Applications ii. Technip, 1996. p. 209–228.
- HOOKER, J. N. **Logic-Based Methods for Optimization: Combining Optimization and Constraint Satisfaction**. New York: John Wiley & Sons, Inc., 2000.

IBM. **Introducing IBM ILOG CPLEX Optimization Studio V12.4**, 2012. Disponível em: <pic.dhe.ibm.com/infocenter/cosinfoc/v12r4/index.jsp>. Acesso em: agosto 2013.

INTERNATIONAL ENERGY AGENCY (IEA). **Key world energy statistics**. [S.l.]. 2011.

JAIN, V.; GROSSMANN, I. E. **Algorithms for hybrid milp/cp models for a class of optimization problems**. *Inform Journal of Computing* v.13 n.4, 2001. p. 258-276.

LI, H.; WOMER, K. **Scheduling projects with multi-skilled personnel by a hybrid milp/cp benders decomposition algorithm**. *Journal of Scheduling* v.12(3), 2009. p. 281-298.

MANNE, A. S. **On the job-shop scheduling problem**. *Operations Research* v.8, 1960. p. 219-223.

MARGOT, F. **Symmetry in integer linear programming. 50 years of integer programming** 1958-2008, 2010. p. 647-686. DOI 10.1007/978-3-540-68279-0_17.

MOURA, A. V.; PEREIRA, R. A.; DE SOUZA, C. C. **Scheduling activities at oil wells with resource displacement**. *International Transactions in Operational Research* v.15(6), 2008. p. 659-683.

NASCIMENTO, J. M. **Hybrid computational tools for the optimization of the production of petroleum in deep waters**. Dissertação (Mestrado), Universidade de Campinas, São Paulo, 2002.

NEMHAUSER, G. L.; WOLSEY, L. A. **Integer and combinatorial optimization**. [S.l.]: Wiley: New York, 1998.

PAPADIMITRIOU, C. H.; STEIGLITZ, K. **Combinatorial Optimization - Algorithms and Complexity**. 1. ed. Mineola, New York: Dover Publications, Inc., 1998.

PEREIRA, R. A.; MOURA, A. V.; DE SOUZA, C. C. **Comparative experiments with grasp and constraint programming for the oil well drilling problem**. *Proceedings of the 4th International Workshop on Experimental and Efficient Algorithms*, 2005. p. 328-340.

PETROBRAS S.A. **Fact sheet 2013**, 2013a. Disponível em: <<http://www.investidorpetrobras.com.br/pt/kit-do-acionista/fact-sheet.htm>>. Acesso em: Setembro 2013. Versão eletrônica.

PETROBRAS S.A. **Plano de negócios 2013-2017**, 2013b. Disponível em: <<http://www.investidorpetrobras.com.br/pt/apresentacoes/apresentacao-do-png-2013-2017.htm>>. Acesso em: Setembro 2013. Versão eletrônica.

PINEDO, M. L. **Scheduling: Theory, Algorithms, and Systems**. 4 ed. ed. Nova Iorque, EUA: Springer, 2011.

RAMAN, R.; GROSSMANN, I. E. **Relation between milp modelling and logical inference for chemical process synthesis**. Computers & Chemical Engineering v.15, 1991. p. 73-84.

RAMAN, R.; GROSSMANN, I. E. **Modelling and computational techniques for logic based integer programming**. Computers & Chemical Engineering v. 18(7), 1994. p. 563-578.

SERRA, T. **Programação por restrições e escalonamento baseado em restrições: Um estudo de caso na programação de recursos para o desenvolvimento de poços de petróleo**. Dissertação (Mestrado), Instituto de Matemática e Estatística - Universidade de São Paulo, 2012.

SERRA, T.; NISHIOKA, G.; MARCELLINO, F. J. M. **A constraint-based scheduling of offshore well development activities with inventory management**. Proceedings of the 43rd Brazilian Symposium on Operations Research. [S.l.]: [s.n.]. 2011.

SHERALI, H. D.; SMITH, J. C. **Improving discrete model representations via symmetry considerations**. Management Science Informs v. 47(10), 2001. p. 1396-1407.

U. S. ENERGY INFORMATION ADMINISTRATION. **International energy outlook 2011**. [S.l.]. 2011. (Report number DOE/EIA-0484(2011)).

VECCHIETTI, A.; LEE, S.; GROSSMANN, I. E. **Modeling of discrete/ continuous optimization problems: characterization and formulation of disjunctions and their relaxations**. Computers & Chemical Engineering v.27(3), 2003. p. 433-448.

WAGNER, H. M. **An integer linear programming model for machine scheduling**. Naval Research Logistics Quarterly v.6, 1959. p. 131–140.

WOLSEY, L. A. **Integer Programming**. [S.l.]: John Wiley & Sons, 1998.