Universidade Federal do Rio de Janeiro

Instituto de Matemática
Núcleo de Computação Eletrônica

Modelos de Controle Ótimo do Scheduling de Petróleo e Derivados em Portos

Dissertação de mestrado

Rio de Janeiro, RJ - Brasil

2005

Modelos de Controle Ótimo do Scheduling de Petróleo e Derivados em Portos

Fabio Dias Fagundez

Dissertação submetida ao Corpo Docente do Instituto de Matemática - Núcleo de Computação Eletrônica da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários para a obtenção do grau de Mestre.

Orientador: João Lauro Dorneles Facó - Dr.Ing. - UFRJ.

Rio de Janeiro, RJ - Brasil
2005

Modelos de Controle Ótimo do Scheduling de Petróleo e Derivados em Portos

Fabio Dias Fagundez

Dissertação submetida ao Corpo Docente do Instituto de Matemática - Núcleo de Computação Eletrônica da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários para a obtenção do grau de Mestre.

Aprovada por:

Orientador: João Lauro Dorneles Facó, Dr.Ing.

José Luiz de Medeiros, D.Sc.

Marco Aurélio C. Pacheco, Ph.D.

Fábio Protti, D.Sc.

Rio de Janeiro, RJ - Brasil
2005

Ficha Catalográfica

Fagundez, Fabio Dias.

Modelos de Controle Ótimo do Scheduling
de Petróleo e Derivados em Portos - Rio de Janeiro, 2005.
Orientador: João Lauro Dorneles Facó
Dissertação (Mestrado) - Universidade Federal
do Rio de Janeiro - UFRJ, Instituto de Matemática - IM, Núcleo
de Computação Eletrônica - NCE, 2005.
1. Otimização Não Linear. 2. Programação de Produção
e Planejamento. 3. Controle Ótimo Discreto. I. Facó, João Lauro
Dorneles (Orient.). II. Universidade Federal do Rio de Janeiro.
Instituto de Matemática. III. Universidade Federal do Rio de
Janeiro. Núcleo de Computação Eletrônica. IV. Título.

Dedico ao leitor.

Agradecimentos

Aos meus pais Sandra e João e minha namorada Carla, que sempre me apoiaram em todos os momentos.

Ao Prof. Facó, que confiou em mim durante todo o tempo em que trabalhamos junto, direcionando meus esforços na direção correta e me aconselhando nas maiores dificuldades.

Aos amigos da UFRJ e da Chemtech, em especial Alessandro Alle, Leonardo Simão e Carlos André, que contribuíram com idéias e revisaram meus manuscritos.

Ao Núcleo de Computação Eletrônica e ao Instituto de Matemática da Universidade Federal do Rio de Janeiro pela oportunidade de desenvolver este trabalho.

Resumo

Problemas de *scheduling* de petróleo e derivados em portos e dutos são discutidos. Apresenta-se a divisão do sistema em subsistemas: porto, tancagem intermediária e planta petroquímica ou refinaria. São apresentadas algumas abordagens para o problema e proposto o uso de Controle Ótimo para o mesmo. É proposto um modelo baseado em equações de estado não lineares, sem variáveis binárias e com o uso de vazões como variáveis de controle. Todas as variáveis são restritas por limites inferiores e superiores. Dificuldades na resolução numérica são resolvidas pelo uso de um método eficiente de Programação Não Linear - o Gradiente Reduzido Generalizado (GRG). Casos de teste, de dificuldade crescente, são executados para validar o modelo e o uso do método de resolução.

Palavras-chave *scheduling*, planejamento, otimização, programação não linear, controle ótimo, GRG, porto, dutos.

Abstract

Pipeline and jetty scheduling of crude oil and derivatives problems are discussed. The underlying system is divided in subsystems: the jetties, intermediate tankage and petrochemical plant or refinery. Different approaches for the problem are presented and Optimal Control models are proposed to solve the problem. A model based on nonlinear state equations, with no binary variable, and the use of flow rates as control variables is introduced. All variables are submitted to lower and upper bounds. Difficulties in the numerical solution of these models are overcome by using an efficient Nonlinear Programming method - Generalized Reduced Gradient (GRG). Test cases of increasing difficulty are discussed in order to validate the model, and the solver.

Keywords scheduling, planning, optimization, nonlinear programming, optimal control, GRG, jetty, pipelines

Conteúdo

1 Introdução **1**

1.1 Motivação 1

1.2 Objetivo do Trabalho 2

1.3 Descrição do Trabalho 3

1.4 Organização da Dissertação 4

2 Descrição do Problema **5**

2.1 Planejamento e Scheduling 5

2.2 Problema de Scheduling de Portos 8

2.3 Sistemas Relacionados 8

2.4 Dutos 9

2.5 Tanques 10

2.6 Navios 10

2.7 Tancagem Intermediária 11

2.8 Refinarias e Plantas Petroquímicas 11

3 Modelo **12**

3.1 Controle Ótimo 12

3.2 Intervalos de Tempo 14

3.2.1 Intervalos de Duração Variável 15

3.2.2 Intervalos de Duração Fixa 17

3.2.3 Modelagem Escolhida 18

3.3 Modelo de Controle Ótimo 19

3.3.1 Variáveis de Estado 19
3.3.2 Variáveis de Controle 20
3.3.3 Equações de Estado 23
3.3.4 Restrições 25
3.3.5 Função de Performance 28

4 Método de Resolução **32**
4.1 Método do Gradiente Reduzido Generalizado 35
4.2 Método do GRG para Problemas de Grande Porte 39
4.3 Método do GRG Especializado a Controle Ótimo de Sistemas Dinâmicos 41
4.4 Programa Utilizado 43

5 Heurística de Inicialização **44**

6 Resultados Computacionais **48**
6.1 Caso 1 49
6.2 Caso 2 53
6.3 Caso 3 56
6.4 Caso 4 58
6.5 Caso 5 63

7 Conclusão **71**

Lista de Tabelas

3.1 Exemplos de duração de atividades 18

3.2 Matriz de Incidência para tanque T1 e navios N1 e N2 24

6.1 Casos de Teste de 1 a 3 49

6.2 Casos de Teste 4 e 5 68

6.3 Pesos para os Casos 1, 4 e 5 69

6.4 Pesos para os Casos 2 e 3 69

6.5 Equipamentos do Caso de Teste 4 70

Lista de Figuras

2.1 Classes de problemas 7
2.2 Representação esquemática do Problema 9

3.1 Intervalos de Duração Variável 16
3.2 Intervalos de Duração Fixa 17
3.3 Vazões como variáveis de controle 21
3.4 Esquema Representativo de Tanque e Navios 23

4.1 Restauração da Viabilidade 39
4.2 Matriz Jacobiana das Restrições de um Problema de Controle Ótimo . . 42

6.1 Inventário do ponto inicial inviável no Caso 1 50
6.2 Inventário da solução do Caso 1 51
6.3 Gráfico de Gantt com vazões ($10^3 m^3$/h) da solução do Caso 1 51
6.4 Inventário do ponto inicial obtido pela heurística no Caso 1 52
6.5 Inventário da solução do Caso 1 a partir da heurística 52
6.6 Gráfico de Gantt com vazões ($10^3 m^3$/h) da solução do Caso 1 a partir da heurística 53
6.7 Inventário do ponto inicial inviável do Caso 2 53
6.8 Gráfico de Gantt com vazões ($10^3 m^3$/h) no ponto inicial do Caso 2 . . . 54
6.9 Inventário da solução do Caso 2 54
6.10 Gráfico de Gantt com vazões ($10^3 m^3$/h) da solução do Caso 2 55
6.11 Inventário do ponto inicial inviável do Caso 3 56
6.12 Gráfico de Gantt com vazões ($10^3 m^3$/h) no ponto inicial do Caso 3 . . . 56

6.13 Inventário da solução do Caso 3 57
6.14 Gráfico de Gantt com vazões ($10^3 m^3$/h) da solução do Caso 3 57
6.15 Inventário do ponto inicial inviável do Caso 4(a) 59
6.16 Gráfico de Gantt com vazões ($10^3 m^3$/h) no ponto inicial do Caso 4(a) . . 59
6.17 Inventário da solução do Caso 4(a) 60
6.18 Gráfico de Gantt com vazões ($10^3 m^3$/h) da solução do Caso 4(a) 60
6.19 Inventário do ponto inicial do Caso 4(b) 61
6.20 Gráfico de Gantt com vazões ($10^3 m^3$/h) no ponto inicial do Caso 4(b) . . 61
6.21 Inventário da solução do Caso 4(b) 62
6.22 Gráfico de Gantt com vazões ($10^3 m^3$/h) da solução do Caso 4(b) 62
6.23 Inventário do ponto inicial do Caso 4(c) 63
6.24 Gráfico de Gantt com vazões ($10^3 m^3$/h) no ponto inicial do Caso 4(c) . . 63
6.25 Inventário da solução do Caso 4(c) 64
6.26 Gráfico de Gantt com vazões ($10^3 m^3$/h) da solução do Caso 4(c) 64
6.27 Inventário do ponto inicial do Caso 4(d) 65
6.28 Gráfico de Gantt com vazões ($10^3 m^3$/h) no ponto inicial do Caso 4(d) . . 65
6.29 Inventário da solução do Caso 4(d) 66
6.30 Gráfico de Gantt com vazões ($10^3 m^3$/h) da solução do Caso 4(d) 66
6.31 Inventário do ponto inicial do Caso 5 67
6.32 Gráfico de Gantt com vazões ($10^3 m^3$/h) no ponto inicial do Caso 5 . . . 67
6.33 Inventário da solução do Caso 5 67
6.34 Gráfico de Gantt com vazões ($10^3 m^3$/h) da solução do Caso 5 69

Capítulo 1

Introdução

1.1 Motivação

Planejamento e programação de tarefas (*scheduling*) são duas atividades de grande importância econômica, que demandam bastante esforço computacional [28]. Uma programação eficiente de tarefas permitindo o uso racional de recursos pode poupar custos e aumentar o lucro operacional. Também evita atrasos em entregas e favorece atendimento de demanda sem perda de qualidade, garantindo o cumprimento de contratos, mantendo a fidelidade dos clientes [6]. Pode-se dizer que, nos dias de hoje, empresas que não atentem para a programação de tarefas têm maiores possibilidades de não cumprir seus compromissos. Entre os muitos problemas de planejamento e *scheduling*, está o de programação de recebimento de petróleo e envio de derivados em portos: o assunto dessa dissertação.

Entre as indústrias que fazem uso de dutos para distribuição por terra de matéria-prima e produtos, está a indústria petroquímica, pois petróleo e vários de seus derivados são materiais líquidos, de modo que os custos de transporte por dutos são menores que por caminhões ou trens. Um mesmo duto pode ser utilizado por mais de um material consecutivamente e continuamente para grandes volumes [26, 36], o que dilui os custos de manutenção, aluguel e operação. Em conjunto com os dutos em terra, a indústria petroquímica utiliza intensamente navios-tanque (*tankers*) para o transporte de

petróleo e derivados, seja por causa da localização de poços de petróleo em alto mar (*off-shore*) ou em países distantes, seja pela tradição e menor custo do transporte marítimo. Assim, deve-se estudar o *scheduling* de navios em conjunto com o de dutos na indústria petroquímica.

Desde os anos 50, a comunidade de pesquisa operacional vem desenvolvendo diversas abordagens e métodos para o problema de *scheduling*, utilizando programação linear, inteira, misto-inteira linear e não linear, dinâmica, métodos estocástico-probabilísticos, heurísticas, filas de Poisson etc. [3, 9, 28, 33, 34, 37]. De fato, pode-se observar que na década de 90 e início dos anos 2000, duas vertentes têm se destacado com resultados significativos para *scheduling* aplicados à indústria petroquímica, programação de dutos e portos: algoritmos genéticos [2, 6, 38, 39] e programação linear misto-inteira [16, 17, 26, 27, 29, 30, 36]. Todavia, nesta dissertação é proposto o uso de outra metodologia para problemas de *scheduling*: controle ótimo de sistemas dinâmicos resolvidos por técnicas de programação não linear.

A motivação para o trabalho, então, pode ser detalhada em três aspectos: a complexidade do problema, a possibilidade de inovação na abordagem do problema, a necessidade de soluções aplicáveis em casos concretos.

1.2 Objetivo do Trabalho

Independentemente dos métodos de resolução, as formas mais populares de modelos de *scheduling* recaem no uso de variáveis binárias para representar tomadas de decisões e variáveis lineares contínuas para quantificar as conseqüências das decisões. Apesar de úteis para representação de tomadas de decisões, váriaveis binárias aumentam a complexidade do problema e requerem algoritmos especializados de programação inteira (ex:*branch and bound*, planos de corte, ponto interior), cuja eficiência depende das características do modelo. Já para as variáveis lineares contínuas existe uma maior

facilidade na manipulação das mesmas e métodos eficientes como o Método Simplex [7] e métodos de pontos interiores [8]. Por isso, modelos misto-inteiros de problemas muito grandes são normalmente aproximados por modelos puramente lineares.

Contudo, deve-se destacar que para modelar misturas químicas de petróleos e derivados petroquímicos, o uso mais comum na indústria é o uso de equações empíricas não lineares, quase sempre não convexas [12, 22, 39]. Assim, aproximações lineares [16, 17, 30], apesar de práticas, podem ser menos efetivas e, portanto, para resultados mais próximos da realidade, o uso de programação não linear deve ser considerado.

O objetivo dessa dissertação é apresentar uma modelagem não linear contínua de controle ótimo para problemas de *scheduling*, sem empregar variáveis binárias de decisão, e mostrar que este tipo de modelo permite um número menor de equações e de variáveis e pode ser resolvido rapidamente por métodos eficientes de Programação Não Linear, em particular pelo GRG [1], bem como pelo algoritmo especializado para Controle Ótimo - o GRECO [10].

É também objetivo da dissertação resolver casos semelhantes aos enfrentados pelos programadores de produção no dia a dia , mostrando os benefícios de adotar uma metodologia baseada em técnicas de otimização no lugar de apenas aplicar regras empíricas tradicionais.

1.3 Descrição do Trabalho

O trabalho foi realizado utilizando um processo de desenvolvimento iterativo. Primeiro, foi realizado um levantamento bibliográfico em artigos, dissertações, livros e empresas. Baseado no levantamento bibliográfico, foi proposto uma formulação inicial do modelo, que foi aplicada a um caso simples de teste. Analisando os resultados do teste, o modelo foi refinado e reaplicado a testes mais complexos, uma heurística de inicialização

foi proposta, até ser obtido um modelo capaz de satisfazer as condições reais de trabalho de um programador de produção.

1.4 Organização da Dissertação

Essa dissertação se divide em mais seis outros capítulos, além do primeiro, conforme descritos a seguir.

O segundo capítulo apresenta o problema de *scheduling* de petróleo e derivados em portos em linhas gerais, enumerando os elementos mais importantes do sistema e o relacionamento com sistemas adjacentes.

O terceiro capítulo apresenta um rápido panorama sobre controle ótimo e suas aplicações e constrói um modelo para o problema que se quer resolver.

O quarto capítulo apresenta o método que será utilizado para resolver o modelo descrito no capítulo anterior e discute as vantagens de utilizar modelos especializados para controle ótimo.

O quinto capítulo apresenta uma heurística de inicialização para o problema de *scheduling* de portos e dutos e faz uma breve discussão das vantagens de se utilizar heurísticas para obter bons pontos iniciais para processos iterativos.

O sexto capítulo apresenta os resultados obtidos em diferentes casos de teste, de diferentes graus de dificuldade e tamanho.

O último capítulo apresenta uma conclusão e aponta possíveis assuntos para uma futura continuidade do trabalho.

Capítulo 2

Descrição do Problema

2.1 Planejamento e Scheduling

Não existe um modelo único ou genérico para problemas de planejamento e de *scheduling*, porém pode-se afirmar que sempre lidam com a alocação de equipamentos, seqüenciamento de atividades e definição do tempo de uso de equipamentos e recursos pelas tarefas processadas [5]. De acordo com a natureza do processo a ser programado ou planejado, pode-se determinar a relação entre alocação de equipamentos, consumo de recursos e tempo despendido para cada atividade.

No jargão técnico do *scheduling*, utiliza-se o termo atividade para determinar a alocação de um ou mais recursos, durante um certo intervalo de tempo, para realizar alguma etapa do processo efetuada no sistema [37]. Exemplos de atividades: transferência de material entre dois equipamentos, atracação e desatracação de navios, período de repouso de um equipamento etc.

Zaccarelli [43] destaca que o processo de programação e controle da produção é necessário para qualquer atividade produtiva, independente do tamanho do processo envolvido: mesmo um artesão, após definir qual produto construir, precisa realizar uma série de etapas de produção, alocando ferramentas e consumindo recursos em cada uma. Em um processo de maior porte, várias pessoas, equipamentos e recursos devem ser utili-

zados, resultando em problemas muito mais complexos para coordenar. Assim, torna-se importante o uso de metodologia científica para encontrar as melhores maneiras de se programar a produção.

A metodologia utilizada nessa dissertação, considerada adequada, é a programação matemática: qualquer problema de *scheduling* ou planejamento pode ser definido como um simples problema de otimização, da forma definida por Papadimitriou e Steiglitz [33]:

Definição 2.1.1 *Define-se por problema de otimização encontrar x tal que:*

Minimizar $f(x)$
Sujeito a: $g_i(x) \geq 0,\ i = 1, ..., m$
$h_j(x) = 0,\ j = 1, ..., p$

Onde f,g_i *e* h_j *são funções de* $x \in \mathbb{R}^n$.

As componentes $x_{k\in\{1,...,n\}}$ de x são variáveis que representam tudo que é mensurável no problema ao longo do horizonte de programação, seja tempo, propriedades de materiais ou equipamentos, recursos consumidos etc. Uma das mais importantes variáveis do problema de *scheduling* é a que representa o tempo, pois pode ser variável ou fixada *a priori* [16, 29]. A primeira representação baseia-se em uma programação dividida em intervalos de tempo de duração variável, com início e fim definidos de acordo com o processamento das atividades, enquanto a segunda baseia-se em intervalos de duração constante. Utilizar intervalos variáveis aumenta a complexidade do problema, enquanto utilizar intervalos constantes impõe uma granularidade à solução do problema.

Dependendo do modelo adotado para representar o problema, a função-objetivo $f(x)$ e as restrições $h_j(x), g_i(x)$ podem ser lineares, não lineares, contínuas, discretas,

diferenciáveis, não diferenciáveis, enquanto as variáveis x_k podem ser contínuas ou discretas (em particular, binárias). Papadimitriou e Steiglitz [33] classificam os problemas de otimização dentro dos ramos da programação matemática: programação não linear; programação convexa; programação linear; redes e fluxos; e programação inteira (Figura 2.1). E de acordo com cada classificação, há uma gama de métodos que podem ser utilizados.

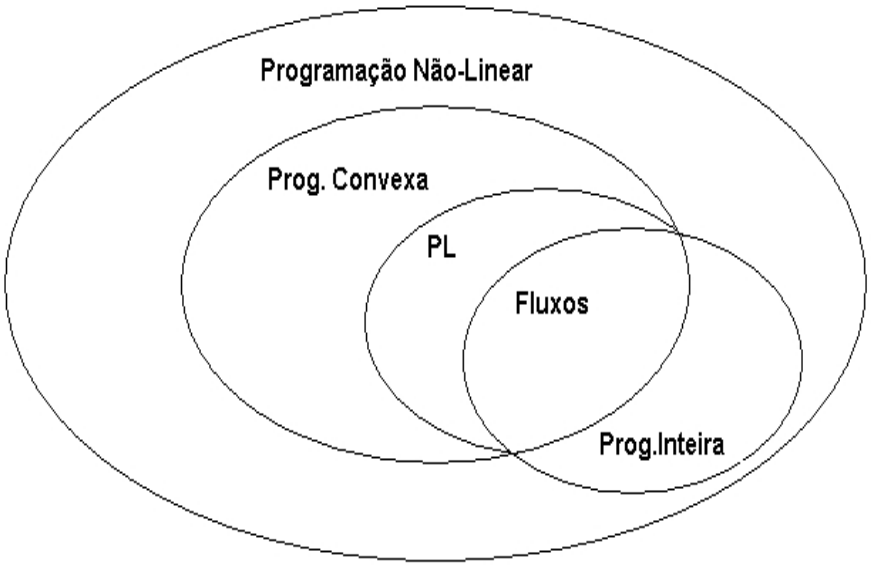

Figura 2.1: Classes de problemas

Os problemas de controle ótimo podem se enquadrar em qualquer uma das classificações, pois qualquer problema de controle ótimo nada mais é que um problema de programação matemática, conforme apresentado por Tabak e Kuo[41], e Canon et alii[4].

2.2 Problema de Scheduling de Portos

O problema de *scheduling* de recebimento de petróleo e entrega de derivados em portos é determinar a ordem de atracação dos navios e as transferências entre os equipamentos envolvidos de forma a minimizar uma função de *performance*, respeitando um conjunto de restrições operacionais, físico-químicas e econômicas.

A importância dos portos na cadeia de transportes da indústria de petróleo é evidente por serem os dutos e a via marítima as duas principais formas de transporte desta indústria. Diversas empresas operam ao redor do mundo no mercado *spot* de óleo cru, negociando carga navio a navio, ou sob contratos de fornecimento que garantem contínuo fornecimento às refinarias. Em ambos os casos, o cru fornecido à refinaria pode variar a cada navio, oriundos de diferentes poços de perfuração. A programação de produção das refinarias é, então, diretamente afetada pela mistura de crus recebida.

2.3 Sistemas Relacionados

O sistema de logística associado à produção e à distribuição de petróleo e seus derivados pode ser dividido em três subsistemas complementares: porto, tancagem intermediária e planta petroquímica ou refinaria [26, 27]. Cada subsistema se comunica com outro por meio de dutos (Figura 2.2), de modo que a programação de um subsistema consegue interferir com a de outro por meio da seqüência de materiais transportados nos dutos.

Dependendo do problema a ser otimizado, dois subsistemas podem ser agrupados em um só, considerando os dutos como tubulações internas. Por exemplo, caso o interesse seja de apenas programar o porto, pode-se considerar a tancagem intermediária como parte integrante da refinaria. Ou ainda, se o porto se encontra contíguo à planta petroquímica, os três subsistemas podem ser agrupados em um só.

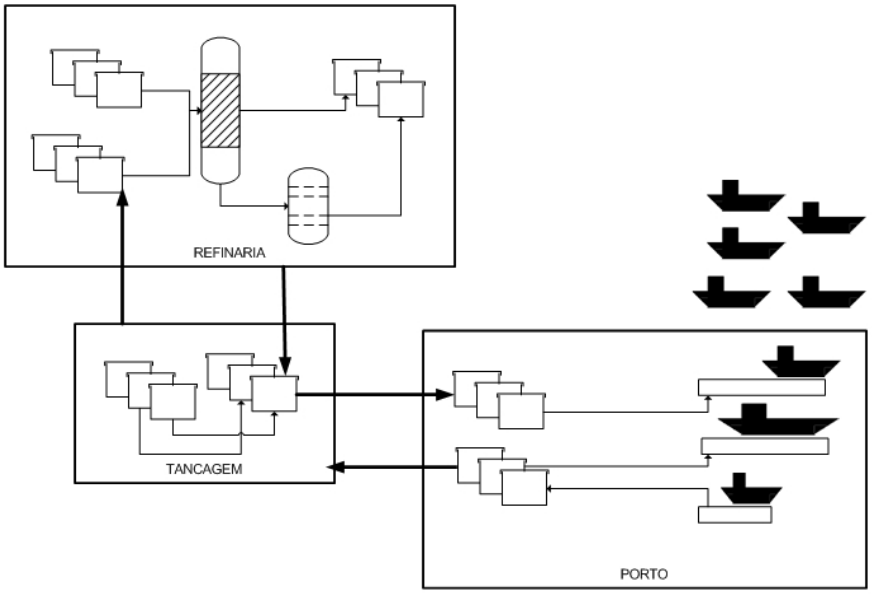

Figura 2.2: Representação esquemática do Problema

2.4 Dutos

O *scheduling* de dutos por si só é um importante assunto de pesquisa. No caso de se ter dutos multi-produtos, deve-se ter o cuidado de evitar a transferência de materiais de características muito diferentes, para diminuir a contaminação de um produto por outro [36]. Apesar de inevitável, a dimensão das interfaces pode ser minimizada com um planejamento racional que considere a natureza de cada fração transportada. O custo de operação de um duto é geralmente pequeno quando comparado a outros custos do sistema, e como eles podem operar continuamente, uma das metas de uma programação eficiente deve ser evitar interrupções no funcionamento dos mesmos.

2.5 Tanques

No porto existe uma tancagem própria, suficiente para o armazenamento do material a ser carregado ou descarregado pelos navios esperados nos próximos dias. Os tanques podem ser alugados ou próprios, e mesmo que o aluguel seja negligenciável, há sempre o custo de imobilização de capital; estoque representa perda de juros referentes ao valor do produto estocado, caso este fosse vendido e aplicado no mercado financeiro. Deste modo, os custos de armazenamento devem sempre ser considerados no problema. Normalmente, os tanques são dedicados a um único tipo de produto, ou seja, um tanque de petróleo não é utilizado para armazenamento de diesel, e vice-versa.

2.6 Navios

Os navios são contratados para carregar ou descarregar material em uma janela de tempo delimitada por uma data de chegada e outra de saída. Conforme as condições marítimas, a janela pode se mover no tempo, mas sempre é conhecida *a priori* para que se realize o *scheduling*. No caso de um navio não ser completamente processado pelo porto até a data de saída, uma multa é aplicada, geralmente por cada dia de atraso. Esta multa é chamada *demurrage*. Este é em princípio o mais importante custo a ser considerado no problema de *scheduling* em portos.

O porto apresenta um número fixo de píeres, com extensão e profundidade definidas, e bombas que podem ser restritivas por material. Logo, nem todos os píeres permitem a atracação de qualquer navio. Após ser processado, um navio ainda despende um tempo de desatracação, de modo que o píer só pode ser novamente utilizado para a atracação de outro navio se este tempo houver transcorrido.

2.7 Tancagem Intermediária

Pontos de tancagem intermediária são utilizados para coordenar a distribuição de material entre diferentes plantas de uma mesma empresa ou consórcio. Um exemplo é o problema estudado por Más [26]: a estrutura da Petrobras no estado de São Paulo, onde a tancagem intermediária distribui petróleo entre as diferentes refinarias deste estado (São Paulo). A programação da tancagem intermediária deve considerar a capacidade de produção das refinarias e o custo de inventário, trabalhando com um estoque suficiente, mas não excessivo. Por ser um subsistema simples, ele pode ser incorporado ao subsistema da refinaria no caso de se realizar a programação do porto.

2.8 Refinarias e Plantas Petroquímicas

O subsistema da planta petroquímica ou refinaria é tão ou mais complexo que o do porto, de modo que a programação geralmente é dividida em áreas: processamento de matéria-prima, transformações, misturas e entrega de produtos finais [38]. Por ter uma miríade de operações unitárias e misturas, e um parque de tancagem muito maior que os outros subsistemas, a programação deste subsistema não deve ser necessariamente considerada no problema de *scheduling* nos portos.

Capítulo 3

Modelo

3.1 Controle Ótimo

A formulação de problemas de controle ótimo é minimizar uma função de *performance* das trajetórias do estado e do controle de um sistema ao longo do tempo, sujeito a equações de estado e limites nas variáveis de estado e de controle, conforme apresentado por Tabak e Kuo [41]:

Definição 3.1.1 *Define-se por problema de controle ótimo, encontrar u, x tal que:*

Minimizar $J = \int_{t_0}^{t_f} g(x(t), u(t), t)dt + G(x(t_f))$

Sujeito a: $\dot{x} = f(x, u, t)$

$$x_{min} \leq x \leq x_{max}$$

$$u_{min} \leq u \leq u_{max}$$

As variáveis de estado geralmente são conhecidas em t_0 e em poucos casos em t_f. É comum definir uma região terminal $s(y(t_f)) = 0$. Restrições adicionais podem ser incorporadas ao problema de controle ótimo, sob a forma de sistemas de equações lineares ou não lineares $h(x, u) = 0$ e inequações $r(x, u) \geq 0$.

A partir da forma contínua do problema de controle, pode-se obter a forma discreta, apropriada a resolução utilizando técnicas de otimização. Tabak e Kuo [41] mostram que pode-se particionar o intervalo de tempo entre t_0 e t_f em n subintervalos tal que:

$\int_{t_0}^{t_f} g(x(t), u(t), t)dt = \lim_{n\to\infty} \sum_{k=1}^{n} g(x(t'_k), u(t'_k), t'_k) * (t_k - t_{k-1})$, onde $t_{k-1} \leq t'_k \leq t_k$.

Nota-se que o instante de tempo final t_f é finito, mas a quantidade de subintervalos n tende ao infinito. Utilizando a mesma partição de intervalos de tempo, obtém-se para as equações de estado:

$lim_{h_k\to 0} \frac{x(t_k)-x(t_{k-1})}{h_k} = f(x(t_{k-1}), u(t_{k-1}), t_{k-1})$, onde $h_k = t_k - t_{k-1}$.

O mesmo artifício pode ser utilizado para restrições adicionais e limites de variáveis, de modo que se obtém um problema de otimização conforme definido no capítulo anterior (definição 2.1.1) de dimensão infinita ($n \to \infty$). Tabak e Kuo [41] destacam que as condições KKT podem ser extendidas a espaços de dimensão infinita, mas que para soluções computacionais é necessário trabalhar em dimensões finitas. Para isso, basta aproximar o problema original por um discretizado em um número finito n de intervalos, definido como problema de controle ótimo em tempo discreto.

A formulação de problemas de controle ótimo em tempo discreto é minimizar uma função de *performance* das trajetórias do estado e do controle ao longo do tempo, sujeito a equações de estado e limites nas variáveis de estado e de controle, conforme apresentado, por exemplo, por Kirk [19]:

Minimizar $\sum_{t=0}^{T-1} g_t(x_t, u_t) + G_T(x_T), g_t \in C^1$
Sujeito a: $x_{t+1} = f_t(x_t, u_t), f_t \in C^1$
$x_{min,t} \leq x_t \leq x_{max,t}$

$$u_{min,t} \leq u_t \leq u_{max,t}$$
$$x_t \in \mathbb{R}^n, u_t \in \mathbb{R}^m$$
$$t \in \{0, 1, ..., T-1\}$$

Esta formulação, onde as variáveis de estado $x_t = x(t)$ e de controle $u_t = u(t)$ podem variar continuamente, permite que o problema de *scheduling* seja modelado de forma natural, sem adição de variáveis binárias de decisão. Ao invés de tratar o escalonamento de atividades (transferências, atracação e desatracação etc.), o modelo de controle ótimo modifica os valores das variáveis de controle continuamente, procurando otimizar a performance do sistema, calculando o estado do mesmo. A natureza combinatória do problema é considerada implicitamente pelo algoritmo de resolução, sujeito às equações de estado e as restrições de limites nas variáveis.

3.2 Intervalos de Tempo

Tendo-se por base a definição a tempo discreto, é importante discutir como calcular a duração de cada intervalo. É fácil observar que quanto maior o número de intervalos de duração tendendo a 0, melhor a aproximação do sistema real contínuo. Entretanto, deve-se trabalhar com um número de intervalos que gerem problemas que possam ser resolvidos em tempo computacional viável. Assim, a duração de cada intervalo não deve tender a zero, mas sim adotar um valor que permita o fracionamento do tempo em um número adequado de intervalos.

Os intervalos de tempo, indexados de 0 a T-1 podem ter durações específicas ou iguais, fixadas a priori ou durante a resolução do modelo. Durante a formulação os termos t_0 e $t = 0$ serão utilizados para designar o primeiro intervalo de tempo, enquanto t_f e $t = T - 1$ para o último intervalo de tempo.

Na literatura [5, 18, 30, 24, 26, 28] são encontradas com freqüência duas abordagens para a modelagem de intervalos de tempos: a discretização do tempo em intervalos pré-fixados de mesma duração e a discretização em intervalos durações específicas, determinadas durante a resolução dos modelos.

Joly et. al [18] apresentam uma discussão sobre o assunto. É discutido que modelos com intervalos de mesma duração devem ser adotados com cuidado, para não se lidar com um número excessivo dos mesmos. Como exemplo, um modelo apresentado no artigo não conseguia ser resolvido em tempo computacional viável, devido ao número de intervalos. Somente com intervalos de duração variável foi possível resolver o modelo. Por outro lado, a definição dos intervalos variáveis era obtida não pela otimização completa do problema, computacionalmente inviável, mas por um pré-processamento, baseado nos diferentes volumes de materiais a serem transportados em dutos. Note que, conforme houvesse diferentes volumes a serem transportados, eram definidas durações específicas para cada intervalo de tempo.

Uma outra abordagem para problemas com transportes em dutos, consiste em dividir o material transportado nos dutos em quantidades indivisíveis (pacotes) de volume fixo (ex. $1000m^3$) e calcular a duração de cada intervalo de tempo a partir da vazão de escoamento. Esse tipo de modelo foi adotado com sucesso por Simão [38] em um *scheduling* de refinaria.

3.2.1 Intervalos de Duração Variável

Nesta abordagem associa-se a cada intervalo de tempo uma duração variável I_t, que é otimizada conjuntamente com o resto do problema. Dentro da nomenclatura do controle ótimo, a variável I_t seria uma variável de controle, limitada por um limite mínimo, geralmente 0 (zero), e um limite máximo, determinado de acordo com o horizonte de *scheduling*:

$$0 \leq I_{min} \leq I_t \leq I_{max}$$

A vantagem desta abordagem é que durante a resolução do *scheduling*, tarefas que requeiram menor tempo poderão alocar os equipamentos somente quando necessário, sem alocação excessiva. Por exemplo, uma atividade que pudesse ocorrer em quinze minutos e outra que necessitasse de duas horas, provavelmente ocorreriam em intervalos de duração I_t diferentes (Figura 3.1).

A desvantagem dessa abordagem é que pode haver um aumento da dimensão do problema, aumentando o tempo computacional de resolução. São adicionadas restrições de máximo e mínimo, e uma variável a ser otimizada para cada intervalo de tempo. Por outro lado, o número total de intervalos pode ser menor. Alguns autores [30, 24] apresentam modelos que utilizam inicialmente esta abordagem, mas que, conforme a dimensão do problema for crescendo, passam a adotar a modelagem de intervalos fixos . Utilizam-se também pré-processamentos baseados no planejamento do transporte de materiais em dutos para determinar *a priori* diferentes intervalos de tempo.

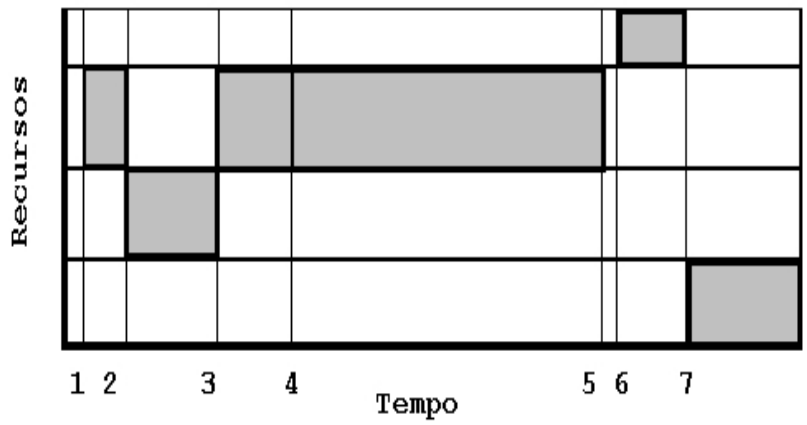

Figura 3.1: Intervalos de Duração Variável

3.2.2 Intervalos de Duração Fixa

Nesta abordagem associa-se a cada intervalo de tempo uma duração constante Δt.

A vantagem dessa abordagem é que a dimensão do problema não é afetada pela duração dos intervalos de tempo, que são considerados não como variáveis, mas sim como parâmetros de configuração do modelo.

Uma desvantagem é que recursos podem permanecer alocados desnecessariamente, se Δt for muito grande (Figura 3.2) para o horizonte de programação. Por exemplo, uma atividade que poderia ocorrer em apenas quinze minutos, pode alocar um recurso por tempo desnecessário, se $\Delta t > 15$ minutos. Por outro lado, deve-se tomar cuidado para não incorrer num modelo com granularidade intensa, pois se Δt for muito pequeno, será necessário considerar um número grande de intervalos, aumentando a dimensão do problema.

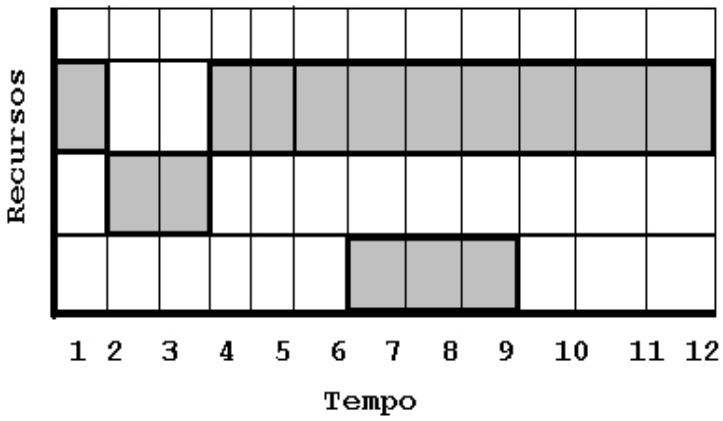

Figura 3.2: Intervalos de Duração Fixa

3.2.3 Modelagem Escolhida

Para a escolha adequada da modelagem de tempo, deve-se considerar as vantagens de cada abordagem, aplicadas aos recursos computacionais disponíveis e às necessidades de detalhamento da solução.

Problemas de *scheduling* geralmente consideram horizontes de alguns dias, de modo que intervalos de menos de 1 hora podem ser considerados pequenos. Por outro lado, podem existir atividades de curta duração, como, por exemplo, a separação da interface entre dois materiais transportados consecutivamente em um duto, que ocorreria em menos de 1 hora. Na tabela 3.1 apresentamos alguns exemplos de atividades e durações comumente encontradas no dia a dia.

No problema escolhido, as principais atividades podem ocorrer em intervalos de tempo de algumas horas, como, por exemplo, o atendimento de um navio em um píer. Assim, concluiu-se ser mais eficiente adotar intervalos de duração constante Δt de algumas horas, dividindo o horizonte em poucos intervalos, evitando incorrer em dimensões exageradamente grandes. Conforme o desenvolvimento dos problemas de teste, pôde-se refinar a duração de Δt, até se ter um *scheduling* que aliasse um bom tempo computacional com o necessário detalhamento da programação.

Exemplo de atividade	**Duração típica (h)**
Transferência entre tanque e duto	> 2
Tempo de Desatracação	> 1
Tempo de Preparação	$4 - 48$
Transferência de interface em duto	< 1
Atendimento de navio	$2 - 24$

Tabela 3.1: Exemplos de duração de atividades

3.3 Modelo de Controle Ótimo

Para modelar o problema de *scheduling*, é preciso identificar as variáveis de estado e de controle. As variáveis de controle são livres para variar dentro de limites estabelecidos, enquanto as de estado são necessariamente calculadas, segundo as equações de estado, dentro de seus limites.

Numa situação real, o operador pode escolher as melhores conexões para carga e descarga dos equipamentos, e controlar a vazão nas conexões, desde que respeitando os limites operacionais. Logo, é natural que as vazões sejam modeladas como variáveis de controle. Decorrente das vazões de entrada e saída em cada equipamento, os volumes e as propriedades dos materiais estocados são modificadas. Logo, é natural que as propriedades físico-químicas e volumes sejam modelados como variáveis de estado.

3.3.1 Variáveis de Estado

Definem-se as variáveis de estado como:

$x_t = [v_t, p_t]^T$, limitadas por:

$0 \leq v_{min} \leq v_t \leq v_{max}$

$p_{min} \leq p_t \leq p_{max}$

Onde v são as variáveis de volume de tanque e de navios e p as de propriedades e composição nos equipamentos e dutos; $x_0 = x_{t=0} = x_{t0}$ é conhecido no princípio do problema, enquanto $x_{T-1} = x_{t_f}$ é livre. Os limites máximo e mínimo são considerados constantes em todos os intervalos.

Pode-se notar que não é necessário considerar volume armazenado nos dutos em um instante de tempo, pois o inventário dos mesmos não consta no subsistema do porto.

Pode-se notar que as composições de misturas de petróleo podem ser modeladas como propriedades, i.e., fração de cru A, fração de cru B etc.

Para navios que apresentem diferentes cargas, consideram-se volume e propriedades distintas para cada carga.

3.3.2 Variáveis de Controle

Para o problema de *scheduling* é suficiente considerar somente conexões lógicas entre os equipamentos, sem detalhar as rotas internas do porto, observando cada ramo de tubulação. Se houver pelo menos uma rota física conectando dois equipamentos, e ambos os equipamentos compartilharem o mesmo tipo de material, é considerado que existe uma conexão lógica entre os dois. O detalhamento de rotas internas e tubulações deve ser considerado pela equipe de operação, e não pela de planejamento ou programação da produção.

Para o modelo, um navio também é um equipamento, que pode se conectar a um píer, que, por sua vez, pode se conectar a um tanque (Figura 3.3). Para identificar se existe uma rota entre um navio e um píer, o navio deve ter dimensões adequadas (comprimento e calado) para atracação no píer. Para identificar se o navio pode então ser carregado ou descarregado em um tanque conectado ao píer, é preciso que o tanque seja dedicado ao armazenamento do material transportado pelo navio.

É fácil notar que um pré-processamento estanque do problema, independente de intervalo de tempo, é capaz de determinar quais as conexões existentes.

Desta forma, definem-se as variáveis de controle como:

$u_t = [u_{tanque-duto,t}, u_{tanque-pier-navio,t}]^T$, limitadas por:

$0 = u_{min} \leq u_t \leq u_{max,t}$

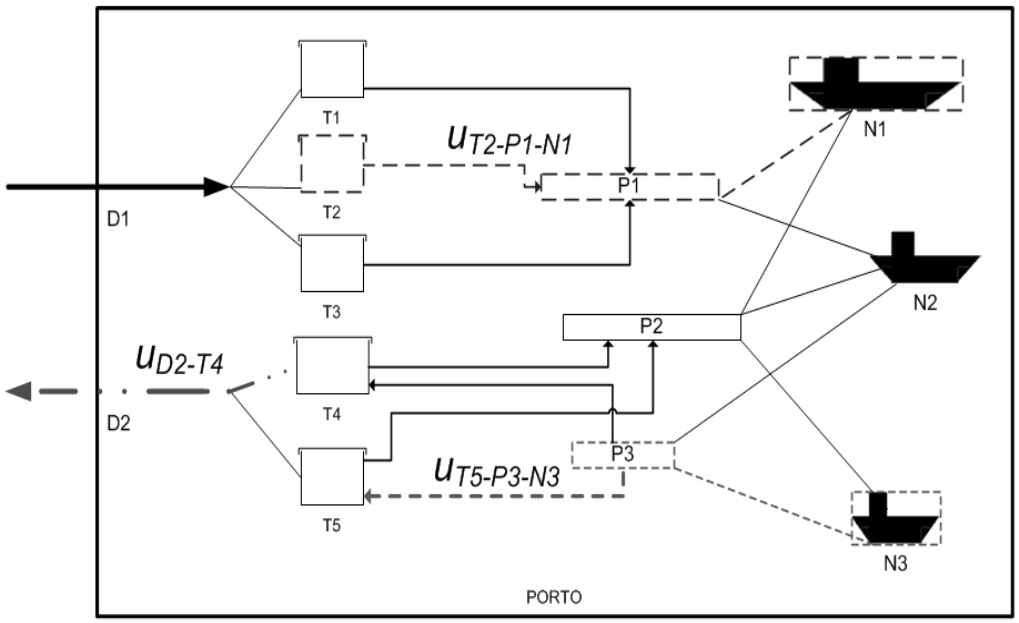

Figura 3.3: Vazões como variáveis de controle

Onde u_t representa as vazões das conexões entre tanques e dutos ($u_{tanque-duto}$) e entre tanques e navios ($u_{tanque-pier-navio}$). Os limites máximo e mínimo são determinados de acordo com as características dos equipamentos e do material transportado. No caso dos dutos, são considerados diferentes variáveis de controle para cada material (ou classe de material), com diferentes limites operacionais. No caso dos navios, são consideradas diferentes variáveis de controle de acordo com os píeres e com os materiais das cargas dos navios. Um mesmo navio pode ter mais de uma carga, e para cada carga, deve haver um conjunto de variáveis de controle. Os limites máximo e mínimo são considerados constantes em todos os intervalos de tempo, exceto nos seguintes casos:

- Sendo at_k a data contratada de chegada de um navio:
  Se $t < at_k \Rightarrow u_{max,k,t} = 0 \ \forall k \in J_{navio-tanque}$, onde $J_{navio-tanque}$ é conjunto dos índices k das conexões entre navios e tanques. Esse limite impede que algum navio

comece a ser atendido antes da sua data contratada de chegada.

- No caso de alguns píeres p ficarem indisponíveis para atracação de um navio n no instante t, tem-se $u_{max,(n,p),t} = 0$. Em alguns portos a disponibilidade dos píeres pode depender das marés, de modo que o acesso pode ser restringido em algumas horas do dia.

- No caso de algum equipamento k ter horários restritos de operação, tem-se $u_{max,k,t} = u_{restrito,k}, \forall t$ no horário restrito. Um exemplo é a operação de dutos apresentada por Magalhães [24], que é restringida diariamente entre às 18:00h e às 21:00h.

- No caso de algum equipamento k ter período de manutenção, tem-se $u_{max,k,t} = 0, \forall t$ no período de manutenção.

Pode-se notar que as vazões são representadas com valores não-negativos, independentemente de estarem carregando ou descarregando os equipamentos. Também deve-se destacar que numericamente as ocorrências de zero nos limites máximos são substituídas por uma tolerância ϵ (ex.10^{-6}) ao se resolver o modelo.

Deve-se destacar que a representação de vazões como variáveis de controle é o que tornou possível representar o problema de *scheduling* como problema de controle ótimo com variáveis contínuas. Modelos tradicionais procuram otimizar discretamente o escalonamento das atividades, levando a problemas combinatórios. O modelo apresentado não lida com escalonamento de atividades, mas sim com valores contínuos para vazões. As atividades de transferência podem ser definidas a partir das vazões em que sejam maiores que zero em cada intervalo de tempo. As outras atividades (tempo de preparação, atracação e desatracação etc.) podem ser determinadas pelos intervalos de tempo entre transferências.

3.3.3 Equações de Estado

A influência das vazões nos volumes de tanques e navios é definida pelas equações de estado:

$$v_t = v_{t-1} + (U\ u_{t-1}) * \Delta t$$
$$U_{i,j} \in \{1, 0, -1\}$$

Onde a matriz U é a matriz de incidência de vazões, definida de modo que $U_{i,j}$ seja o fator de incidência da conexão entre os equipamentos i e j. Por exemplo, suponhamos a configuração apresentada na Figura 3.4.

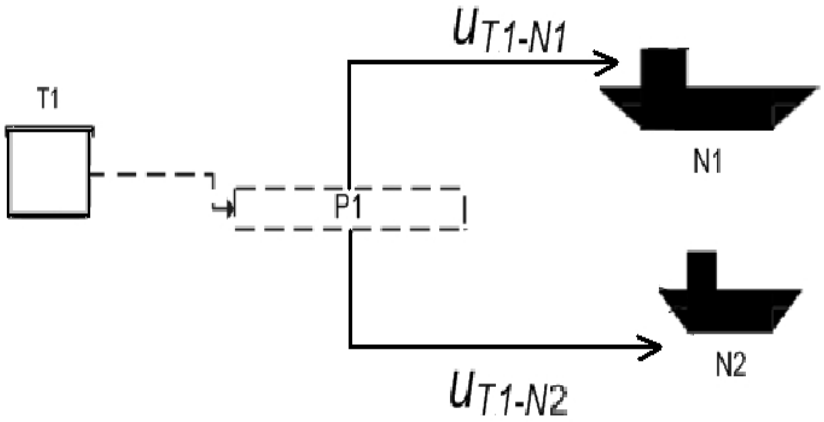

Figura 3.4: Esquema Representativo de Tanque e Navios

O tanque T1 pode ser esvaziado pelas vazões $u_{T1,N1}$ e $u_{T1,N2}$, enquanto os navios N1 e N2 podem ser carregados pelas mesmas vazões. Ora, isso significa que, para o tanque T1 as células de incidência referentes às vazões $u_{T1,N1}$ e $u_{T1,N2}$ valem -1 (esvaziamento), enquanto para os navios N1 e N2 valem 0 (vazão não se aplica ao equipamento), 1 (carregamento), conforme a Tabela 3.2.

A influência das vazões nas propriedades dos equipamentos é definida da seguinte

Equipamento	$u_{T1,N1}$	$u_{T1,N2}$
T1	-1	-1
N1	1	0
N2	0	1

Tabela 3.2: Matriz de Incidência para tanque T1 e navios N1 e N2

forma: se não houver vazão de entrada no equipamento, as propriedades se mantêm com o mesmo valor do intervalo anterior, pois não ocorreu qualquer mistura no último intervalo de tempo. Entretanto, se houver vazão de entrada, significa que alguma mistura ocorreu:

$$p_{v,q,t} = p_{v,q,t-1}, \text{ se } u_{v-entrada,t-1} = 0$$
$$p_{v,q,t} = f_{mix-q,t}, \text{ senão.}$$

Onde $p_{v,q}$ é o valor de uma propriedade q no equipamento v, v_v o volume, $u_{v-entrada}$ a vazão de entrada. Os valores das propriedades nos tanques, nas cargas dos navios e dutos em t_0 é considerado conhecido para a programação do *scheduling*, seja por medições realizadas nos equipamentos, seja por tabelamento, inferência ou simulação. Também é considerado que os produtos que chegam ao porto, via dutos ou navios, apresentam propriedades conhecidas (tabeladas ou medidas), e que somente os materiais que saem do porto, resultantes das misturas nos tanques é que têm propriedades calculadas.

Como diferentes propriedades podem ter regras distintas de mistura, o modelo representa isso com a função f_{mix-q}, que pode variar para cada propriedade q. A única exigência do modelo de controle ótimo é que f_{mix-q} seja diferenciável em cada intervalo de tempo, o que geralmente ocorre nas equações utilizadas na prática [39].

Algumas propriedades importantes para a indústria petroquímica como densidade ou teor de enxofre, são aditivas em base volumétrica, de modo que a função f_{mix-q} é:

$$f_{mix-q,t} = \frac{v_{v,t-1} * p_{v,q,t-1} + u_{v-entrada,t-1} * \Delta t * p_{v-entrada,q,t-1}}{v_{v,t}}$$

3.3.4 Restrições

Além das equações de estado utilizadas para calcular volume e propriedades de misturas, o sistema precisa atender a uma série de outras restrições, decorrentes da operação de portos e de tanques de petróleo e derivados. Para representar essas restrições no modelo de controle ótimo, utiliza-se do artifício de criar novas variáveis de estado.

Na solução do scheduling dos portos é desejado que nenhum equipamento opere em *pulmão*, enviando e recebendo simultaneamente, e nem que envie para ou receba de dois outros equipamentos simultaneamente. Essas operações resultam em instabilidade nas misturas, sendo difíceis de controlar na prática, portanto devem ser evitadas.

Para adequar esta restrição ao modelo de controle ótimo, pode-se criar uma variável de estado artificial w, tal que:

$0 \leq w_{k,t+1} = \max(u_{i,t}) - \sum_{j \in J_k} u_{j,t} \leq \epsilon, \forall i \in J_k$, onde:

J_k = conjunto de índices das vazões que envolvem o equipamento k

ϵ = tolerância (ex: 10^{-6})

Um navio só pode ser processado por um píer se o navio processado anteriormente no mesmo píer houver desatracado. Para garantir tal restrição, o píer precisa estar vazio, sem vazões, durante o tempo necessário para desatracação do navio anterior.

Para adequar esta restrição ao modelo de controle ótimo, pode-se criar uma variável de estado artificial z, tal que:

$0 \leq z_{p,n,t+1} = u_{p,n,t} * \sum_{j=t-T_p}^{t} u_{p,n*,j} \leq \epsilon$, onde:

$u_{p,n,t} = \sum_{i \in J_{p,n}} u_{i,t}$

$u_{p,n*,t} = \sum_{i \in J_{p,n*}} u_{i,t}$

$J_{p,n*}$ = conjunto de índices de vazões do píer p, mas não do navio n

$J_{p,n}$ = conjunto de índices de vazões do píer p e do navio n

T_p = tempo de desatracação no píer p

ϵ = tolerância (ex: 10^{-6})

Um tanque só pode enviar para outro equipamento se houver ocorrido desde o último recebimento o tempo necessário para o assentamento de impurezas, chamado de *tempo de preparação*. O tempo de preparação normalmente é associado ao tipo de material, logo, como estamos trabalhando com tanques dedicados, o tempo de preparação pode ser associado diretamente ao tanque. Para garantir tal restrição, o tanque deve estar em repouso, sem vazões, durante o tempo de preparação.

Para adequar esta restrição ao modelo de controle ótimo, pode-se criar uma variável de estado artificial y, tal que:

$0 \leq y_{v,t+1} = u_{v-saida,t} * \sum_{j=t-T_v}^{t} u_{v-entrada,j} \leq \epsilon$, onde:

$u_{v-saida,t} = \sum_{i \in J_{v-saida}} u_{i,t}$

$u_{v-entrada,t} = \sum_{i \in J_{v-entrada}} u_{i,t}$

$J_{v-saida}$ = conjunto dos índices de vazões que saem do tanque v

$J_{v-entrada}$ = conjunto dos índices de vazões que entram no tanque v

T_v = tempo de preparo do tanque v

ϵ = tolerância (ex: 10^{-6})

Ao final do *scheduling*, todos os navios devem ser atendidos corretamente.

Para adequar esta restrição ao modelo de controle ótimo, pode-se criar uma variável de estado artificial r, tal que para cada navio n:

$r_{n,t} = 0$ se $t < t_f$,

$0 \leq r_{n,t_f} = (v_{n,t_f} - vlt_n)^2 \leq \epsilon$, onde:
vlt_n = volume da carga do navio n quando for processado
ϵ = tolerância (ex: 10^{-6})

É possível que esta restrição não seja atendida, de modo que o algoritmo não consiga convergir para um ponto viável. Neste caso, deve-se relaxar o problema, incorporando a restrição à função-objetivo, sob a forma de custo penalizado, chamado de *custo de perda de navios*.

Ao final do scheduling, todos os dutos devem ser atendidos corretamente.

Para adequar esta restrição ao modelo de controle ótimo, pode-se criar uma variável de estado artificial s, tal que para cada duto n e material m:

$s_{n,m,t} = 0$ se $t < t_f$,
$0 \leq s_{n,m,t_f} = (M_{n,m,t_f} - Mf_{n,m})^2 \leq \epsilon$, onde:
$M_{n,m,t_f} = \sum_{t=t_0}^{t_f} u_{n,m,t} * \Delta t$
$u_{n,m,t}$ = vazão de material m no duto n em t
$Mf_{n,m}$ = total de m que deve ser transportado por n entre t_0 e t_f
ϵ = tolerância (ex: 10^{-6})

É possível que esta restrição não seja atendida, de modo que o algoritmo não consiga convergir para um ponto viável. Neste caso, deve-se relaxar o problema, incorporando a restrição à função-objetivo, sob a forma de custo penalizado, chamado de *custo de desvio do planejamento*.

Deve-se destacar que as restrições apresentadas nesse tópico poderiam ser representadas sob a forma de restrições de igualdade, ao invés de restrições de caixa, limitadas inferior e superiormente. Entretanto, é numericamente interessante adotar limites, pois

possibilitam uma maior flexibilidade à resolução do problema. As variáveis de estado criadas para representar as restrições podem variar mais livremente com a consideração dos limites durante a otimização, impedindo uma rigidez na variação das variáveis de controle baseadas nas quais se calculam as de estado.

3.3.5 Função de Performance

A função de *performance* do sistema é uma função de custo, composta pela soma ponderada de parcelas distintas. Quanto maior o número de parcelas, maior é a possibilidade de exercerem um efeito amortecedor na função-objetivo, por isso custos como de energia elétrica, de bombeamento e de aquecimento de petróleo não foram considerados. O uso de pesos serve para adequar as ordens de grandeza dos diferentes custos e hierarquizar os objetivos.

Definição 3.3.1 *Define-se a função de performance tal que:*

Minimizar $\sum_{t_0}^{t_f} g(x_t, u_t) + G(t_f)$, *onde:*

$$g(x_t, u_t) = p_1 * C_{demurr}(t) + p_2 * C_{vazao}(t) + p_3 * C_{invent}(t) + p_4 * C_{interf}(t)$$

$$G(t_f) = p_5 * C_{planej} + p_6 * C_{navio}$$

Para o problema de *scheduling* em portos um importante objetivo é evitar a ocorrência de multas (*demurrage*):

$$C_{demur}(t) = cdemurr^T * atraso(t)$$

$$atraso_k(t) = \max(0, t - lt_k) * \Delta t * (v_k(t) - vlt_k)^2$$

lt_k = data contratual de saída do navio

$v_k(t)$ = volume do navio k em t

vlt_k = volume do navio k quando for completamente processado

Onde *cdemurr* é o vetor de custos de demurrage para cada navio e $atraso(t)$ é um vetor com fatores associados a cada navio k, que se mantêm zero, caso o navio não esteja atrasado, mas que crescem conforme o tempo de atraso e o volume necessário para o término de processamento daquele navio.

Para evitar que a resolução do modelo incorra em programações com mudanças desnecessárias de vazões, pode-se associar um custo arbitrário às suas variações:

$$C_{vazao}(t) = cvaz^T * deltavaz(t)$$
$$deltavaz_k(t) = \frac{(u_k(t)-u_k(t-1))^2}{1+u_k(t-1)}, \text{ se } t > t_0$$
$$deltavaz_k(t_0) = 0$$

Onde *cvaz* é o vetor de custos de mudança de vazão para cada vazão e $deltavaz(t)$ é um vetor com medidas quadráticas de variação de cada vazão k. O denominador apresenta um termo "1"para impedir divisões por zero.

Na solução do problema há o desejo de se obter um nível de estoque suficiente apenas para atender os navios sem atraso, evitando que um número grande de tanques seja utilizado na programação. O fator de ponderação para o custo de inventário deve ser pequeno para não afetar a função de performance de modo a obter soluções com atrasos nos navios ou mudanças desnecessárias de vazão.

Para evitar que a resolução do modelo incorra em programações com altos estoques:

$$C_{invent}(t) = cinvent^T * v(t)$$

Onde *cinvent* é o vetor de custos de inventário para cada tanque e $v(t)$ é o vetor com os volumes dos tanques.

Caso seja de interesse do programador de produção, pode-se considerar também o inventário dos navios, bastando para isso estipular custos de inventários e incluir os volumes armazenados nos navios. Entretanto, não é recomendado incluir inventário de navios no modelo pelas seguintes razões: navios a ser carregados vão necessariamente aumentar seu inventário; navios a ser descarregados armazenam material que ainda não faz parte do estoque da companhia que vai recebê-lo.

A escala de materiais nos dutos pode ser considerada, mas não deve obrigar que os estoques fiquem altos ou que os navios atrasem. Além disso, como para cada material transportado em duto existe um conjunto próprio de variáveis de controle, o custo de variação de vazão já inibiria naturalmente a mudança de material transportado a cada intervalo de tempo. O fator de ponderação deve ser, portanto, muito pequeno:

Define-se o custo de interface como:

$$C_{interf}(t) = cinterf^T * deltaq(t)$$
$$deltaq_k(t) = (q_k(t) - q_k(t-1))^2$$

Onde $cinterf$ é o vetor de custos de interface para cada duto k e $deltaq(t)$ é o vetor com medidas quadráticas da variação de propriedade q do duto k. A propriedade q deve ser uma propriedade que identifique bem o material que está no duto e que faça sentido para todos os materiais, como por exemplo, a densidade.

Quando é impossível atender as restrições nas variáveis de estado $s(t)$, deve-se penalizar a função de *performance* com o custo de desvio de planejamento:

$$C_{planej} = \sum_{n \in J_n} \sum_{m \in J_m} s_{n,m}(t_f)$$
J_n = conjunto de índices de dutos
J_m = conjunto de índices de materiais

Quando é impossível atender as restrições nas variáveis de estado $s(t)$, deve-se penalizar a função de *performance* com o custo de perda de navios:

$C_{navio} = \sum_{n \in J_n} r_n(t_f)$

J_n = conjunto de índices de navios

Capítulo 4

Método de Resolução

Para a resolução de problemas de programação não-linear com restrições, há uma grande variedade de métodos eficientes. Nocedal e Wright [31] definem que algoritmos de otimização são processos iterativos, inicializados em uma primeira estimativa, e que, em cada iteração, obtém uma nova estimativa, tentando melhorar o valor da função-objetivo do problema, e obedecendo possíveis restrições presentes. O modo como cada algoritmo obtém uma nova estimativa a cada iteração é o que os diferencia. Entretanto, independentemente de como as iterações são executadas, um algoritmo para ser considerado bom deve apresentar as seguintes características: robustez, eficiência computacional (armazenamento e processamento) e precisão (estabilidade numérica).

Como o modelo de controle ótimo apresentado no capítulo anterior é um modelo de programação não linear restrita, devemos procurar um método adequado que reúna as característica destacadas acima. Pinto [35] apresenta quatro principais classes de métodos para programação não linear restrita, todas apresentando diversos graus de robustez, eficiência e precisão, conforme descritas a seguir:

- Métodos de Penalidades: consiste em transformar os problemas com restrições em problemas irrestritos equivalentes, por meio da incorporação das restrições à função objetivo. São utilizados três formas de incorporação, gerando três tipos

desses métodos: penalidade, barreira, lagrangeano aumentado. Esta classe de métodos é uma das mais tradicionais dentro da programação não linear, porém apresenta algumas dificuldades na sua utilização: as penalidades são aumentadas conforme aproxima-se da solução ótima, onde finalmente as restrições penalizadas são zeradas. Desta forma, a cada iteração a função-objetivo piora, e somente no ótimo alcança um valor adequado, exigindo bastante cuidado na implementação dos parâmetros de penalização. Como estes convergem para infinito na proximidade do ótimo, podem surgir dificuldades numéricas que venham a impedir uma convergência adequada.

- Métodos de Programação Linear Sucessiva: consiste em resolver a cada iteração k um problema de programação linear gerado a partir de aproximações em séries de Taylor de primeira ordem (lineares) da função objetivo e das restrições. Baseado na solução do problema de programação linear, é definida a direção de descida e atualizada a estimativa de solução. Segundo Pinto [35], esta classe de método surgiu de aplicações para indústria petroquímica e refinarias, sendo popular nesse meio até os dias de hoje. A grande vantagem dessa classe é que pode-se utilizar o Simplex [7] para resolver as aproximações. As desvantagens são que se o ótimo não estiver em um dos vértices a convergência é lenta, e também que durante as iterações as restrições não lineares podem estar sendo sempre violadas.

- Métodos de Programação Quadrática Sucessiva: consiste em resolver a cada iteração k um problema de programação quadrática gerado a partir da incorporação da hessiana do lagrangeano, estimada por um método quase-Newton (BFGS ou DFP). Quando a hessiana estimada coincide com a hessiana exata do problema quadrático, o método equivale ao de Newton, de modo que pode-se obter altas taxas de convergência. A desvantagem desse método é que ele pode ser extremamente ineficiente para problemas de grande porte, devido à atualização e armazenamento da hessiana a cada iteração. Por outro lado, considera-se que o método é eficiente para

problema de pequeno porte ou caixa-preta, quando não é trivial explicitar os gradientes analiticamente. Pinto [35] destaca que nas últimas décadas tem havido um esforço para se obter implementações eficientes de Programação Quadrática Sucessiva para problemas de grande porte, mas ainda sem resultados definitivos.

- Métodos de Gradiente Reduzido: consiste em resolver o problema de otimização por meio da eliminação de variáveis, tratando problemas menores que os originais. Dividem-se as variáveis de em dois grupos, as básicas e as independentes. Efetua-se a otimização em cima do grupos das variáveis independentes, e a cada iteração, pode-se mover variáveis de um grupo para outro. O teorema da função implícita garante a relação entre os dois grupos de variáveis, definida pelas restrições de igualdades. Luenberger [23] destaca que sob o ponto de vista computacional o método do Gradiente Reduzido é relacionado ao Convex-Simplex, enquanto do ponto de vista teórico, com o método do Gradiente Projetado. Quando a função-objetivo e as restrições são lineares, o método equivale ao Simplex. Quando somente as restrições são lineares, tem-se o método do Gradiente Reduzido de Wolfe. Quando a função-objetivo e as restrições são não lineares, tem-se o Gradiente Reduzido Generalizado (GRG), de Abadie e Carpentier. Existem diversas implementações para o GRG, algumas especializadas para problemas de grande porte, como a apresentada por Lasdon et al.[40]. Os métodos de Gradiente Reduzido possuem muito bom desempenho para problemas que tenham restrições lineares, e em casos onde as derivadas analíticas das não lineares podem ser supridas. Tipicamente, conseguem resolver problemas com várias centenas de variáveis e restrições. Pinto [35] destaca que, comparado com os métodos de Programação Quadrática Sucessiva, os métodos de Gradiente Reduzido requerem um número maior de iterações, porém com um menor tempo computacional por iteração.

Considerando as características das diferentes classes de métodos, pode-se afirmar que a utilização do GRG se adequa perfeitamente ao modelo de controle ótimo

apresentado no capítulo anterior. Há restrições lineares e não lineares, e os problemas podem ser de grande porte, conforme seja expandido o horizonte de programação e o grau de discretização do tempo. Martinez e Santos [25], inclusive, destacam que o GRG vem sendo utilizado historicamente por pesquisadores na área de controle, pois, além da facilidade na representação das variáveis de controle e estado, ele mantém a viabilidade em todas as iterações. O GRG, portanto, apresenta todas as características necessárias para ser considerado um bom método para o problema que se deseja resolver.

4.1 Método do Gradiente Reduzido Generalizado

Abadie [1] descreve o GRG como uma generalização do método do Gradiente Reduzido de Wolfe [42], onde tanto as restrições, quanto a função-objetivo são não lineares. Ele resolve problemas de programação matemática da seguinte forma:

Minimize $f(x)$
Sujeito a: $h(x) = 0$
$a \leq x \leq b$
$x, a, b \in \mathbb{R}^n$
$f : \mathbb{R}^n \rightarrow \mathbb{R},\ f \in C^1$
$h : \mathbb{R}^n \rightarrow \mathbb{R}^m,\ h_i \in C^1,\ \forall i \in \{1, ..., m\}$

Qualquer problema de otimização pode ser representado na forma acima. Para representar problemas de limites irrestritos, basta considerar $a'_j s$ e $b'_j s$ como $+\infty$ ou $-\infty$. Para representar restrições de desigualdade, basta adicionar variáveis de folga.

O GRG é inicializado com um ponto viável x_0, e supõe-se que para todo ponto x obtido pelo método, a condição de *não degenerescência* da matriz é válida. Desse modo, existe uma partição de x tal que:

1. $x = (y, z)$, onde $y \in \mathbb{R}^m$ é o vetor de variáveis básicas e $z \in \mathbb{R}^{n-m}$, o de vetor de variáveis independentes;

2. Se $a = (a_y, a_z)$ e $b = (b_y, b_z)$, então $a_y < y < b_y$;

3. A matriz (m x m) $\nabla_y h(y, z)$ é não singular em $x = (y, z)$.

O conceito por trás dessa divisão é que o problema deve apresentar um conjunto de restrições ativas de dimensão menor que o de todas as restrições, mais fácil de resolver. De fato, mostra-se que a partir das variáveis independentes, pode-se calcular as dependentes (básicas), pois pelo teorema da função implícita, há uma única função $y = F(z)$ em uma vizinhança U_0 ao redor de z_0, tal que:

1. F é contínua em U_0;

2. $h(x) = h(F(z), z) = 0, \ \forall z \in U_0$;

3. $y_0 = F(z_0)$;

4. $\exists \ F'(z)$ contínuo em U_0, sendo $F'(z_0) = g_0$ (gradiente reduzido em z_0).

Deste modo, o problema original de minimizar $f(x)$, é reduzido ao problema mais simples de minimizar $f(z) = f(F(z), z)$, sujeito apenas a $a_z \leq z \leq b_z$. As variáveis dependentes são determinadas a partir das variáveis independentes. Pensando-se no caso das restrições lineares, é fácil de ver que y pode ser diretamente determinado a partir de z, por meio da resolução do sistema linear $h(x) = Ax = By + Cz = h(y, z) = 0$, de modo que $y(z) = -B^{-1}Cz$. No caso mais genérico das restrições não lineares as variáveis são projetadas em planos tangentes à superfície de restrições, e em seguida retornam ao espaço de soluções ao se resolver o sistema não linear $h(x) = 0$ iterativamente.

Algoritmo 4.1.1 *De posse dessas informações, podemos rapidamente descrever o algoritmo do GRG como:*

1. *Iniciar com* $k = 0$, *tal que* $h(x^k) = 0$.

2. *Calcular a Jacobiana em* x^k *e separar* x^k *em variáveis básicas* (y^k) *e não-básicas* (z^k) *de maneira a respeitar a hipótese de não-degenerescência:*
   a) $a_y < y^k < b_y$
   b) $\nabla_y h(y^k, z^k)$ *é não singular.*

3. *Cálculo da direção de descida das variáveis independentes:*
   a) Cálculo dos multiplicadores de Lagrange:
   $\lambda = -\nabla_y f(y^k, z^k)[\nabla_y h(y^k, z^k)]^{-1} = -c_y B^{-1}$
   b) Cálculo do Gradiente Reduzido:
   $g = \nabla_z f(y^k, z^k) + \lambda \nabla_z h(y^k, z^k) = c_z + \lambda A$
   c) Cálculo do Gradiente Reduzido Projetado:
   Definir vetor P *tal que:*
   Se $g_j > 0$ *e* $z_j^k = a_j$ *ou* $g_j < 0$ *e* $z_j^k = b_i$ *então* $P_i = 0$.
   Senão, $P_i = -g_i$.
   d) Verificar fim:
   Se $P_i = 0 \; \forall i$, *encontrou um ponto KKT: fim do algoritmo.*
   e) Calcular direção a partir do Gradiente Reduzido Projetado:
   Fazer $d_z = P$ *ou utilizar algum método quase-Newton ou de Gradiente Conjugado para determinar* d_z.

4. *Cálculo da direção de descida das variáveis básicas:*
   $d_y = -B^{-1} A d_z$

5. *Busca linear inexata para melhorar solução. Encontrar* α *até atender uma condição de Armijo ou Wolfe, tal que diminua (minimize):*
   $f(x^k + \alpha d)$, *onde* $d = (d_y, d_z)$.

6. *Mover variáveis básicas e não básicas:*
   $\tilde{x} = x^k + \alpha d = (y^k + \alpha d_y, z^k + \alpha z_y) = (\tilde{y}, \tilde{z})$

7. *Restaurar viabilidade, recalculando as variáveis básicas, pois o ponto $\tilde{x}$ pode ser inviável (Figura 4.1).*
   Isso é feito por um método pseudo-Newton para resolução do sistema de equações não lineares $h(y, \tilde{z}) = 0$:
   a) Calcular $J = [\nabla_y h(\tilde{x})]^{-1}$.
   b) Fazer $j = 0$.
   c) Iterar $y^{j+1} = y^j - Jh(y^j, \tilde{z})$ até obter convergência.
   d) Fazer $\tilde{y} = y^j$

   Durante a resolução do sistema, algumas variáveis básicas podem ficar saturadas (nos limites), sendo então necessária a realização de troca de variáveis da base por variáveis não básicas com folgas.

8. *Fazer $x^{k+1} = \tilde{x} = (\tilde{y}, \tilde{z})$.*

9. *Se $f(x^{k+1}) > f(x^k)$, retornar para (5) e utilizar um α menor.*

10. *Se $f(x^{k+1}) < f(x^k)$, retornar para (5) e utilizar um α maior.*

Alguns pontos devem ser destacados no uso do GRG. Primeiro, deve-se inicializar o método com um ponto inicial viável x_0, o que pode não ser trivial de encontrar. No caso de não se conhecer um x_0 viável, deve-se proceder adicionando variáveis artificiais e realizar um procedimento de Fase 1 (semelhante ao Fase 1 do Simplex) ou de penalidade. Diferentes versões do GRG são obtidas de acordo com os métodos utilizados em cada um dos passos. As variantes mais comuns se referem ao cálculo da direção de descida (passo 3(e)), que pode ser feito por um método quase-Newton ou gradiente conjugado;

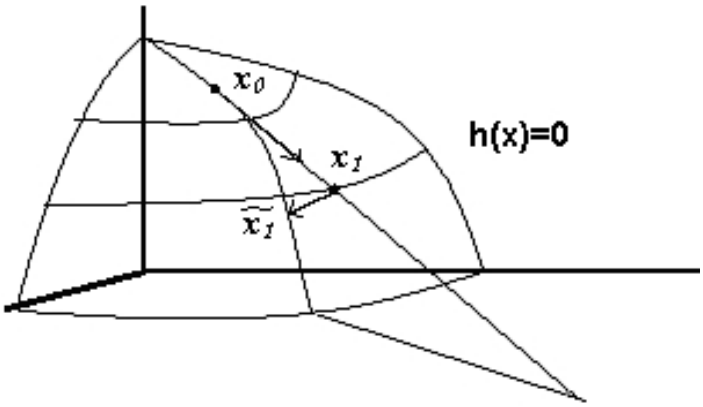

Figura 4.1: Restauração da Viabilidade

à resolução do sistema não linear, que pode ser feito pelo método de Newton (passo 7); ao cálculo do passo α, que pode ser obtido por diferentes métodos de busca linear (passo 5); à definição de variáveis para troca de base, que pode ser feita por regras adaptadas de programação linear (passo 7).

Deve-se observar também que o GRG encontra um ponto que atenda as condições KKT, ou seja, um ótimo local, a partir do ponto inicial fornecido. Em problemas de engenharia, procura-se obter soluções que melhorem os processos já existentes, mas que não exijam mudanças radicais, pois poderiam precisar de investimentos financeiros e tempo para serem implementadas. Assim, pode-se concluir que o GRG aplica-se perfeitamente à resolução de problemas de engenharia.

4.2 Método do GRG para Problemas de Grande Porte

Lasdon et. al [20] implementaram, no final da década de 70, uma versão do GRG, que segundo testes realizados pelos autores no artigo citado, seria a mais robusta e acurada até então. Entretanto, a implementação se mostrava eficiente apenas para problemas de tamanho até médio porte, ainda não capaz de resolver problemas de grande porte.

Porém, na conclusão do artigo, os autores destacam que trabalhavam num método GRG eficiente para problemas não lineares de grande porte, desde que apresentassem esparsidade e um bom número de equações lineares.

Problemas reais de planejamento e *scheduling*, como o modelo de controle ótimo apresentado nessa dissertação, apresentam tais características. Cada restrição envolve um conjunto reduzido de variáveis, pois se aplica a um intervalo de tempo e a um número pequeno de equipamentos. Logo, a matriz Jacobiana das restrições é esparsa. A relação entre vazões e volumes é linear, pois considera-se vazão constante em um intervalo de tempo. Esse fato torna o sistema rico em restrições lineares, o que facilita o cálculo da matriz Jacobiana, apresentando coeficientes constantes.

Em meados da década de 80, Lasdon e Waren apresentam uma nova versão de seu código GRG2 [21], que se utiliza da detecção de equações lineares para facilitar a otimização, de fornecimento de derivadas analíticas ou estimativas por diferenças finitas e uma implementação robusta do BFGS. Fylstra et al. apresentam em [15] o desenvolvimento do *solver built-in* do aplicativo Microsoft Excel, utilizando o GRG2 como método de otimização.

Para problemas de grande porte é essencial que sejam utilizados métodos eficientes com uso limitado de memória. No início da década de 90 foi apresentado por Lasdon e Smith o LSGRG2 [40], uma versão do GRG2 para problemas de larga escala (*Large Scale* GRG2), que utiliza estruturas de armazenamento esparsas e apresenta métodos de gradiente conjugado e BFGS de memória limitada. Essa versão é utilizada pela empresa Frontline Systems [14] no código da versão *Premium Platform* do *solver* do Microsoft Excel.

Oliveira [32] discute a implementação do LSGRG2 e introduz o uso de paralelismo, com resultados animadores. O método do gradiente conjugado para determinar a direção de descida e a fatoração LU, realizada para resolver o sistema de equações

não lineares, foram paralelizados com sucesso. Esse trabalho é mais um indicativo de que o GRG pode ser ainda aperfeiçoado computacionalmente, permitindo resolução de problemas maiores e mais complexos.

4.3 Método do GRG Especializado a Controle Ótimo de Sistemas Dinâmicos

Abadie [1] mostrou que o GRG pode ser aplicado a solução de problemas de controle ótimo. Conforme descrito nos capítulos anteriores, um problema de controle ótimo divide as variáveis em dois grupos: de controle e de estado. As variáveis de estado de cada intervalo de tempo são calculadas a partir das de controle e de estado do intervalo anterior, baseado em equações de estado: $x_t = f(x_{t-1}, u_{t-1})$.

Ora, o GRG adota a divisão em dois grupos de variáveis, as básicas e as independentes, e realiza a otimização baseada em problemas formulados em relação às independentes, limitadas por limites inferior e superior. Esse procedimento pode ser diretamente estendido para o controle ótimo, adotando uma base (chamada de *simplest basis* por Abadie) formada pelas variáveis de estado, enquanto as variáveis de controle aparecem como variáveis independentes. Durante as iterações do GRG, uma ou mais variáveis de estado podem violar seus limites, e neste momento devem ser retiradas da base, trocadas por variáveis de controle, escolhidas de acordo com critérios bem definidos. No caso do artigo de Abadie, o critério é escolher variáveis de controle, do mesmo período das de estado, que estejam com folga, ou seja, distante de seus limites inferior e superior.

De posse das variáveis básicas e não básicas, o GRG realiza a etapa de resolução do sistema de equações não lineares que relacionam ambos os grupos de variáveis. Enquanto no GRG geral essa relação era garantida pelo teorema da função implícita, no

controle ótimo o sistema é explícito, composto pelas equações de estado. Cabe então, mais uma observação: as equações de estado podem ser resolvidas ordenadamente conforme os instantes de tempo t, de t_0 a t_f. Isso torna o método extremamente eficiente computacionalmente: ao invés de resolver o problema de otimização por inteiro, resolvem-se pequenos sistemas não lineares, um por vez, obtendo economia em processamento e armazenamento. A Figura 4.2 mostra a topologia da matriz Jacobiana de restrições para problemas de controle ótimo. Nela pode-se ver que as variáveis de estado e do controle de um instante, impactam o instante posterior, e que pode-se resolver o problema consecutivamente por instantes de tempo.

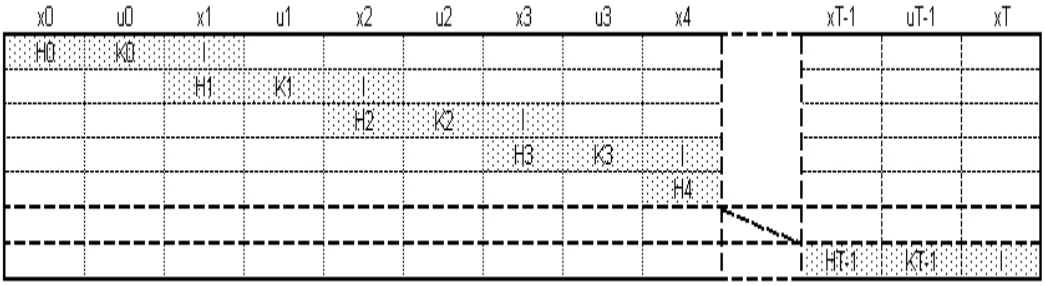

Figura 4.2: Matriz Jacobiana das Restrições de um Problema de Controle Ótimo

Facó [10], a partir do trabalho de Abadie, criou uma especialização do GRG para controle ótimo, o Gradiente Reduzido Especializado a Controle Ótimo (GRECO), que permite resolver problemas de controle ótimo com e sem atraso, além de estender as possibilidades de escolha de variáveis básicas no caso das variáveis básicas estarem em um de seus limites. Um problema com atraso é aquele em que uma variável de estado é função das variáveis de estado e controle de mais de um instante de tempo anterior. O GRECO constrói uma base, de modo a só precisar armazenar uma pequena matriz triangular inferior por blocos ao invés da base completa. Esta é a chamada base de trabalho, formada segundo os seguintes critérios:

1. Adicione as variáveis artificiais não nulas à base de trabalho prioritariamente, no

caso de se realizar uma Fase 1;

2. Adicione a base de trabalho o máximo possível de variáveis de estado de cada intervalo t que estejam com folga;

3. No caso de haver variáveis de estado saturadas (sem folga), variáveis de controle com folga do mesmo período t substituem estas na base;

4. No caso de não haver tais variáveis de controle, a variável de estado saturada continua na base, e uma variável de controle de um intervalo anterior é adicionada à base do GRG;

5. Em cada matriz triangular inferior em blocos, substituir os blocos pela fatoração LU, sem alterar o outros trechos da matriz.

A base de trabalho é, então, utilizada para resolver os sistemas de equações do GRG. Por ter um tamanho reduzido e por sua estrutura, os cálculos tornam-se mais simples e estáveis, menos sujeitos a erros de arredondamento.

4.4 Programa Utilizado

Apesar das vantagens do GRECO em relação ao GRG para problemas de controle ótimo, optou-se por utilizar nos casos de teste desta dissertação o programa LSGRG2. Isto é explicado pelo fato do LSGRG2 estar disponível sob a forma de um programa comercial, acoplado ao *software* Microsoft Excel [14]. Além do algoritmo em si, o LSGRG2 do *solver* apresenta outras funcionalidades como derivadas analíticas de equações que utilizam funções do Excel, o que diminuiu em muito o tempo gasto no desenvolvimento dos casos de teste.

Entretanto, é intenção do autor dar continuidade ao trabalho desta dissertação, implementando uma versão do GRECO e utilizando-a para resolver o problema de *scheduling* aqui apresentado.

Capítulo 5

Heurística de Inicialização

Conforme apresentado por Morton e Pentico [28], heurísticas podem ser aplicadas a problemas de *scheduling*, obtendo pontos que violem nenhuma ou poucas restrições. Observando que o GRG poderia obter soluções melhores se inicializado com uma solução quase viável, foi implementada uma heurística.

A heurística procura alocar os tanques disponíveis mais adequados para os dutos no início do *scheduling*, sem se preocupar com o atendimento inicial dos navios. Isso é explicado pelo fato de que deve existir um conjunto mínimo de tanques para a operação simultânea de dutos e navios, senão o porto não conseguiria operar no dia a dia. Entretanto, não é possível atender *a priori* todos os dutos completamente, pois os níveis dos tanques vão variar conforme os navios sejam atendidos. Logo, a heurística deve tentar atender somente os dutos que forem possíveis inicialmente com os dados conhecidos no estado inicial do sistema.

Para os atendimento dos navios são escolhidos os píeres que estiverem inativos um tempo suficiente para desatracar um navio qualquer que houvesse sido anteriormente atendido. Os tanques devem estar em repouso, mas sem se preocupar com o *tempo de preparação*, para diminuir a possibilidade de algum navio não ser atendido pela heurística. Os tanques utilizados atendem os navios completamente, sem se preocupar com os limites operacionais dos mesmos. Isto porque, durante a otimização do sistema, as restrições

serão forçosamente atendidas.

Consolidando as considerações apresentadas anteriormente, foi construída a seguinte heurística:

Algoritmo 5.0.1 *Heurística H1*

(1) Para cada duto n faça:

$t = t_0$;

Se n é duto de recebimento do porto

S = conj dos tanques que recebem de n, em repouso em t_0;

Se S = $\emptyset$*: escolha outro duto: volte para (1);*

k = tanque de S que estiver mais cheio;

vol = $vol_k(t)$ - $vol_{min,k}$;

Senão

S = conj dos tanques que enviam para n, em repouso em t_0;

Se S = $\emptyset$*: escolha outro duto: volte para (1);*

k = tanque de S que estiver mais vazio;

vol = $vol_{max,k}$ - $vol_k(t)$;

Fim Se

Enquanto $t < tf$ *e* $vol > 0$ *faça:*

$u_{k,n}(t) = \min(u_{max,k,n}, \frac{vol}{\Delta t})$;

vol = vol - $u_{k,n}(t) * \Delta t$;

$t = t + 1$;

Fim Enquanto

Fim Para

L = lista de navios n ordenada por data de chegada at_n;

(2) Para cada navio n de L faça:

$t = at_n$;

Se $\nexists$ *p conectado a n tal que* $P_p = \{t_p | t_p \geq t, \int_{t_p - T_p}^{t_p} u_p(t)dt = 0\} \neq \emptyset$

Escolha outro navio: volte para (2);

Fim Se

(3) Escolha píer p tal que $t_p = \min\{t_p \in P_p\}$*;*

$t = t_p$*;*

$m = \lceil \frac{vol_n}{\Delta t} \rceil + t$ *,onde* vol_n *é o volume a ser processado pelo navio n;*

Se n é navio a ser descarregado

S = conjunto dos tanques em repouso em t que podem descarregar n pelo píer p;

Se $S = \emptyset$*: escolha outro píer: volte para (3);*

k = tanque de S que estiver mais vazio;

Senão

S é conjunto dos tanques em repouso em t que podem carregar n pelo píer p;

Se $S = \emptyset$*: escolha outro píer: volte para (3);*

k = tanque de S que estiver mais cheio;

Fim Se

Enquanto $t \leq m$ *e* $vol_n > 0$ *faça:*

$u_{k,n}(t) = \min(u_{max,k,n}, \frac{vol}{\Delta t})$*;*

$voln = vol_n$ - $u_{k,n}(t) * \Delta t$*;*

$t = t + 1$*;*

Fim Enquanto

Fim Para

É importante destacar que a heurística de inicialização se propõe a encontrar um ponto próximo à viabilidade, facilitando a etapa de otimização com o GRG. Portanto, a existência de restrições violadas não é um ponto impeditivo. A grande vantagem de utilizar uma heurística para inicializar o problema é que ela tem um custo computacional reduzido comparado ao da etapa de otimização.

A partir do ponto obtido pela heurística, pode-se aplicar um rearranjo manual das transferências, obtendo um *scheduling* viável. Para isso, deve-se observar quais as

transferências que fazem os tanques operarem fora de seus limites. Estas devem ser divididas em mais tanques, sempre respeitando seus limites de operação. Os navios que porventura não consigam ser atendidos ou o sejam sem respeitar o tempo necessário de desatracação, devem ser movidos para intervalos de tempo posteriores, quando houver píeres vagos para eles. E, finalmente, os dutos devem ser atendidos em intervalos de tempo onde haja tanques disponíveis para tal.

Durante a etapa de testes, foram utilizados pontos obtidos pela heurística e pontos obtidos a partir do rearranjo manual de pontos obtidos pela heurística.

Capítulo 6

Resultados Computacionais

Para validar o modelo, foram construídos cinco casos de testes, com dificuldade crescente de resolução. Desta forma foi possível acompanhar o comportamento do algoritmo conforme os casos iam se tornando mais próximos das condições reais de trabalho (Tabelas 6.1 e 6.2).

Todas as variáveis de controle e estado são limitadas inferior e superiormente. O primeiro caso foi resolvido com o GRG2 do Solver do *software* comercial Microsoft Excel [15]. Os outros foram resolvidos pela versão 6.0 do Solver da Frontline Systems para MS Excel [14], que implementa o LSGRG2. Todos os casos foram executados em um desktop AMD XP 2.6 GHz, com 512MB de memória RAM.

C. Teste	Equip.	P. Inicial	Solução	Dimensão
1(a) 48h $\Delta t = 4$h	Tanques: 2 Navios: 2 Píeres: 1 Dutos: -	F.Perform.: $1.4 * 10^6$ Viável: Não Sem heurística	F.Perform.: 7.66 Viabilidade(iter.): 63 Otimização(iter.): 6 tempo(s): 10	Controle: 48 Estado: 56 Lineares: 54
1(b) 48h $\Delta t = 4$h	Tanques: 2 Navios: 2 Píeres: 1 Dutos: -	F.Perform.: 4.47 Viável: Sim Heurística	F.Perform.: 1.54 Viabilidade(iter.): 0 Otimização(iter.): 8 tempo(s): < 1	Controle: 48 Estado: 56 Lineares: 54
2 39h $\Delta t = 3$h	Tanques: 3 Navios: 3 Píeres: 1 Dutos: 1	F.Perform.: 247.79 Viável: Não Heurística	F.Perform.: 7.60 Viabilidade(iter.): 12 Otimização(iter.): 8 tempo(s): 2	Controle: 126 Estado: 92 Lineares: 88
3 39h $\Delta t = 3$h	Tanques: 3 Navios: 4 Píeres: 2 Dutos: 1	F.Perform.: 246.89 Viável: Não Heurística	F.Perform.: 5.65 Viabilidade(iter.): 36 Otimização(iter.): 1 tempo(s): 8	Controle: 196 Estado: 316 Lineares: 84

Tabela 6.1: Casos de Teste de 1 a 3

6.1 Caso 1

O caso consistiu em realizar o descarregamento de dois navios de cru, em um porto composto por apenas um píer e dois tanques de cru, com um horizonte de programação de 48 horas, dividido em intervalos de 4 horas. Ambos os tanques podem receber as cargas de ambos os navios, bem como o píer permite a atracação dos mesmos. Como não havia dutos no cenário de testes, os custos de interface, desvio de planejamento e as restrições de planejamento não foram calculados (p4 = p5 = 0), enquanto os outros custos foram ponderados conforme a Tabela 6.3.

Todas as restrições foram avaliadas, exceto as de propriedades, pois se sabia de

antemão que elas seriam respeitadas em qualquer solução viável: os dois navios seriam descarregados nos tanques, e todos tinham o mesmo tipo de petróleo, logo, não haveria como obter valores inviáveis de propriedades. Como as restrições de atendimento de navios foram incluídas no caso de teste, o custo de perda de navios não o foi na função-objetivo.

O ponto inicial fornecido ao algoritmo era completamente inviável, considerando vazões com valor de $1.0 * 10^3 m^3/\text{h}$, sem atender as restrições de volume dos tanques e dos navios. Os tanques extravasavam enquanto os navios eram descarregados simultaneamente, atingindo volumes negativos ao final do cenário (Figura 6.1).

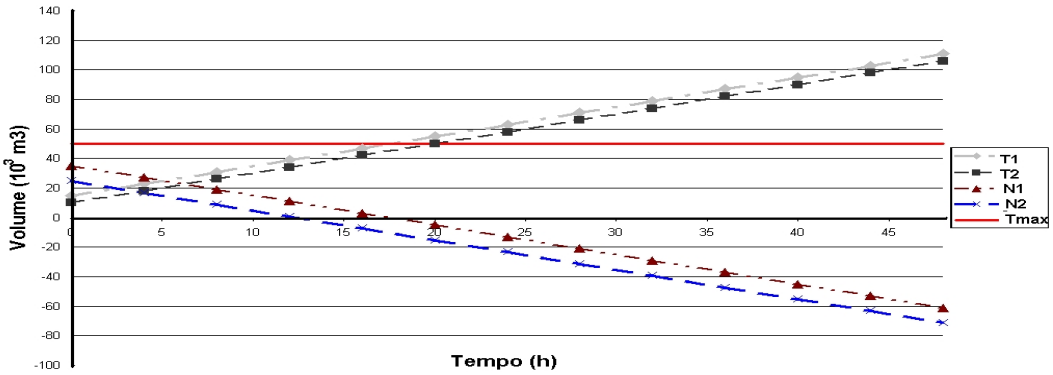

Figura 6.1: Inventário do ponto inicial inviável no Caso 1

O GRG convergiu em 69 iterações, sendo que 63 foram despendidas para obter o primeiro ponto viável e 6 para encontrar um ótimo local. O valor da função de performance era inicialmente $1.45 * 10^6$no ponto inicial inviável, 1631.91 no primeiro ponto viável encontrado e 7.66 no ótimo local.

As Figuras 6.2 e 6.3 apresentam a solução encontrada pelo GRG, onde ambos os navios são descarregados no prazo, sem incorrer em multa, e ambos os tanques se mantêm nos limites operacionais. Ao contrário do ponto inicial, não ocorreu sobreposição no descarregamento dos navios e o tempo de desatracação no píer foi respeitado. Pode-se

observar que os navios foram liberados somente na data programada de saída, evitando um maior custo de inventário.

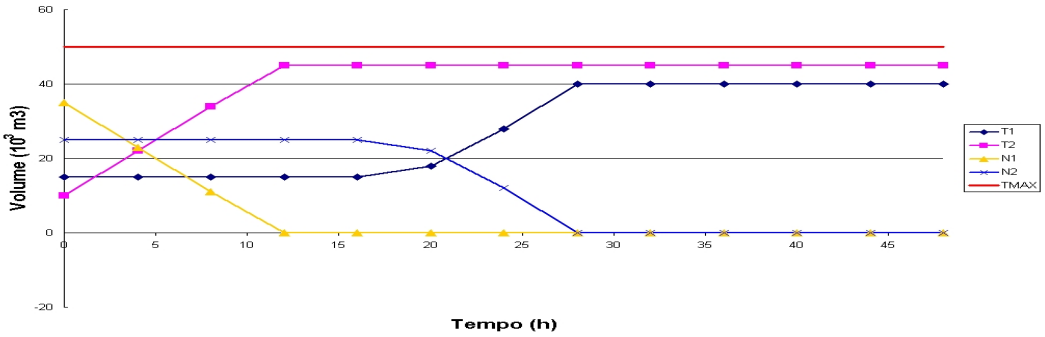

Figura 6.2: Inventário da solução do Caso 1

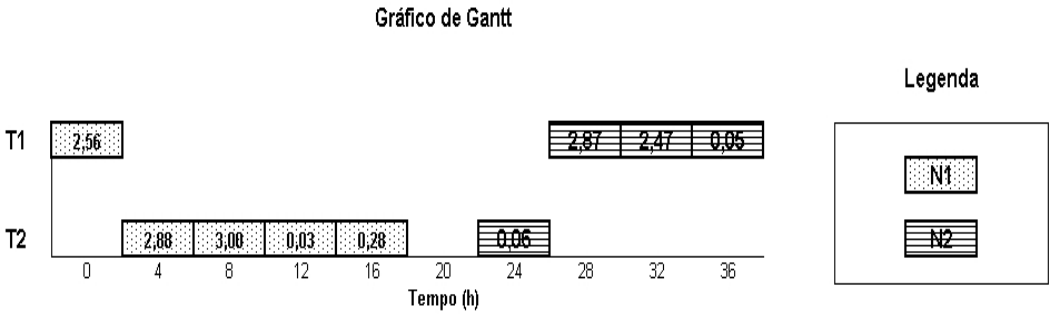

Figura 6.3: Gráfico de Gantt com vazões ($10^3 m^3$/h) da solução do Caso 1

Analisando a solução obtida no caso de teste, pode-se notar que o número de iterações necessárias para obter o primeiro ponto viável foi aproximadamente 10 vezes o número de iterações gastas na otimização dentro da região de viabilidade. Observando que o GRG poderia obter soluções melhores se iniciado com um *schedule* quase viável, foi fornecido um novo ponto inicial, obtido pela aplicação da heurística H1 (Figura 6.4).

A heurística H1 encontrou um ponto inicial viável, com função de performance 4.47. A partir desse ponto o GRG convergiu em 8 iterações para o ótimo local de valor

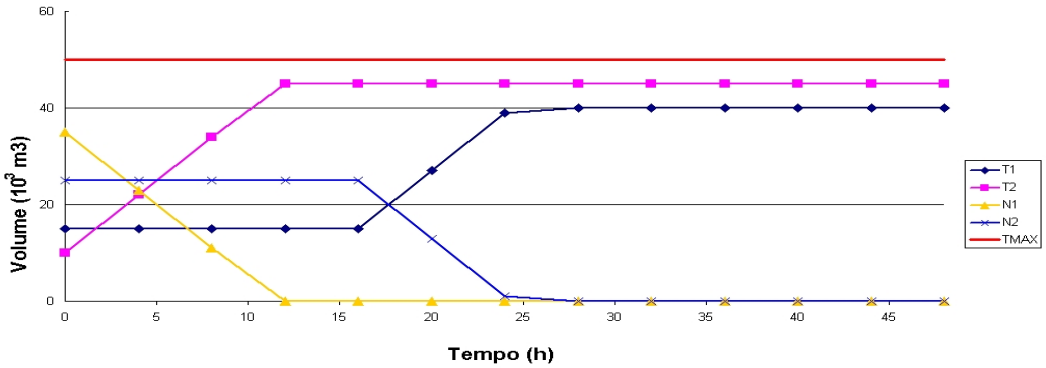

Figura 6.4: Inventário do ponto inicial obtido pela heurística no Caso 1

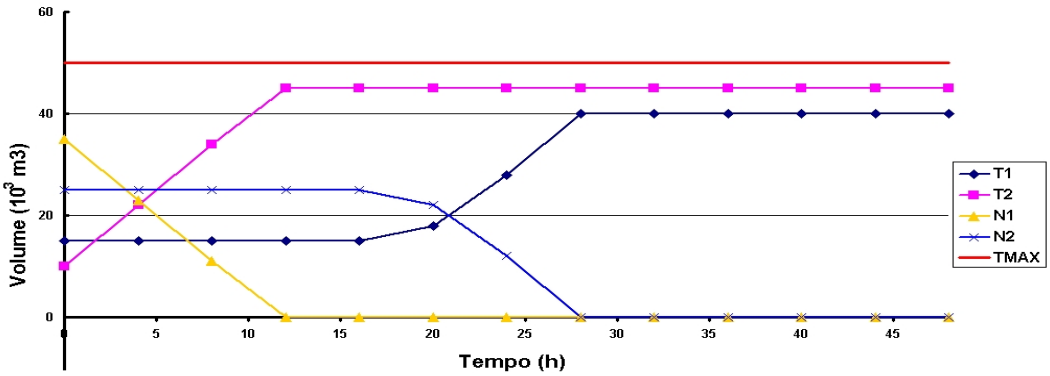

Figura 6.5: Inventário da solução do Caso 1 a partir da heurística

de performance 1.54 (Figuras 6.5 e 6.6).

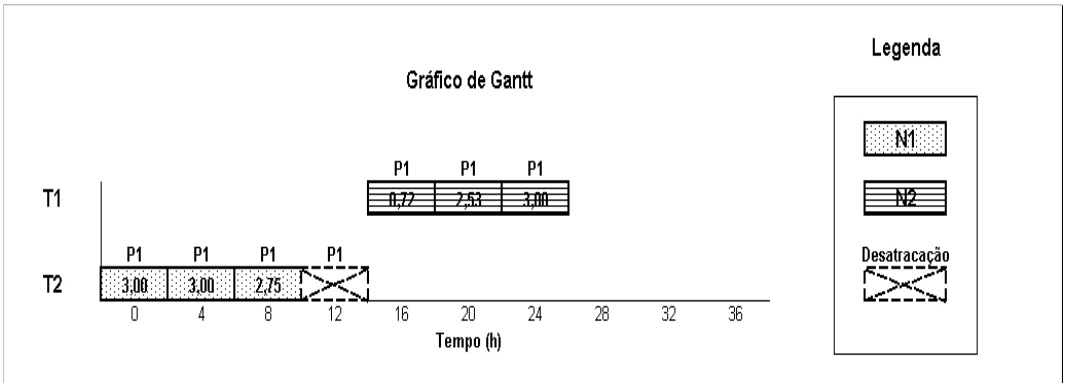

Figura 6.6: Gráfico de Gantt com vazões ($10^3 m^3$/h) da solução do Caso 1 a partir da heurística

6.2 Caso 2

O caso 2 apresentou um problema um pouco mais complicado que o do caso anterior, com atendimento de duto e descarregamento de navios. Foi aplicada a heurística para inicializar o problema, que obteve um ponto quase viável. Os navios foram atendidos no prazo, sem atraso, e nenhum tanque violou seus limites operacionais. Entretanto, o duto não foi atendido completamente, devido o fato da própria heurística não obrigar o atendimento completo de dutos.

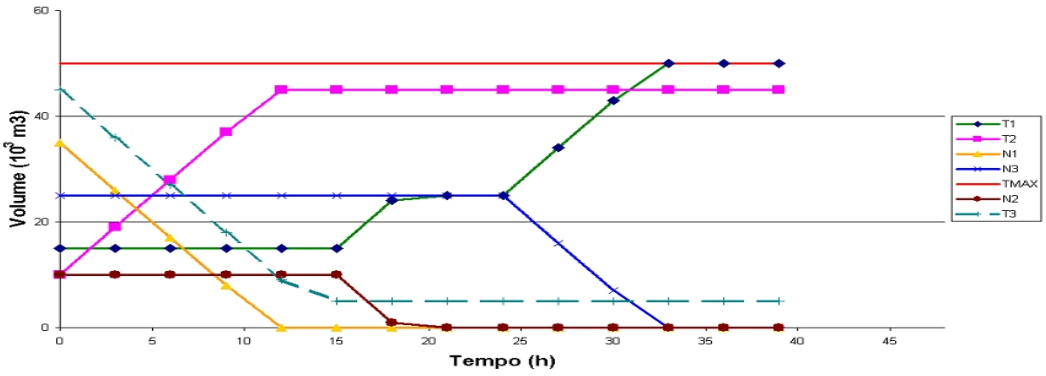

Figura 6.7: Inventário do ponto inicial inviável do Caso 2

O caso foi rodado com pesos diferentes do Caso 1, penalizando a função de

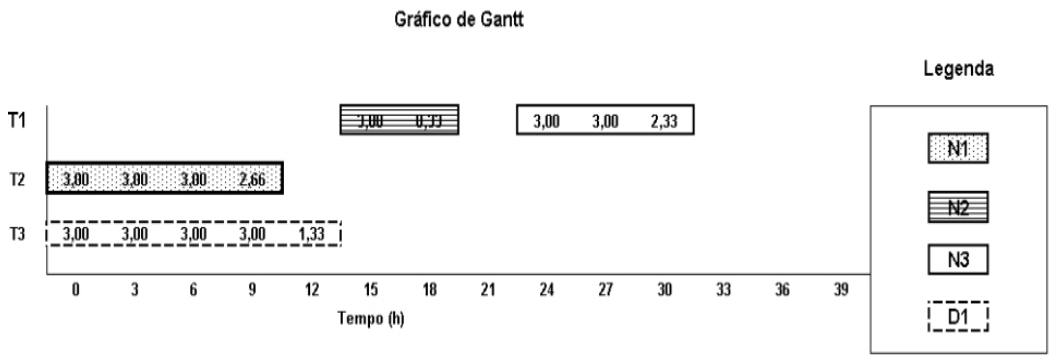

Figura 6.8: Gráfico de Gantt com vazões ($10^3 m^3$/h) no ponto inicial do Caso 2

performance com a incorporação da restrição de planejamento de dutos, conforme os pesos indicados na Tabela 6.4. Isto ocorreu para se obter uma melhor sensibilidade de qual seria o impacto na função-objetivo das penalidades por não atendimento de dutos. Entretanto, a restrição na formulação do problema foi mantida, pois direciona o método a atingir uma solução que obrigatoriamente atenda os dutos.

O LSGRG2 aplicou uma etapa Fase 1 para obter o primeiro ponto viável, despendendo apenas 12 iterações. A partir do primeiro ponto viável, foi encontrada um ótimo local em apenas 8 iterações, mantendo o atendimento dos navios sem atraso e conseguindo atender o duto completamente.

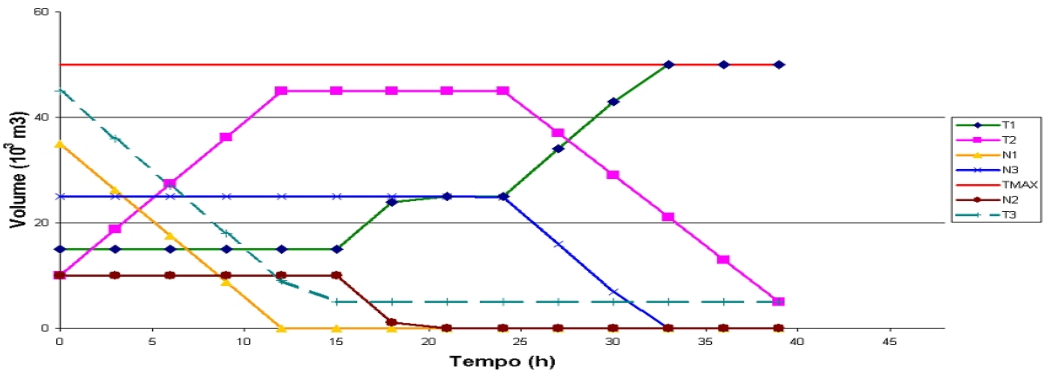

Figura 6.9: Inventário da solução do Caso 2

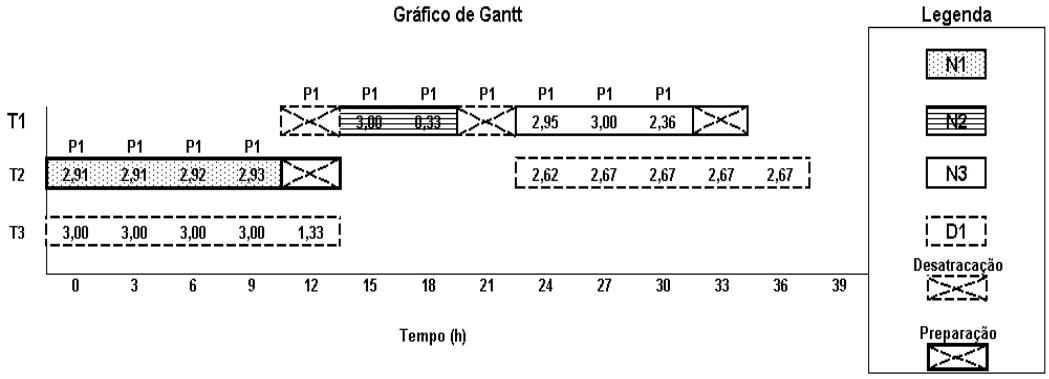

Figura 6.10: Gráfico de Gantt com vazões ($10^3 m^3$/h) da solução do Caso 2

Utilizou-se o *solver* com LSGRG2, já antecipando que o número de variáveis passaria a crescer ao ponto de não ser suportado pelo GRG *default* nos casos seguintes.

6.3 Caso 3

O caso 3 apresentou um problema um pouco mais complicado que o do caso anterior, com atendimento de duto e descarregamento e carregamento de navios em 2 píeres. Foi aplicada a heurística para inicializar o problema, que obteve uma solução inviável. Os navios foram atendidos no prazo, sem atraso, e nenhum tanque violou seus limites operacionais. Entretanto, o duto não foi atendido completamente, devido o fato da própria heurística não obrigar o atendimento completo de dutos. Considerando todos os intervalos de tempo, o modelo do caso 3 apresenta 196 variáveis de controle e 321 variáveis de estado e restrições, já não sendo um problema pequeno.

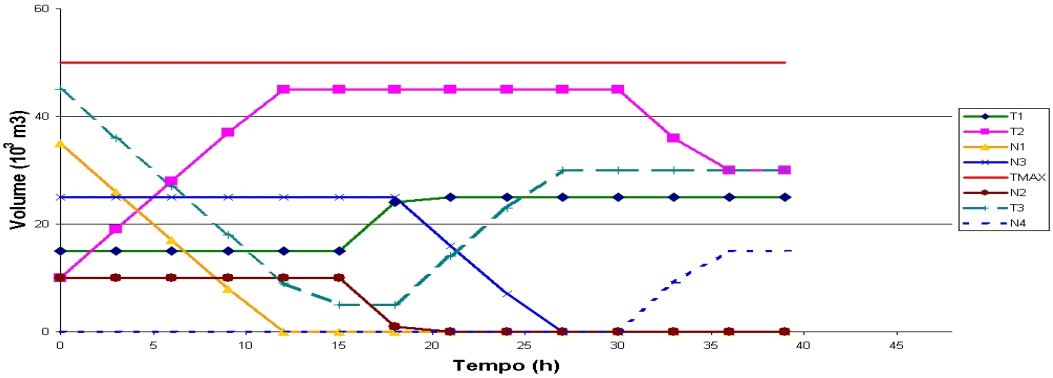

Figura 6.11: Inventário do ponto inicial inviável do Caso 3

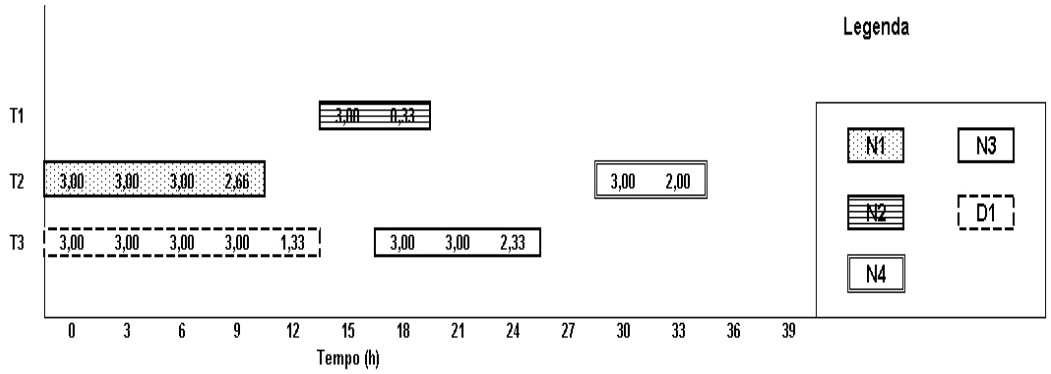

Figura 6.12: Gráfico de Gantt com vazões ($10^3 m^3$/h) no ponto inicial do Caso 3

O caso foi resolvido com as restrições de atendimento de dutos e atendimento de navios. Além disso, a primeira foi incorporada a função de *performance*, conforme a Tabela 6.4.

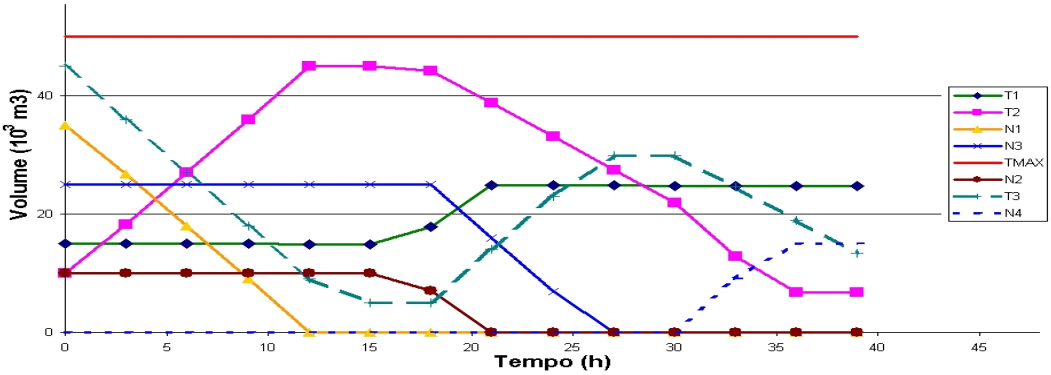

Figura 6.13: Inventário da solução do Caso 3

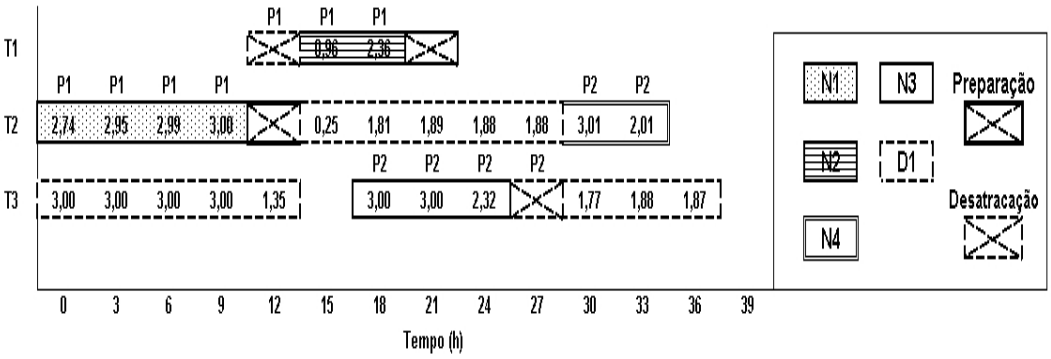

Figura 6.14: Gráfico de Gantt com vazões ($10^3 m^3$/h) da solução do Caso 3

O GRG rapidamente convergiu, e a solução encontrada atendeu todos os navios no prazo, o duto completamente e não violou qualquer limite operacional. Nesse caso, todas as iterações foram gastas para obter uma primeira solução viável, que, quando encontrada, já era um ótimo local, de modo que não houve novas iterações na região viável.

6.4 Caso 4

O caso 4, uma situação próxima da realidade do programador de produção (Tabela 6.5), onde o número de navios é grande para a quantidade de recursos disponíveis (tanques, píeres e dutos), de modo que uma solução obtida manualmente não é trivial e provavelmente longe de ótima. Considerando todos os intervalos de tempo, o modelo do caso 4 apresenta 650 variáveis de controle e 774 variáveis de estado e restrições, sendo um problema de porte razoável.

De fato, pode-se observar que o ponto fornecido pela heurística H1 só conseguiu atender todos os navios, ao atrasar dois deles, ou seja, não foi possível pela heurística obter um *schedule* que mantivesse os navios no prazo. Além disso, restrições de volume de tanques, tempo de preparação e de desatracação de píer foram violadas.

O caso 4 foi modelado com todas as restrições e os pesos utilizados conforme a Tabela 6.3. As restrições de atendimento de dutos e de navios não foram relaxadas na função objetivo. Pode-se notar que o objetivo mais importante era o atendimento dos navios no prazo, pois a solução inicial não o conseguia.

Como este caso era semelhante a uma situação real, decidiu-se observar o comportamento do modelo de acordo com diferentes pontos iniciais. Foram utilizados 4 diferentes pontos iniciais para o caso 4:

a) Ponto inicial obtido pela heurística H1: não houve atendimento completo do duto e dois navios foram atrasados. Neste caso, os limites de tanques foram desrespeitados para atender todos os navios (Figuras 6.15 e 6.16). Pode-se observar na Tabela 6.2 que o GRG despendeu um número considerável de iterações para obter a viabilidade e depois um número pequeno de iterações para convergir ao ótimo local. Na solução encontrada nenhum navio deixou de ser atendido, todos os limites operacionais foram respeitados e o duto atendido. Entretanto, um dos navios continuou

ligeiramente atrasado (Figuras 6.17 e 6.18).

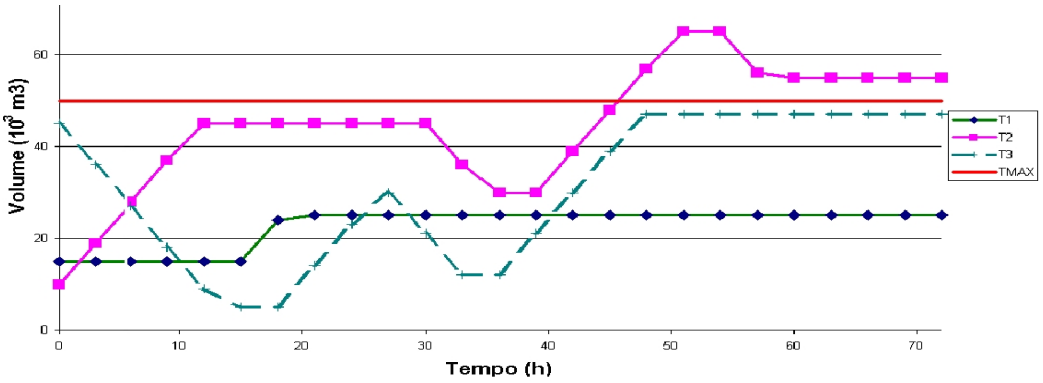

Figura 6.15: Inventário do ponto inicial inviável do Caso 4(a)

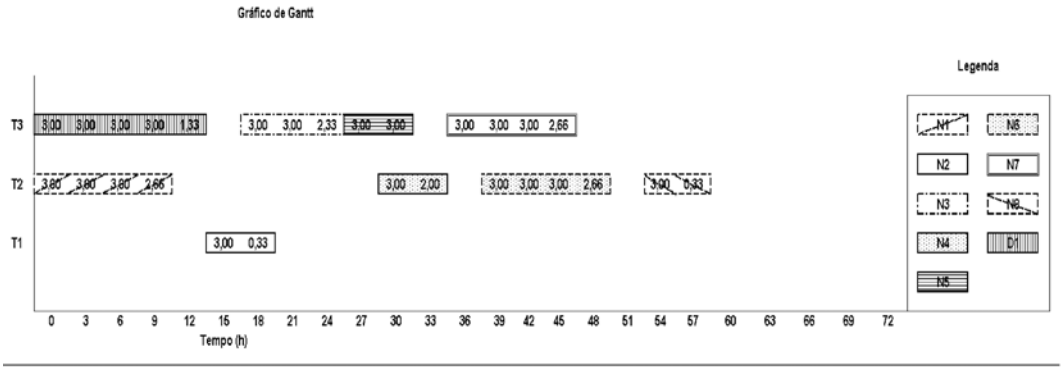

Figura 6.16: Gráfico de Gantt com vazões ($10^3 m^3$/h) no ponto inicial do Caso 4(a)

b) Modificação do ponto obtido pela heurística: houve atendimento completo do duto e somente um navio foi atrasado. Os limites operacionais foram respeitados (Figuras 6.19 e 6.20). Pode-se observar na Tabela 6.2 que o GRG despendeu um número pequeno de iterações para convergir ao ótimo local. A solução encontrada gerou um *schedule* muito semelhante ao do ponto inicial, onde nenhum navio deixou de ser atendido, todos os limites operacionais foram respeitados e o duto atendido. Somente as vazões foram modificadas, diminuindo o custo de variação de vazões, mas quase não modificando o de *demurrage*, pois o navio que estava atrasado, con-

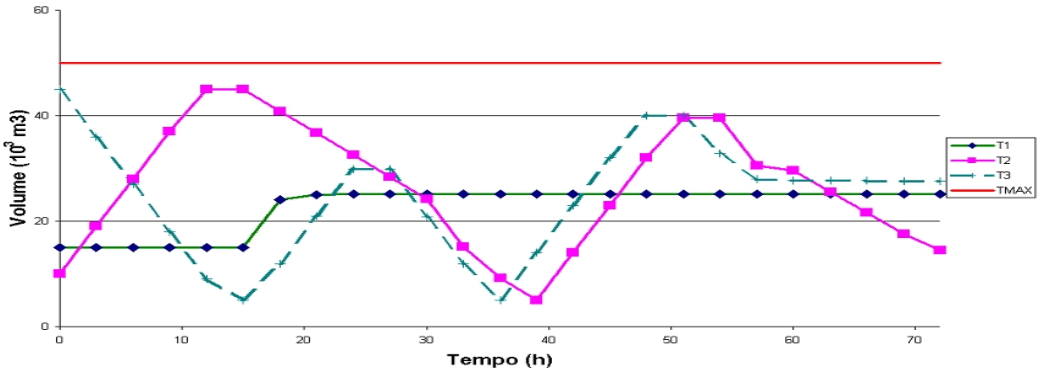

Figura 6.17: Inventário da solução do Caso 4(a)

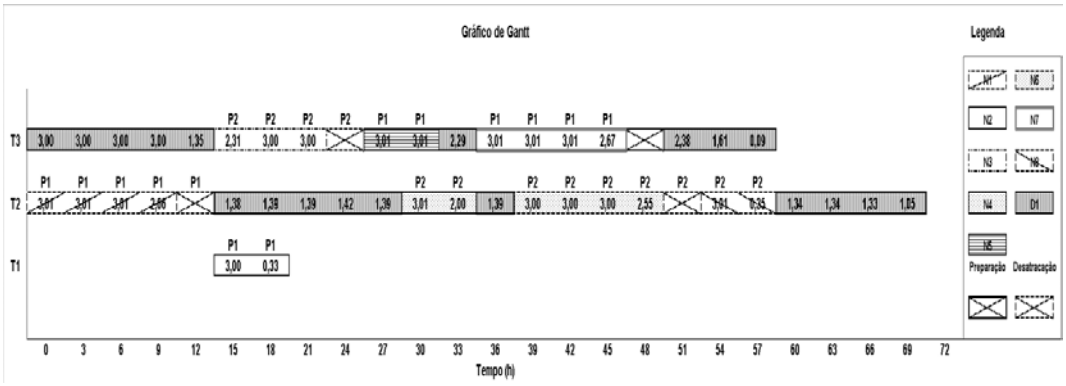

Figura 6.18: Gráfico de Gantt com vazões ($10^3 m^3$/h) da solução do Caso 4(a)

tinuou assim (Figuras 6.21 e 6.22).

c) Modificação do ponto obtido pela heurística: Apesar da configuração diferente obtida no ponto inicial, o comportamente foi semelhante ao do caso 4(b): somente um navio atrasou, todos os limites operacionais foram respeitados (Figuras 6.23 e 6.24). Pode-se observar na Tabela 6.2 que o GRG despendeu um número pequeno de iterações para convergir ao ótimo local. A solução encontrada gerou um *schedule* muito semelhante ao do ponto inicial, onde nenhum navio deixou de ser atendido, todos os limites operacionais foram respeitados e o duto atendido. Somente as vazões foram modificadas, diminuindo o custo de variação de vazões, mas quase não modificando o de *demurrage*, pois o navio que estava atrasado, continuou as-

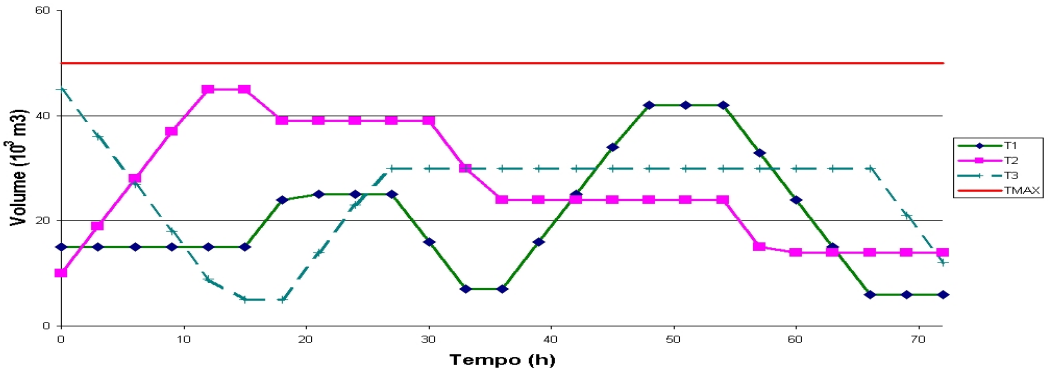

Figura 6.19: Inventário do ponto inicial do Caso 4(b)

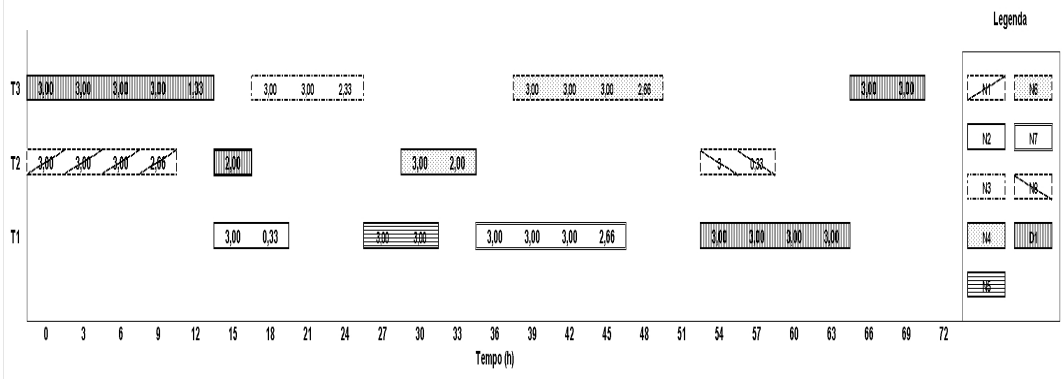

Figura 6.20: Gráfico de Gantt com vazões ($10^3 m^3$/h) no ponto inicial do Caso 4(b)

sim (Figuras 6.25 e 6.26).

d) Modificação do ponto obtido pela heurística: Neste caso foi fornecido um ponto inicial com atraso de dois navios, sendo que um deles, era parcialmente atendido, depois liberado do porto e finalmente completado no final (Figuras 6.27 e 6.28). Os limites operacionais foram todos respeitados. O GRG conseguiu convergir em um número de iterações compatível com o do caso 4(a) para uma solução muito próxima das obtidas nos outros casos, com apenas um navio atrasando (Figuras 6.29 e 6.30).

Pode-se observar a partir do Caso 4 que o modelo e o método foram capazes

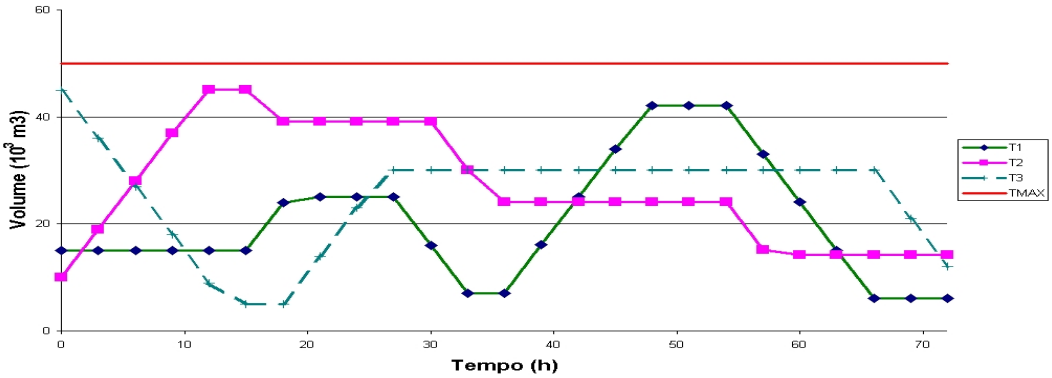

Figura 6.21: Inventário da solução do Caso 4(b)

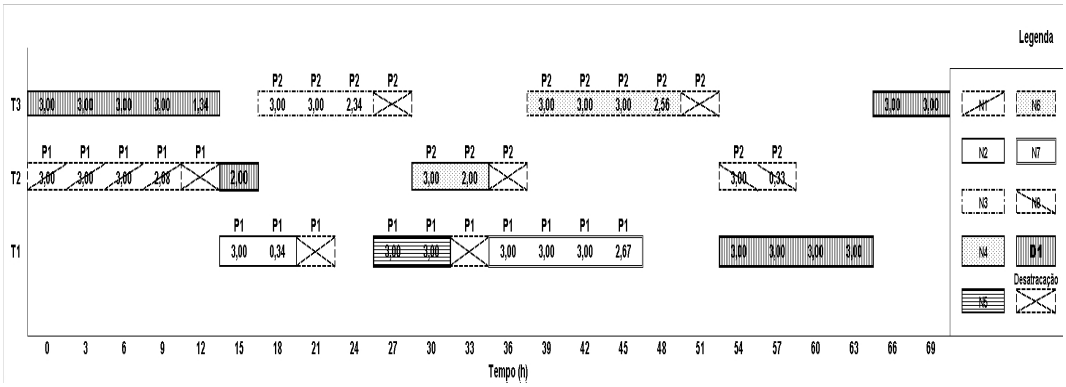

Figura 6.22: Gráfico de Gantt com vazões ($10^3 m^3$/h) da solução do Caso 4(b)

de rapidamente melhorar as soluções já existentes e convergir para soluções muito boas na prática. Deve-se destacar que esta característica é das mais importantes para o uso em problemas reais.

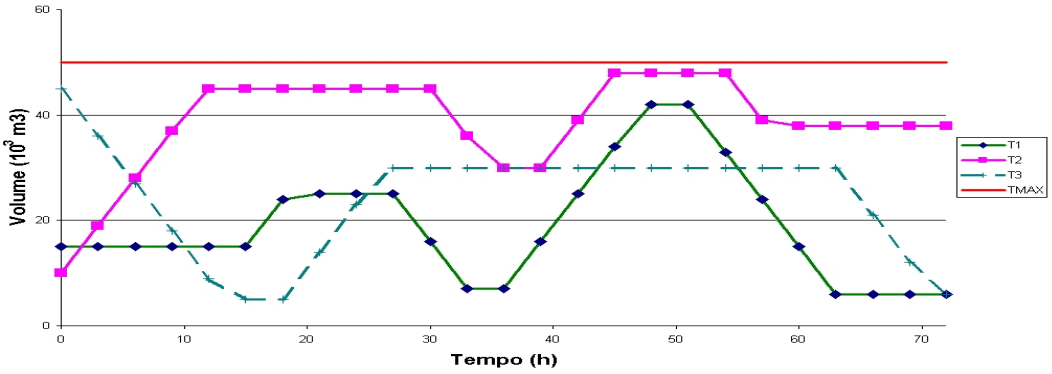

Figura 6.23: Inventário do ponto inicial do Caso 4(c)

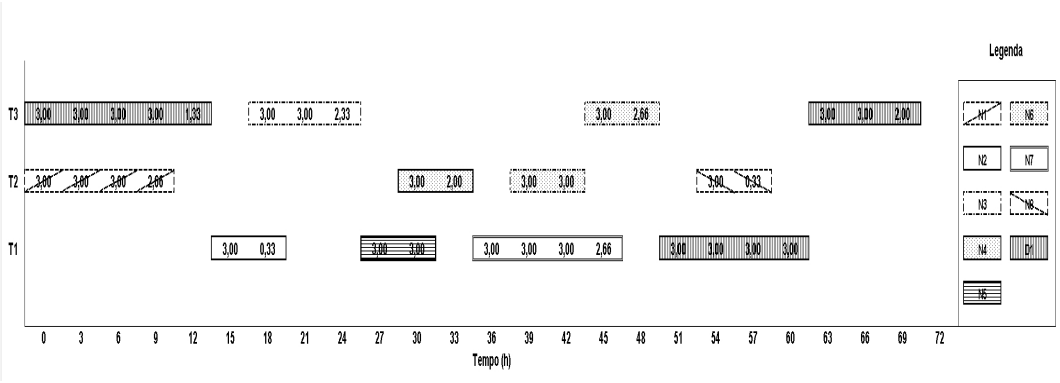

Figura 6.24: Gráfico de Gantt com vazões ($10^3 m^3$/h) no ponto inicial do Caso 4(c)

6.5 Caso 5

O caso 5 apresenta uma modificação do caso 4: a adição de mais um duto. Para manter o mesmo número de variáveis, a conexão entre o tanque T3 e o navio N6 foi removida do problema e adicionada uma conexão entre o duto D2 e o tanque T3. A inclusão desta conexão foi obrigar o algoritmo a encontrar uma solução que utilizasse o tanque T3, mesmo sendo ele bastante utilizado para atendimento dos navios e do duto D1 nas soluções do caso 4. O duto D1 exigiu uma movimentação total de $60 * 10^3 m^3$ ao longo do cenário, enquanto o duto D2 um total de $40 * 10^3 m^3$.

O ponto inicial fornecido pela heurística H1 era inviável, pois não atendia o duto

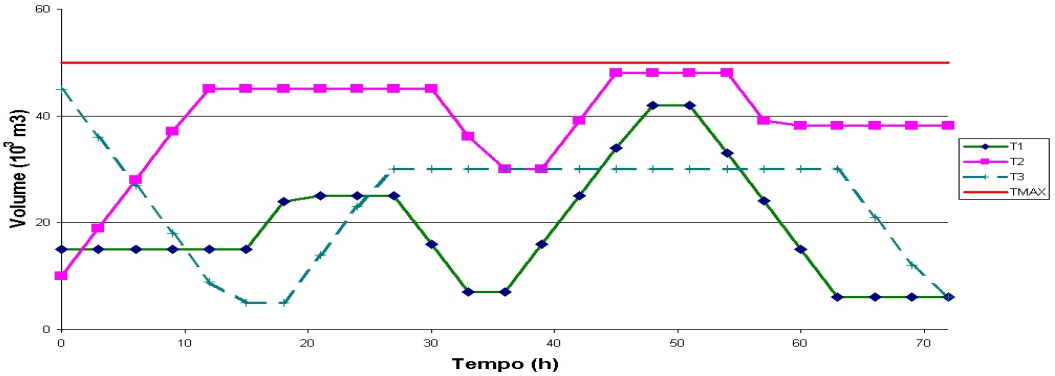

Figura 6.25: Inventário da solução do Caso 4(c)

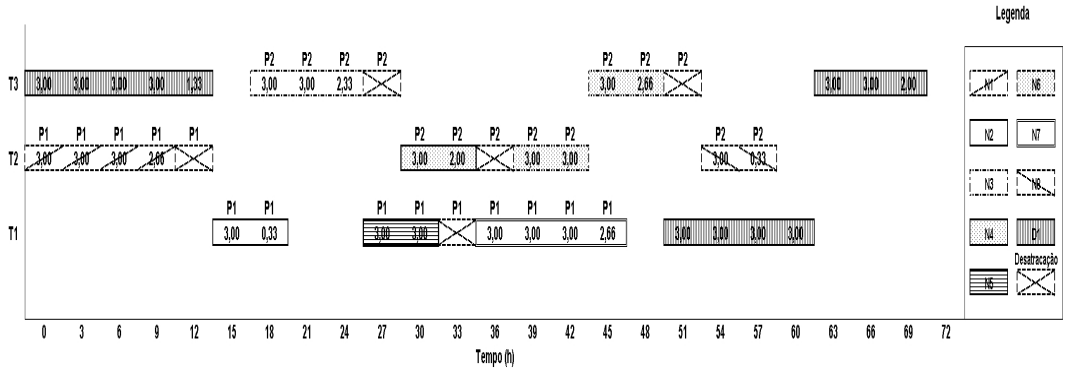

Figura 6.26: Gráfico de Gantt com vazões ($10^3 m^3$/h) da solução do Caso 4(c)

D1 e não respeitava os tempos de preparação nem os limites operacionais dos tanques. Os pesos utilizados na função-objetivo estão apresentados na Tabela 6.3. Considerando todos os intervalos de tempo, o modelo do caso 5 apresenta 650 variáveis de controle e 774 variáveis de estado e restrições, sendo um problema de porte razoável.

O GRG despendeu 120 iterações para obter o primeiro ponto viável (função de *performance* $3.3 * 10^3$) e mais 380 para convergir a um ótimo local. O grande número de iterações pode ser um indicativo de que o espaço viável de soluções se encontra bastante restrito. Uma forma de relaxar o conjunto de restrições e permitir um maior espaço viável para a busca do GRG seria incorporar parte das restrições sob forma de penalidade na função objetiva.

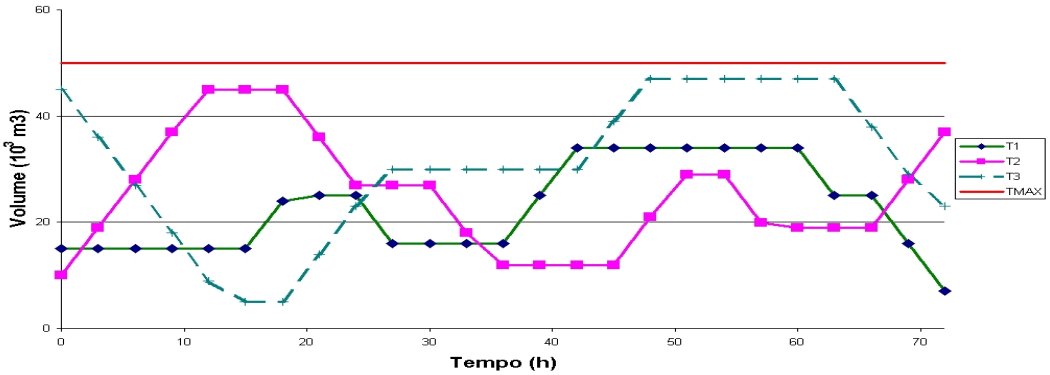

Figura 6.27: Inventário do ponto inicial do Caso 4(d)

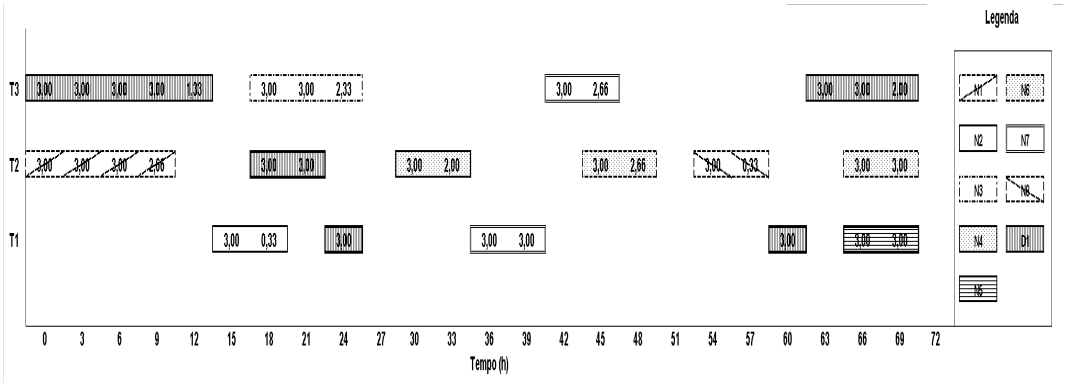

Figura 6.28: Gráfico de Gantt com vazões ($10^3 m^3$/h) no ponto inicial do Caso 4(d)

Pode-se notar que o ótimo encontrado apresenta um valor praticamente idêntico ao do ponto inicial inviável, pois mantém o atraso de um dos navios (N6). Isso é um indício de que dificilmente haveria uma solução sem atrasos e que valores menores de *performance* só ocorreriam em *schedules* inviáveis.

A partir do caso 5 pode-se concluir que, mesmo em casos de difícil obtenção de viabilidade, o GRG é capaz de obter bons pontos viáveis e convergir para um ótimo em pouco tempo computacional. Isso é importante para o uso em situações reais, quando procura-se encontrar soluções boas para problemas difíceis em pouco tempo computacional.

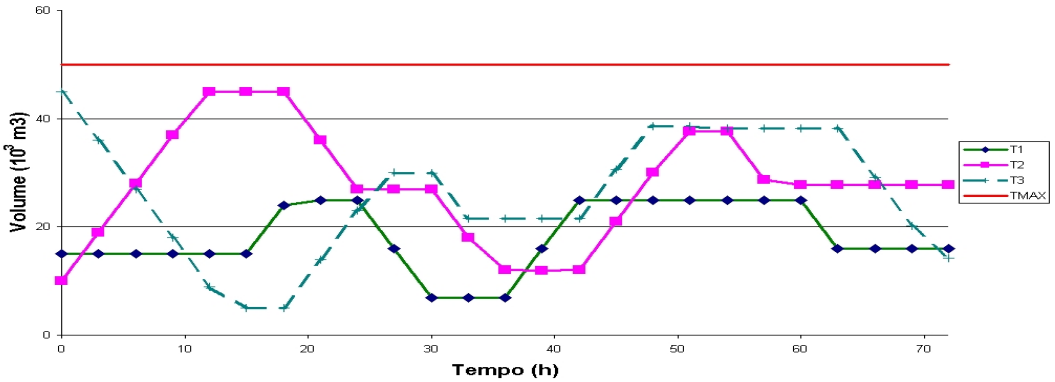

Figura 6.29: Inventário da solução do Caso 4(d)

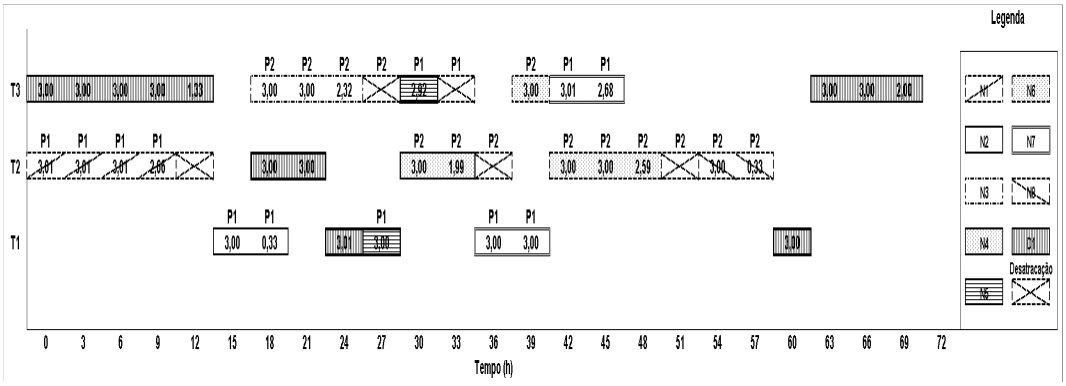

Figura 6.30: Gráfico de Gantt com vazões ($10^3 m^3$/h) da solução do Caso 4(d)

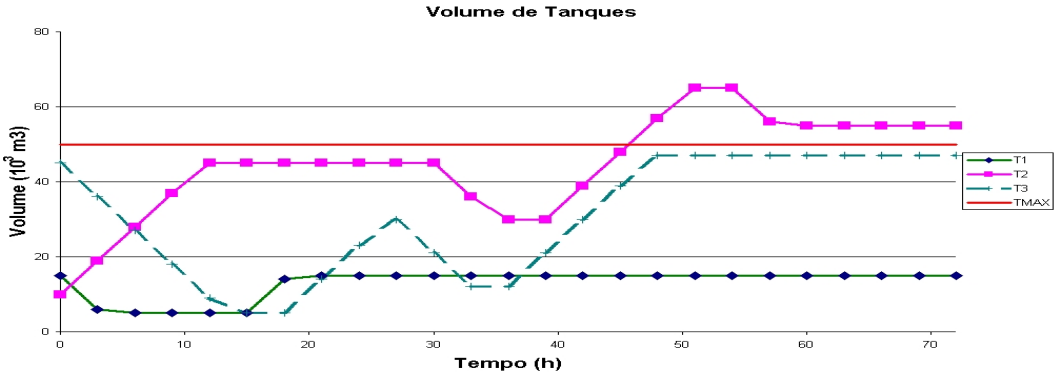

Figura 6.31: Inventário do ponto inicial do Caso 5

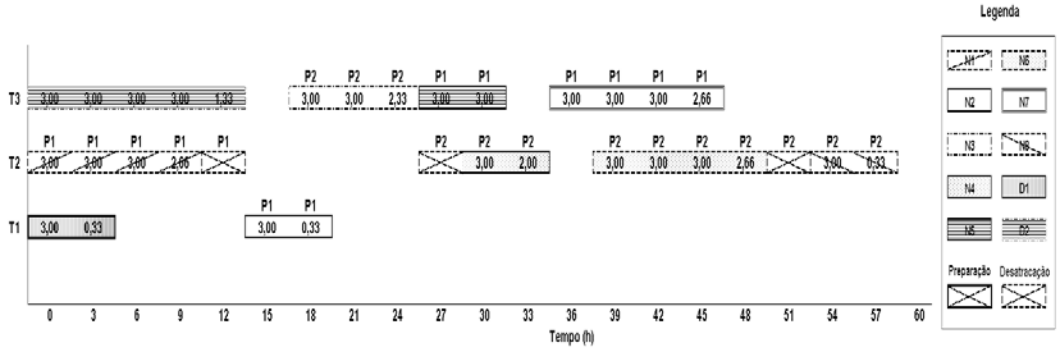

Figura 6.32: Gráfico de Gantt com vazões ($10^3 m^3$/h) no ponto inicial do Caso 5

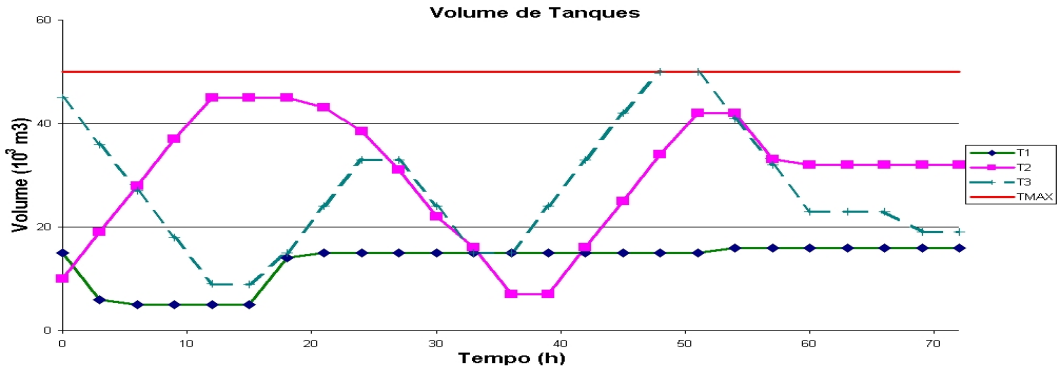

Figura 6.33: Inventário da solução do Caso 5

C. Teste	**Equip.**	**P. Inicial**	**Solução**	**Dimensão**
4(a)	Tanques: 3	F.Perform.: 331.89	F.Perform.: 314.32	Controle: 650
72h	Navios: 8	Viável: Não	Viabilidade(iter.): 80	Estado: 774
$\Delta t = 3$h	Píeres: 2	Heurística	Otimização(iter.): 221	Lineares: 198
	Dutos: 1		tempo(s): 85	
4(b)	Tanques: 3	F.Perform.: 340.99	F.Perform.: 313.24	Controle: 650
72h	Navios: 8	Viável: Sim	Viabilidade(iter.): 0	Estado: 774
$\Delta t = 3$h	Píeres: 2	Rearranjo Heurística	Otimização(iter.): 46	Lineares: 198
	Dutos: 1		tempo(s): 12	
4(c)	Tanques: 3	F.Perform.: 343.14	F.Perform. 316.57	Controle: 650
72h	Navios: 8	Viável: Sim	Viabilidade(iter.): 0	Estado: 774
$\Delta t = 3$h	Píeres: 2	Rearranjo Heurística	Otimização(iter.): 50	Lineares: 198
	Dutos: 1		tempo(s): 11	
4(d)	Tanques: 3	F.Perform.: $2.9 * 10^5$	F.Perform.: 346.41	Controle: 650
72h	Navios: 8	Viável: Sim	Viabilidade(iter.): 0	Estado: 774
$\Delta t = 3$h	Píeres: 2	Rearranjo Heurística	Otimização(iter.): 250	Lineares: 198
	Dutos: 1		tempo(s): 36	
5	Tanques: 3	F.Perform.: 331.97	F.Perform.: 331.65	Controle: 650
72h	Navios: 8	Viável: Não	Viabilidade(iter.): 120	Estado: 774
$\Delta t = 3$h	Píeres: 2	Heurística	Otimização(iter.): 380	Lineares: 198
	Dutos: 2		tempo(s): 90	

Tabela 6.2: Casos de Teste 4 e 5

Custo	Peso	Valor
Demurrage	p1	0.59
Variação de Vazão	p2	0.40
Inventário	p3	0.01
Interface	p4	0
Dutos	p5	0
Navios	p6	0

Tabela 6.3: Pesos para os Casos 1, 4 e 5

Custo	Peso	Valor
Demurrage	p1	0.5
Variação de Vazão	p2	0.1
Inventário	p3	0.1
Interface	p4	0
Dutos	p5	0.3
Navios	p6	0

Tabela 6.4: Pesos para os Casos 2 e 3

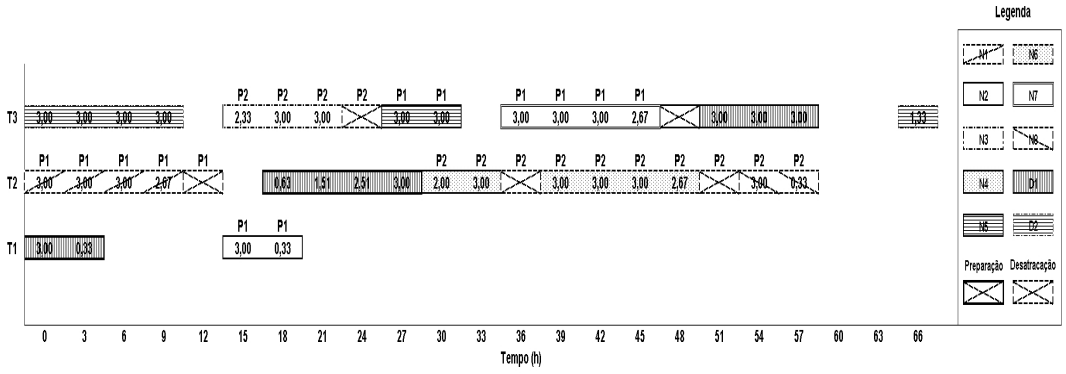

Figura 6.34: Gráfico de Gantt com vazões ($10^3 m^3$/h) da solução do Caso 5

Parâmetro					
Tanque	Vol_{min} (10^3 m^3)	Vol_{max} (10^3 m^3)	Vol_0 (10^3 m^3)	C_{invent}(10^3 \$/m^3)	Tv (h)
T1	5	50	15	0.05	3
T2	5	50	10	0.05	3
T3	5	50	45	0.05	3
Navio	$Vol_{inicial}$ (10^3 m^3)	Vol_{final} (10^3 m^3)	$at(h)$ / $lt(h)$	C_{demur}(10^3 \$/dia)	
N1	0	35	0/24	48	
N2	0	10	8/27	48	
N3	0	25	12/36	48	
N4	15	0	21/37	48	
N5	18	0	27/42	48	
N6	0	35	33/47	48	
N7	0	35	33/47	48	
N8	10	0	51/62	48	
Duto	Mm (10^3 m^3)				
D1	100				
Píeres	$Tp(h)$				
P1	3				
P2	3				
P3	3				
Vazões	u_{max} (10^3 m^3/h)	**Vazões**	u_{max} (10^3 m^3/h)	**Vazões**	u_{max} (10^3 m^3/h)
$U_{T1\text{-}D1}$	3	$U_{T2\text{-}P1\text{-}N2}$	3	$U_{T3\text{-}P2\text{-}N3}$	3
$U_{T2\text{-}D1}$	3	$U_{T2\text{-}P1\text{-}N1}$	3	$U_{T3\text{-}P2\text{-}N4}$	3
$U_{T3\text{-}D1}$	3	$U_{T2\text{-}P1\text{-}N3}$	3	$U_{T3\text{-}P2\text{-}N6}$	3
$U_{T1\text{-}P1\text{-}N1}$	3	$U_{T3\text{-}P1\text{-}N5}$	3	$U_{T3\text{-}P2\text{-}N7}$	3
$U_{T1\text{-}P1\text{-}N3}$	3	$U_{T3\text{-}P1\text{-}N6}$	3	$U_{T2\text{-}P2\text{-}N4}$	3
$U_{T1\text{-}P1\text{-}N2}$	3	$U_{T3\text{-}P1\text{-}N7}$	3	$U_{T2\text{-}P2\text{-}N8}$	3
$U_{T1\text{-}P1\text{-}N5}$	3	$U_{T3\text{-}P1\text{-}N8}$	3	$U_{T2\text{-}P2\text{-}N5}$	3
$U_{T1\text{-}P1\text{-}N6}$	3	$U_{T1\text{-}P2\text{-}N4}$	3	$U_{T2\text{-}P2\text{-}N6}$	3
$U_{T1\text{-}P1\text{-}N7}$	3	$U_{T3\text{-}P2\text{-}N2}$	3		

Tabela 6.5: Equipamentos do Caso de Teste 4

Capítulo 7

Conclusão

Os modelos apresentados se mostraram capazes de representar o problema utilizando um número pequeno de variáveis e de restrições, sem utilizar variáveis de decisão binárias. Em modelos tradicionais de programação misto-inteira, como, por exemplo, os apresentados por Más [26, 27] e Rejowski e Pinto [36], haveria para cada variável de controle pelo menos uma variável de decisão associada. Um exemplo é o Problema 1 apresentado por Más [27], composto por um cenário de 7 dias, 13 petroleiros, 7 classes de petróleo, 4 píeres, 17 tanques e 2 dutos: apresenta 1093 variáveis binárias, 1996 contínuas e 7093 restrições. Comparando com os modelos caixa-preta das abordagens de algoritmos genéticos, como os apresentados por Almeida [2], Cruz [6] e Simão [38], pode-se observar que a otimização do sistema de controle ótimo aparentemente requer menos iterações para convergir a um ótimo, utilizando o GRG. Uma linha futura de pesquisa deve ser a associação do o GRG a métodos metaheurísticos na busca da solução ótima global (ou quase). O GRG realizaria uma etapa de busca local, enquanto outro algoritmo faria a busca global pela região de viabilidade.

Como um número pequeno de variáveis contínuas foi suficiente para descrever os sistemas, a obtenção de um ótimo local na programação *a priori* ou a simples melhoria de uma programação já existente podem ser obtidas requerendo apenas uma rápida configuração do sistema, levantando um conjunto pequeno de dados de operação. A etapa de otimização também ocorre rapidamente a partir de um ponto viável ou obtido pela

heurística H1 (algoritmo 5.0.1).

É importante notar que não foram realizados testes comparativos ou *benchmarks* em termos de tempo computacional, complexidade de algoritmo ou consumo de CPU, de modo que não se pode afirmar que um método é melhor que outro. Contudo, pode-se afirmar que a modelagem e o método de solução apresentados nesta dissertação são bastante eficientes e podem obter respostas em tempo computacional reduzido.

A função de *performance* utilizada pelo modelo considera uma série de custos de acordo com seus graus de importância, podendo ser adaptada para cada caso que se queira resolver. Esta flexibilidade permite que as soluções sejam encontradas privilegiando os aspectos de maior interesse do programador de produção no momento da formulação do *scheduling*.

Foi possível observar que o algoritmo converge para um ótimo local, mesmo quando iniciado em um ponto distante da região de viabilidade em problemas de pequeno e médio porte. Entretanto devido à natureza não convexa do problema é importante iniciar a otimização a partir de um *schedule* viável ou levemente inviável. É recomendado o uso de heurísticas, pois fornecem bons pontos e têm custo computacional baixo. A heurística apresentada nesta dissertação se mostrou eficiente, fornecendo *schedules* viáveis ou levemente inviáveis.

O uso de métodos eficientes de Programação Não Linear como o LSGRG2 (*Large Scale GRG*) permite que sejam resolvidos problemas de grande porte em tempo computacional satisfatório. A aplicação de um algoritmo especializado para Controle Ótimo como GRECO [10] deve apresentar resultados ainda melhores, com convergência mais rápida, pois se aproveita a estrutura particular da matriz Jacobiana das restrições. Para trabalhos futuros, deve ser testada a utilização do GRECO e verificar a possibilidade de resolver o problema com todos os subsistemas associados simultaneamente (planta e tancagem intermediária).

Bibliografia

[1] Abadie, J.; *Application of the GRG algorithm to Optimal Control problems* in Abadie, J. (ed.), Integer and Nonlinear Programming, pp.191-212, North-Holland, Amsterdam (1970).

[2] Almeida, M.R.; *Programação Automática da Produção em Refinarias de Petróleo utilizando Algoritmos Genéticos*. Dissertação de Mestrado, Pontífice Universidade Católica, Rio de Janeiro (2001).

[3] Asperen, E.v. et al; *Arrival processes in port modeling Insights from a case study*. Erasmus University, Rotterdam.
www.few.eur.nl/few/research/eurfew21/m&s/article/jetty (2004).

[4] Canon, M.D., Cullum Jr., C.D., Polak, E.; *Theory of Optimal Control and Mathematical Programming*. McGraw-Hill, Nova York(1970).

[5] Conway, R.W., Maxwell, W.L., Miller, L.W; *Theory of Scheduling*. Addison-Wesley, Nova York(1967).

[6] Cruz, A.V.A.; *Otimização de Planejamento com Restrições de Precedência usando Algoritmos Genéticos e Co-Evolução Cooperativa*. Dissertação de Mestrado, Pontífice Universidade Católica, Rio de Janeiro (2003).

[7] Dantzig, G.B.; *Linear Programming and Extensions*. Princeton, Nova Jersey (1963).

[8] Dantzig, G.B., Thapa, M.N.; *Linear Programming 1 Introduction*. Springer-Verlag, Nova York (1997).

[9] Ehrlich, P.J.; *Pesquisa Operacional: Curso Introdutório 5ª ed*. Atlas Editora, São Paulo (1985).

[10] Facó, J.L.D.; *A Generalized Reduced Gradient Algorithm for Solving Large-scale Discrete-time Nonlinear Optimal Control Problems,* in Siguerdidjane, H.B., Bernhard, P. (eds.), Control Applications of Nonlinear programming and Optimization 1989, IFAC Workshop Series #2, pp.45-50, Pergamon Press, Oxford (1990).

[11] Fagundez, F., Faco, J.L.D.; *Modelos de Controle Ótimo do Scheduling de Petróleo e Derivados em Portos*. Aceito como trabalho completo para o XXXVII SOBRAPO (setembro de 2005).

[12] Farah, M.A.; *Caracterização do petróleo e seus derivados: Parte 1 - Combustíveis*. Petrobras, Rio de Janeiro (1985).

[13] Floudas, C.A.; *Nonlinear and Mixed-Integer Optimization: Fundamentals and Applications*. Oxford, Nova York (1995).

[14] *Premium Solver Platform Solver DLL Platform Field-Installable Solver Engines User Guide. Version 6.0*. Frontline Systems, Inc., NV.
www.solver.com(2004).

[15] Fylstra, D., Lasdon, L., Watson, J., Waren, D.; *Design and Use of the Microsoft Excel Solver*. Interfaces 28:5, pp.29-55, Institute for Operations Research and the Management Sciences(1998).

[16] Jia, Z., Ierapetritou, M., Kelly, J.D.; *Refinery Short-Term Scheduling Using Continuous Time Formulation: Crude Oil Operations*. Ind. Chem. Eng. Res. 42, pp.3085-3097, American Chemical Society (2003).

[17] Jia, Z., Ierapetritou, M.; *Mixed-Integer Linear Programming Models for Gasoline Blending and Distribution Scheduling*. Ind. Chem. Eng. Res. 42, pp.825-835, American Chemical Society (2003).

[18] Joly, M., Moro, L.F.L., Pinto, J.M.; *Planning and Scheduling for Petroleum Refineries using Mathematical Programming*. Brazilian Journal of Chemical Engineering Vol.19, No.2, pp.207-228 (2002).

[19] Kirk, D.E.; *Optimal Control Theory: An Introduction*. Prentice-Hall, Nova York (1970).

[20] Lasdon, L.S., Waren, A.D., Jain, A., Ratner, M.; *Design and Testing of a Generalized Reduced Gradient Code for Nonlinear Programming*. ACM Transactions on Mathematical Software, Vol. 4, No. 1, pp.34-50 (1978).

[21] Lasdon, L.S., Waren, A.D.; *GRG2 User's Guide*. Department of Computer and Information Science. Cleveland State University, Cleveland, Ohio (1986).

[22] Leffler, W.L.; *Petroleum Refining in Nontechnical Language, 3rd ed*. PennWell, Oakland (2000).

[23] Luenberger, D.G.; *Introduction to Linear and Nonlinear Programming*. Addison Wesley, Massachusetts (1973).

[24] Magalhães, M.V.O. ; *Refinery Scheduling*. Tese de Doutorado, Imperial College, Londres (2004).

[25] Martínez, J.M. e Santos, S.A.; *Métodos Computacionais de Otimização*. Departamento de Matemática Aplicada. Unicamp, Campinas(1998).

[26] Más, R.; *Otimização da Programação de Suprimento de Petróleo*. Dissertação de Mestrado, Universidade de São Paulo, São Paulo (2001).

[27] Más, R.; *Técnicas de Otimização de Suprimento de Petróleo em Complexos de Distribuição*. Boletim Técnico 45(2), Petrobras, Rio de Janeiro (2002).

[28] Morton, T.E., Pentico, D.W.; *Heuristic Scheduling Systems with Applications to Production Systems and Project Management*. John Wiley & Sons, Nova York, (1993).

[29] Moro, L.F.L., Pinto, J.M.; *A Mixed Integer Model for Short Term Crude Oil Scheduling*. AIChE J. Annual National Meeting, 241c, Florida (1998).

[30] Moro, L.F.L.; *Técnicas de Otimização Mista Inteira para o Planejamento e Programação de Produção em Refinarias de Petróleo*. Tese de Doutorado, Universidade de São Paulo, São Paulo (2000).

[31] Nocedal, J., Wright, S.J.; *Numerical Optimization*. Springer-Verlag, Nova York (1999).

[32] Oliveira, L.A.; *Aplicação do Paralelismo à Resolução Numérica de Problemas de Otimização Não Linear*. Dissertação de Mestrado, Universidade Federal do Rio de Janeiro, Rio de Janeiro(2002).

[33] Papadimitriou, C.H., Steiglitz, K.; *Combinatorial Optimization Algorithms and Complexity*. Prentice-Hall, Nova Jersey (1982).

[34] Pinedo, L.F.L.; *Scheduling Theory, Algorithms, and Systems*. Prentice-Hall, Nova Jersey (1995).

[35] Pinto, J.M.; *Otimização de Processo Químicos - Aplicações em Planejamento e Scheduling*. Notas de Aula. Laboratório de Simulação e Controle de Processos, Universidade de São Paulo (2002).

[36] Rejowski Jr., R., Pinto, J.M.; *An MILP Formulation for the Scheduling of Multiproduct Pipeline Systems*. Brazilian Journal of Chemical Engineering, vol. 19, No.4, pp.467-474 (2002).

[37] Sasieni, M., Yaspan, A., Friedman, L.; *Operations Research: Methods and Problems*. John Willey & Sons, Nova York (1959).

[38] Simão, L.; *Otimização da Programação de Produção em Refinarias de Petróleo utilizando Algoritmos Genéticos e Co-evolução Cooperativa*. Dissertação de Mestrado, Pontífice Universidade Católica, Rio de Janeiro (2004).

[39] Simão, L., Pacheco, M.A.C., Vellasco, M.M.B.R.; *Otimização da Mistura de Produtos na Indústria de Petróleo utilizando Algoritmos Genéticos*. VI Simpósio Brasileiro de Automação Inteligente, Bauru (2003).

[40] Smith, S. e Lasdon, L.S.; *Solve Large-Sparse Nonlinear Programs using GRG*. ORSA Journal of Computing, 4, pp.1-15(1992).

[41] Tabak, D., Kuo, B.C.; *Optimal Control by Mathematical Programming*. Prentice-Hall, Nova Jersey (1971).

[42] Wolfe, P.; *Convergence Theory in Nonlinear Programming*. in Abadie, J. (ed.), Integer and Nonlinear Programming, pp.1-36, North-Holland, Amsterdam (1970).

[43] Zaccarelli, S.B.; *Programação e Contrôle de Produção*, Coleção de Engenharia de Produção e Administração Industrial. Livraria Pioneira Editôra, São Paulo (1973).