Modelos Computacionais para o Escalonamento de Tarefas em Redes de Dutos

Este exemplar corresponde à redação final da Dissertação devidamente corrigida e defendida por André Augusto Ciré e aprovada pela Banca Examinadora.

Campinas, 18 de agosto de 2008.

Prof. Dr. Arnaldo Vieira Moura (Orientador)

Prof. Dr. Cid Carvalho de Souza
(Co-orientador)

Dissertação apresentada ao Instituto de Computação, UNICAMP, como requisito parcial para a obtenção do título de Mestre em Ciência da Computação.

FICHA CATALOGRÁFICA ELABORADA PELA BIBLIOTECA DO IMECC DA UNICAMP
Bibliotecária: Maria Júlia Milani Rodrigues CRB8a / 2116

Ciré, André Augusto
C496m Modelos computacionais para o escalonamento de tarefas em redes de dutos / André Augusto Ciré -- Campinas, [S.P. :s.n.], 2008.

Orientadores : Arnaldo Vieira Moura ; Cid Carvalho de Souza
Dissertação (mestrado) - Universidade Estadual de Campinas, Instituto de Computação.

1. Restrições (Inteligência artificial). 2. Pesquisa operacional. 3. Otimização combinatória. 4. Petróleo - Transporte. I. Moura, Arnaldo Vieira. II. Souza, Cid Carvalho de. III. Universidade Estadual de Campinas. Instituto de Computação. IV. Título.

Título em inglês: Computational models for task scheduling in pipeline networks

Palavras-chave em inglês (Keywords): 1. Constraints (Artificial intelligence. 2. Operational research. 3. Combinatorial optimization. 4. Petroleum transportation.

Área de concentração: Pesquisa operacional

Titulação: Mestre em Ciência da Computação

Banca examinadora:

Prof. Dr. Arnaldo Vieira Moura (IC-UNICAMP)
Prof. Dr. Eduardo Camponogara (DAS-UFSC)
Prof. Dr. Zanoni Dias (IC-UNICAMP)
Prof. Dr. Nizam Omar (Univ.Presbiteriana Mackenzie)
Profa. Dra. Anamaria Gomide (IC-UNICAMP)

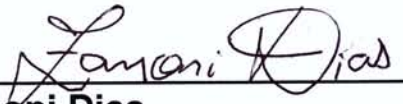
Data da defesa: 18/08/2008

Programa de pós-graduação: Mestrado em Ciência da Computação

TERMO DE APROVAÇÃO

Dissertação Defendida e Aprovada em 18 de agosto de 2008, pela Banca examinadora composta pelos Professores Doutores:

Prof. Dr. Eduardo Camponogara
DAS - UFSC



Prof. Dr. Zanoni Dias
IC – UNICAMP

Prof. Dr. Arnaldo Vieira Moura
IC – UNICAMP

Instituto de Computação
Universidade Estadual de Campinas

Modelos Computacionais para o Escalonamento de Tarefas em Redes de Dutos

André Augusto Ciré[1]

Julho de 2008

Banca Examinadora:

- Prof. Dr. Arnaldo Vieira Moura (Orientador)
- Prof. Dr. Eduardo Camponogara
  Departamento de Automação e Sistemas, UFSC
- Prof. Dr. Zanoni Dias
  Instituto de Computação, UNICAMP
- Prof. Dr. Nizam Omar (Suplente)
  Universidade Prebisteriana Mackenzie
- Profa. Dra. Anamaria Gomide (Suplente)
  Instituto de Computação, UNICAMP

[1]Suporte financeiro de: Fundação de Amparo à Pesquisa do Estado de São Paulo (processo 05/57344-7), 2006-2008.

Resumo

Esta dissertação de Mestrado trata de um problema real de escalonamento, no qual uma complexa *rede de dutos* é utilizada para distribuição de derivados de petróleo e bio-combustíveis de refinarias a mercados locais. Dutos constituem a alternativa de transporte mais vantajosa em termos econômicos e ambientais, mas trazem consigo um amplo conjunto de restrições operacionais difíceis, envolvendo seqüenciamento de produtos, capacidade de tanques, controle de taxa de vazão, controle de estoque e muitas outras. O objetivo do problema está em escalonar operações de bombeamento nos dutos de forma a satisfazer as demandas locais em cada órgão de distribuição, dentro de um horizonte de planejamento pré-definido.

Para resolvê-lo, este trabalho propõe uma nova abordagem híbrida composta por duas fases. Primeiramente, uma fase de *planejamento* define os volumes de produto que devem ser transmitidos entre órgãos para que as demandas sejam completamente atendidas. Em seguida, uma fase de *escalonamento* é responsável por criar e escalonar as operações de bombeamento, de forma a garantir que os volumes definidos na fase anterior sejam efetivamente enviados. Esta dissertação foca na fase de escalonamento, e duas formulações em *Programação por Restrições* (PR) são apresentadas para modelá-la. Conforme foi verificado, a flexibilidade de PR é fundamental para representar e satisfazer restrições que, usualmente, são desconsideradas na literatura, mas que são essenciais para a viabilidade operacional das soluções. A estratégia completa foi implementada e produziu resultados adequados e promissoras para 5 instâncias reais fornecidas pela Petrobras. Tais instâncias contêm 30 dutos, mais de 30 produtos e 14 órgãos de distribuição que contemplam cerca de 200 tanques.

Abstract

This dissertation deals with a very difficult overly-constrained scheduling challenge: how to operate a large pipeline network in order to adequately transport oil derivatives and biofuels from refineries to local markets. Pipeline network systems are considered the major option for transporting these product types, in view of their many economic and environmental advantages. However, they pose serious operational difficulties related to product sequencing, flow rates and tank capacities.

The challenge is how to schedule individual pumping operations, given the daily production and demand of each product, at each location in the network, over a given time horizon. In order to tackle this problem, we propose a novel hybrid approach which comprises two phases. Firstly, a *planning* phase decides the necessary volume transmission among depots to satisfy the given demands. Finally, a *scheduling* phase generates and schedules the pumping operations that guarantee the required volume transmission. This dissertation focuses on the scheduling phase, in which two new Constraint Programming (CP) models are proposed. The CP flexibility plays a key role in modeling and satisfying operational constraints that are usually overlooked in literature, but that are essential in order to guarantee viable solutions. The full strategy was implemented and produced adequate and promising results when tested over 5 large real instances from Petrobras. These instances have a complex topology with around 30 interconnecting pipelines, over 30 different products in circulation, and about 14 distribution depots which harbor more than 200 tanks.

Agradecimentos

Agradeço primeiramente à minha família, Antonio, Maria Antonia e Danielle, pelo carinho, amor, paciência e pelo apoio incondicional à minha educação, sem a qual não teria chegado até aqui.

Agradeço também aos meus orientadores, Prof. Arnaldo e Prof. Cid, pela oportunidade, amizade, por me iniciarem na área de pesquisa e por constituírem o modelo de profissional que almejo ser no futuro.

Agradeço a todos os meus amigos pela convivência, pelos ótimos momentos nesses últimos anos e por todo o apoio que me deram. Em especial ao Tony, pela amizade, paciência e grandes idéias nos dias, noites e madrugadas acordadas que passamos desenvolvendo este projeto.

Agradeço à FAPESP, pelo suporte financeiro ao meu projeto de pesquisa. À Petrobras, particularmente Fernando Marcellino, André Lima, Leandro Barcelos e Enio Medeiros, pelo fornecimento dos dados e, principalmente, pela oportunidade de participar deste projeto. Também agradeço aos funcionários do Instituto de Computação da UNICAMP, sempre dispostos a ajudar nos momentos que precisamos.

Força sempre!

Sumário

Lista de Tabelas

Lista de Figuras

Capítulo 1

Introdução

Problemas operacionais envolvendo agendamento e planejamento têm obtido grande destaque nos últimos anos, e sua importância deriva da necessidade da tomada rápida de decisões em ambientes de incerteza e de requerimentos operacionais extremamente complexos. Neste contexto, o planejamento da distribuição de álcoois e produtos derivados do petróleo adquire uma especial importância. Primeiramente, por envolver uma grande quantidade de recursos materiais e humanos, como conseqüência da dificuldade em encontrar agendamentos capazes de suprir toda a demanda sem violar restrições operacionais da distribuição. E, em segundo lugar, o fato do cenário petrolífero brasileiro ter deixado de ser um monopólio estatal, a partir de 1994, faz com que uma logística eficiente seja condição essencial neste ambiente de alta competitividade.

Esta distribuição pode ser realizada por diversos meios, tal como marítimo, rodoviário e ferroviário. Contudo, a principal forma de distribuição, no Brasil e no mundo, é via uma *rede de dutos*, em decorrência principalmente de sua viabilidade econômica e segurança ambiental [32]. De acordo com dados da própria PETROBRAS, há no Brasil mais de 11.300 km de dutos, que distribuem cerca de 1.600 barris por dia de derivados do petróleo para o mercado consumidor [31].

Atualmente, o agendamento da distribuição nas malhas de oleodutos no Brasil é essencialmente manual, caracterizando-se como uma atividade custosa e lenta, o que dificulta a simulação de cenários que possam reduzir custos e atender mais eficientemente a demanda por tais produtos. Desta forma, ferramentas computacionais de otimização exerceriam um papel fundamental no apoio à decisão deste planejamento, pois são capazes de lidar rapidamente com um grande conjunto de variáveis e restrições, fornecendo alternativas de soluções mais diversificadas, com menor custo e mais detalhadas. Conseqüentemente, mesmo que as soluções não fossem totalmente aproveitadas, já seriam excelentes pontos de partida para um posterior refinamento e análise.

1.1 Objetivos do Projeto

Este projeto de mestrado trata do estudo de modelos para a distribuição de álcoois e derivados do petróleo por meio de uma rede de dutos. Ele surge de um problema prático, originado na PETROBRAS, parceira na especificação dos requisitos e no fornecimento de dados. Sua importância, desta forma, não se limita apenas à pesquisa acadêmica, mas também envolve o esforço de criar metodologias úteis e aplicáveis para problemas de transporte em dutos de grande complexidade.

Contudo, a complexidade operacional inerente às topologias dutoviárias torna-o um problema de difícil resolução. Não há atualmente uma modelagem matemática suficientemente compacta e abrangente que aborde todos os requisitos mais fundamentais do problema. Conseqüentemente, os modelos de otimização ou satisfação existentes hoje na literatura, apesar de eficientes, geram soluções muito aquém daquelas operacionalmente factíveis. Isto porque são fortemente baseadas em relaxações do problema, tornando-o assim mais adequado para metodologias de otimização clássicas, como meta-heurísticas e Programação Linear Inteira [40].

Portanto, diferentemente dos estudos existentes, o objetivo do projeto é propor modelos capazes de representar todo o amplo número de requisitos operacionais do problema em formulações mais compactas, de forma que uma solução válida destes modelos esteja bastante próxima de uma solução factível para os operadores da rede de dutos. Tais modelos devem contemplar três outras características: mostrarem-se promissores para a resolução de instâncias reais, fornecidas pela própria PETROBRAS; serem flexíveis para a modelagem de outros requisitos operacionais; e, por fim, serem computáveis por diferentes algoritmos de resolução, servindo como base para futuras pesquisas.

A fim de cumprir esses objetivos, montou-se um grupo de pesquisa formado por professores e estudantes da UNICAMP, além de funcionários da própria PETROBRAS, em um projeto de longo prazo denominado SCONSUELO[1]. O grupo era inicialmente composto pelos professores Arnaldo V. Moura, Cid C. de Souza, pelos estudantes de mestrado André A. Ciré, Tony M.T. Lopes, e pelos funcionários da PETROBRAS, Fernando Marcellino, André Lima, Leandro Barcelos e Enio Medeiros. No término dessa dissertação, o grupo completou dois anos de pesquisa, incluindo coleta de dados, especificação do problema, modelagem e testes computacionais. Os avanços eram periodicamente fornecidos aos responsáveis pelo planejamento de dutos da PETROBRAS, os quais relatavam as características operacionais das soluções deveriam ser melhor abordadas pelos solucionadores.

A arquitetura de resolução desenvolvida pelo grupo propõe uma divisão do problema em duas fases, *planejamento* e *escalonamento*, ambas independentes e que permitem a

[1] *Solucionador para* CONSUELO, já que os resultados dos modelos devem ser compatíveis com o sistema de simulação da PETROBRAS, denominado CONSUELO.

aplicação de diferentes métodos para resolvê-las. Este projeto de mestrado foca na parte de *escalonamento*, por estar relacionada à satisfação dos requisitos operacionais da rede de dutos. Para então atingir os objetivos de modelagem descritos anteriormente, o trabalho apresenta uma formulação exata em *Programação por Restrições* [23], ferramenta que provê uma grande liberdade de representação e resolução.

Atualmente, o projeto SCONSUELO conta com diversos alunos de iniciação científica e novos estudantes de mestrado, os quais utilizarão os métodos desenvolvidos até então como base para algoritmos mais robustos e capazes de lidar com um maior número de restrições operacionais. A idéia inicial é que a fase de escalonamento seja resolvida utilizando técnicas de relaxação e metaheuristicas sobre os modelos PR aqui descritos, o que reforça a contribuição deste projeto de mestrado para futuras pesquisas relacionadas ao problema de escalonamento em dutos.

1.2 Organização do Texto

A dissertação de Mestrado está organizada como segue. O capítulo 2 descreve em detalhes o problema de Transporte em Dutos e seus parâmetros, e o capítulo 3 apresenta as principais referências da literatura de programação de dutos. Utilizando-se dos trabalhos anteriores como base, o capítulo 4 descreve a proposta de resolução do problema, a qual será desenvolvida em detalhes nos capítulos 6 e 7. Por fim, o capítulo 8 exibe as conclusões e capítulo 9 os possíveis trabalhos futuros, seguido pelas referências bibliográficas.

Capítulo 2

O Problema do Transporte em Redes de Dutos

Empresas de petróleo dispõem usualmente de um conjunto de *refinarias* e *terminais*, denominados *órgãos*, os quais são dedicados à produção e distribuição de derivados de petróleo e bio-combustíveis aos seus mercados consumidores. As campanhas de produção das refinarias e de entrega aos mercados são traduzidas, respectivamente, em valores estimados de produção e demanda de produtos em cada órgão da rede [4, 6].

Contudo, os órgãos não são auto-suficientes no suprimento de seus mercados locais. Torna-se necessário, portanto, o uso de algum meio de distribuição para evitar a falta de produto nos terminais e para escoar a produção em excesso nas refinarias. Tal transporte é predominantemente feito por meio de uma *rede de dutos*, a alternativa mais vantajosa em termos econômicos, operacionais e ambientais [32].

Os dutos possuem capacidades, sentidos permitidos de fluxo, vazões de bombeamento e podem ser usados para movimentar diferentes tipos de produto consecutivamente. Note que alguns dutos permitem vazão em ambos os sentidos, com inversões de fluxo. Outra característica operacional é que os dutos devem estar sempre completamente preenchidos, ou seja, a transmissão de um volume implica que necessariamente outro volume deve ser bombeado em seguida.

Os produtos, por sua vez, estão associados a um conjunto de *restrições de interface*, isto é, apenas podem ser bombeados consecutivamente em um duto se forem *compatíveis* entre si, condição essencial para a manutenção de um nível de qualidade aceitável. Cada órgão também contém restrições locais de bombeamento e recebimento de produtos, decorrentes das conexões internas entre tanques e dutos.

O *Problema de Transporte em Redes de Dutos* (PTRD), desta forma, consiste em determinar as operações a serem realizadas em uma rede de dutos a fim de escoar a produção e atender um dado conjunto de demandas, considerando um horizonte tem-

poral pré-estabelecido. Esta programação deve satisfazer todas as restrições operacionais particulares dos órgãos e dutos, além de buscar reduzir os custos de transporte do sistema.

2.1 Caracterização do PTRD

A seguir, são descritos os dados e restrições que formalizam o domínio do PTRD, necessários para definir as instâncias e suas soluções válidas. Tal caracterização pode ser dividida em quatro classes: *parâmetros e constantes gerais do problema*, *restrições dos órgãos*, *restrições dos dutos* e *dados dinâmicos da rede*. As três primeiras classes estão relacionadas à topologia física e pouco mutável da rede, enquanto a última varia conforme o horizonte de planejamento.

2.2 Parâmetros e Constantes Gerais

Há quatro parâmetros fundamentais para o PTRD, conforme relação abaixo.

1. *Unidade de Tempo (**u.t.**)*: designa a unidade de tempo real que será utilizada para a discretização da instância e precisão dos modelos desenvolvidos. Nos cenários reais, é comum que seja utilizado *minuto* como a u.t. básica.

2. *Unidade de Volume (**u.v.**)*: analogamente à u.t., designa a unidade de volume real que será utilizada para as vazões e capacidades. Usualmente são *metros cúbicos*.

3. *Horizonte (**H**)*: um número inteiro que designa o horizonte de execução em unidades de tempo, isto é, o tamanho da janela de tempo da instância na qual o planejamento e escalonamento devem ser aplicados. Considera-se também que o instante inicial de planejamento é **1**. Por exemplo, dados $H = 14400$ e u.t. em minutos, têm-se um horizonte de 10 dias em tempo real.

4. *Conjunto de Produtos (**P**)*: o conjunto $P = \{p_1, p_2, ..., p_{np}\}$ designa os produtos que são armazenados e transportados pela rede. Um *grupo de produtos* é um subconjunto de P formado por produtos com certas características químicas semelhantes, tais como os grupos de *gasolinas*, *dieseis* e *álcoois*. Algumas restrições, tal como as de interface, podem ser definidas sobre grupos ao invés de produtos individuais, facilitando a representação do problema.

   Produtos no PTRD são considerados *fungíveis* ou *intercambiáveis* [14], ou seja, não há diferenciação entre volumes de produtos p distribuídos na rede. Esta hipótese surge do fato de que, na prática, as quantidades de um mesmo produto p provenientes de origens distintas possuem também qualidades diferenciadas, apesar de sua

equivalência química. Caso esta diferenciação seja necessária, é possível explicitá-la nesta formulação criando um novo produto para cada nível de qualidade desejado.

Um exemplo no cenário real ocorre com a *gasolina de exportação* e a *gasolina comum*. Apesar de na prática serem o mesmo produto, ambas contêm especificações de qualidade bastante distintas. Desta forma, é importante que sejam consideradas como produtos distintos, com dados de produção e demandas desagregados.

2.2.1 Restrições dos Órgãos

Uma rede de dutos possui um conjunto de órgãos $Org = \{o_1, o_2, ..., o_{no}\}$, responsáveis pela produção, armazenamento e distribuição de produtos. Um número significativo das restrições de um determinado órgão $o \in Org$ está relacionado com seu complexo de tanques, dado pelo conjunto $T(o) = \{t_1, t_2, ..., t_{nto}\}$. Tais restrições são descritas a seguir, com $Tq = \bigcup_{o \in Org} T(o)$.

C1: Para um dado tanque $t \in Tq$, deve-se respeitar sua capacidade máxima de armazenamento $cap(t)$ e nunca violar sua capacidade mínima, $\mathbf{0}$.

C2: Um tanque $t \in Tq$ pode armazenar apenas um tipo de produto $prod(t) \in P$ durante todo o horizonte de planejamento, condição imposta como um requisito de qualidade. Além disso, usualmente há mais de um tanque por produto em cada órgão, mas não necessariamente um órgão contém tanques para todos os produtos da rede. Se $T(o) = \emptyset$ para um dado $o \in Org$, o órgão é denominado *entroncamento*, isto é, utilizado apenas para a passagem de produtos.

No cenário real, há casos em que certos tanques podem ser utilizados para o armazenamento de produtos diferentes dos originalmente atribuídos, o que envolve a aplicação de um custo fixo relativo à limpeza e a demais detalhes operacionais. Como são casos indesejáveis e muitas vezes provenientes da dificuldade do planejamento manual, a restrição C2 é inclusive requisitada pelos operadores da Petrobras.

C3: Um produto pode ser tanto bombeado para um tanque ou dele retirado, desde que essas operações não sejam simultâneas. Isto significa que, ao iniciar uma inserção ou retirada em um tanque, toda a operação deve ser completada antes que qualquer outra seja realizada neste tanque.

Uma exceção à este caso ocorre em órgãos que representam terminais portuários. Devido ao período restrito em que os navios podem ficar parados nos portos, muitas vezes volumes de produtos devem ser enviados diretamente dos dutos para os navios sem pré-estocagem. Para tanto, realiza-se uma operacão *pulmão*, onde o produto é simultaneamente bombeado nos dutos para um tanque, e do tanque para o navio

com uma vazão menor. No entanto, a operação pulmão é apenas possível para alguns órgãos que possuem esta capacidade hidráulica, definido pelo conjunto $Pulm \subseteq Org$.

Além disso, como hipótese, um tanque que esvaziou pode ser reabastecido sem que seja necessário considerar um tempo para limpeza e preparo para o armazenamento de volume adicional.

C4: Se o produto a ser inserido em um tanque é produzido localmente, isto é, no mesmo órgão, o tanque só pode receber este produto se estiver completamente vazio, condição imposta também por questões de qualidade.

C5: Caso ocorra a mistura de dois volumes produzidos em órgãos distintos num certo tanque, a qualidade do produto deverá ser reverificada antes da entrega ao mercado consumidor ou bombeamento para outros órgãos. Isto é representado aqui como um *tempo de certificação* T_c, considerado entre o instante final da mistura e o instante inicial das demais operacões no tanque.

Uma vez definidas as condições de armazenamento de produto nos órgãos, é possível listar as restrições referentes às necessidades de estocagem impostas pelos mercados, além daquelas relacionadas aos limites de envio e recebimento.

C6: O valor de *estoque* de um órgão $o \in Org$, produto $p \in P$ e instante i é dado somando-se o volume em i de todos os tanques $T(o)$ que armazenam p. Em todos os órgãos e instantes, tal estoque deve respeitar limitantes de *estoque mínimo*, $est\_min(o, p)$, e *estoque máximo*, $est\_max(o, p)$. Esta restrição é proveniente da necessidade de se manter estoques de segurança para incentivar o escoamento da produção e para eventuais problemas na distribuição, tal como manutenções emergenciais de dutos.

C7: Devido às restrições sobre o conjunto de bombas e válvulas hidráulicas em um certo órgão $o \in Org$, deve ser observado um *número máximo de operações simultâneas de envio*, $env\_max(o)$. Como são relativas aos bombeamentos, não há uma restrição análoga para recebimento de produtos em o.

C8: *Alinhamentos proibidos*: o complexo de conexões internas aos órgãos, principalmente refinarias, não permite que determinadas operações em oleodutos ocorram simultaneamente. Cada configuração não-passível de uso para transporte de produtos é associada a um *alinhamento proibido*, um código que agrupa as operações proibidas de serem exercitadas simultaneamente no órgão.

Supondo que $Align(o)$ sejam os alinhamentos proibidos de $o \in Org$, cada alinhamento $alin \in Align(o)$ é formado por um conjunto de duplas (p, s), onde $p \in P$

e s (sentido) indica se a operação é relativa ao envio ou recebimento. Por exemplo, o alinhamento $alin = \{(gasolina, envio), (diesel, recebimento)\}$ significa que as operações de envio de gasolina e recebimento de diesel não podem ser simultâneas. Um certo alinhamento pode envolver diversos produtos e diferentes direções. Além disso, como está relacionado às conexões internas de um órgão, não depende dos dutos de entrada ou saída de produto.

Essa forma de representação facilita a modelagem do PTRD, evitando a necessidade de explicitar detalhes da topologia dos circuitos hidráulicos e dutos internos ao órgão.

C9: Devido principalmente ao custo mais alto da energia elétrica em certos períodos do dia, um órgão deve obededer a *períodos de sazonalidade*, momentos em que nenhuma operação pode ser iniciada. Para um dado órgão $o \in Org$, períodos de sazonalidades são dados como um conjunto de intervalos de tempo, $saz(o) = \{(inicio_1, fim_1), ..., (inicio_{nsaz(o)}, fim_{nsaz(o)})\}$.

C10: Operações também não podem ser nem iniciadas, nem terminadas, durante as *trocas de turno* de trabalho nos vários órgãos. Estas trocas são dadas de forma análoga aos períodos de sazonalidade, $tt(o) = \{(inicio_1, fim_1), ..., (inicio_{tt(o)}, fim_{tt(o)})\}$, $o \in Org$.

2.2.2 Restrições dos Dutos

A rede está associada a um conjunto $Dt = \{d_1, d_2, ..., d_{nd}\}$ de dutos, onde cada $d \in Dt$ conecta um par de órgãos $orgs(d) = (o_i, o_j)$, com $o_i, o_j \in Org$. No caso, o_i é denominado *órgão* ou *extremidade de origem* e o_j, *órgão* ou *extremidade de destino* do duto d. O sentido de fluxo da extremidade de origem à de destino corresponde ao *sentido principal* do duto, ou $p(d)$. Já o sentido de fluxo contrário corresponde ao *sentido reverso* do duto, ou $r(d)$. Outra característica é que dois órgãos podem estar conectados por mais de um duto, o que é usual nas instâncias reais. Volumes também podem ser transmitidos diretamente de um duto para outro, sem a necessidade de serem armazenados temporariamente em órgãos intermediários.

A partir dos órgãos e dutos, define-se também o conceito de *rotas*. Uma *rota* r é composta por uma seqüência de pares de órgãos separados por dutos, representando os percursos pelos quais qualquer fluxo da rede pode transitar. As rotas possíveis são prefixadas e informadas como parâmetro: se um certo volume de produto v for transmitido de um órgão o_i da rede até um órgão o_j, passando por um ou mais órgãos e dutos intermediários $d_1, o_{j+1}, ..., o_{j-1}, d_n$, $j = i + n$, necessariamente a rota $r = (o_i, d_1, o_{i+1}, ..., o_{j-1}, d_n, o_j)$ deve existir. Note que os volumes são transmitidos diretamente entre os dutos intermediários de uma rota, sem perdas ou armazenamento nas conexões intermediárias. Contudo, existem

casos em que parte do volume pode ser depositado em órgãos intermediários, e o movimento do volume restante do duto segue na rota com uma vazão menor. Esta operação é denominada *sangria*, que também deve ser prevista no modelo.

As restrições referentes aos dutos são listadas a seguir.

C11: Os dutos são pressurizados e, portanto, devem sempre estar completamente preenchidos com produtos. A capacidade volumétrica de um duto $d \in Dt$ é definida por $vol(d)$.

Também é considerada uma conservação de massa ideal: para se retirar um determinado volume v de uma extremidade do duto, deve-se inserir o mesmo volume v em sua outra extremidade. Tal volume pode ser proveniente de um tanque ou mesmo de um outro duto, interligado ao mesmo órgão através de uma de suas extremidades.

C12: As *vazões de bombeamento* em um duto são limitadas por produto, devido às suas viscosidades, e pelo sentido de fluxo, decorrente das diferenças na capacidade das bombas entre os órgãos de origem e destino e inclinação do terreno. Uma intuição relativa à vazão por sentido é que dutos geograficamente inclinados possuem uma vazão de descida maior que a de subida. Os limites inferior e superior das vazões para um duto $d \in Dt$, produto $p \in P$ e sentido $s \in \{p(d), r(d)\}$ são dados, respectivamente, por $vz\_inf(d, p, s)$ e $vz\_sup(d, p, s)$.

Assim, para se bombear um certo volume em um duto, deve-se considerar todos os produtos e sentidos dos dutos nas rotas que serão movimentadas como conseqüência do bombeamento. A vazão máxima de bombeamento será, portanto, o mínimo das vazões superiores destes produtos nos seus respectivos sentidos. Já a vazão mínima de bombeamento será dada pelo máximo das vazões inferiores.

Os parâmetros $vz\_inf(d, p, s)$ e $vz\_sup(d, p, s)$ também indicam os sentidos possíveis de fluxo para os produtos em um duto. Por exemplo, se um produto p só puder ser trafegado no sentido principal do duto, então $vz\_inf(d, p, r(d)) = vz\_sup(d, p, r(d)) = 0$.

C13: Devido às questões operacionais das bombas de cada órgão, deve-se respeitar também uma *quantidade mínima de bombeamento* ao se injetar produtos nos dutos. O volume bombeado em um duto também pode ser denominado de *batelada*, e esta restrição também é comumente referenciada como *tamanho mínimo de batelada* na literatura. Tal quantidade é definida por duto $d \in Dt$, produto $p \in P$ e sentido $s \in \{p(d), r(d)\}$, dada por $qtde\_min(d, p, s)$.

C14: Dois produtos em contato num certo duto devem ser *compatíveis*. Pares incompativeis são dados pelo conjunto de duplas $Incomp = \{(g_i, g_j), ..., (g_m, g_n)\}$, onde $g_k \subset P$ é um grupo de produtos.

Esta restrição está ligada à manutenção da qualidade dos produtos, de forma a evitar a degradação decorrente da mistura (ou *interface*) entre tipos químicos diferenciados. Caso seja absolutamente necessário que sejam enviados dois produtos incompatíveis em seqüência, deve-se interpor entre eles um terceiro produto, compatível com ambos, chamado de *selo*. O volume de selo depende do oleoduto, dos produtos em questão e do sentido do fluxo, e é dado por $selo(d, p, q, s)$, para $d \in Dt$, $p, q \in P$ e $s \in \{p(d), r(d)\}$.

C15: Os dutos possuem *períodos de manutenção* específicos, momentos em que são realizados reparos e limpeza. Os períodos de manuntenção são dados por $manut(d) = \{mnut_1, mnut_2, ..., mnut_{nm(d)}\}$, $d \in Dt$. Cada elemento $mnut_i = (inicio, fim, \alpha)$ indica que, durante os instantes $inicio$ e fim, as vazões estão limitadas a um parâmetro α vezes as vazões máximas dos produtos e sentidos do duto, com $\alpha < 1$.

2.2.3 Dados Dinâmicos da Rede

Os *dados dinâmicos* compõem os parâmetros que não se referem à topologia física da rede, mas específicos da instância a ser tratada pelo algoritmo. Podem ser divididos em três grupos: dados relativos ao estado inicial da rede, relativos à produção e relativos à demanda.

Os dados do estado inicial da rede são dois, dados a seguir.

1. *Estoque inicial dos tanques*, indicando o produto e o volume inicial nos tanques.

2. *Conteúdo inicial dos dutos*, dado para todos os dutos por uma seqüência da forma $DIni(d) = \{(p_m, q_m, r_m), .., (p_n, q_n, r_n)\}$, $d \in Dt$. A tripla (p_j, q_j, r_j) representa, respectivamente, um produto $p_j \in P$, sua quantidade q_j e a rota que deve seguir, r_j. Desta forma, considera-se que os produtos inicialmente nos dutos já possuem um destino pré-estabelecido, que obrigatoriamente deve ser respeitado pelas soluções geradas. A ordem das tuplas na seqüência $DIni(d)$ refere-se ao sentido principal de fluxo no duto.

As campanhas de produção para cada órgão $o \in Org$ são definidas pelo conjunto $Pr(o) = \{pr_1, pr_2, ..., pr_{npr}\}$, onde cada elemento $pr_i \in Pr(o)$ é formado pela tupla $(p, v, inicio\_pr, fim\_pr)$. No caso, $p \in P$ é o produto que será refinado, v um inteiro não-negativo que representa o volume produzido e, por fim, $[inicio\_pr, fim\_pr]$ representa o intervalo de tempo em que a produção ocorrerá no órgão.

O volume v produzido é distribuído uniformemente entre as $t = fim\_pr - inicio\_pr + 1$ unidades de tempo, isto é, a cada instante serão produzidas $\lceil v/t \rceil$ unidades de volume

(u.v.). A produção pode ser distribuída em diferentes tanques durante este intervalo, contanto que estejam vazios imediatamente antes de a receberem (restrição C4).

Já as demandas são dadas de forma simétrica às produções. Para o órgão $o \in Org$, é definido o conjunto $Dem(o) = \{dem_1, dem_2, ..., dem_{ndem}\}$, onde cada elemento $dem_i \in Dem(o)$ é formado pela tupla $(p, v, inicio\_dem, fim\_dem)$. Têm-se que $p \in P$ é o produto da demanda, v é um inteiro não-negativo que representa o volume demandado e, por fim, $[inicio\_dem, fim\_dem]$ representa o intervalo de tempo em que a demanda ocorrerá no órgão o. O volume v demandado é distribuído uniformemente entre as $t = fim\_dem - inicio\_dem + 1$ unidades de tempo, isto é, a cada instante serão extraídas $\lceil v/t \rceil$ unidades de volume (u.v.). Tal como a produção, demandas podem ser absorvidas de diferentes tanques durante este intervalo.

2.3 Solução e Objetivos

Uma solução é formada por uma programação de *movimentos* de entrada nos dutos. Cada movimento é designado pela tupla $m = (p, r, v, inicio\_m, fim\_m, t_s, t_e)$, isto é, por um produto $p \in P$, uma rota r cadastrada, o volume v positivo do produto, os instantes de início, $inicio\_m$, e fim, fim_m, de bombeamento do produto no primeiro duto da rota e, por fim, pelos tanques de saída $t_s \in Tq$ e de entrada $t_e \in Tq$, de onde o volume será extraído e onde será armazenado, respectivamente. O conjunto de movimentos deve ser tal que respeite todas as restrições nos órgãos e dutos, além de satisfazer as campanhas de produção e demanda em cada órgão.

No modelo proposto neste projeto, o foco será dado à satisfação de restrições ao invés da otimização de uma certa função objetivo, devido à alta complexidade de se obter soluções viáveis para o PTRD. Assim, custos relativos à utilização dos dutos e tanques são desconsiderados. Isto é condizente com a prática atual realizada pela Petrobras, na qual o correto atendimento da demanda é prioritário frente aos demais custos.

2.4 Modelo da Rede de Dutos

A figura 2.1 apresenta um modelo esquemático simples de uma rede de dutos tratada neste projeto, também mostrada no capítulo 7. Cada órgão B_i apresenta seu próprio conjunto de tanques, e note que mais de um duto pode ser utilizado para conectar dois órgãos distintos, tal como ocorre entre B_2 e B_3.

Para que dois produtos fiquem em contato no duto, como em D_2, ambos devem ser compatíveis entre si, conforme restrição C14. Se o órgão B_0 possuir um número máximo de envios simultâneos igual a 1 (restrição C7), então produtos não podem ser injetados

simultaneamente em D_0 e D_1 a partir de B_0. Além disso, caso houver um alinhamento proibido (restrição C8) de recebimento de produtos P_1 (cor cinza) e P_2 (cor escura) no órgão B_3, ele não poderá receber produtos dos dutos D_3 e D_4.

Por fim, um exemplo de rota seria $r = (B_1, D_2, B_2, D_4, B_3)$. Se o destino do produto P_1 (cor cinza) do duto D_2 for o órgão B_3 pela rota r, a vazão aplicada para empurrar o duto D_2, no caso a partir de B_1, deve ser consistente com as vazões dos produtos tanto de D_2 como do duto D_4, que também será empurrado.

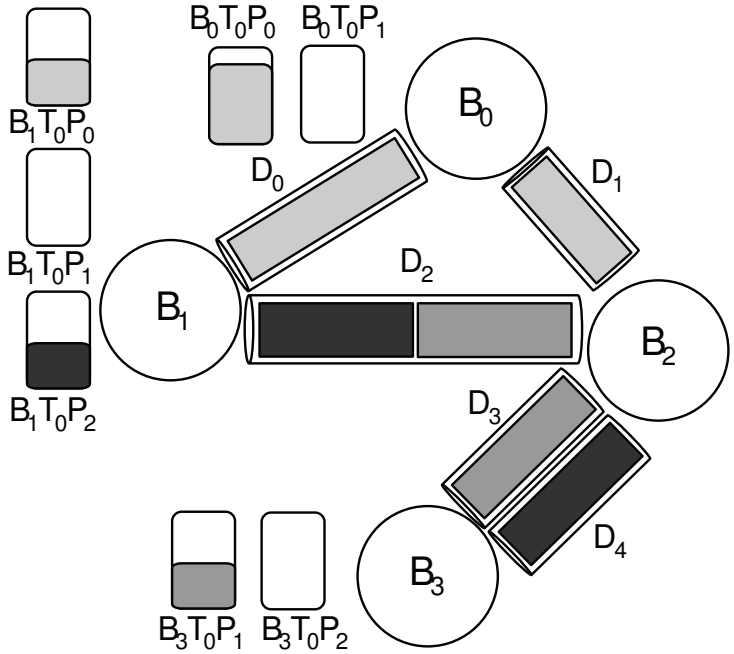


Figura 2.1: Modelo esquemático de uma rede de dutos.

2.5 Topologia Brasileira

A especificação do PTRD aqui descrita agrega as principais restrições abordadas na literatura do problema, além de representar com certa fidelidade a realidade enfrentada pelos planejadores e operadores de dutos. Ela foi concebida a partir da análise bibliográfica do PTRD e, principalmente, com diversas reuniões juntos aos gerentes da Petrobras.

O projeto focará na rede de dutos da região Sudeste do Brasil, gerenciada pela Petrobras e representada na figura 2.2. Esta rede é composta por 14 órgãos, 29 dutos, cerca de 240 tanques e mais de 30 produtos.O horizonte de planejamento é de cerca de 10 dias.

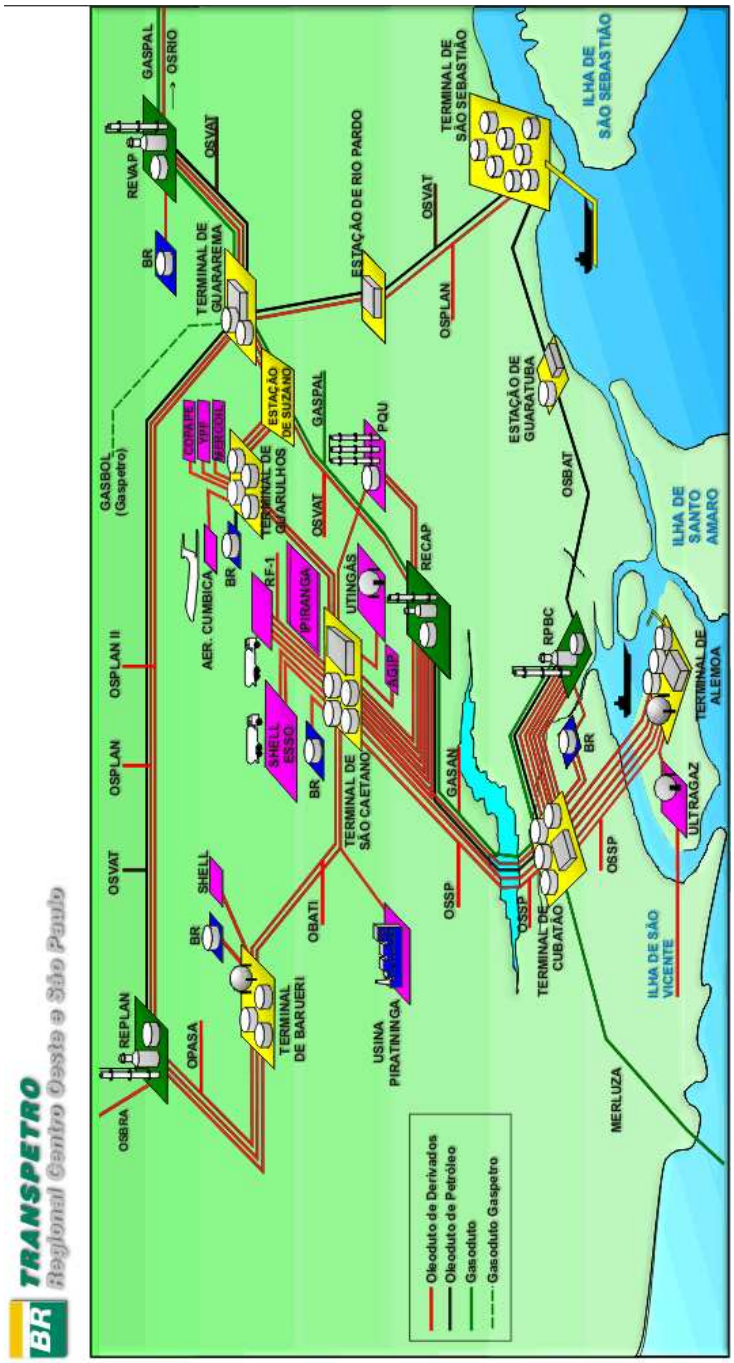


Figura 2.2: Representação do sistema de dutos da região Sudeste de São Paulo (figura fornecida pela Petrobras).

Capítulo 3

Revisão Bibliográfica

O problema abordado neste projeto engloba um conjunto de restrições bastante complexo e diversificado, relacionando fatores como capacidade de estocagem, seqüenciamento de produtos nos dutos e condições de balanceamento de massa em sua especificação mais geral. O trabalho de [25] demonstra que basta considerar a restrição C14, de interfaces entre produtos, para que o PTRD seja classificado como *NP-Difícil* [30].

Na literatura, o problema é usualmente conhecido como *Multiproduct Pipeline Scheduling* e diversos trabalhos se propuseram a apresentar abordagens heurísticas ou exatas para sua resolução. Contudo, em decorrência da grande aplicabilidade prática do problema, os trabalhos existentes diferem bastante na forma como definem e adaptam restrições às suas respectivas realidades. Desta forma, três pontos são essenciais para a compreensão de cada proposta de modelagem: a topologia da rede, o método de resolução aplicado e, por fim, os resultados computacionais e a operacionalidade das soluções.

Há pesquisas dedicadas à revisão bibliográfica do PTRD, tais como [13, 39], que apresentam uma análise bastante detalhada das abordagens existentes na área. Assim, a revisão aqui descrita procurará destacar as propostas mais relevantes e que serviram como base para o desenvolvimento deste projeto. A tabela 3.1 sintetiza as características de tais propostas e a compara com este trabalho.

Dentre tais trabalhos, o estudo mais completo da área é realizado por Camponogara [6], cujas características mais se aproximam da realidade quando comparadas aos demais projetos. Nele, define-se um *Problema de Transporte de Derivados de Petróleo Simplificado*, ou PTDPS, no qual as vazões são fixas por duto, todas as bases são capazes de armazenar todos os produtos (eliminando-se, assim, a transmissão de produtos duto-duto), e não há trocas de turno ou manutenções programadas. Também desconsideram-se estoques mínimos por base e quantidades mínimas por operação. Os tanques, por sua vez, são *agregados*, isto é, há um tanque por órgão com capacidade equivalente à soma de todos os tanques que armazenam o produto em questão.

	[1]	[3,7,18]	[4]	[6]	[11]	[19,20,21]	**PTRD**
Órgãos	8	5	12	7	2	2	14
Dutos	8	4	28	7	1	1	29
Tanques	40	25	84	28	16	12	242
Produtos	5	4	7	4	8	6	32
Horizonte	14 dias	3 dias	5 dias	80 u.t.	5 dias	30 dias	10 dias
Tempo Contínuo		√	√				√
Vazões Variáveis		√			√	√	√
Tanques Individuais					√		√
Topologia Genérica	√		√	√			√
Restrições de Interface		√	√		√	√	√
Técnica	GA	MILP,VNS	A-Teams	GA,MILP	MILP,CP	MILP	Heurística+CP
Resultados	Viável	Ótimo	Inviável	Viável	Ótimo	Ótimo	Viável

Tabela 3.1: Comparação entre abordagens anteriores.

Para a resolução do PTDPS, o trabalho de [6] propôs duas técnicas distintas, uma exata e outra heurística. A primeira, modelada como uma Programação Linear Inteira (PLI) [40], buscava reduzir o PTDPS a um problema de *Fluxo em Redes Multiperíodos* [3], com dutos e órgãos como vértices e transmissões de quantidade como arestas. Apesar da existência de algoritmos com bom desempenho para problemas deste tipo, verificou-se que a resolução do modelo PLI do PTDPS era impraticável, como consequência do tamanho e densidade do grafo resultante da topologia da rede. Para as últimas versões dos resolvedores comerciais de PLI até a publicação desta dissertação, o tempo médio para se obter uma solução para a relaxação linear do modelo foi de 50 horas.

A segunda técnica proposta em [6] caracterizava-se pela aplicação de *Times Assíncronos* ao PTDPS, heurística na qual diversos agentes autônomos tratam de partes distintas do problema, trocando de tempos em tempos informações relevantes para a sua resolução. Para viabilizar a aplicação da heurística, o problema foi modelado como um *Job Shop Scheduling* [6], composto por dois agentes: um para a geração de bombeamentos de quantidades entre bases e dutos, os *jobs*, e outro para o escalonamento e simulação de tais bombeamentos nos dutos, equivalente ao processamento dos *jobs* em máquinas (dutos).

Apesar de mais eficiente, a aplicação de Times Assíncronos proposta por [6] não foi capaz de gerar soluções viáveis para as instâncias apresentadas, em decorrência das estratégias essencialmente gulosas atribuídas aos agentes. Com esta perspectiva, Laber *et al* [27] utilizaram os Times Assíncronos como a fase gulosa de uma metaheurística *GRASP* [38] para o PTDPS, desenvolvendo uma busca local para o reparo das soluções. Como conseqüência, foram capazes de satisfazer todas as restrições consideradas no PTDPS, mas o tempo necessário para a ressimulação após cada aplicação da busca local inviabilizava sua utilização.

Posteriormente, Braconi *et al* [4] relaxou o modelo PLI de Fluxo em Redes Multi-períodos de [6], denominando a resolução deste novo modelo *Etapa de Planejamento*. Esta solução parcial era então utilizada como parte de uma segunda heurística, a *Etapa de Escalonamento*, na qual atribuía-se tempos aos fluxos criados na primeira etapa. Esta técnica permitiu obter soluções rapidamente, mas as relaxações intrínsecas à definição do PTDPS, tal como tanques para todos os produtos em todos os órgãos, tornavam as soluções encontradas pouco aplicáveis na prática.

Outro trabalho fundamentado na aplicação de heurísticas é de Crane *et al* [10], no qual um Algoritmo Evolutivo [15] foi aplicado para uma versão bastante simplificada do PTRD. Nela, a rede de dutos é tratada como uma árvore direcionada com 8 nós, e os estoques dos tanques possuem apenas três estados possíveis: alto, médio e baixo. Restrições de interface e tamanho mínimo de bateladas são também desconsideradas. O algoritmo proposto foi capaz de gerar soluções para horizontes de até 3 dias mas, devido à forma como as soluções eram representadas, o algoritmo era capaz de lidar apenas com instâncias pequenas com poucos tanques e órgãos.

A técnica desenvolvida por De La Cruz *et al* [11] também aplica um Algoritmo Evolutivo para uma versão simplificada do PTRD, sem restrições de interface e com todos os dutos possuindo um mesmo diâmetro e vazão. No entanto, são considerados dutos bidirecionais e uma função multi-objetivo, buscando reduzir interfaces e prazo de entrega dos produtos. Adicionalmente em [12], o autor também implementa um PLI para o problema, usado para gerar parte das soluções que compõe a população do Algoritmo Evolutivo. Em todos os casos, o algoritmo foi capaz de gerar soluções para instâncias pequenas rapidamente, especialmente com o uso de resolvedores PLI e como decorrência das simplificações.

Há ainda o trabalho de Alves [1], que também aplica um Algoritmo Evolutivo para uma variação do PTRD. No entanto, o procedimento desenvolvido foca na utilização de uma rede específica da Petrobras denominada *rede de escuros*, utilizada para o tráfego de produtos de grande viscosidade. O problema é similar ao PTDPS de [6], com tanques agregados e vazões constantes, mas considerando também tamanho das bateladas e uso de selos para separar produtos incompatíveis. Além disso, a função objetivo buscava minimizar o não-atendimento da demanda. O algoritmo obteve bons resultados para um horizontes de 7 e 14 dias, discretizados em períodos de algumas horas.

Uma abordagem alternativa é descrita em Sasikumar *et al* [20], onde uma técnica baseada em Inteligência Artifical, denominada *Beam Search*, é agreada a heurísticas fortemente baseadas na experiência humana. O projeto é aplicado em uma rede indiana, com restrições de seqüenciamento de produtos e estoques nos órgãos. Apesar das simplificações, são geradas boas programações mensais para as instâncias em questão.

Ainda, os trabalho de Liporace [14] e Milidiú e Liporace [26] tratam o PTRD como

um problema de *Planejamento em Inteligência Artificial* (PIA). Tal modelagem consiste em formular o problema genericamente como um conjunto de proposições lógicas, cuja composição de seus valores binários representam um *estado*. A partir de um estado inicial, aplicam-se *ações* que irão modificá-lo até que se atinja um *estado-objetivo*. O objetivo de um planejador, desta forma, é identificar uma seqüência de ações válidas que leve o sistema ao estado-objetivo. Além de um planejador específico para o problema, a especificação do PTRD como um PIA por Liporace foi incluída no *benchmark* oficial da *International Planning Competition* [9], uma competição internacional de planejadores genéricos. Contudo, numa perspectiva prática, os planejadores existentes resolvem apenas instâncias muito limitadas, com horizontes curtos, poucos tanques e restrições.

Outro foco de pesquisa está em topologias simples, compostas por apenas um duto e um conjunto de órgãos conectados a ele. A motivação é que ela já representa diversos casos práticos existentes no mundo, incluindo sub-redes da própria Petrobras. Além disso, esta topologia permite considerar restrições mais complicadas referentes às vazões, inventário, perdas de volume decorrentes da criação de interfaces e custos de energia.

Neste contexto, encontram-se os trabalhos de Rejowski e Pinto [18, 24]. Ambos apresentam uma formulação PLI de tempo discreto para o sistema OSBRA da Petrobras, composto por 5 órgãos, 4 produtos e um duto que liga os estados de São Paulo e Brasília. São consideradas diversas restrições, tais como inventário nos órgãos e seqüências permitidas de bombeamento, além de uma função objetivo referente à minimização de custos de estoque e de interface. Para um horizonte de 3 dias, os autores não obtiveram a solução ótima nos testes realizados. Contudo, em [33], o modelo é fortalecido pela introdução de cortes e se consegue, em um tempo computacional razoável, soluções ótimas para todas as instâncias anteriores.

Para a mesma especificação proposta por [33], Cafaro e Cerdá [5] apresentam uma formulação baseada em uma representação contínua do tempo e dos volumes. Conforme explicado pelos autores, esta representação permite uma diminuição significativa do tamanho da formulação e do número de variáveis binárias. Os resultados são comparados ao trabalho de [33], e mostram uma redução de cerca de três ordens de magnitude dos requisitos computacionais necessários para resolver o problema. A representação por tempos contínuos também foi implementada por Rejowski e Pinto [34], considerando adicionalmente vazões variáveis de bombeamento e decisões relativas ao controle de inventário. Para tanto, o problema foi modelado como uma Programação Não-linear Inteira, baseado na formulação em [24]. O algoritmo resultante gerou boas soluções para poucos dias.

Um outro direcionamento consiste em decompor o PTRD em subproblemas mais fáceis de serem tratados, ao invés de resolvê-lo completamente com um modelo único. O trabalho de Magatão *et al* [21] propõe uma divisão em que três submodelos são processados seqüencialmente, projetada para uma rede composta por um duto conectando uma refi-

naria a um terminal portuário. A execução inicia-se com um procedimento denominado *Tanque Bound*, caracterizado como um modelo em PLI responsável pela determinação dos recursos (*i.e.*, tanques) a serem utilizados em todo o escalonamento. Em seguida, a *Rotina Auxiliar*, basicamente heurística, define alguns parâmetros de entrada para o próximo submodelo, como limites temporais que devem ser respeitados pelas tarefas no oleoduto. Por fim, o *Modelo Principal*, escrito como um PLI e baseado nos parâmetros fornecidos pelo Tanque Bound e pela Rotina Auxiliar, determina o seqüenciamento e a temporização das atividades de bombeamento no duto. A técnica foi testada com sucesso na topologia em questão, considerando restrições de tancagem e de interface, além de vazão constantes. Os autores propõe também em [22] uma abordagem integrada entre PLI e Programacão por Restrições para resolver as mesmas instâncias, gerando boas soluções.

Mais recentemente, um conjunto de trabalhos de Relvas *et al* apresenta um estudo de uma rede de dutos de Portugal, formada por um duto e dois órgãos. Em [35, 37], os autores apresentam uma formulação PLI que considera diversas restrições de controle de inventário, baseada em [5]. Já em [36], o modelo anterior é melhorado de forma que sejam consideradas paradas em dutos e vazões variáveis, além do conceito de *re-escalonamento*, em que o modelo é resolvido após perturbações na solução corrente.

Capítulo 4

Arquitetura do Modelo de Resolução

O estudo dos artigos e teses apresentados no capítulo 3 evidencia que, como decorrência da grande dificuldade do PTRD, as abordagens existentes se apóiam fortemente em simplificações dos requisitos do problema. Por exemplo, são tratadas topologias de rede bastante simples na maioria dos casos, com apenas um duto de sentido único e terminais dispostos de forma seqüencial [5, 20, 34]. Ou, ainda, não são consideradas incompatibilidade entre produtos [1, 12] e os tanques são representados com capacidade agregada [4, 6].

Tais hipóteses permitem modelagens mais intuitivas e diretas do PTRD, fundamentando seu tratamento pelas técnicas de otimização mostradas no capítulo anterior. Em contrapartida, elas também impossibilitam a representação de grande parte das restrições mais importantes para a viabilidade prática das soluções, o que torna as abordagens existentes muito pouco aplicáveis para os cenários reais. Um exemplo é quando se considera um tanque único de capacidade agregada. Apesar de simplificar as decisões relativas ao controle de inventário, inviabiliza a aplicação de restrições como não-simultaneidade (C3), essencial para um cálculo mais realista dos tempos de bombeamento. Note também que a maioria dos trabalhos existentes considera cerca de 6 órgãos, 4 produtos e poucos tanques. Por outro lado, a instância a ser tratada aqui agrega 14 órgãos com aproximadamente 300 tanques, além de 28 dutos, 30 produtos, u.t. em minutos e diversas outras restrições.

É clara, portanto, a necessidade de modelos mais abrangentes para a obtenção de soluções factíveis para apresentar aos operadores da rede de dutos. Contudo, constata-se que a resolução do PTRD como um grande problema único e integrado é pouco promissora. Além do alto número de restrições complicadoras que devem ser satisfeitas simultaneamente, muitos dos trabalhos baseados em resoluções integradas não obtiveram resultados satisfatórios para uma topologia complexa, mesmo após diversas relaxações do modelo [6].

Desta forma, propõe-se uma arquitetura de resolução do problema composta por duas fases, executadas seqüencialmente: a fase de *planejamento* e a fase de *escalonamento*, sendo a última o objeto de estudo desta dissertação. Tal arquitetura é baseada nas

decomposições recorrentes do PTRD encontradas na literatura, as quais refletem os procedimentos manuais atualmente adotados. Cada fase é descrita com detalhes nas seções seguintes.

4.1 Fase de Planejamento

O problema de *planejamento* trata da movimentação geral na rede de dutos, onde são programados quais órgãos irão receber e enviar produtos para o atendimento das demandas e escoamento das produções. Uma vez fixados os órgãos de destino e de origem, define-se então quais serão os dutos e as bases intermediárias pelos quais os produtos irão trafegar, determinando também os volumes que serão transmitidos nas rotas escolhidas.

As trocas de produto geradas pela fase de planejamento são representadas por uma estrutura denominada *plano de entrega*, que representa uma transmissão de volume de um órgão origem a um órgão destino, passando por uma determinada rota. Cada plano contém um *prazo*, isto é, o instante máximo em que todo o volume já deve estar armazenado no órgão destino. Um plano de entrega é formalmente definido pela tupla

$$pe = (t_i, o_i, t_d, o_d, p, v, r, pz),$$

onde $t_i \in T(o_i)$ é o tanque no órgão de origem $o_i \in Org$, $t_d \in T(o_d)$ é o tanque no órgão de destino $o_d \in Org$, v é o volume do produto $p \in P$, r é a rota de transmissão do plano e pz é o prazo do plano dentro do horizonte de programação. Considera-se que um plano de entrega é *indivisível*, ou seja, seu volume não pode ser fragmentado em dois ou mais planos. Contudo, um plano pode *parar* no duto, isto é, ser injetado de forma intermitente, o que acarreta também a parada de toda a rota envolvida no bombeamento. Operações de parada são essenciais, por exemplo, para satisfazer o numero máximo de envios simultâneos (restrição C7) e os alinhamentos proibidos (restrição C8). Além disso, também são necessárias quando se necessita aguardar esvaziamento de tanques por demanda para que tais tanques recebam quantidades proveniente do duto.

Assim, dadas as campanhas de produção e demanda em cada órgão, o objetivo da fase de planejamento é gerar um conjunto de planos de entrega de tal forma que, se todos forem *satisfeitos*, *i.e.*, entregues no prazo, garante-se uma solução que atenda todas as demandas da rede e escoe os volumes em excesso das produções.

Diferentes métodos heurísticos ou exatos podem ser aplicados para a criação dos planos de uma certa instância, os quais considerariam critérios específicos para a determinação das rotas, volumes e tanques. As técnicas heurísticas aplicadas neste trabalho baseiam-se em escolher os órgãos com as demandas mais *criticas*, isto é, com maior volume e mais

próximas do início do horizonte. Uma vez escolhidos, definem-se então os órgãos que vão prover volume para satisfazê-los, utilizando como critério aqueles mais *próximos*, isto é, conectados por um menor número de dutos. Por fim, são selecionadas as rotas de maior vazão e os volumes para fixar os demais parâmetros dos planos.

Além disso, visto que planos são também uma representação intuitiva do planejamento da rede, operadores dos dutos também são capazes de inserir novos planos ou alterar os existentes, muitas vezes para a representação de preferências de movimentação não previstos pelo algoritmo.

A fase de planejamento encapsula as decisões de quais rotas e volumes utilizar para a resolução de uma instância do PTRD, mas não detalha as operações de bombeamento e seus respectivos tempos. O desafio da fase de planejamento é, portanto, criar conjuntos de planos com uma grande probabilidade de serem viáveis, sem contudo aumentar a complexidade do problema a ponto de torná-lo não-resolvível.

Este projeto não focará na fase de planejamento, considerando-o como um módulo à parte já existente. O algoritmo utilizado para tal módulo será o de [19], o qual foi desenvolvido juntamente com esta pesquisa. Contudo, é possível que diversos outras metodologias sejam utilizadas tanto para a fase de planejamento como a de escalonamento, incentivando mais pesquisas sobre esse aspecto.

4.2 Fase de Escalonamento

Uma vez definidos os órgãos que irão enviar e receber produtos e as respectivas rotas por onde os volumes irão trafegar, deve-se resolver o problema de *escalonamento*, ou seja, determinar o número de operações de envios por plano de entrega e, principalmente, a ordem em que os bombeamentos serão realizados. Nesta fase, consideram-se as restrições de incompatibilidade e de capacidade dos tanques, além do cálculo de limitantes de tempo para envio e chegada dos volumes nos órgãos. Por fim, é necessário atribuir os exatos momentos em que ocorrerão as operações de bombeamento, compondo a solução final do problema. Este último deve garantir as restrições referentes às vazões de bombeamento, troca de turno, manutenção de dutos e demais restrições análogas.

Dado um conjunto de planos de entrega $PE = \{pe_1, pe_2, ..., pe_{npl}\}$, o objetivo da fase de escalonamento é gerar movimentos $m_1, ..., m_{nm}$, conforme definidos na seção 2.3, de forma que todos os planos sejam *cobertos* no prazo. Isto equivale a sequenciar e escalonar os planos nas suas respectivas rotas e tanques, garantindo todos requisitos do problema relacionadas no capítulo 2, como restrição de interface, capacidade de tanques e não-simultaneidade de operações. Note que $nm \geq npl$, já que planos podem ser bombeados por uma ou mais operações de envio.

Apesar do grande número de restrições, a pré-atribuição de rotas da fase de planeja-

mento abre a possibilidade de modelagens mais intuitivas e compactas do problema de escalonamento, já que as decisões ficam concentradas em como ordenar e temporizar os planos nos seus respectivos dutos.

Como conseqüência da dificuldade em garantir heuristicamente soluções operacionalmente viáveis, será utilizado um **modelo exato** como módulo resolvedor desta fase. Este modelo, por sua vez, será implementado usando *Programação por Restrições* (PR) [23], por duas principais razões. A primeira motivação decorre de restrições não-lineares inerentes ao seqüenciamento e às vazões variáveis, impossibilitando a modelagem do problema usando Programação Linear Inteira. Já a segunda razão vem do fato de focarmos, prioritariamente, a obtenção de soluções viáveis ao invés de ótimas. As técnicas em PR têm obtido grande sucesso neste quesito. Mais detalhes sobre Programação por Restrição são apresentados no capítulo 5.

Capítulo 5

Conceitos Básicos de Programação por Restrições

A Programação por Restrições (PR) [2, 23] é uma ferramenta de programação declarativa reconhecida, principalmente, pela sua eficiência na modelagem e resolução de problemas combinatórios difíceis [16]. Apesar de ter origem nos anos 70, seu uso intensificou-se bastante durante as décadas de 80 e 90, agregando hoje centenas de pesquisadores das áreas de Inteligência Artificial, Linguagens de Programação, Computação Simbólica, Lógica Computacional e Otimização Combinatória.

Há atualmente diversos cenários práticos em que o uso de PR possui grandes vantagens em comparação às outras tecnologias mais tradicionais. Entre tais casos de sucesso [2], é possível citar problemas de análise e síntese de circuitos, agendamento de tarefas em microprocessadores, sequenciamento de DNA em biologia molecular, alocação de funcionários em empresas aéreas e muitos outros problemas relacionados à agendamento em Pesquisa Operacional, o que motiva o estudo de PR para sua aplicação no PTRD.

De forma geral, os conceitos básicos de PR giram em torno de *restrições*. Uma restrição é definida como uma relação lógica entre um conjunto de variáveis, cada qual contendo um *domínio* de possíveis valores. As restrições, portanto, irão limitar os domínios de valores que as variáveis podem assumir simultaneamente. Por exemplo, dada duas variáveis $x \in \{1, ..., 5\}$ e $y \in \{1, ..., 5\}$, a restrição $x + y \leq 3$ impede, por exemplo, que tanto x como y tenham valores 3, 4 ou 5.

O poder de modelagem de PR recai nas propriedades interessantes que a definição genérica de restrição traz. Primeiramente, restrições especificam informações *parciais* e, portanto, não necessitam definir unicamente o valor de suas variáveis; $x + y \leq 3$, por exemplo, representa um relacionamento que possui por si só um conjunto infinito de soluções para x e y. Além disso, restrições também são *não-direcionais*. Do mesmo exemplo anterior, temos que $x \leq 3 - y$ ou $y \leq 3 - x$, isto é, é possível inferir novas

restrições sobre um literal a partir das outras variáveis que se relacionam com ele.

A principal das propriedades, entretanto, está no fato de que restrições são *declarativas*, ou seja, elas especificam o relacionamento entre variáveis sem necessariamente impor um procedimento computacional para garantir tal relacionamento. Conseqüentemente, é virtualmente possível definir restrições de qualquer natureza, lineares ou não. Um exemplo típico consiste em definir variáveis como *tarefas* e restrições que impedem que um certo número máximo de tarefas seja executado ao mesmo tempo em uma máquina. Por fim, restrições também são *aditivas*: não importa a ordem em que sejam postadas, contanto que a conjunção delas seja satisfeita pela solução final.

Uma vez definidas as restrições, o estudo de PR, portanto, recai em sistemas computacionais *resolvedores* que sejam capazes de gerar rapidamente soluções que satisfaçam todas as condições impostas no modelo. Para tanto, as estratégias de busca são baseadas essencialmente no conceito de *propagação de restrições*. A idéia consiste em reduzir os domínios das variáveis a partir da consistência *parcial* das restrições existentes, diminuindo, desta forma, o espaço de busca a ser explorado. No caso anterior em que $x+y \leq 3$ e $x, y \in \{1, ..., 5\}$, o sistema de propagação poderia eliminar $3, 4, 5$ do domínio de ambas as variáveis ao inferir que, para tais valores, teríamos $x + y > 3$. Por outro lado, a consistência é parcial porque ainda há uma combinação de valores no domínio resultante, $x, y \in \{1, 2\}$, que podem violar a restrição. O uso de técnicas de propagação é inerente às técnicas de busca aplicadas e será descrito nas subseções seguintes.

Podemos dividir a PR em dois ramos que, apesar de compartilharem a mesma terminologia, diferem na origem e nas tecnologias de busca de soluções: *satisfação de restrições* e *resolução de restrições*. A primeira trata de problemas cujas variáveis pertencem a domínios finitos e, assim, as soluções derivam basicamente de técnicas combinatórias. Já o segundo ramo lida com domínios infinitos ou mais complexos, e cujas metodologias de resolução provém do cálculo e da matemática contínua.

Neste projeto, será focado o ramo de *satisfação de restrições*, já que os domínios das variáveis do problema são finitos e bem delineados no horizonte a ser tratado. Além disso, tal ramo abrange a maior parte das aplicações industriais e possui um grande conjunto de métodos e políticas para agilizar o processo de busca.

5.1 Satisfação de Restrições

Define-se um problema de PR deste tipo a partir dos seguintes elementos [2]:

1. Um conjunto de variáveis de decisão $X = \{x_1, x_2, ..., x_n\}$.
2. Um conjunto finito D_i associado a cada x_i, com os possíveis valores que a variável pode assumir. Note que este domínio não é necessariamente numérico; pode, por

exemplo, representar valores booleanos ou estruturas específicas do problema.

3. Um conjunto de *restrições*, isto é, relações entre as variáveis que impedem a ocorrência simultânea de certos valores.

Uma *solução válida* é definida como uma atribuição de valor a cada variável x_i que respeite os domínios D_i e não viole nenhuma restrição do modelo. Portanto, uma formulação PR pode conter diversas soluções válidas, todas igualmente aceitáveis dentro do contexto de satisfação de restrições. Contudo, caso cada solução estivesse associada a um valor real representando um determinado *custo*, encontrar a solução de *melhor custo* caracterizaria um problema de *otimização de restrições*.

O processo de uso da tecnologia de PR é dividido em duas etapas. Em primeiro lugar, deve-se *modelar* o problema adequadamente, definindo as variáveis, seus domínios e as restrições que as relacionam. Uma vez criada a formulação PR, é necessário definir *estratégias de busca* para a atribuição das variáveis sem que nenhuma restrição seja violada. Cada uma dessas etapas em PR será descrita com mais detalhes a seguir.

5.1.1 Modelagem em PR

As técnicas de PR induzem naturalmente a uma modelagem do problema mais intuitiva e próxima da realidade, principalmente em decorrência da maior expressividade das restrições. Devido à sua origem estar intrinsecamente relacionada com Programação Lógica, formulações podem ser colocadas de forma *declarativa*, tal como em PROLOG [8], ou também de forma análoga às formulações em Programação Linear.

Para exemplificar a modelagem em PR, será mostrado um problema clássico de criptoaritmética, extraído de [23] e apresentado abaixo.

$$\begin{array}{cccccc} & \mathtt{S} & \mathtt{E} & \mathtt{N} & \mathtt{D} \\ + & \mathtt{M} & \mathtt{O} & \mathtt{R} & \mathtt{E} \\ \hline \mathtt{M} & \mathtt{O} & \mathtt{N} & \mathtt{E} & \mathtt{Y} \end{array}$$

O objetivo da criptoaritmética consiste em atribuir um número $i \in \{0, 9\}$ a cada uma das letras para que a operação de soma seja consistente. Uma das possíveis formas de modelar o problema seria atribuir diretamente uma variável inteira para cada letra, todas com domínio no intervalo de $[0, 9]$. Contudo, além da expressão aritmética para representar a consistência da soma, também é necessário impor que as variáveis (letras) assumam valores distintos. De forma declarativa no estilo de PROLOG, a formulação seria facilmente escrita como:

$$
\begin{aligned}
crypto(Vars) :- \\
&Vars = [\mathtt{S}, \mathtt{E}, \mathtt{N}, \mathtt{D}, \mathtt{M}, \mathtt{O}, \mathtt{R}, \mathtt{Y}], \\
&Vars :: [0, 9], \\
&\mathtt{M} \neq 0, \\
&\mathtt{S} \neq 0, \\
&1000.\mathtt{S} + 100.\mathtt{E} + 10.\mathtt{N} + \mathtt{D} \\
&+ 1000.\mathtt{M} + 100.\mathtt{O} + 10.\mathtt{R} + \mathtt{E} \\
&= 10000.\mathtt{M} + 1000.\mathtt{O} + 100.\mathtt{N} + 10.\mathtt{E} + \mathtt{Y}, \\
&\mathtt{all\_different}(Vars).
\end{aligned}
$$

Inicialmente, as variáveis do problema ($Vars$) e seus domínios são definidos. Em seguida, condiciona-se que `M` e `S` sejam não-nulos, pois são os primeiros dígitos dos números, e se escreve a expressão aritmética da soma. Por fim, impõe-se a condição `all_different`, pertencente a um conjunto de restrições denominadas *restrições globais*, para garantir que as variáveis tenham valores distintos entre si.

As restrições globais compõem um dos pilares de pesquisa em PR. Uma restrição global caracteriza-se, intuitivamente, como um conjunto de restrições mais elementares cuja estrutura global pode ser facilmente explorada pelos resolvedores [16]. No exemplo acima, a restrição global é equivalente a escrever, para cada par de variáveis $v_i, v_j \in Vars$, a desigualdade $v_i \neq v_j$.

Além de permitirem uma modelagem mais compacta do problema, também é possível implementar propagadores específicos para tais restrições, mais eficientes em restringir os domínios das variáveis. No caso de `all_different`, constrói-se usualmente um *grafo bipartido* para cada restrição global desse tipo, aplicando-se algoritmos específicos para explorar tal estrutura [7]. Catálogos de restrições globais e seus propagadores podem ser encontrados em [7] e [16].

A flexibilidade de modelagem de PR, principalmente pelo uso das restrições globais, será essencial para a modelagem mais completa do PTRD e para a obtenção de soluções válidas, como será descrito com detalhes nos capítulos 6 e 7.

5.2 Estratégias de Busca

A eficiência de um resolvedor PR está intrinsicamente relacionada ao modo como a busca de uma solução válida é realizada. A forma mais trivial baseia-se em, para cada variável, escolher e atribuir um valor arbitrário de seu domínio. Assim que todas as variáveis

estiverem atribuídas, deve-se então checar se alguma restrição foi violada. Tal método é denominada *gerar-e-testar* e, claramente, possui o menor desempenho.

Outra forma de busca sistemática, conhecida como *backtracking*, consiste em estender incrementalmente soluções válidas parciais, escolhendo-se repetidamente valores para as variáveis ainda livres. Assim que todas as variáveis pertinentes a uma certa restrição estiverem atribuídas, verifica-se então se ela foi violada. Caso positivo, o algoritmo retorna para a última variável que ainda possua valores alternativos em seu domínio, seleciona um novo valor e o procedimento é repetido. Apesar de ser mais eficiente que o procedimento de gerar-e-estar, o algoritmo de backtracking básico ainda é bastante ineficiente, principalmente pelo fato de que a violação de restrições é detectada muito tardiamente.

Desta forma, mecanismos de *propagação de restrições*, também chamados de técnicas de *consistência local*, são geralmente acoplados ao backtracking para contornar tal problema. Estes mecanismos buscam remover valores inconsistentes dos domínios a cada atribuição nova de valor, reduzindo o tamanho da árvore de backtracking e, assim, agilizando o processo de busca. Usualmente, os propagadores baseiam-se em um *grafo de restrições*, onde vértices representam variáveis e, as arestas, restrições do problema. A partir de tal grafo, são analisadas as inconsistências dos domínios usando algoritmos como *nó-consistência*, *arco-consistência* e *caminho-consistência* [23]. Além disso, restrições globais também implementam algoritmos de propagação mais específicos para os subproblemas que representam, tal como mostrado anteriormente com a restrição global `all_different`.

Para exemplificar a estratégia de backtracking, suponha, por exemplo, a formulação de criptoaritmética mostrada anteriormente. Antes do início da busca, o algoritmo de propagação já é capaz de eliminar valores dos domínios de todas as variáveis, como mostrado em [41]. Seja $v_1 \in \{0, 1\}$ o *vem-um* da coluna correspondente às variáveis $\mathtt{E}, \mathtt{O}, \mathtt{N}$ e $v_2 \in \{0, 1\}$ o *vai-um* da coluna $\mathtt{S}, \mathtt{M}, \mathtt{O}$. Temos, pela restrição aritmética, que:

$$v_1 + \mathtt{S} + \mathtt{M} = \mathtt{O} + 10.v_2.$$

Portanto, o primeiro dígito da palavra `MONEY`, `M`, é claramente v_2. Como $v_2 \leq 1$ e $\mathtt{M} \neq 0$, então o domínio de `M` pode ser reduzido para [1]. Por `all_different`, o valor 1 é removido de todos os demais domínios. A equação anterior torna-se, desta forma,

$$v_1 + \mathtt{S} = \mathtt{O} + 9.$$

Isto implica que $S \geq 8$ e, assim, o domínio de S é reduzido para $[8, 9]$. Como $v_1 + \mathtt{S} \leq 10$, $\mathtt{O} \leq 1$ mas, como o valor 1 foi removido pela restrição de `all_different`, o domínio de

O torna-se [0] e, conseqüentemente, $v_2 = 0$ e o domínio de S reduz-se para [9]. As demais variáveis, D, E, N, R, Y, têm como domínio resultante da propagação inicial o intervalo $[2, 8]$.

Assim, o mecanismo de backtracking se inicia com os domínios das variáveis já reduzidos. A próxima etapa consiste em escolher uma variável com domínio não-unitário e um valor de atribuição. Após escolhidos, a propagação de restrições é realizada novamente e, se alguma inconsistência for encontrada, escolhe-se um novo valor até que uma atribuição válida seja encontrada ou se prove que o problema não possui solução. Por exemplo, é possível escolher a variável N e o valor 6. Em seguida, a propagação removeria o valor 6 de todos os domínios, e o processo assim continuaria. Uma solução válida para o problema seria S = 9, E = 5, N = 6, D = 7, M = 1, O = 0, R = 8 e Y = 2.

As políticas de busca por atribuições válidas são essencialmente heurísticas e variam bastante quanto à ordem em que as variáveis ou domínios são testados [23]. Todo o espaço de busca pode ser coberto ou não, e também podem incluir mecanismos de propagação de restrições auxiliares.

Como exemplos, podemos citar as políticas de *first-fail*, que atribui inicialmente para as variáveis com os menores domínios; *most constrained*, que seleciona as variáveis relacionadas em mais restrições; e *split-domain*, que explora os domínios tal como um algoritmo de divisão-e-conquista. O método de busca ideal deve ser escolhido de acordo com as características do problema.

5.3 Otimização em PR

A qualidade de uma solução em PR é quantificada por meio de uma *função objetivo*, e as técnicas devem buscar maximizá-la ou minimizá-la, conforme o caso. Assim, um problema de *Otimização em PR*, além do conjunto de restrições a serem satisfeitas, também possui uma função objetivo que mapeia cada atribuição válida para um valor numérico, sendo que o objetivo é minimizar (ou maximizar) este valor.

A principal técnica para otimização em PR é o *Branch and Bound* [2], que utiliza uma função de avaliação heurística subestimadora (em problemas de minimização) para classificar atribuições parciais de soluções. O algoritmo se comporta tal como uma busca em largura: dentro de um conjunto de atribuições para uma variável, as soluções parciais com valores (heurísticos) acima de um determinado limite (ou *bound*) são descartadas. Este limite é inicialmente colocado como muito alto, e é substituído quando uma atribuição completa válida é encontrada.

No modelo do PTRD aqui estudado, o foco será dado apenas à satisfação de restrições, tanto devido à dificuldade do problema, como pela falta de dados de entrada referentes aos custos de bombeamento e de criação de interfaces entre produtos. A otimização do problema é relacionada, desta forma, como sugestão para trabalho futuro no capítulo 9.

Capítulo 6

Modelo Inicial para o Problema de Escalonamento

Prólogo

O artigo a seguir concentra-se na primeira versão da formulação aplicada à fase de escalonamento do PTRD. Também é realizada uma descrição sucinta das heurísticas utilizadas para a geração dos planos.

Inicialmente, descreve-se o problema de forma geral e as restrições a serem tratadas. As heurísticas para a fase de planejamento são então citadas, seguidas da formulação da fase de escalonamento em PR. Por fim, alguns resultados para as instâncias reais fornecidas pela PETROBRAS são mostrados. Para auxilar a compreensão da formulação matemática do problema, a tabela 6 apresenta uma síntese da notação utilizada para as variáveis e constantes do modelo.

Este artigo foi aceito na conferência internacional *IEEE 11th International Conference on Computational Science and Engineering* (CSE08, `http://www.icmc.usp.br/~cse08/`), que ocorreu em São Paulo, Brasil, nos dias 16 a 18 de julho de 2008. Neste ano, cerca de 30% de mais de 190 trabalhos foram aceitos.

O artigo foi publicado nos anais da conferência, *Proceedings of the 11th IEEE International Conference on Computational Science and Engineering - CSE'08* [28], e eletronicamente, pela editora do *IEEE*. No decorrer deste capítulo, segue sua transcrição na íntegra, em inglês.

Notação	**Significado**
$Del*$	Prefixo para variáveis de planos de entrega (*delivery*)
$Dem*$	Prefixo para variáveis de planos de demanda
$Det*$	Prefixo para planos determinados
$Pd*$	Prefixo para variáveis de produto
$Pr*$	Prefixo para variáveis de planos de produção
$Und*$	Prefixo para planos não-determinados
$*Acc$	Sufixo para variáveis de acúmulo de volume
$*DueDate$	Sufixo para prazo de um plano de entrega
$*Dir$	Sufixo para variáveis de direção de fluxo nos dutos
$*FR$	Sufixo para variáveis de vazão (*flow rate*)
$*R$	Sufixo para o conjunto de dutos de uma rota
$*Pos$	Sufixo para variável de posição de um plano
$*PosOr$	Sufixo para variável de posição de um plano em seu tanque origem
$*PosDest$	Sufixo para variável de posição de um plano em seu tanque destino
$*TqOr$	Sufixo para tanque de origem de um plano
$*TqDest$	Sufixo para tanque de destino de um plano
$*T_i$	Sufixo para variáveis de tempo inicial
$*T_f$	Sufixo para variáveis de tempo final
$*V$	Sufixo para os volumes pré-definido de planos
$*Vol$	Sufixo para variáveis de volume
Adj_p	Pares de produtos/operações compatíveis no duto p
$DelRP_{i,j}$	Posição do plano i no duto j
Dem^{ord}	Conjunto de planos de demanda
Del^{ord}	Conjunto de planos de entrega
Del_p^{ord}	Conjunto de planos de entrega
$MaxFR_{i,j}$	Vazão máxima de um plano i no duto j
Pr^{ord}	Conjunto de planos de produção

Tabela 6.1: Notação utilizada no modelo PR.

Heuristics and Constraint Programming Hybridizations for a Real Pipeline Planning and Scheduling Problem

Arnaldo V. Moura, Cid C. de Souza, Andre A. Cire, Tony M.T. Lopes
Institute of Computing - University of Campinas
13081-970, Campinas, SP
{arnaldo, cid}@ic.unicamp.br, {andre.cire, tony.lopes}@gmail.com

Abstract

Pipeline network systems are considered the major option for transporting petroleum derivatives from refineries to local markets, in view of their many economic and environmental advantages. This article deals with a large real-world pipeline system planning and scheduling problem, in which different products should be transported in a pipeline network in order to supply market demands, while also satisfying hard operational constraints related to product sequencing, flow rates and tank capacities. We propose a novel hybrid approach based on two iterative phases comprised by a heuristic strategy and a Constraint Programming model. The resulting algorithm was tested with real-world instances yielding feasible solutions for all of them.

Keywords: constraint satisfaction, heuristic methods, scheduling, combinatorial algorithms, decision support.

1 Introduction

Planning and scheduling problems have been receiving increasing attention in the last few years, mainly due to the need to deal efficiently with large, complex real-world settings in order for companies to persevere in highly competitive markets. A notable example stems from the oil industry, in which ethanol and several petroleum derivatives, like gasoline, diesel, and naphtha must be transported from refineries to depots where consumer markets are located. In this context, pipeline networks are considered the principal transportation mode in comparison to rail, road, and vessels, since they are environmentally safer and much more economical. For instance, the Brazilian system managed by PETROBRAS has more than 11,300 km of pipelines and supplies 1,600 barrels per day [15].

A multiproduct pipeline network is composed of depots linked by bidirectional ducts. In order to operate the network, one must determine how product volumes should be transferred among depots, as well as their exact pumping time intervals, and how inventories should be managed. Local markets demands and production schedules, both represented by daily volumes at each depot, must be satisfied at all times. Furthermore, several conditions regarding pumping flow rates, product contamination and tank allocation must be taken into account.

Even though research in this field can lead to a significant operational cost reduction, the transportation planning and scheduling procedures are, typically, still executed manually in most oil companies. Moreover, studies from the literature usually focus on more restrictive and smaller pipeline topologies, each with a reduced set of operational constraints that, despite making the problem more tractable, are far from representing real-world scenarios. This is a consequence of the large number of hard sequencing and timing constraints involved, aggravated when large and more generic topologies are considered. In face of these difficulties, manual planning is usually deficient, and down-level operators must frequently apply corrections in pumping plans in order to make the system work appropriately.

We propose a new algorithm for generating feasible solutions for a very-large pipeline planning and scheduling problem, considering most of the hardest real-world constraints. Our approach has two phases: the *planning phase* is implemented as a constructive heuristic that generates *orders*, representing necessary transfers between two depots; and the *scheduling phase*, a Constraint Programming (CP) model, is used to assign time intervals to orders. The resulting algorithm, specially tailored to deal with large instances, generates more reliable pumping plans and can also be used to validate production and demand scenarios.

This project was a joint work with PETROBRAS, the Brazilian oil company, which provided both problem specifications and instances, as well as analyzed the proposed solutions. The prototype developed is being considered for use as an auxiliary tool to aid planners at PETROBRAS. This strategy proved to be very flexible, permitting the easier addition or removal of operational requirements, and facilitating the test of new search heuristics.

The article is organized as follows. Section 2 presents a literature review. Section 3 describes the problem and Section 4 discusses the solver. Finally, section 5 shows the computational results and section 6 presents some conclusions and points to further work.

2 Literature Review

In the literature, one of the most studied settings involves transporting petroleum derivatives from a single origin to multiple destinations. One pipeline connects an oil refinery to depots distributed in series along its segments. For this case, [9] developed a knowledge-based heuristic search for an Indian pipeline, composed of three depots, four products and a 1-month time horizon. Considering a more constrained problem, [16] presented two large-size discrete MILP formulations, strongly based on disjunctive programming, for scheduling a real-world Brazilian instance with five depots and four products. Later, [17] improved this model by adding integer cuts and redundant constraints.

In addition to the discrete formulations, [3] developed a continuous time and continuous volume model representation based on a MILP approach studied by [16]. Recently, another continuous-time model with non-linear constraints (MINLP) for this same configuration was proposed by [18], who covered pumping flow rate variation for a 130-hours time horizon.

Another studied topology was composed of a refinery, a pipeline, and a depot with a tank farm [10]. For this case, a decomposition strategy similar to the one proposed by [13] was developed, involving a MILP submodel, an auxiliary routine, and a main discrete MILP model. Improvements appeared in [11], now using CP techniques. A recent work [19, 21] presented a single continuous-time MILP for a case study proposed by a Portuguese oil distribution company. Although simpler in the operational details, it added inventory management constraints for a 1-month time horizon. An expansion was published in [20], with variable flow rates, intermittent pipeline operation, variable settling periods, and rescheduling.

There are also several approaches dealing with multiple origins and destinations within a complex pipeline network, albeit neglecting most of the hard constraints in order to make the problem tractable by the methods used. In [4], an heuristic method, known as *A-Teams*, was applied to a relaxed real-world pipeline system, but was unable to find any feasible solutions. A much more simplified problem version was solved by [5], which implemented a Genetic Algorithm for a directed tree network topology with 8 depots and only a few operational constraints. A similar work using evolutionary algorithms and MILP models is presented in [6].

A different approach is discussed in [14], which addressed the pipeline scheduling as a general-purpose artificial intelligence planner problem, albeit not effective for large instances. A more recent application of metaheuristics to more general topologies, but with relaxed operational constraints, can be found in [1, 7].

3 Problem Description

In this section, we give a cursive description of the problem, as may be conveyed by field engineers. In the sequel, a more formal discussion will be presented. As an illustration, Figure 1 shows a sample network with 4 depots, B_0, B_1, B_2, and B_3, interconnected by 5 pipelines. Between depots B_2 and B_3, there are 2 pipelines, which is a common scenario in practice. Each depot also has its own tank farm. For instance, depot B_1 has storage tanks for products P_0, P_1 and P_2. Each tank contains an initial volume, shown in standardized units. Ducts are shown completely filled with product volumes, also given in standardized units. In this network, products P_0 and P_2 cannot make contact since they are not compatible.

3.1 Problem Constraints

1. Pipelines operate in an intermittent fashion and must always be completely filled. In order to pump out a certain volume from one side, the same volume must be pumped in from the other extremity. No interface losses are considered. Volumes pumped-out can either enter a tank or move directly into another pipeline. An initial list of products inside each pipeline, with their respective volumes and destinations is known.

2. At each depot there are tanks to store certain products, but it is not necessary to have tanks for all products. A tank can store only one pre-defined product and its capacity must always be respected. Additionally, a tank is not allowed to pump into more than one pipeline at a time, and all operations in a tank must be disjunctive in time. The initial product level in each tank is known.

3. There is a maximum flow rate per pipeline, per pumping direction, and for each product. Flows must respect the least maximum rate for all products in a pipeline.

4. Some pairs of products can not make contact, else a quality loss will ensue. Products that can not make contact are called *incompatible*. It is possible, nevertheless, to use a third product, called a *plug*, to separate two incompatible products. Of course, the plug itself must be compatible with both products it separates.

5. A *route* is a sequence of depots and non-repeating pipelines that is traversed by a certain product volume. A volume in transit can only leave its route at the final destination tank. Although new routes can be created, choices dictated by human experience are to be preferred, since they might account for other implicit operational restrictions. A list of known routes is given.

6. Productions and demands are defined per depot and per product, each with its own time interval. However, since future estimates for such data are speculative, a production (demand) is redefined in a daily basis by linearly distributing its volume during the days that comprise its active time interval. The algorithm is at liberty to inject a production (extract a demand) from the system at any instant and at any rate during a day. Furthermore, a certain given percentage of such daily production (demand) can be postponed to the next day, as long as it does not violate its original time interval of activation.

3.2 Solution and Goals

A solution is defined as a set of continuos and nonpreemptive *pumping operations*. Each such operation contains information about the product, volume, route, origin and destination tanks, as well as start and end pumping times. Once a pumping operation is concluded, its corresponding volume must follow the designated route until it reaches the destination tank. However, a volume in transit may stop at any instant and at any point along its route, as long as it is not interleaved with other products. The main goal is to find a solution that satisfies both all operational as well as all production and demand constraints.

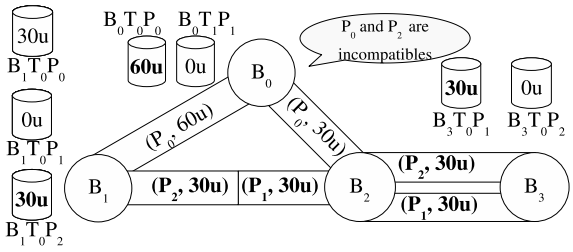


Figure 1: A pipeline network example.

4 The Solver Framework

When facing a complex and real sequencing problem, it is very common to decompose it [4, 10, 11]. One frequently used decomposition scheme, also applied here, divides the problem into two parts:

1. *Planning and Routing:* This sub-problem aims at satisfying all productions and demands by creating a set of *orders* that specify routes and volumes.

2. *Sequencing and Scheduling:* Upon receiving a set of *orders*, this sub-problem defines the sequence and exact times for pumping operations at depots, including those special operations used to store production and extract demands.

The accumulated experience at PETROBRAS showed that the first part is the easiest to handle manually. We followed some of their guidelines and human expertise when developing heuristic strategies for our *planning phase*.

The second part, the so called *scheduling phase*, is the hardest. At PETROBRAS it is solved manually by trial-and-error using a proprietary simulation software that can check simple constraints. In order to properly solve this part, we used CP model.

Although the planning phase is designed to produce a good set of orders, on occasion it might create infeasible sets. Or the scheduling phase may not be able to find a feasible solution in a reasonable time. In any case, the planning phase must be activated again, since it uses randomization and can produce a new and different set of orders. The interaction between the two phases is shown in figure 2.

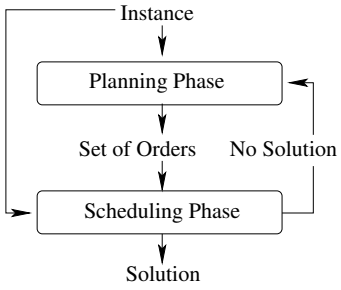


Figure 2: Interaction between phases in the solver.

4.1 The Planning Phase

The planning phase defines a set of orders necessary to satisfy all products demands within the time horizon. To this end, we developed a heuristic that incrementally builds a set of orders in a randomized and constructive way.

For any single order, its product, volume, origin depot, destination depot, origin tank, destination tank, route, and due date must be determined. These characteristics are determined sequentially, as described below:

1. *Product, Destination Depot, and Due Date:* For every triple of product p, destination depot bd and day t, we define $V_{p,bd,d} = H - (lastMinute(t) - TN_{p,bd})$, where H is the last minute in the horizon, and $TN_{p,bd}$ is the minute when enough of the

appropriate product is available, in all depots, to fully satisfy the demand at day t. The idea is to satisfy any pressing demands first. We randomly select some triple within the best α_1 percent triples for the most critical $V_{p,bd,t}$ values.

2. *Origin Depot and Route:* For every pair of origin depot bo and route r starting at bo and ending at a selected depot bd, we define $V_{bo,r} = (lastMinute(t) - (TD_{p,bo} + Tmin_{r,p}))/(1 + NR_r)$, where $TD_{p,bo}$ is the minute when there is any quantity of the product p available at bo, and $Tmin_{r,p}$ gives the minimum amount of minutes necessary for p to travel route r. Hence, the route with the greatest difference between earliest arrival and due times, for any volume occurring in it, will be selected. Since we endured to avoid flow direction reversals, the term NR_r captures the number of such occurences if route r is used. A pair among the best α_2 percent $V_{bo,r}$ values is randomly selected.

3. *Origin and Destination Tanks:* Let $V_{tq} = Cap_{tq}/(TAloc_{tq} + 1)$, where tq is a tank, Cap_{tq} is its capacity and $TAloc_{tq}$ is the volume so far assigned to it. A big V_{tq} indicates a large tank with few allocations. The origin tank to and the destination tank td are selected within the α_3 percent tanks with larger V_{tq} values inside the origin depot and the destination depot, respectively.

4. *Volume:* The volume of an order is determined by its origin and destination tanks. Assume that in some tank t all orders allocated before this new one have already been fulfilled, leaving a quantity $TQde_t$ in the tank. This way $Volume = \min\{Cap_{to} - TQde_{to}, Cap_{td} - TQde_{td}\}$. By executing orders in this sequence, no undertankage or overtankage will occur.

Three types of orders are created. There are *production* and *demand* orders that select tanks to accommodate productions in the origin depot and to satisfy demands in the destination depot, respectively. There are also *delivery orders*, which represent volume transfers throughout the chosen route. Delivery orders, in turn, are of two types: *determined*, which are orders generated by the heuristic, and *undetermined orders*. The later are created randomly and, although they are assigned a route, they do not carry information about the origin or destination tanks. Undetermined orders represent volumes that are not transfered to satisfy demands or store production, but are only used to push other volumes along the pipeline. They are also used to free space in tanks in order to accommodate future productions.

The planning phase ends when there is no more pairs of product and destination to be chosen, meaning that every demand has been satisfied. In real cases, at this moment there could be some production that was left unused. This production can usually be stored in its origin.

4.2 The Scheduling Phase

Given that the planning phase already selects routes and tanks, the scheduling phase must gather additional information to control the size of the model, incorporating special structures geared to effectively explore the search space. Notably, knowing which pipelines a delivery order will traverse provides lower bounds on the corresponding pumping volumes and flow rates. This eliminates incompatible order sequences in advance and makes it possible to infer special conditions to constrain time variables.

However, taking advantage of these structures using discrete MILP models, as studied in [10, 17], turned out to be impracticable for two main reasons. Firstly, most of the problem restrictions are computationally costly, or impossible, to model as linear constraints, specially those related with variable flow rates and direct flow connections between pipelines. Secondly, the MILP models would have to deal with multiple pipelines and depots, and investigation showed that the number of integer variables and constraints would increase at an unacceptable rate.

On the other hand, heuristics and meta-heuristics strategies per se, as described in [5, 6], are greatly impaired when too many operational constraints are considered. This is particularly disturbing when slight modifications in a solution give rise to collateral effects on the problem structure as a whole. For example, products can flow directly from one pipeline to another connecting pipeline. Thus, changing a single pumping start time may delay the arrival of a number of other products in many connected pipelines, which can turn the solution into an infeasible one. Repairing procedures could be an elegant choice to ameliorate this situation, but their bias can make the model loose robustness.

In face of all these issues, we propose a new CP model that explicitly takes advantage of the problem's diverse structures, while also providing the flexibility to consider new operational requirements or implement new searching heuristics. CP fundamentals can be found in [2, 12]. The CP programming paradigm is distinguished by a powerful modeling language, reinforced by *global constraints*, like `alldiff` and `cumulative`, that exploit specific problems patterns and provide specialized constraint propagation mechanisms [8]. CP is thus well-suited for the scheduling phase. In particular, it can easily model multiple representations in a compact way, which are then interconnected by *element* global constraints, thus taking advantage of each problem structure.

Our model comprises two different CP perspectives, or views, containing both specific variables and constraints to deal with the corresponding structures they focus on, as well as additional redundancy used to enforce constraint propagation. Firstly, the *orders* view will handle order restrictions. In the sequel, their variables for production, demand and delivery, respectively, will be identified by the prefixes *Pr**, *Dem** and *Del**. Secondly, the *resources* view will represent pipeline and tank restrictions, and related variables will be identified by the prefix *Op**. In addition, the following notation is used across

the mathematical models detailed below. For common variables suffixes we use $*Vol$ for volume; $*T_i$ and $*T_f$ for start and ending times; $*FR$ for flow rate; $*Dir$ for flow direction in pipelines; $*V$ for orders' pre-defined volumes; $*R$ for pipelines that occur in a route; and $*TqOr$ and $*TqDest$ for orders' pre-defined origin and destination tanks, respectively. Other prefixes include $Pd*$, for product; and $Det*$ and $Und*$, for marking delivery orders as determined or undetermined. Finally, Pip denotes the pipeline set and Tnk denotes the tank set. Note that $*T_i$ and $*T_f$ are continuous variables, while all others are integer variables. Domains are easily inferred.

The Order View

Constraints (1) to (4) below guarantee the satisfaction of production and demand orders. In this case, Pr^{ord} and Dem^{ord} stand for the set of orders and $[*T^{min}, *T^{max}]$ denotes their time interval.

$$PrT_i^{min} \leq PrTi_i, PrTf_i \leq PrT_i^{max}, \tag{1}$$

$$PrVol_i = PrV_i,\ \forall i \in Pr^{ord}. \tag{2}$$

$$DemT_i^{min} \leq DemTi_i, DemTf_i \leq DemT_i^{max}, \tag{3}$$

$$DemVol_i = (-1).DemV_i,\ \forall i \in Dem^{ord}. \tag{4}$$

Volumes can be pumped in a staccato manner into the pipeline network. The waiting time between two consecutive pump actions determines a pipeline stoppage time. The parameter **ns** limits the number of pump actions a delivery order can be broken into. Although this limits the maximum number of pipeline stoppages, it helps to avoid an excessive number of small pump actions for each order.

We consider now two sets of variables: the *entering* (overscript $^{(e)}$) and *leaving* (overscript $^{(l)}$) sets. They correspond, respectively, to pumping operations that inject or extract derivatives from a duct along a route. The constraints for the internal relationship in each set of variables is represented by constraints (5) to (9) below. In these equations, $MaxFR_{i,j}$ is the maximum flow rate for order i in pipe j, and the superscript $^{(*)}$ indicates that the constraint is defined for both sets of variables.

$$DelTi_{i,j,k}^{(*)} \leq DelTf_{i,j,k}^{(*)}, \tag{5}$$

$$DelVol_{i,j,k}^{(*)} = DelFR_{i,j,k}^{(*)}(DelTf_{i,j,k}^{(*)} - DelTi_{i,j,k}^{(*)}), \tag{6}$$

$$DelFR_{i,j,k}^{(*)} \leq MaxFR_{i,j}, \tag{7}$$

$$DelTf_{i,j,k}^{(l)} \leq DelDueDate_i, \forall i \in Del^{ord}, \tag{8}$$

$$\begin{aligned} &\forall j \in \{1, ..., \mathbf{ns}\}, \forall k \in DelR_i. \\ \sum_{j=1}^{\mathbf{ns}} DelVol_{i,j,k}^{(e)} &= \sum_{j=1}^{\mathbf{ns}} DelVol_{i,j,k}^{(l)} = DetV_i, \\ &\forall i \in Det^{ord}, \forall k \in DelR_i. \end{aligned} \tag{9}$$

The channeling between these two sets of variables along route segments is posed by constraints (10) to (12):

$$DelTi_{i,j,k}^{(l)} = DelTi_{i,j,k+1}^{(e)}, \tag{10}$$

$$DelTf_{i,j,k}^{(l)} = DelTf_{i,j,k+1}^{(e)}, \tag{11}$$

$$DelVol_{i,j,k}^{(l)} = DelVol_{i,j,k+1}^{(e)}, \tag{12}$$

$$\forall i \in Del^{ord}, \forall j \in \{1, ..., \mathbf{ns}\}, \forall (k+1) \in DelR_i.$$

The Resource View: Pipelines

Pipelines are represented as two time-ordered operations sequences: the *send* sequence (overscript $^{(s)}$) relates to operations entering a pipeline, and the *receive* sequence (overscript $^{(r)}$) relates to operations leaving a pipeline. An operation is considered to be a continuous pumping action and, thus, if a sending operation has finished before a receiving operation, the next sending operation must start immediately after the previous one, in order to push the volume along the route. The exact number of operations in each pipeline p is given by $n_p = 2.\mathbf{ns}|Del_p^{ord}|$.

Constraints which regulate time, flow direction and product incompatibilities are posed by constraint (13) to (16). Here, $^{(*)}$ means that they are defined for both types of sequences, and Adj_p is the set of operation pairs which may make contact inside a duct.

$$OpTf_{i,p}^{(*)} \leq OpTi_{i+1,p}^{(*)}, i \leq n_p - 1, \forall p \in Pip. \tag{13}$$

$$\mathbf{disjunctive}([OpTi_{i,p}^{(*)}, OpTf_{i,p}^{(*)}]_{i=1}^{n_p}), \forall p \in Pip. \tag{14}$$

$$(OpDir_{i,p}^{(*)}, OpDir_{i+1,p}^{(*)}) \in ValidDirPairs \tag{15}$$

$$\begin{aligned} &(OpPd_{i,p}^{(*)}, OpPd_{j,p}^{(*)}) \notin \{(pd_m, pd_n), (pd_n, pd_m)\}, \\ &\forall p \in Pip, \forall (i,j) \in Adj_p, \forall (m,n) \in IncompatPd \end{aligned} \tag{16}$$

The necessary channeling constraints between the in and out sequences is detailed next. First, we define the *accumulate* variables ($*Acc$), which represent the sum of the volumes sent and received until the end of each operation. The accumulate relations are

defined by constraints (17) to (18) below. The channeling and flow control constraints are expressed by equations (20) to (21).

$$OpAcc_{0,p}^{(*)} = OpVol_{0,p}^{(*)}, \tag{17}$$

$$OpAcc_{i,p}^{(*)} = OpAcc_{i-1,p}^{(*)} + OpAcc_{i,p}^{(*)}, \tag{18}$$

$$\forall i \in \{2, ..., n_p\}, \forall p \in Pip. \tag{19}$$

$$\begin{aligned} &OpAcc_{i,p}^{(s)} - (OpAcc_{j,p}^{(r)} - OpVol_{j,p}^{(r)}) > Vol_p \\ &\wedge\ (OpAcc_{i,p}^{(s)} - OpVol_{j,p}^{(s)}) - OpAcc_{j,p}^{(r)} < Vol_p \\ &\Longrightarrow OpTi_{i,p}^{(s)} = OpTi_{j,p}^{(r)} + \\ &+\mathbf{max}(0, (OpAcc_{i,p}^{(s)} - OpVol_{j,p}^{(s)}) - \\ &-(OpAcc_{j,p}^{(r)} - OpVol_{j,p}^{(r)}) - Vol_p)/OpFR_{j,p}^{(r)} \\ &\wedge\ OpTi_{j,p}^{(r)} = OpTi_{i,p}^{(s)} + \\ &+\mathbf{max}(0, Vol_p - ((OpAcc_{i,p}^{(s)} - OpVol_{j,p}^{(s)}) - \\ &-(OpAcc_{j,p}^{(r)} - OpVol_{j,p}^{(r)}))/OpFR_{i,p}^{(s)} \\ &\wedge\ OpFR_{i,p}^{(s)} = OpFR_{j,p}^{(r)} \\ &\forall i, j \in \{1, ..., n_p\}, \forall p \in Pip. \end{aligned} \tag{20}$$

$$\begin{aligned} &(OpAcc_{i,p}^{(s)} - OpVol_{j,p}^{(s)}) - OpAcc_{j,p}^{(r)} < Vol_p \\ &\Longrightarrow OpFR_{i,p}^{(s)} \leq OpFRMax_{j,p}^{(r)} \\ &\forall i, j \in \{1, ..., n_p\}, \forall p \in Pip. \end{aligned} \tag{21}$$

The Resource View: Tanks

For each tank t, only one operation sequence is needed, and its size is given by $n_t = |Del_t^{ord}|$. For undetermined delivery orders, an operation is created for each tank to which it might be assigned and only one such operation does not have a null volume. Tank constraints are represented by equations (22) to (26):

$$OpTf_{i,t} \leq OpTi_{i+1,t}, \tag{22}$$

$$OpAcc_{0,t} = OpVol_{0,t}, \tag{23}$$

$$OpAcc_{i+1,t} = OpAcc_{i,t} + OpVol_{i+1,t}, \tag{24}$$

$$\forall i \in \{1, ..., n_t - 1\},$$

$$0 \leq OpAcc_{i,t} \leq Capacity_t, \forall i \in \{1, ..., n_t\}, \tag{25}$$

$$\textbf{disjunctive}([OpTi_{i,t}, OpTf_{i,t}]_{i=1}^{n_t}), \forall t \in Tnk. \tag{26}$$

Channeling Constraints

In order to relate pipeline and order variables, we use `element` global constraints over *positional* variables. Each of these variables indicates a position in an order sequence associated with both a tank (suffixes $*PosOr$ for origin tanks and $*PosDest$ for destination tanks), as well as with pipelines along a route (suffix $*Pos$). Constraints (27) to (29) below govern pipeline sequencing, and constraints (30) to (42) relate to tank sequencing. In these equations, $DelP_p$ indicates delivery orders assigned to pipeline p and $DelRP_{i,p}$ denotes the position of p in order i.

$$\textbf{alldiff}([DelPos_{i,k}^{(*)}]_{i \in DelP_p, k \in DelRP_{i,p}}) \tag{27}$$

$$\begin{aligned}
\bigwedge_{j=1}^{\mathbf{ns-1}} \quad & DelTi_{i,j,k}^{(e)} = OpTi_{[\mathbf{ns}(DelPos_{i,k}^{(e)})+j],k}^{(s)} \\
\wedge \quad & DelTf_{i,j,k}^{(e)} = OpTf_{[\mathbf{ns}(DelPos_{i,k}^{(e)})+j],k}^{(s)} \\
\wedge \quad & DelVol_{i,j,k}^{(e)} = OpVol_{[\mathbf{ns}(DelPos_{i,k}^{(e)})+j],k}^{(s)} \\
\wedge \quad & DelFR_{i,j,k}^{(e)} = OpFR_{[\mathbf{ns}(DelPos_{i,k}^{(e)})+j],k}^{(s)}
\end{aligned} \tag{28}$$

$$\begin{aligned}
\bigwedge_{j=1}^{\mathbf{ns-1}} \quad & DelTi_{i,j,k}^{(l)} = OpTi_{[\mathbf{ns}(DelPos_{i,k})^{(l)}+j],k}^{(r)} \\
\wedge \quad & DelTf_{i,j,k}^{(l)} = OpTf_{[\mathbf{ns}(DelPos_{i,k}^{(l)})+j],k}^{(r)} \\
\wedge \quad & DelVol_{i,j,k}^{(l)} = OpVol_{[\mathbf{ns}(DelPos_{i,k}^{(l)})+j],k}^{(r)} \\
\wedge \quad & DelFR_{i,j,k}^{(l)} = OpFR_{[\mathbf{ns}(DelPos_{i,k}^{(l)})+j],k}^{(r)} \\
& \forall i \in Del^{ord}, \forall k \in DelR_i
\end{aligned} \tag{29}$$

$$DelTi_{i,0,0}^{(e)} = OpTi_{DelPosOr_i, DelTqOr_i}^{(s)} \tag{30}$$

$$DelTf_{i,\mathbf{ns},0}^{(e)} = OpTf_{DelPosOr_i, DelTqOr_i}^{(s)} \tag{31}$$

$$DelVol_i = OpVol_{DelPosOr_i, DelTqOr_i}^{(s)} \tag{32}$$

$$DelTi_{i,0,|DelR_i|}^{(l)} = OpTi_{DelPosDest_i, DelTqDest_i}^{(r)} \tag{33}$$

$$DelTf_{i,\mathbf{ns},|DelR_i|}^{(l)} = OpTf_{DelPosDest_i, DelTqDest_i}^{(r)} \tag{34}$$

$$DelVol_i = OpVol^{(r)}_{DelPosDest_i, DelTqDest_i} \quad \forall i \in Del^{ord}. \tag{35}$$

$$PrTi_i = OpTi_{PrPos_i, PrTq_i}, \tag{36}$$

$$PrTf_i = OpTf_{PrPos_i, PrTq_i}, \tag{37}$$

$$PrVol_i = OpVol_{PrPos_i, PrTq_i}, \forall i \in Pr^{ord}. \tag{38}$$

$$DemTi_i = OpTi_{DemPos_i, DemTq_i}, \tag{39}$$

$$DemTf_i = OpTf_{DemPos_i, DemTq_i}, \tag{40}$$

$$DemVol_i = OpVol_{PrPos_i, DemTq_i}, \forall i \in Dem^{ord}. \tag{41}$$

$$\textbf{alldiff}([DelPosOr_i, DelPosDest_j, PrPos_k, DemPos_l]_{(i,j,k,l) \in Ord_t}). \tag{42}$$

The Search Procedure

The strategy for searching a solution is based on a *backtrack* mechanism [12] in which the order for selecting variables is as follows. First, pipelines that are involved in a greater number of operations are chosen, and delivery orders with the earliest due dates are sequenced first in these pipelines (assignment of $*Pos$ variables). Then, the volumes and tanks for undetermined delivery orders that were sequenced first in these pipelines are assigned. This is followed by randomly choosing the sequence of tank orders (including production and demand orders as well). Finally, start and end times are assigned for all orders, which are again chosen by the earliest delivery deadline.

5 Results

The experiments were carried out on a *Pentium D* 3.40 Ghz CPU, with 4GB of memory. The planning and scheduling phases were coded in C++ and compiled using *GCC-4.0* with optimization flag O3. The CP model was solved using *ILOG Solver 6.2* and *ILOG Scheduler 6.2*, with medium to high propagation enforcement.

Part of a typical solution is presented in table 3. As described earlier, a solution is a sequence of pumping actions and each line in the table describes one such action. Column headings are as follows: T_i and T_f are start and end times (in minutes), respectively; Vol is the pumped volume (in m^3); P_d is the product code (G for gasoline, N for naphtha and D for diesel); Tk_{Or} and Tk_{Dt} are origin and destination tank codes, respectively; and *Route* is the route code. A table showing a full solution would continue for several hundred lines.

A summary of the results for two real instances are shown in table 2. Both instances share the same network topology composed of 14 depots, 29 pipelines, 32 products and

242 tanks distributed among the depots. As can be seen from section 2, these are far more complex instances than others that can be found in any previously reported work. Pipelines volumes range from 30 to 8,000 m^3, and most of the tank capacities are between 4,000 and 30,000 m^3. Instance 1 had a 10-day scheduling time horizon, while instance 2 had a 7-day scheduling horizon. The maximum number of pumping operations for each delivery order was set as $\mathbf{ns} = 7$. Also, parameters α_1, α_2 and α_3, described in section 4.1, were set to 10%. As shown in table 2, the algorithm quickly found a feasible solution, despite the size of the problems. The planning phase is the fastest and uses little memory, generating more than 800 orders for each instance. Of these, about 25% were delivery orders and the rest were production and demand orders. The scheduling phase consumed a large amount of memory and was the overall bottleneck. This reflects the sizable number of variables and constraints in the CP model. Experimental results showed that only a small fraction of the orders used up to **ns** pumping operations. In all cases the solver found a solution in a reasonable time, *e.g.*, within 5 minutes. The number of choice points (*i.e.*, variables assignment) and *fails* during the solver execution were greatly reduced by appropriately choosing a sequence for variable assignments, as discussed in section 4.2. The majority of variables were assigned as a result of constraint propagation, and this was greatly improved by the use of different model views linked by channeling constraints.

Figure 3 shows the total tankage for some product groups of instance 2. A group is formed by gathering compatible products. The figure exposes volume variations over time in a typical solution, covering the entire horizon of 7 days. Note that, in certain cases, there was a reduction of more than 15,000 m^3, attesting to a intense use of the pipeline network. Mainly, this volume variation indicates refilling activities necessary to maintain inventory levels, specially for those lines connected to depots with low-capacity tanks. For this same reason, some pipelines did not show the typical intermittent usage, but were continuously used throughout the time horizon (not shown in the figure).

The heuristic guidance provided in the planning phase was also instrumental to improve other important aspects of the solution quality, as noticed by logistic engineers. For instance, usually a typical solution showed only a very small number of pipeline flow reversions, a kind of operation that engineers prefer to keep to a minimum. Also, new and interesting routes were identified for product flow through the mesh, some of them surprising the logistic engineers, who were biased to use almost always the same routes.

6 Conclusions and Further Work

We proposed a novel algorithm for generating feasible solutions for a very-large multiproduct pipeline network, considering hard real-world constraints including product incompatibility, direct transmission between pipelines, flow rate limitations and tank capacities.

Table 2: Solver Results

Instance	1	2
Horizon	10 days	7 days
Orders	924	845
Planning Phase Time	1 min	1 min
Planning Phase Memory	350MB	309MB
Integer Variables	92,548	90,619
Continuous Variables	9,752	7,612
Constraints	417,215	401,363
Choice Points	2,561	2,451
Fails	5,232	4,952
Scheduling Phase Time	3 min	4 min
Scheduling Phase Memory	3.4 GB	3.2 GB
Total Time	4 min	5 min

Table 3: Solution Example

$\mathbf{T_i}$	$\mathbf{T_f}$	**Vol.**	**Pd**	$\mathbf{Tk_{Or}}$	$\mathbf{Tk_{Dt}}$	**Route**
2075	2362	858	*G*	T004	T005	SUG03
4857	4868	30	*N*	T160	T087	GUG03
4870	5111	722	*D*	T008	T005	BUG03
5111	5693	582	*D*	T111	T098	GRC01
6323	6753	2139	*D*	T004	T077	TRC01
6753	6871	582	*G*	T009	T012	SUG03
7211	7889	2030	*G*	T001	T007	TRC03
7890	8309	1256	*G*	T003	T005	SUG03
8309	8505	582	*D*	T023	T215	KDD2
...	...	...	...	...	...	...

The algorithm has two phases. The *planning phase* generates the so called *orders*, which are related to how product volumes should be transmitted between depots. An order contains information regarding product, volumes, origin and destination tanks, as well as delivery deadlines. In this phase, a number of randomized heuristics work together in order to select appropriate depots, tanks and volumes so as to satisfy demand and production schedules.

The *scheduling phase*, on the other hand, assigns start and end times for each order, satisfying all deadlines and operational constraints. Several reasons motivated the use of CP techniques to model the actions in this phase. The scheduling problem is highly over-constrained and have several non-linear constraints, easily modeled in the CP paradigm. Besides, the primary goal was to search for a feasible solution. The choice of variables for value assignment in the CP engine prioritizes variables related to orders with earliest

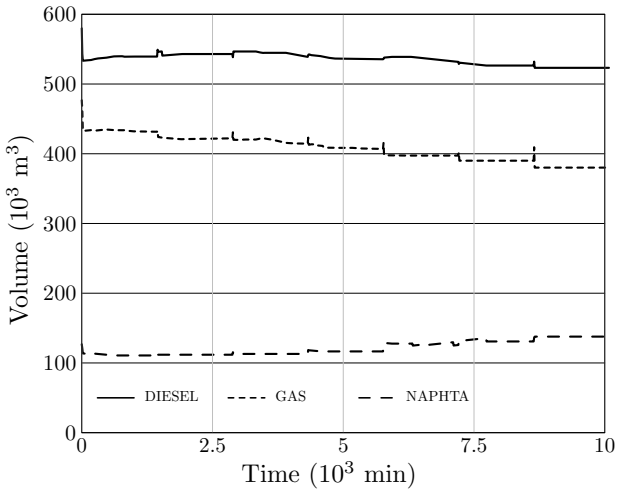


Figure 3: Instance 2 Tankage Evolution

delivery deadlines. When no solution was found in this phase, a new set of orders was requested from the planning phase.

As a result, the algorithm was able to find solutions for two real-world instances composed of 14 depots, 29 pipelines, 32 products, and more than 200 tanks, satisfying all hard operational constraints and offering a more reliable pumping plan to the pipeline operators. The prototype developed is being considered by PETROBRAS as an automatic assistant that could be used by the logistic engineers who have the responsibility of generating operational plans to run the entire pipeline network on a daily basis.

There are several opportunities for further research related to this problem. First, new real-world constraints are being considered to improve the adequacy of the overall model. These include inventory management restrictions and energy hourly limitations at the depots. Also, one can implement more sophisticated search heuristics for both the planning and scheduling phase, making the overall approach capable of dealing with more specific real instances classes. Finally, one can consider objective functions that would help guide the heuristics. This would provide a yardstick, based on a uniform cost grid, that could be used to gauge solution quality, thus allowing the search for better solutions across several problem instances.

Acknowledgment The authors gratefully acknowledge financial support from FAPESP under grants 05/57343-0 and 05/57344-7, from CNPq under grant 305781/2005-7, and from PETROBRAS under grant TI-SPS Nº 0200-0012829-05-2. The cooperation of Fernando Marcellino, Enio Medeiros, André Lima and Leandro Barcelos were also instrumental.

References

[1] V. Alves and V. J. Filho. Pipeline scheduling of petroleum derivatives using genetic algorithm. In *IV Congresso Brasileiro de Pesquisa e Desenvolvimento em Petróleo e Gás*, Campinas, Brazil, 2007.

[2] R. Bartak. Constraint programming: In pursuit of the holy grail. In *Proceedings of Week of Doctoral Students (WDS99)*, pages 555–564, Prague, Czech Republic, 1999.

[3] D. C. Cafaro and J. Cerdá. Optimal scheduling of multiproduct pipeline systems using a non-discrete MILP formulation. *Computers & Chemical Engineering*, 28(10):2053–2058, 2004.

[4] E. Camponogara and P. S. Souza. A-Teams for oil transportation problem through pipelines. In *Information Systems Analysis and Synthesis*, Orlando, United States, 1996.

[5] D. S. Crane, R. L. Wainwright, and D. A. Schoenefeld. Scheduling of multi-product fungible liquid pipelines using genetic algorithms. In *Proceedings of the 1999 ACM Symposium on Applied Computing*, pages 280–285, San Antonio, USA, 1999.

[6] J. de la Cruz, A. Herrán-González, J. Risco-Martín, and B. Andrés-Toro. Hybrid heuristic and mathematical programming in oil pipelines networks: Use of immigrants. *Journal of Zhejiang University SCIENCE*, 6A(1):9–19, 2005.

[7] E. M. S. Filho, V. J. Filho, and L. S. de Lima. Variable neighborhood search (VNS) applied to pipeline distribution problem with capacity constraints. In *IV Congresso Brasileiro de Pesquisa e Desenvolvimento em Petróleo e Gás*, Campinas, Brazil, 2007.

[8] J. N. Hooker. *Integrated Methods for Optimization (International Series in Operations Research & Management Science)*. Springer-Verlag, Secaucus, USA, 2006.

[9] M. M. Sasikumar, P. R. Prakash, S. M. Patil, and S. Ramani. Pipes: A heuristic search model for pipeline schedule generation. *Knowledge-Based Systems*, 10(3):169–175, 1997.

[10] L. Magatao, L. Arruda, and F. Neves. A mixed integer programming approach for scheduling commodities in a pipeline. *Computers & Chemical Engineering*, 28(1):171–185, 2004.

[11] L. Magatao, L. Arruda, and F. Neves. Using CLP and MILP for scheduling commodities in a pipeline. *Computer-Aided Chemical Engineering*, 20B:1027–1032, 2005.

[12] K. Marriot and P. Stuckey. *Programming with Constraints: An Introduction*. MIT Press, Cambridge, Massachusetts, 1^a edition, 1998.

[13] R. Más and J. M. Pinto. A mixed-integer optimization strategy for oil supply in distribution complexes. *Optimization and Engineering*, 4(1):23–64, 2003.

[14] R. Milidíu, F. dos Santos Liporace, and C. J. P. de Lucena. Pipesworld: Planning pipeline transportation of petroleum derivatives. In *Proceedings of ICAPS'03 - Workshop on the Competition: Impact, Organization, Evaluation, Benchmarks*, Trento, Italy, 2003.

[15] PETROBRAS. Relatório anual de atividades, 2004.

[16] R. Rejowski and J. M. Pinto. Scheduling of a multiproduct pipeline system. *Computers & Chemical Engineering*, 27(8):1229–1246, 2003.

[17] R. Rejowski and J. M. Pinto. Efficient MILP formulations and valid cuts for multiproduct pipeline scheduling. *Computers & Chemical Engineering*, 28(8):1511–1528, 2004.

[18] R. Rejowski and J. M. Pinto. A novel continuous time representation for the scheduling of pipeline systems with pumping yield rate constraints. *Computers & Chemical Engineering*, 32:1042–1066, 2008.

[19] S. Relvas, A. P. F. D. Barbosa-Póvoa, H. A. Matos, J. Fialho, and A. S. Pinheiro. Pipeline scheduling and distribution centre management - a real-world scenario at CLC. In *Proceedings of the 16th European Symposium on Computer Aided Process Engineering and 9th International Symposium on Process Systems Engineering*, pages 2135–2140, Garmisch-Partenkirchen, Germany, 2006.

[20] S. Relvas, H. A. Matos, A. P. F. D. Barbosa-Póvoa, and J. Fialho. Reactive scheduling framework for a multiproduct pipeline with inventory management. *Industrial & Engineering Chemistry Research*, 46(17):5659–5672, 2007.

[21] S. Relvas, H. A. Matos, A. P. F. D. Barbosa-Póvoa, J. Fialho, and A. S. Pinheiro. Pipeline scheduling and inventory management of a multiproduct distribution oil system. *Industrial & Engineering Chemistry Research*, 45(23):7841–7855, 2006.

Epílogo

O modelo PR apresentado neste trabalho ainda não modela as seguintes restrições do problema: tanques vazios quando recebem produção local (C4), tempo de certificação (C5), número máximo de envios simultâneos (C7), alinhamentos proibidos (C8) e manutenção de dutos (C15). Mesmo assim, apresenta um modelo inovador e mais abrangente para o PTRD, sendo capaz de gerar soluções para instâncias reais fornecidas pela PETROBRAS.

Cada instância era composta por 14 órgãos, 29 dutos, 32 produtos e cerca de 240 tanques distribuídos na rede. O horizonte resolvido variava de 7 a 10 dias. Devido à dificuldade de coletar dados de entrada, apenas algumas instâncias foram testadas e resultados para duas foram apresentados, já que as demais ainda apresentavam inconsistências nas tabelas de entrada.

Dentre as dificuldades enfrentadas, destacava-se o fato de que as soluções ainda estavam pouco operacionalizáveis na prática, apesar do grande número de restrições consideradas. Um fator que dificultou a expansão do modelo estava relacionado ao parâmetro ns, o qual representa o número máximo de operações de bombeamento para cada plano de entrega. Quando ns é pequeno, há poucas possibilidades de parada dos planos nos dutos, o que sempre força bombeamentos maiores e elimina diversas possibilidades de solução. Por outro lado, um valor alto para ns cria diversas operações desnecessárias nos dutos, aumentando excessivamente o número de variáveis e o tempo de propagação na execução do modelo.

Desta forma, o modelo é mais adequado para topologias menores e cenários reais nos quais as restrições de alinhamento e envios simultâneos não sejam críticas. Sua implementação também é relativamente simples quando comparado ao modelo descrito no capítulo 7 e, assim, torna-se mais vantajoso quando a rede de dutos não apresenta todas as restrições operacionais do PTRD.

Capítulo 7

Modelo Aprimorado para o Problema de Escalonamento

Prólogo

O artigo a seguir apresenta em detalhes o novo modelo desenvolvido para a fase de escalonamento, o qual utiliza estruturas de dados mais eficientes e faz extenso uso de *restrições globais* específicas para escalonamento em PR. Além disso, novos métodos de busca de solução também são propostos.

Neste trabalho, as restrições do problema são descritas e o uso de PR é fundamentado. Em seguida, explica-se a arquitetura geral do modelo e a fase de Planejamento é citada, para então a formulação em PR ser extensamente detalhada.

Este artigo foi aceito em uma das conferência internacionais mais importantes na área de Programação por Restrições, a *The 14^{th} Principles and Practice of Constraint Programming* (CP'08), que ocorreu em Sydney, Austrália, nos dias 14 a 18 de setembro de 2008. O artigo foi publicado nos anais do congresso, na série *Lecture Notes in Computer Science* [29]. Além disso, também recebeu o prêmio de *Best Paper Award* no *Applications Track* nesta mesma conferência.

A versão em inglês aqui transcrita refere-se ao texto final submetido à conferência. A notação das variáveis na formulação matemática do texto seguem um padrão simples descrito com detalhes no texto.

Planning and Scheduling the Operation of a Very Large Oil Pipeline Network

Arnaldo V. Moura, Cid C. de Souza, Andre A. Cire, Tony M.T. Lopes
Institute of Computing - University of Campinas
13081-970, Campinas, SP
{arnaldo, cid}@ic.unicamp.br, {andre.cire, tony.lopes}@gmail.com

Abstract

Brazilian PETROBRAS is one of the world largest oil companies. Recurrently, it faces a very difficult over-constrained planning challenge: how to operate a large pipeline network in order to adequately transport oil derivatives and biofuels from refineries to local markets. In spite of being more economical and environmentally safer, the use of a complex pipeline network poses serious operational difficulties. The network has a complex topology, with around 30 interconnecting pipelines, over 30 different products in circulation, and about 14 distribution depots which harbor more than 200 tanks, with a combined capacity for storing up to 65 million barrels. The problem is how to schedule individual pumping operations, given the daily production and demand of each product, at each location in the network, over a given time horizon. We describe a solution based on a two-phase problem decomposition strategy. A novel Constraint Programming (CP) model plays a key role in modeling operational constraints that are usually overlooked in literature, but that are essential in order to guarantee viable solutions. The use of CP was crucial, since it allowed the modeling of complex constraints, including nonlinearities. The full strategy was implemented and produced very adequate results when tested over large real instances. In contrast, other approaches known from the literature failed, even when applied to much less complex networks.

1 Introduction

PETROBRAS is ranked as the 14th largest oil company in the world (see `www.energyintel.com`). One of the major sources of costs faced by PETROBRAS is related to transportation, specially regarding petroleum derivatives, such as gasoline, and biofuel, like ethanol. In this context, pipeline networks are considered the main inland transportation mode in contrast to rail and road, since they are much more economical and environmentally safer.

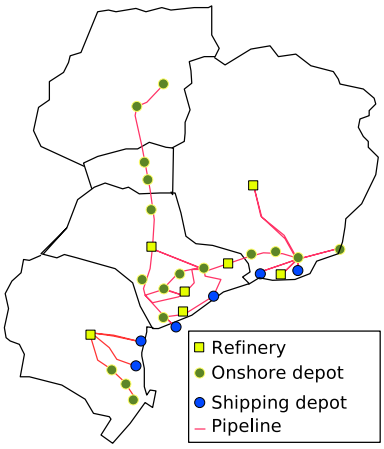


Figure 1: PETROBRAS pipeline networks.

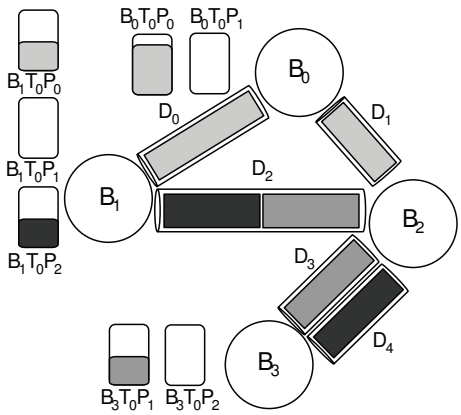


Figure 2: A pipeline network example.

However, these advantages ensue a very high operational complexity. For instance, the Brazilian pipeline network owned and operated by PETROBRAS has an extension of 7,000 kilometers, comprising 29 individual interconnecting pipelines in which more than 30 different types of products are in circulation. There are 14 distribution depots that can store up to 65 millions barrels of these products, stocked in more than 200 tanks located at such depots. A partial illustration of the Brazilian southeastern network is shown in Figure 1. Pipelines must always be completely filled with products, meaning that a volume must be *pushed* into a duct in order to pump out the same volume at the other extremity. Moreover, due to chemical properties, certain products can not make contact with each other - they are called *incompatibles*. Also, each product has its own flow rate interval, and that depends on the flow direction and on the particular pipeline being used. At depots, not all departing and arriving operations can be simultaneous, due to restrictions imposed both by the internal valve and ducts layout, as well as by the number of local pumps. Tanks can store just one type of product, and extraction or injection of volumes can not be simultaneous.

The problem is how to schedule all individual *pumping operations* in order to fulfill market demands and store all the planned production. Each pumping operation is defined by origin and destination tanks, a pipeline *route*, start and end times, a specific product and its respective volume. The operations must obey all constraints over the given time horizon. The management of all these resources gives rise to a complex planning and scheduling problem.

Currently, the problem is solved manually by executing a trial-and-error process with the aid of a proprietary simulator that checks whether some simple physical constraints are being satisfied. This process is very time consuming and, not rarely, the final results still violate some of the more complex restrictions. Clearly, this manual process is far

from optimal and limits the efficiency of the network operation. In fact, it is common for the company to use trucks for transporting pending volumes, thus increasing the overall transportation costs, a situation that could be avoided by a more intelligent use of the pipeline network.

Due to its size and complexity, as well as to its financial impact, the efficient operation of this large oil pipeline network is one of the most strategic problems faced by logistics at PETROBRAS today. As will be discussed later, CP was at the core of a computational model devised and used to find good operational solutions for real problem instances, in an adequate amount of computer time.

Problem Description. As an illustration, Figure 2 shows a sample network with 4 depots, B_0, B_1, B_2, and B_3, interconnected by 5 pipelines. Between depots B_2 and B_3, there are 2 pipelines, which is common to occur in practice. Each depot also has its own tank farm. For instance, depot B_1 has storage tanks for products P_0, P_1 and P_2. Each tank contains an initial volume. Ducts must always be completely filled. All of these quantities are measured in standardized units.

The following constraints must be satisfied:
(1) During the whole planning horizon, a tank can store only a pre-defined product and its capacity must always be respected. But a depot not necessarily contains tanks to store all types of products. All injection and extraction operations in a tank must be disjunctive in time.
(2) Pipelines operate in an intermittent fashion and must always be completely filled. No interface losses between products are considered. Furthermore, volumes pumped out can either enter a tank or move directly into another pipe in an assigned route. The initial sequence of products inside each pipe is given.

Flows in pipelines can change direction dinamically, an event called *pipeline flow reversal*. An example of reversal is illustrated in Figure 3 for a single pipeline topology. From instants $t = 0$ to $t = 2$, a product extracted from tank $B_0T_0P_2$ in depot B_0 is being used to push another product into the tank $B_1T_0P_0$ in depot B_1. As soon as the first product is completely injected into its destination tank at $t = 3$, the second volume must return to the first depot, since there is no tank for it in depot B_1. This is done by using the product from tank $B_1T_0P_1$ to push it back to the origin tank, changing the pipeline direction at $t = 4$ and $t = 5$.
(3) Depending on the internal arrangement of a depot, certain operations can not be active simultaneously. Such sets of operations are called *forbidden alignment configurations*. Also, each depot has an upper limit on the number of outgoing pump operations, which depends on the number of available pumps.
(4) A *route* is an alternating sequence of depots and non-repeating connecting ducts.

For example, the sequence $(B_0, D_1, B_2, D_3, B_3)$ represents a valid route in figure 2. Each product in circulation must have a route assigned to it, and a volume can only leave its route at the final destination tank. Although there is no restriction barring the creation of new routes, the most common choices obtained from human experience should be preferred.
(5) Least maximum flow rates among all products in any route must be enforced.
(6) To separate two incompatible products, it is possible to use a third product, called a *plug*, compatible with both products it separates.
(7) Production and demand volumes are defined per depot and per product, each with its own duration interval.

A solution is defined by a set of *pumping operations*. Each such operation is taken as a continuous and atomic pumping stream. An operation is defined by specifying information about the product, volume, route, origin and destination tanks, as well as start and end pumping times. Once a pumping operation starts, the volume must follow its designated route until it reaches the destination tank. However, a pumping operation can be stopped at any time, as long as no pumped volume (*i.e.* a volume that composes a whole operation) is interleaved with other products at any intermediate depot along its assigned route. The main goal is to find a solution that respects all operational and physical constrains of the network, as well as that uses stocks and productions to satisfy all local demands, while storing away any remaining production.

2 Why CP and Related Work

Previous studies from the literature frequently have focused on more restricted or much smaller network topologies. Usually, they consist of a single pipe connecting one origin (a refinery) to multiple depots. Different problem decompositions together with several MILP formulations [1, 2, 3, 4] were proposed for these cases. Some studies also deal with variable pumping flow rates and other non-linear constraints [5, 6]. Other approaches handle multiple origins and destinations within a more realistic network, albeit neglecting most of the hard constraints in order to make the problem tractable [7]. In [8], a MILP based on a network flow model was created to solve a relaxed version of the problem, but it took more than 50 hours of computer time only to find the LP initial basis.

As our research indicated, taking advantage of the problem structure using single MILP models is not practical for two reasons. First, most of the problem restrictions are computationally costly, or even impossible, to model as linear constraints, specially those related with variable flow rates and transmission between pipelines. Besides, MILP models would have to deal with multiple pipelines and depots, and investigation showed that the number of integer variables and constraints would increase at an unacceptable rate. On

the other hand, heuristic and meta-heuristic strategies *per se* are greatly impaired when too many operational constraints are considered. This is particularly disturbing when slight modifications in a solution give rise to serious collateral perturbations over the problem structure as a whole. For example, since products can flow directly from one pipeline to another, changing a single pumping start time may delay the arrival of a number of other products that pass through connected pipelines. This can easily render a candidate solution into an infeasible one.

In face of all these issues, the use of CP was seen to offer great advantages for modeling and solving this problem. Firstly, its powerful modeling language allowed for the implementation of operationally crucial constraints, besides providing enough flexibility to extend the model if new restrictions were risen by pipeline operators. Secondly, and most importantly, it was possible to exploit specific problems patterns explicitly. This is done, for instance, by modeling multiple subproblem representations in order to use specialized and adequate constraint propagation mechanisms to solve each of the subproblems. In fact, such multiple perspectives played an important role in the final model, greatly improving domain reductions. Furthermore, a preliminary study [9] already indicated that CP would be flexible and powerful enough to treat the real problem faced by PETROBRAS. Finally, the use of CP was further fostered by its well-known good performance when treating scheduling problems [10]. In addition, CP is more suitable for our case since any feasible solution is enough.

3 How CP ?

The complete problem was solved using a hybrid approach that combined a randomized constructive heuristic and a novel CP model. The hybridization main cycle is schematically presented in Figure 4. The *planning phase*, implemented as a constructive heuristic,

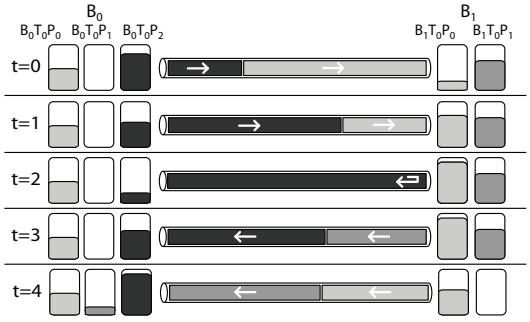


Figure 3: Example of a flow reversal.

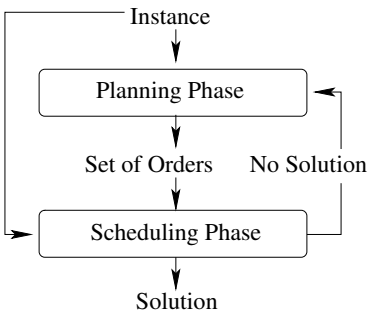


Figure 4: Solver Framework.

is responsible for creating a set of *delivery orders*. Each such order is defined by a volume,

origin and destination tanks, product type, route and a delivery deadline. The planning phase must guarantee that, if all delivery orders are completed within their respective deadlines, local market demands will be fulfilled and the excess production will be correctly stored away. The *scheduling phase* takes the set of delivery orders generated by the planning phase. It must both sequence the pumping operations at the initial pipeline in each route present in a delivery order, as well as determine the start times of each of the pumping operations, while ensuring that no network operational constraint is violated at any time. The scheduling phase represents the problem's central decision process and it was implemented as a CP model. In the sequel, each phase will be discussed, with the CP model described in more detail.

Planning and Routing. To generate delivery orders, we created a randomized constructive heuristic that makes use of the accumulated experience at PETROBRAS. The purpose of the randomization is to generate diversified sets of orders in case the main cycle restarts, increasing the chance of finding solutions. Also, it takes into consideration other criteria that are difficult to handle manually, such as estimating the time for product volumes to arrive at depots.

Delivery orders are created incrementally as follows: **(1)** randomly select a local product demand in any depot, giving higher priority to demands that must be fulfilled earlier in time; **(2)** randomly choose depots that could supply volumes of the required products, as well as the routes that these volumes should traverse. In order to do so, consider factors such as pipeline occupation rate, production schedules, present product stocks and estimated time of arrivals; **(3)** select origin and destination tanks, setting order volumes accordingly. Also, set order deadlines so as to guarantee demand fulfillment.

As soon as there is no more demands to choose from, the planning phase ends. At this point network operators can interfere adding, modifying, or removing orders according to their particular needs. This flexibility is interesting since sudden needs might unexpectedly arise. For example, there might be exceptional cases where the operators want to empty certain tanks for emergency maintenance purposes. This could be achieved by issuing new orders that remove products from those particular tanks.

All demands are guaranteed to be satisfied if the resulting orders can be scheduled to arrive by their respective deadlines. Of course, at this stage, it is not possible to know if the whole pipeline network can be operated in a way that meets all delivery order deadlines, while satisfying all problem constraints.

Orders are *indivisible*, i.e., once a volume starts to be pumped in, no other pump operation, at that same origin, can be started before the first one completes. However, orders are *preemptive* in the sense that they can be interrupted and be resumed at a later time. For instance, it is possible that a segment of the route that is being used in this

pumping must also carry other products, with more pressing deadlines, along another route that has that segment in common. In such cases, it may be necessary to interrupt the present pumping operation, allowing for the more urgent products to circulate in the common pipeline segment, later resuming the first pumping.

Sequencing and Scheduling Orders. The scheduling phase must determine the pumping parameters in order to meet all delivery order deadlines, also taking into account the network operational constraints. Or it must prove that the present set of delivery orders can not be sequenced nor have their start times set in a way that observe all their assigned deadlines. At this point, orders already have their routes, volumes and origin/destination tanks assigned by the planning and routing phase, besides their deadlines.

In a typical scenario comprising 14400 minutes (*i.e.* 10 days), the model is expected to deal with around 900 delivery orders, involving dozens of products, leaving and reaching several tanks, circulating through many interconnected pipelines, and subject to thousands of constraints. In order to cope with this problem complexity, the CP model was further divided into two steps. A first model treats the sequencing of delivery orders, generating *time intervals* for the start of the respective pumping operations. After one such sequencing is completed, the most difficult constraints are guaranteed to be satisfied. Then a second, simpler, CP model takes over and determines the number of pumping operations for each delivery order (*i.e.* taking into account possible preemptions), as well as the start time of each operation.

All time variables represent minutes, the unit currently adopted by network operators. Therefore, all variables have integer domains. Time value roundings, *e.g.* due to some particular combination of flow rate and pipeline extension, can be safely neglected given the large volumes that are involved. Variable domains are easy to infer from the input data instances and are not further detailed here.

The Sequencing Model. This model must take into account product pair incompatibilities, tank capacities, pipeline flow direction restrictions and other essential operational constraints, such as no two products being pumped into a pipeline simultaneously. Furthermore, it must consider order deadlines and flow rates, in order to determine valid time bounds for the pumping operations.

The model interrelates two different *viewpoints* [11]. Firstly, the *order viewpoint* provides a *global* view of the problem, dealing mainly with routes and volume transmission between pipelines. In contrast, the *operations viewpoint* captures a *local* view of the problem, representing the pumping operation constraints in each pipeline. Both viewpoints are connected by *channeling* constraints.

The Order Viewpoint. The *order viewpoint* handles the problem globally, focusing on the relationship between orders and the pipelines that occur in their assigned routes. It also enforces constraints related to flow rates, deliver deadlines, disjunctions of pipeline operations, product incompatibilities, and tanks.

Let $\mathbf{P}$ be the pipeline set, $\mathbf{T}$ the tank set and $\mathbf{O} = \{o_1, \ldots, o_n\}$ the set of delivery orders received from the planning phase. For each $o_i \in \mathbf{O}$, let $\texttt{route}(o_i) = (p_l, \ldots, p_m)$ be the sequence of pipelines that order o_i must traverse. For each $p \in \texttt{route}(o_i)$, the volume specified by o_i can have one of four possible pipeline flow attributes when traversing pipeline p: N, if it follows the normative, or preferred, pipeline flow direction; R, if it follows the reverse direction; NR, if it starts in the normative direction, but later changes to the reverse direction, thus leaving the pipeline through the same extremity it was pumped into; and RN, similar to NR but starting in the reverse direction. Let variable $\texttt{direct}_{i,p}$ specify one among such possibilities. Finally, let $\texttt{origin}(o_i)$, $\texttt{destin}(o_i) \in \mathbf{T}$ be the origin and destination tanks, respectively, for order o_i.

For each $o_i \in \mathbf{O}$ and $p \in \texttt{route}(o_i)$, we define two *activities* [12], $snd_{i,p}$ and $rcv_{i,p}$, each composed by start and end time variables and an inferred non-negative duration variable. The first activity represents the time interval during which order o_i is being pumped into p, while the second represents the time interval during which the order is being pumped out of p. Using these activities, we can give bounds on flow rates and state delivery deadline constraints for each $o_i \in \mathbf{O}$ and each $p \in \texttt{route}(o_i)$:

$$EndTime(rcv_{i,p}) \leq \texttt{deadline}(o_i), \tag{1}$$

$$Duration(snd_{i,p}).\texttt{max\_flow\_rate}_{p,\texttt{direct}_{i,p}} \geq \texttt{volume}(o_i), \tag{2}$$

$$Duration(rcv_{i,p}).\texttt{max\_flow\_rate}_{p,\texttt{direct}_{i,p}} \geq \texttt{volume}(o_i). \tag{3}$$

Before an order exits a pipeline, it must first traverse all the pipeline extension. Thus, for each $o_i \in \mathbf{O}$ and $p \in \texttt{route}(o_i)$, we require:

$$StartTime(rcv_{i,p}) \geq StartTime(snd_{i,p}) + \left\lfloor \frac{\texttt{volume}(p)}{\texttt{max\_flow\_rate}_{p,\texttt{direct}_{i,p}}} \right\rfloor. \tag{4}$$

When an order is being pumped out of a pipeline, it is immediately pumped into the next pipeline in its route, without volume loss. This can be done by *unifying* [13] send and receive activity variables in the following way. For each $o_i \in \mathbf{O}$ and for each pair (p_l, p_m) of consecutive pipeline pairs in $\texttt{route}(o_i)$, let:

$$StartTime(rcv_{i,p_l}) = StartTime(snd_{i,p_m}), \tag{5}$$

$$EndTime(rcv_{i,p_l}) = EndTime(snd_{i,p_m}). \tag{6}$$

Order activities in a pipeline must all be *disjunctive* with respect to time; a send (or receive) activity from a certain order must not overlap with the send (or receive) activity of other orders in the pipe. In order to guarantee this, for each $p \in \mathbf{P}$, we define two *unary resources*[1]: $SndResource_p$ and $RcvResource_p$, and we associate the send and receive activities to these resources, respectively. Since an unary resource defines a mutually exclusive relationship between activities that are linked to it, each resource constraint is ranked during the solving process, *i.e.*, it is ordered along the time line. This ranking is explicitly represented in our model using positional variables $sndPos_{i,p}$ and $rcvPos_{i,p}$, accounting for, respectively, the send and receive activities positions of order o_i in pipeline $p \in \texttt{route}(o_i)$. The positional variables are connected directly to the resource's *precedence graph* [10, 12], expressed by the constraints:

$$snd_{i,p} \quad \textbf{startsBefore} \quad snd_{j,p} \Longleftrightarrow sndPos_{i,p} < sndPos_{j,p}, \tag{7}$$

$$rcv_{i,p} \quad \textbf{startsBefore} \quad rcv_{j,p} \Longleftrightarrow rcvPos_{i,p} < rcvPos_{j,p}, \tag{8}$$

$$\forall o_i, o_j \in \mathbf{O},\ \forall p \in \texttt{route}(o_i) \cap \texttt{route}(o_j).$$

We also add redundant *all different* global constraints [13]. For each $p \in \mathbf{P}$,

$$\textbf{all\_diff}(sndPos_{i,p}) \wedge \textbf{all\_diff}(rcvPos_{i,p}), \forall o_i \in \mathbf{O} \text{ s.t. } p \in \texttt{route}(o_i). \tag{9}$$

In case two orders $o_i, o_j \in \mathbf{O}$ share at least one common consecutive pipeline pair $(p_l, p_m) \in \texttt{route}(o_i) \cap \texttt{route}(o_j)$, the activities precedence relations must be preserved in both pipelines. Here, we present the restrictions for flow directions N and R, the other cases being similar.

$$\begin{aligned} sndPos_{i,p_l} > sndPos_{j,p_l} &\iff sndPos_{i,p_m} > sndPos_{j,p_m} \\ \wedge\ rcvPos_{i,p_l} > rcvPos_{j,p_l} &\iff rcvPos_{i,p_m} > rcvPos_{j,p_m}, \\ &\forall o_i, o_j \in \mathbf{O},\ \forall (p_l, p_m) \in \texttt{route}(o_i) \cap \texttt{route}(o_j). \end{aligned} \tag{10}$$

Positional variables also help discarding sequences that violate product incompatibilities. Given two orders $o_i, o_j \in \mathbf{O}$, if $\texttt{product}(o_i)$ and $\texttt{product}(o_j)$ are incompatible, then they can not make contact in a pipeline. A necessary condition for contact is that both orders enter consecutively at the same pipeline extremity, and this can only happen if they have the same entering (or leaving) pipeline flow direction. This scenario is represented by the following constraints, for each $o_i, o_j \in \mathbf{O}$, $p \in \texttt{route}(o_i) \cap \texttt{route}(o_j)$ and

[1] An *unary resource* is a resource that allows for only one activity at a time [12].

$\texttt{product}(o_i)$ **incompatible** $\texttt{product}(o_j)$.

$$|sndPos_{i,p} - sndPos_{j,p}| > 1 \texttt{ if } \quad (\texttt{direct}_{i,p} \neq N \vee \texttt{direct}_{j,p} \neq RN) \tag{11}$$
$$\wedge \quad (\texttt{direct}_{i,p} \neq R \vee \texttt{direct}_{j,p} \neq NR),$$
$$|rcvPos_{i,p} - rcvPos_{j,p}| > 1 \texttt{ if } \quad (\texttt{direct}_{i,p} \neq N \vee \texttt{direct}_{j,p} \neq NR) \tag{12}$$
$$\wedge \quad (\texttt{direct}_{i,p} \neq R \vee \texttt{direct}_{j,p} \neq RN).$$

Next, we define two new activities: $ext_{i,\texttt{origin}(\texttt{o}_\texttt{i})}$ and $inj_{i,\texttt{destin}(\texttt{o}_\texttt{i})}$, representing volume extraction and injection, respectively, from the assigned tanks associated with order o_i. The relationship between send (receive) variables and tanks activities is the same as those for pipeline volume transmissions. For each $o_i \in \mathbf{O}$, letting p_0 and p_m be the first and last pipeline in $\texttt{route}(o_i)$, we state:

$$StartTime(snd_{i,p_0}) = StartTime(ext_{i,\texttt{origin}(\texttt{o}_\texttt{i})}), \tag{13}$$
$$EndTime(snd_{i,p_0}) = EndTime(ext_{i,\texttt{origin}(\texttt{o}_\texttt{i})}), \tag{14}$$
$$StartTime(rcv_{i,p_m}) = StartTime(inj_{i,\texttt{destin}(\texttt{o}_\texttt{i})}), \tag{15}$$
$$EndTime(rcv_{i,p_m}) = EndTime(inj_{i,\texttt{destin}(\texttt{o}_\texttt{i})}). \tag{16}$$

Injecting and extracting volumes from tanks must not overlap in time as well. Hence, activities $ext_{i,t}$ and $inj_{i,t}$ are associated with a new unary resource $TkDisj_t$, created for each tank $t \in \mathbf{T}$. However, capacities must also be taken into account in this case, requiring the combined use of a different type of resource $TkRes_t$, $t \in \mathbf{T}$, called a *reservoir* [12]. Such activities can both increase capacity (volume injection) or deplete capacity (volume extraction) from reservoirs.

Finally, we must also consider production and demand volumes in order to appropriately represent the behavior of tank capacities. Let **Dem** and **Pr** be, respectively, the sets of demands and productions given as input. For each $d \in \mathbf{Dem}$, an activity dem_d is created, with its associated constraints, considering demand time bounds and volume extraction from tanks. Similarly, an activity $prod_p$ is created for each $p \in \mathbf{Pr}$, but now considering volume injection instead of extraction. These activities are associated with the unary resources $TkDisj_t$ and reservoirs $TkRes_t$. Additional constraints are stated as follows.

$$StartTime(dem_d) \geq \texttt{DemandMinStartTime}(d), \tag{17}$$
$$EndTime(dem_d) \leq \texttt{DemandMaxEndTime}(d), \ \forall d \in \mathbf{Dem}, \tag{18}$$
$$StartTime(prod_p) \geq \texttt{ProductMinStartTime}(p), \tag{19}$$
$$EndTime(prod_p) \leq \texttt{ProductMaxEndTime}(p), \ \forall p \in \mathbf{Pr}. \tag{20}$$

The Operations Viewpoint. The main intuition for this viewpoint is to consider each pipeline individually (a *local* vision), since time variables domains will be automatically propagated by force of constraints defined in the order viewpoint. Although time bounds and disjunctions were already established, it is still necessary to model the fact that, in order for a certain volume to leave a pipeline, the exact amount of volume must be pumped in from the other extremity. Besides that, restrictions such as variable flow rates which depend on the products inside a pipeline, must also be considered. We will also use some ideas from previous studies [2, 5, 6], that treated the case of a single pipeline.

Given a pipeline $p \in P$, two time-ordered sets of *operation activities* are defined, $SndPipe_p$ and $RcvPipe_p$, where $|SndPipe_p| = |RcvPipe_p| = |\{o_i : o_i \in \mathbf{O}, p \in \texttt{route}(o_i)\}|$. As in the order viewpoint, they represent send and receive activities in p, respectively, but now with new precedence relations of the form

$$i < j \iff sndOp_{p,i} \quad \textbf{startsBefore} \quad sndOp_{p,j},\ \forall sndOp_{p,i}, sndOp_{p,j} \in SndPipe_p,$$
$$i < j \iff rcvOp_{p,i} \quad \textbf{startsBefore} \quad rcvOp_{p,j},\ \forall rcvOp_{p,i}, rcvOp_{p,j} \in RcvPipe_p.$$

A *volume* and a *product* variables are additionally associated with each activity belonging to $SndPipe_p$ and $RcvPipe_p$. We thus say that both sequences represent a valid ranking of *undetermined* delivery orders; they will only be *determined* when orders are ranked in their unary resources. However, since they are already time-ordered, we are able to create a more intuitive and compact model to represent pipeline flow behavior, in which constraints will also enforce propagation in the order viewpoint variable domains.

Let $rcvOp_{p,j} \in RcvPipe_p$ be an activity. A certain volume associated with it can only be received when an activity $sndOp_{p,i} \in SndPipe_p$ is being pumped at the other extremity of the pipe. In order to define which send activity i pushes a receive activity j, it is necessary to consider three factors: the pipeline volume, the volumes of the activities and the volumes between activities $sndOp_{p,i}$ and $rcvOp_{p,j}$, *i.e.*, the volume still in the pipeline before sending $sndOp_{p,i}$ and after receiving $rcvOp_{p,j}$. For the latter, a new variable $acc_{p,i,j}$ is created for each $sndOp_{p,i} \in SndPipe_p$, $rcvOp_{p,j} \in RcvPipe_p$, for $i \leq j$, as follows:

$$acc_{p,i,j} = \sum_{k<i} volume(sndOp_{p,k}) - \sum_{k \leq j} volume(rcvOp_{p,k}). \tag{21}$$

It can be shown that $sndOp_{p,i}$ never pushes $rcvOp_{p,j}$ off the pipeline, for $i > j$, due to the time-ordering of the relation. If $acc_{p,i,j} \geq volume(p)$, then it is not possible for $sndOp_{p,i}$ to push $rcvOp_{p,j}$, since a quantity greater or equal than the pipeline volume was already injected between activities i and j. On the other hand, if $acc_{p,i,j} + volume(sndOp_{p,i}) + volume(rcvOp_{p,j}) \leq volume(p)$, then the volume in $sndOp_{p,i}$ is not

enough to push $rcvOp_{p,j}$ out of the pipeline. Thus, a necessary and sufficient condition for activity i to push activity j out of the pipeline (a $\mathbf{push}_{i,j}$ event), can be stated as:

$$\begin{aligned} \mathbf{push}_{i,j} \iff & \; acc_{p,i,j} < volume(p) \\ \wedge & \; acc_{p,i,j} + volume(sndOp_{p,i}) + volume(rcvOp_{p,j}) > volume(p). \end{aligned} \tag{22}$$

Similar ideas can be used to determine the exact amount of volume involved in the pumping. Let $flow_{i,j}$ be the volume used in $sndOp_{p,i}$ to push the same volume $flow_{i,j}$ from $rcvOp_{p,j}$ out of the pipeline. We have:

$$\begin{aligned} \mathbf{push}_{i,j} \implies & \; flow_{i,j} = \texttt{min}[volume(rcvOp_{p,j}), volume(rcvOp_{p,j}) \\ & + volume(sndOp_{p,i}) + acc_{p,i,j} - volume(p)] \\ & - \texttt{max}[0, volume(rcvOp_{p,j}) + acc_{p,i,j} - volume(p)], \end{aligned} \tag{23}$$

$$\neg\mathbf{push}_{i,j} \implies \; flow_{i,j} = 0. \tag{24}$$

These flow variables can be seen as flow edges in a capacitated network. Assuming that each operation activity represents a node, the amount of volume (flow) pushed into a pipe must be equal to the volume pushed out it, and both are equal to the operation's total volume. To model this restriction, we can use a **flow** global constraint [10] for send and receive variable sequences.

In order to ensure flow rate bounds consistency, it is necessary to limit the flow rate of send and receive activities according to the products that are inside the pipeline when the pumping activity occurs. This can now be easily done using the earlier condition $acc_{p,i,j} + volume(sndOp_{p,i}) + volume(rcvOp_{p,j}) \leq volume(p)$, which is true if $sndOp_{p,i}$ and $rcvOp_{p,j}$ are both inside the pipeline at the moment of pumping. Let $MaxFl_i$ be a variable representing the maximum flow rate for activity $sndOp_{p,i}$, related to the variable $product(sndOp_{p,i})$, and let $MaxFl_j$ stand similarly for activity $rcvOp_{p,j}$. We state:

$$\begin{aligned} & acc_{p,i,j} + volume(sndOp_{p,i}) + volume(rcvOp_{p,j}) \leq volume(p) \\ \implies & \left\lfloor \frac{volume(sndOp_{p,i})}{(EndTime(sndOp_{p,i}) - StartTime(sndOp_{p,i})} \right\rfloor \leq MaxFl_j \\ \wedge & \left\lfloor \frac{volume(rcvOp_{p,j})}{(EndTime(rcvOp_{p,j}) - StartTime(rcvOp_{p,j})} \right\rfloor \leq MaxFl_i. \end{aligned} \tag{25}$$

Finally, flow directions in the pipeline must be consistent as well. For instance, if an activity has its direction attribute set to N, the next activity along the pipe must necessarily have its direction attributes set to N or NR. Direction attributes such as R and RN are only consistent after a sequence of NR activities whose volume sum is equal

to the pipeline volume. Attribute RN is treated similarly. These valid pairs are enforced using a *Table Constraint* [12].

For the pipeline reversal, a special constraint **reversal** was created, encapsulating the rules for the reversal of flow direction. This global constraints also controls the relation between the sequences $sndPos_{i,p}$ and $rcvPos_{i,p}$, since orders do not enter and live a pipe in the same order when there is a flow reversal, as showed in figure 3. If there is no flow reversal in a pipe p, then a constraint $sndPos_{i,p} = rcvPos_{i,p}$ is added to the model.

The Channeling Constraints. The order and operation viewpoints can be easily connected using the *element* constraint [13] and positional variables. Notice that a similar set of constraints is applied to the receive sequence.

$$StartTime(snd_{i,p}) = StartTime(sndOp_{\,sndPos_{i,p},\,p}), \quad (26)$$

$$EndTime(snd_{i,p}) = EndTime(sndOp_{\,sndPos_{i,p},\,p}), \quad (27)$$

$$\texttt{volume}(o_i) = Volume(sndOp_{\,sndPos_{i,p},\,p}), \quad (28)$$

$$\texttt{direct}(o_i) = Direct(sndOp_{\,sndPos_{i,p},\,p}), \quad (29)$$

$$\texttt{product}(o_i) = Product(sndOp_{\,sndPos_{i,p},\,p}), \quad (30)$$

$$\forall o_i \in \mathbf{O}, p \in \texttt{route}(o_i).$$

The Scheduling Model. The second part of the complete CP model is a simpler model which is responsible for assigning the exact times to pumping operations, respecting forbidden alignment configurations and avoiding simultaneous pipe usage. The pumping operations are created by checking the $flow_{p,i,j}$ variables values for each activity i and pipeline p. If $flow_{p,i,j} > 0$, for a certain j, then there is a pumping operation of volume $flow_{p,i,j}$ with flow rate and time bounds already established by the sequencing step. In that case, a new activity, *pumpOp*, is created and its time constraints are included in the model. Note that the precedence among activities can be inferred from the orders' sequence.

Let **Dep** be the set of depots, and let $PumpOps_d$ give the pumping operations that will start at depot d, for each $d \in$ **Dep**. The simultaneous sending constraint can be implemented using a *discrete resource*, $DiscSending_d$, a resource which limits the number of consecutive operations by a certain capacity [12]. Thus, we associate each operation in $PumpOps_d$ in its respective resource $DiscSending_d$, limited by the input $\texttt{DepotMaxSimultaneousOperations}_d$. Similarly, the forbidden alignment configurations are enforced with discrete resources $AlignDisc_{a,d}$, created for each alignment restriction a. The operations associated with each resource are easily identifiable by checking their product type and flow direction.

Free Delivery Orders. In certain scenarios, the orders created by the planning phase are not enough to guarantee a valid pumping solution. For instance, suppose that only two orders need to be scheduled, and they have incompatible products. Consequently, one can not push the other in a pipeline. A third product must be used between them. For that, *free delivery orders* are arbitrary created before entering the scheduling phase. In contrast to regular orders, their volumes, products, and origin/destination tanks are treated as variables instead of constants, and they do not have a deadline. Note also that free orders may have a null volume associated with them. Furthermore, their routes are previously determined by choosing among the ones typically used by pipeline operators. Operators can change such routes by editing a configuration file.

Free orders are also used to represent products that remain in the pipeline at the end of the process, for the purpose of ensuring that all pipes are always completely filled. These orders do not have a destination tank, and constraints are used to indicate they are the last ones to be pumped into the pipelines.

Only minor changes to the previous model are necessary in order to accommodate free orders. Among them, in the order viewpoint, constraints where volume and product were constants should be changed to variables, and an *Alternative Resource Set* [12] can be used to indicate that an origin and destination tank must be assigned to free orders. The operation viewpoint remains unchanged.

Search Strategy. Different types of search strategies were tested for solving both the sequencing and the scheduling models. The currently implemented version is shown as Algorithm 1. It combines a *backtracking* mechanism [13] with a special variable ordering, being divided into three consecutive parts: *disjunctive components determination*, *adaptive backtracking* and *time assignment*. In the *Disjunctive Components Determination*, a disjunctive component is defined as a subset of the network which can be scheduled separately, without affecting other regions. Two pipelines belong to the same component if they are both contained in at least one order's route. The same reasoning applies to tanks.

For the *Adaptive Backtracking*, we implemented backtracking using positional variables for each pipeline. The term *adaptive* comes from the fact that it is based on a *restart* strategy [14]. As such, the pipeline sequencing order is changed dynamically according to the number of fails that occurred during the search. The values of positional variables are randomly chosen, giving higher probabilities to orders with the earliest deadlines. For free orders, volumes, products and tanks are set after their respective positional variables are labeled.

The initial sequencing is constructed as follows. Firstly a *pipe graph* is created, in which pipelines are nodes and there is a direct arc from node p to q if there is a consecutive

Algorithm 1: Procedure for search strategy

```
1  begin
2      Identify network disjunctive components C
3      for each c ∈ C do
4          Build pipe graph G(c) and sort it topologically, obtaining order N
5          N' := N; k := initial_k
6          while N' ≠ ∅ do
7              p := first element from sequence N'; N' := N'/{p}
8              Label positional variables, and volumes/tanks in case of free orders
9              if fails in labeling ≥ k and not cyclic condition then
10                 k := k+incremental factor; N' := N
11                 Move p to the beginning of sequence N'
12     while no scheduling solution found do
13         Create scheduling model and assign times as earliest as possible
14         if no solution then request next sequencing solution
15 end
```

pair (p, q) in some order's route. In case there are two arcs (p, q) and (q, p), only the one associated with the order having the earliest deadline is maintained. Secondly, the graph is topologically sorted, the result being the desired initial sequencing. Clearly, this strategy considers first those pipelines with the least number of orders that come directly from other pipelines.

After the occurrence of k fails involving a pipeline, the backtrack tree is reinitialized with that pipeline as the first element in the topological ordering, and k is incremented by a constant. This implementation was motivated by the fact that, during test runs, it was observed that a fair number of fails were caused by earlier decisions taken when instantiating variables in related pipelines in the given sequencing. We empirically determined $k = 150$ and 100 as the increment.

Finally, in *Time Assignment*, executed after the sequencing is completed, the CP scheduling model is created and the time variables are instantiated with the least possible value in their domains. This forces pumping to start as soon as possible. In case a failure ensues, a new sequencing solution is requested, most certainly a different one due to the randomization present in the model.

4 Results

Solutions were obtained on a *Intel Pentium D* 3.40 Ghz CPU platform, with 4GB of memory. The planning and scheduling phases were coded in C++ and compiled using

Table 1: Solution Example

$\mathbf{T_i}$	$\mathbf{T_f}$	**Vol.**	**Pd**	$\mathbf{Tk_{Or}}$	$\mathbf{Tk_{Dt}}$	**Route**
2075	2362	858	G	T004	T005	SUG03
4857	4868	30	N	T160	T087	GUG03
4870	5111	722	D	T008	T005	BUG03
...	...	...	...	...	...	...

GCC-4.0. The CP model was solved using *ILOG Solver 6.2* and *ILOG Scheduler 6.2*, with medium to high propagation enforcement. Part of a typical solution is presented in Table 1. As described earlier, a solution is a sequence of pump operations and each line in the table describes one such operation. Column headings are as follows: T_i and T_f are the start and end times (in minutes), respectively; Vol is the pumped volume (in m^3); P_d is the product code (G is gasoline, N is naphtha and D is diesel); Tk_{Or} and Tk_{Dt} are origin and destination tank codes, respectively; and *Route* is the route code. A full solution table would contain several hundred such lines.

We used four real field instances to test the models. The first two rows in Table 2 indicate the time horizon and the number of deliver orders generated by the planning phase, respectively, for each of the test instances. The remaining lines give details of typical runs. All instances share the same network topology of 14 depots, 29 pipelines, 32 different product types and 242 tanks distributed among the depots. Pipelines volumes range from 30 to 8,000 m^3, and most of the tank capacities are between 4,000 and 30,000 m^3.

In all cases the solver found a solution in a reasonable amount of computer time, *e.g.*, within 10 minutes. Most variables were instantiated as a result of constraint propagation. The search heuristic, which proved crucial in the planning phase, was also instrumental to improve other important aspects of the solution quality, as noticed by logistic engineers. For instance, usually, a typical solution showed only a very small number of pipeline flow reversions, the kind of operation that engineers prefer to keep to a minimum. Also, new and interesting routes were identified. Some of them came as a surprise to logistic engineers, who were biased towards the same traditional routes they were using when manually planning the network operation.

5 Added Value and Conclusions

We proposed a novel procedure for generating feasible solutions for real instances stemming from planning and scheduling the operation of a very-large pipeline network used do move petroleum derivatives. The operation of such a network is subject to a complex set of physical and operational constraints, and it makes possible the delivery of oil and

Table 2: Solver Results

Instance	1	2	3	4
Horizon	10 days	7 days	7 days	7 days
Orders	924	645	724	693
Planning Phase Time	4 min	5 min	4 min	6 min
Planning Phase Peak Memory	78MB	61MB	67MB	63MB
Sequencing Model Variables	37,326	21,381	25,938	24,315
Sequencing Model Constraints	382,565	148,075	160,302	155,409
Sequencing Choice Points	3,355	2,462	3,417	2,518
Sequencing Fails	2,301	1,291	987	1,902
Sequencing Time	2 min	1 min	1 min	1 min
Sequencing Peak Memory	450 MB	240 MB	310 MB	270 MB
Scheduling Model Variables	12,350	7,530	8,931	8,032
Scheduling Model Constraints	27,088	16,768	19,231	18,292
Scheduling Choice Points	1,516	1,164	801	1,810
Scheduling Fails	301	429	210	120
Scheduling Time	2 min	1 min	1 min	1 min
Scheduling Peak Memory	450 MB	250 MB	290 MB	280 MB
Total Time	8 min	7 min	6 min	8 min

biofuel to local markets, as well as the storing of the excess production from refineries. Using the CP paradigm, these constraints were adequately modeled. Problems of this size and complexity, as known by the authors, would not be solved by other approaches reported in the literature to date, in which much of the difficult constraints and topologies are overlooked.

The procedure is already integrated with a proprietary flow simulation tool and the company is currently considering it for routine use on a daily basis. The tool has already proved its value, showing that it can save many valuable work hours of skilled engineers. Also, using the tool, many different planning and scheduling scenarios can be easily setup and quickly tested, by varying local demand needs and production schedules at refineries.

The present modeling and implementation stage was reached after 2 years of problem specification, data gathering, model development, and testing. As work progresses, it is expected that new constraints will be introduced. Such could include inventory management restrictions, limitations on energy use at specific time intervals and at specific depots, and shutdown periods or partial operation intervals for tanks and pipelines. When modeling such new constraints, we feel that the flexibility of the CP paradigm will again prove to be crucial.

As for future directions, one could implement more sophisticated search heuristics for both the planning and scheduling phases, making the overall approach capable of dealing with more specific instance classes. Finally, one could consider objective functions that

would help guide the heuristics. This would provide a yardstick that could be used to gauge solution quality.

Acknowledgments. This research was supported by grants 05/57343-0 and 05/57344-7 from FAPESP and grants 301732/07-8, 478470/06-1, 472504/07-0, and 473726/07-6, 305781/2005-7 from CNPq. The authors are also grateful to Fernando Marcellino and the team of engineers from PETROBRAS-TI/SP.

References

[1] Rejowski, R., Pinto, J.M.: Efficient MILP formulations and valid cuts for multiproduct pipeline scheduling. Computers & Chemical Engineering **28**(8) (2004) 1511–1528

[2] Cafaro, D.C., Cerdá, J.: Optimal scheduling of multiproduct pipeline systems using a non-discrete MILP formulation. Computers & Chemical Engineering **28**(10) (2004) 2053–2058

[3] Relvas, S., Barbosa-Póvoa, A.P.F.D., Matos, H.A., Fialho, J., Pinheiro, A.S.: Pipeline scheduling and distribution centre management - a real-world scenario at CLC. In: Proceedings of the 16th European Symposium on Computer Aided Process Engineering and 9th International Symposium on Process Systems Engineering, Garmisch-Partenkirchen, Germany (2006) 2135–2140

[4] Relvas, S., Matos, H.A., Barbosa-Póvoa, A.P.F.D., Fialho, J., Pinheiro, A.S.: Pipeline scheduling and inventory management of a multiproduct distribution oil system. Industrial & Engineering Chemistry Research **45**(23) (2006) 7841–7855

[5] Rejowski, R., Pinto, J.M.: A novel continuous time representation for the scheduling of pipeline systems with pumping yield rate constraints. Computers & Chemical Engineering **32** (2008) 1042–1066

[6] Relvas, S., Matos, H.A., Barbosa-Póvoa, A.P.F.D., Fialho, J.: Reactive scheduling framework for a multiproduct pipeline with inventory management. Industrial & Engineering Chemistry Research **46**(17) (2007) 5659–5672

[7] Crane, D.S., Wainwright, R.L., Schoenefeld, D.A.: Scheduling of multi-product fungible liquid pipelines using genetic algorithms. In: Proceedings of the 1999 ACM Symposium on Applied Computing, San Antonio, USA (1999) 280–285

[8] Camponogara, E.: A-Teams para um problema de transporte de derivados de petróleo. Master's thesis, Instituto de Matemática, Estatística e Ciência da Computação, Universidade Estadual de Campinas, Campinas, Brazil (1995)

[9] Moura, A., de Souza, C., Cire, A., Lopes, T.: Heuristics and constraint programming hybridizations for a real pipeline planning and scheduling problem. In: Proceedings of the 11th IEEE International Conference on Computational Science and Engineering - CSE'08. (Julho 2008) 455–462

[10] Hooker, J.N.: Integrated Methods for Optimization (International Series in Operations Research & Management Science). Springer-Verlag, Secaucus, USA (2006)

[11] Cheng, B.M.W., Choi, K.M.F., Lee, J.H.M., Wu, J.C.K.: Increasing constraint propagation by redundant modeling: an experience report. Constraints **4**(2) (1999) 167–192

[12] ILOG: ILOG Scheduler 6.2: User's Manual. ILOG. (2006)

[13] Marriot, K., Stuckey, P.: Programming with Constraints: An Introduction. 1^a edn. MIT Press, Cambridge, Massachusetts (1998)

[14] Kautz, H., Horvitz, E., Ruan, Y., Gomes, C., Selman, B.: Dynamic restarts policies. In: Proceedings of the AAAI-2002, Edmonton, Alberta (2002)

Epílogo

As principais diferenças em relação às duas formulações de escalonamento existentes está na forma como os bombeamento são representados. No primeiro caso, descrito no capítulo 6, as variáveis $Op*$ modelam explicitamente as operações de envio e recebimento nos dutos e tanques, o que acarreta a necessidade de tratar o parâmetro ns que descreve o número máximo de operações por plano.

No modelo aprimorado, os planos são vistos como *atividades*, que são uma estrutura para aplicação de restrições globais mais específicas para problemas de escalonamento [17]. Cada atividade é, de forma geral, composta por um tempo de ínicio t_i e tempo final t_f, os quais modelam os *intervalos* de ocorrência do plano em cada duto, ao invés de explicitamente os bombeamentos em si.

Esta representação de intervalos de tempo permite uma formulação mais compacta da rede de dutos, descartando o uso do parâmetro ns e, adicionalmente, facilitando a aplicação de restrições globais relativas às disjunções temporais, como as restrições *cumulative* e *disjunctive* apresentadas no texto. Como há também um menor número de variáveis, também é possível aumentar o nível de propagação das restrições, o que reduz o tamanho da árvore de *backtrack* do PR.

As movimentações por intervalos de tempo, contudo, não caracterizam uma solução do PTRD, conforme descrito no capítulo 2. Assim, um segundo modelo PR de *atribuição de tempos* é utilizado para determinar exatamente os tempos de bombeamentos. Tais operações são calculadas a partir do seqüenciamento dos planos, obtido pela solução do modelo anterior.

As variáveis do modelo de atribuição de tempos, desta forma, representam exatamente as operações de envio e recebimento que ocorrerão nos dutos, determinadas a partir de uma solução parcial do problema de Escalonamento. As restrições faltantes, tal como envios simultâneos e alinhamentos, são facilmente implementadas neste segundo submodelo. O artigo não apresenta a modelagem dos requisitos C5, C10 e C15 em decorrência da falta de dados na instância, mas estas podem ser implementadas utilizando-se restrições de disjunção temporal simples.

O artigo também apresenta um método de busca diferenciado de soluções, denominado *backtrack adaptativo*. Ao invés de fixar uma ordem de variáveis para atribuição de valores, o algoritmo procura identificar os dutos mais críticos a partir do número de *fails* durante o seqüenciamento dos planos. Espera-se que a ocorrência de muitos *fails* seja devida à decisões equivocadas no passado e, portanto, os dutos problemáticos ganham prioridade na ordem de seqüenciamento, quando uma nova árvore de *backtrack* é construída.

Capítulo 8

Conclusão

Este projeto de mestrado trata do problema de distribuição de derivados de petróleo e álcoois em redes de duto, ou PTRD, atualmente enfrentado pela PETROBRAS. Seu objetivo é definir movimentos de transporte de produtos entre órgãos de tal forma que as campanhas de produção e os mercados locais, representados por valores estimados de demandas, sejam totalmente satisfeitos. Para tanto, um amplo conjunto de restrições operacionais nos dutos e tanques deve ser respeitado, envolvendo seqüenciamentos válidos de produtos, capacidade de tanques e limites de envio simultâneo.

As técnicas até então empregadas para resolver o PTRD são fortemente baseadas em relaxações de seus principais requisitos ou lidam apenas com topologias de rede restritas, tornando-o assim tratável para ser resolvido eficientemente com técnicas de otimização clássicas. Por outro lado, tais relaxações acarretam soluções pouco aplicáveis para os cenários reais ao descartarem restrições fortes para os operadores de duto, tais como interface de produtos e não-simultaneidade de operações em tanques.

O trabalho aqui desenvolvido propõe formulações capazes de representar a maior parte das restrições fundamentais do PTRD, com potencial para serem utilizadas em instâncias de tamanho real. Com este objetivo, a resolução é dividida em duas fases, *planejamento* e *escalonamento*. A dissertação foca na segunda fase: dado um conjunto de *planos de entrega* a serem seqüenciados e escalonados, deve-se gerar operações de bombeamento para cobrí-lo sem que nenhum requisito operacional seja violado.

São propostas duas formulações em *Programação por Restrições* para a fase de escalonamento do PTRD. Tal técnica foi utilizada devido à sua flexibilidade em modelar problemas de escalonamento e dos fortes mecanismos de propagação existentes, os quais permitem encontrar soluções factíveis rapidamente. O primeiro modelo proposto já considera um número bastante alto de restrições e uma topologia real, mas possui limitações decorrentes da forma como as paradas em dutos são tratadas. Já o segundo modelo representa um aprimoramento significativo do primeiro, resolvendo a questão das paradas

dos dutos, contemplando mais restrições e sendo capaz de acomodar um horizonte maior.

As formulações foram testadas com 5 instâncias reais fornecidas pela PETROBRAS, coletadas ao longo de dois anos de especificação e implementação dos modelos. Vale ressaltar a grande dificuldade de coletar dados reais, visto que esta é a primeira vez em que o problema é formalizado junto aos operadores e gerentes da PETROBRAS. Assim, considera-se que uma contribuição adicional do trabalho é prover uma especificação do PTRD em que as soluções obtidas são bastante próximas das realizáveis, conforme exigência dos próprios operadores. Pela mesma razão de falta de dados consolidados pela empresa, ainda não foi possível comparar as soluções obtidas pelos modelos com as manualmente criadas. Contudo, todas as soluções foram executadas com sucesso no simulador de dutos da PETROBRAS, denominado CONSUELO. Por fim, devido ao alto número de restrições, gerar instâncias artificiais caracteriza-se como um procedimento extremamente difícil, o qual foi deixado como sugestão para trabalho futuro.

O objetivo de se pesquisar modelos flexíveis e abrangentes para o PTRD foi alcançado, os quais apresentaram grande potencial para gerar soluções efetivas para o problema. Espera-se que este seja um primeiro passo para uma resolução consistente do PTRD, inspirando novas técnicas e, ainda, idéias aplicáveis em problemas similares de escalonamento.

Capítulo 9

Trabalhos Futuros

Propõe-se diversas extensões deste trabalho, entre as quais:

- Alterar o modelo de *satisfação* para uma formulação capaz de lidar com *otimização* de parâmetros. Assim, o algoritmo seria capaz de gerar soluções parciais em relação ao atendimento da demanda e escoamento das produções, criando soluções úteis para guiar os operadores da rede de dutos. Além disso, também seria possível considerar custos de energia de bombeamento, o que exigiria também um esforço adicional para a coleta de tais dados.

- Novas formas de atribuição de valores às variáveis dos modelos em PR apresentados. Um possível exemplo seria aplicar heurísticas de busca local, as quais aproveitariam o sistema de propagação de PR para a redução do tamanho da vizinhança.

- Testes mais extensos com instâncias reais fornecidas pela Petrobras. Experimentos de grande porte ainda não foram realizados devido à dificuldade em se obter um conjunto de dados de entrada confiáveis, visto que esta é a primeira vez que o problema completo é formalizado junto com os operadores da rede. Além disso, também é essencial que um gerador de instâncias artificiais seja desenvolvido para verificação da robustez do algoritmo, o que forneceria um primeiro *benchmark* mais extenso para comparação de trabalhos, tal como em [26].

- Formas de resolução diferenciadas para a fase de escalonamento. Por exemplo, por meio de meta-heurísticas, ou hibridizações entre PLI, PR e heurísticas.

- Formas para que a fase de escalonamento seja capaz de analisar soluções com falhas, *i.e.* sem solução, e retornar um *feedback* para que a fase de planejamento crie conjuntos de planos de entrega com mais chance de viabilidade.

- Elaborar teorias que regulam o funcionamento dos dutos a partir de, por exemplo, Fluxo em Redes, definindo formalmente a noção de *empurramento*. Tal teoria pode ser essencial para criar algoritmos eficientes para o PTRD.

Referências Bibliográficas

[1] Vanessa Rennó Frota Moraes Alves. Programação de transferência de derivados de petróleo em rede dutoviária usando algoritmo genético. Dissertação de Mestrado, COPPE - Universidade Federal do Rio de Janeiro, Rio de Janeiro, RJ, Brasil, 2007.

[2] R. Bartak. Constraint programming: In pursuit of the holy grail. *Proceedings of Week of Doctoral Students (WDS99)*, páginas 555–564, Prague, Czech Republic, 1999.

[3] M.S. Bazaraa, J.J. Jarvis, e H.D. Sherali. *Linear Programming and Network Flows*. John Wiley and Sons, 1990.

[4] Viviane Monteiro Braconi. Heurísticas multifluxo para roteamento de produtos em redes dutoviárias. Dissertação de Mestrado, Departamento de Informática, PUC-Rio, Rio de Janeiro, RJ, Brasil, 2002.

[5] Diego C. Cafaro e Jaime Cerdá. Optimal scheduling of multiproduct pipeline systems using a non-discrete MILP formulation. *Computers & Chemical Engineering*, 28(10):2053–2058, 2004.

[6] Eduardo Camponogara. A-Teams para um problema de transporte de derivados de petróleo. Dissertação de Mestrado, Instituto de Matemática, Estatística e Ciência da Computação, Universidade Estadual de Campinas, Campinas, Brazil, 1995.

[7] Global Constraint Catalog. http://www.emn.fr/x-info/sdemasse/gccat/index.html.

[8] W.F. Clocksin e C.S. Mellish. *Programming in Prolog*. Springer, 1994.

[9] 5^{th} International Planning Competition. http://zeus.ing.unibs.it/ipc-5/.

[10] D. Scott Crane, Roger L. Wainwright, e Dale A. Schoenefeld. Scheduling of multiproduct fungible liquid pipelines using genetic algorithms. *Proceedings of the 1999 ACM Symposium on Applied Computing*, páginas 280–285, San Antonio, USA, 1999.

[11] J.M. de la Cruz, B. Andrés-Toro, A. Herrán-González, E. Besada Porta, e P. Fernandez Blanco. Multiobjective optimization of the transport in oil pipelines networks. *Proceedings of the IEEE Conference on Emerging Technologies and Factory Automation*, volume 1, páginas 566–573, 2003.

[12] J.M. de la Cruz, A. Herrán-González, J.L. Risco-Martín, e B. Andrés-Toro. Hybrid heuristic and mathematical programming in oil pipelines networks: Use of immigrants. *Journal of Zhejiang University SCIENCE*, 6A(1):9–19, 2005.

[13] Érito Marques de Souza Filho, Vanessa Rennó Frota Moraes, e Virgílio José Martins Ferreira Filho. Utilização de técnicas de pesquisa operacional em problemas de distribuição dutoviária: Uma revisão. *XXXVIII Simpósio Brasileiro de Pesquisa Operacional*, páginas 1873–1880, Goiania, Brazil, 2006.

[14] Frederico dos Santos Liporace. *Planejadores para transporte em polidutos*. Tese de Doutorado, Departamento de Informática, PUC-Rio, Rio de Janeiro, RJ, Brasil, 2005.

[15] D. E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley, 1989.

[16] John N. Hooker. *Integrated Methods for Optimization (International Series in Operations Research & Management Science)*. Springer-Verlag, Secaucus, USA, 2006.

[17] ILOG. *ILOG Scheduler 6.2: User's Manual*. ILOG, 2006.

[18] R. Rejowski Jr. e José M. Pinto. An MILP formulation for the scheduling of multiproduct pipeline systems. *Brazilian Journal of Chemical Engineering*, 19(4):467–474, 2002.

[19] Tony Lopes. Planejamento em rede de dutos. Dissertação de Mestrado, Instituto de Computação, Universidade Estadual de Campinas, Campinas, SP, Brasil, 2008 (em andamento).

[20] M. M. Sasikumar, P. R. Prakash, S. M. Patil, e S. Ramani. Pipes: A heuristic search model for pipeline schedule generation. *Knowledge-Based Systems*, 10(3):169–175, 1997.

[21] L. Magatao, L.V.R. Arruda, e F. Neves. A mixed integer programming approach for scheduling commodities in a pipeline. *Computers & Chemical Engineering*, 28(1):171–185, 2004.

[22] L. Magatao, L.V.R. Arruda, e F. Neves. Using CLP and MILP for scheduling commodities in a pipeline. *Computer-Aided Chemical Engineering*, 20B:1027–1032, 2005.

[23] K. Marriot e P.J. Stuckey. *Programming with Constraints: An Introduction*. MIT Press, Cambridge, Massachusetts, 1^a edição, 1998.

[24] Rodrigo Más e José M. Pinto. A mixed-integer optimization strategy for oil supply in distribution complexes. *Optimization and Engineering*, 4(1):23–64, 2003.

[25] R.L. Milidíu e Frederico dos Santos Liporace. Planning of pipeline oil transportation with interface restrictions is a difficult problem. Relatório Técnico 56, Departamento de Informática, PUC-Rio, Rio de Janeiro, RJ, Brasil, 2003.

[26] R.L. Milidíu, Frederico dos Santos Liporace, e Carlos José P. de Lucena. Pipesworld: Planning pipeline transportation of petroleum derivatives. *Proceedings of ICAPS'03 - Workshop on the Competition: Impact, Organization, Evaluation, Benchmarks*, Trento, Italy, 2003.

[27] R.L. Milidíu, A.A. Pessoa, V. Braconi, E.S. Laber, e Rey P.A. Um algoritmo GRASP para o problema de transporte de derivados de petróleo em oleodutos. *Proceedings of the XXXIII Brazilan Symposium on Operations Research*, páginas 237–246, 2001.

[28] A.V. Moura, C.C. de Souza, A.A. Cire, e T.M.T. Lopes. Heuristics and constraint programming hybridizations for a real pipeline planning and scheduling problem. *Proceedings of the 11th IEEE International Conference on Computational Science and Engineering - CSE'08*, páginas 455–462, Julho de 2008.

[29] A.V. Moura, C.C. de Souza, A.A. Cire, e T.M.T. Lopes. Planning and scheduling the operation of a very large oil pipeline network. *Lecture Notes in Computer Science - Proceedings of the 14th International Conference on Principles and Practice of Constraint Programming*, volume 5202, páginas 36–51, Setembro de 2008.

[30] C.H. Papadimitriou e K. Steiglitz. *Combinatorial Optimization: Algorithms and Complexity*. Dover Publications, 1998.

[31] PETROBRAS. Relatório anual de atividades, 2004.

[32] PETROBRAS. RIMA - Relatório de Impacto Ambiental, Plano Diretor de Dutos de São Paulo PDD/SP, Setembro, 2007.

[33] R. Rejowski e José M. Pinto. Efficient MILP formulations and valid cuts for multiproduct pipeline scheduling. *Computers & Chemical Engineering*, 28(8):1511–1528, 2004.

[34] R. Rejowski e José M. Pinto. A novel continuous time representation for the scheduling of pipeline systems with pumping yield rate constraints. *Computers & Chemical Engineering*, 32:1042–1066, 2008.

[35] Susana Relvas, Ana Paula F. D. Barbosa-Póvoa, Henrique A. Matos, João Fialho, e António S. Pinheiro. Pipeline scheduling and distribution centre management - a real-world scenario at CLC. *Proceedings of the 16th European Symposium on Computer Aided Process Engineering and 9th International Symposium on Process Systems Engineering*, páginas 2135–2140, Garmisch-Partenkirchen, Germany, 2006.

[36] Susana Relvas, Henrique A. Matos, Ana Paula F. D. Barbosa-Póvoa, e João Fialho. Reactive scheduling framework for a multiproduct pipeline with inventory management. *Industrial & Engineering Chemistry Research*, 46(17):5659–5672, 2007.

[37] Susana Relvas, Henrique A. Matos, Ana Paula F. D. Barbosa-Póvoa, João Fialho, e António S. Pinheiro. Pipeline scheduling and inventory management of a multiproduct distribution oil system. *Industrial & Engineering Chemistry Research*, 45(23):7841–7855, 2006.

[38] M.G.C. Resende e C.C. Ribeiro. Greedy randomized adaptive search procedures. F. Glover e G. Kochenberger, editors, *Handbook of Metaheuristics*, páginas 219–249. Kluwer Academic Publishers, 2002.

[39] Álvaro García Sánchez e Miguel Ortega Mier. Programación de oleoductos: presentación del problema y revisión de enfoques. *Anales del IX Congreso de Ingeniería de Organización (CIO 2005)*, 2005.

[40] L. A. Wolsey. *Integer Programming*. Wiley-Interscience, 1998.

[41] Tallys Hoover Yunes. Problemas de escalonamento no transporte coletivo: Programação por restrições e outras técnicas. Dissertação de Mestrado, Instituto de Computação, Universidade Estadual de Campinas, 2000.