XXI ENMC
ENCONTRO NACIONAL DE MODELAGEM COMPUTACIONAL
IX ECTM
ENCONTRO DE CIÊNCIA E TECNOLOGIA DE MATERIAIS

08 a 11 de Outubro de 2018
Instituto Federal Fluminense
Búzios - RJ

PROGRAMAÇÃO DE PETRÓLEO UTILIZANDO OTIMIZAÇÃO BILEVEL EM PROGRAMAÇÃO GENÉTICA COM INSPIRAÇÃO QUÂNTICA

Cristiane Salgado[1,3] - cristiane.salgado@petrobras.com.br
Douglas M Dias[2] - douglas.dias@uerj.br
Marley Vellasco[3] - marley@ele.puc-rio.br
[1]Petrobras S.A, Centro de Pesquisa e Desenvolvimento, RJ, Brazil
[2]Universidade do Estado do Rio de Janeiro, Depto de Eng. Eletrônica e Telecomunicações, RJ, Brazil
[3]Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Engenharia Elétrica, RJ, Brazil

Resumo. *A programação de petróleo em refinaria é um problema de alta complexidade que envolve uma sequência de decisões que buscam otimizar a alocação de recursos, o sequenciamento de atividades e a realização temporal dessas atividades, respeitando um conjunto de restrições de diferentes naturezas e visando ao atendimento de múltiplos objetivos. Possui uma natureza combinatória e apesar de sua complexidade, a atividade hoje carece de sistemas de otimização que auxiliem o processo de tomada de decisão. Este trabalho propõe a utilização de um modelo de otimização Bilevel, ou seja, dividindo a solução em dois níveis de decisão, onde um dos níveis, chamado Seguidor, ignora as restrições do problema e outro, chamado Líder, as considera. O algoritmo é baseado em um modelo de programação genética com inspiração quântica. Em relação aos múltiplos objetivos, são avaliadas abordagens a priori e a posteriori. Os resultados são comparados com dois modelos de apenas um nível, um baseado em priorização dos objetivos (a priori) e outro que usa a dominância de soluções como técnicas multiobjetivo. É observado benefício potencial do Bilevel quando comparado à priorização. Pelos resultados, consideramos que o desenvolvimento da pesquisa seja continuado.*

Keywords: *Otimização Bilevel, Otimização multiobjetivo, Programação genética, Programação genética com inspiração quântica, Programação de petróleo em refinaria*

1. INTRODUÇÃO

De acordo com Sinha (2009), a indústria do petróleo é um exemplo de cadeia de suprimentos onde estão integrados os processos de planejamento, fornecimento, produção e comercialização de produtos que abrange desde os fornecedores de seus fornecedores até os clientes de seus clientes, alinhando estratégia operacional com fluxo material, produtivo e de informação. Magalhães (2004) e Moro (2000) argumentam que esta indústria movimenta bilhões de dólares anualmente em um ambiente onde, muitas vezes, as margens são reduzidas. Portanto, é razoável admitir que soluções que melhorem os processos de tomada de decisão devem se traduzir

em aumento da rentabilidade da indústria e, portanto, representem potencial interesse para as organizações.

Bodington (1995) define a programação de refinaria como a área que conecta o planejamento da produção e as operações da planta de processo. Enquanto o primeiro foca as decisões considerando um viés econômico e semanas a frente, o segundo foca nas atividades de curto prazo, com uma visão bem detalhada sobre elas. A programação de produção (também conhecida como *scheduling*) pode ser definida como a especificação de quais tarefas cada estágio da produção deve fazer ao longo de um horizonte de tempo que pode ser de turnos até semanas. Isto significa que ela busca respostas às questões como "Quando iniciar e por quento tempo manter a produção de um perfil de derivados?", "Qual o sequenciamento de tanques na carga da unidade de processo?", "Que correntes em que composições devem ser usadas para compor o produto final?", "Quais serão as vazões no período?" etc. Todas essas questões devem ser respondidas de modo a gerar pelo menos um modo viável de operar a refinaria, compatibilizando a programação de chegada de crus e a de entrega de produtos.

Nas refinarias brasileiras, é comum a programação de produção ser dividida em, pelo menos, duas áreas: programação de petróleo e programação de derivados. No primeiro caso, as perguntas devem ser respondidas considerando desde o recebimento de petróleo (por exemplo, por descarregamento de navio ou bombeio de terminal) até as correntes de intermediários oriundas das unidades de destilação (CDU). No segundo caso, o escopo começa a partir destas unidades e se extende pelas demais unidades de processo até a mistura de produtos finais. Neste trabalho, é abordada uma proposta de solução para a programação de petróleo em refinaria.

De acordo com Masood (2016), problemas de programação de produção são *NP-Hard*. Cruz (2007) e Moro & Pinto (2004) explicitam que se trata de um problema de programação não linear inteira-mista (*Mixed-Integer Non Linear Problem* – MINLP) devido aos componentes de alocação de recursos (inteiro), transferência de volumes entre equipamentos (contínuo) e cálculos de propriedades (não linearidade).

Uma revisão bibliográfica comprova que já foram propostos trabalhos relevantes utilizando tanto abordagens de solução matemática quanto metaheurísticas, como por exemplo, Shah (2015); Mouret et al (2011); Wu et al. (2009); Cruz (2007). No entanto, a maioria desses trabalhos utiliza funções objetivo com apenas um objetivo, como a maximização do lucro usada em Xu et al (2017) ou técnicas de agregação para converter múltiplos objetivos em apenas uma expressão, usada em Oliveira et al (2008). O objetivo principal das indústrias em geral é obter a máxima rentabilidade de suas operações. No entanto, nas refinarias não é comum haver informação suficiente para balizar uma função objetivo financeira no nível de decisão da programação. Dessa forma, programador trabalha com múltiplos objetivos que representam componentes da missão de manter a refinaria operando continuamente e da forma mais estável possível, ou seja, minimizar variações da carga programada nas unidades de processo, receber os itens de petróleo conforme previsto, minimizar o número de operações de movimentação entre tanques, minimizar sobrespecificação nos produtos, entre outros.

Este trabalho apresenta um estudo sobre o uso de otimização em dois níveis hierárquicos de decisão (*Bilevel Optimization Problem* – BOP) para tratar problemas com mais de um objetivo aplicados à programação de petróleo. Essa abordagem é combinada com técnica de priorização entre os múltiplos objetivos do problema ou com o uso de algoritmos evolutivos multiobjetivo (*Multiobjective Optimization Evolutionary Algorithm* – MOEA). A Seção 2. apresenta os conceitos principais sobre as técnicas de otimização *Bilevel* e MOEA. A Seção 3. apresenta uma descrição do funcionamento do modelo evolutivo de acordo com as técncias. A Seção 4. apresenta os resultados obtidos e a seção 5. apresenta a discussão e principais conclusões.

2. OTIMIZAÇÃO MULTIOBJETIVO E OTIMIZAÇÃO BILEVEL

2.1 Otimização Multiobjetivo

Resolver um problema de otimização implica em buscar a melhor solução que minimize (ou maximize) os objetivos em questão, respeitando as restrições que definem o espaço de busca desta solução. Quando estes objetivos podem ser representados por apenas 1 elemento, configura-se uma classe de problemas chamada *single-objective problem* (SOP), onde é natural determinar se uma solução é melhor do que outra. No entanto, há diversos exemplos de problemas caracterizados por buscar soluções que atendam objetivos tipicamente conflitantes. Por exemplo, em investimentos financeiros se deseja maximizar o retorno e minimizar o risco. Esta classe de problemas é chamada de problemas multiobjetivo (*Multiobjective Problem* – MOP). De acordo com Deb (2001), há duas formas de lidar com o dilema dos problemas de natureza multiobjetiva: uma *a priori* pois o decisor precisa estabelecer uma preferência entre os objetivos (por exemplo através da definição de pesos multiplicadores para cada termo) e outra *a posteriori* pois um algoritmo irá buscar o conjunto de soluções que representa a melhor relação de compromisso entre os objetivos e apresentá-las ao decisor para que ele defina qual adotar.

A formulação de problemas multiobjetivos é dada por Deb (2001):

$$\min_{(x)} \quad F(x) = (F_1(x), \cdots, F_M(x)) \tag{1}$$

sujeito a:

$$G(x) \geq 0, \qquad H(x) = 0$$

onde $F(x)$ é a função multiobjetivo composta pelos M diferentes objetivos do problema, representados, cada um, através de sua $F_{(i)}$, $G(x)$ e $H(x)$ representam, respecitvamente, os conjuntos de restrições de desigualdade e de igualdade. As variáveis do problema são representadas por x. Nas abordagens *a posteriori*, a função objetivo é mantida como um vetor de objetivos com valores independentes.

Ainda de acordo com Deb (2001), ao longo dos anos, os algoritmos evolutivos receberam maior atenção em problemas multiobjetivos porque sua característica de manipular uma população de indivíduos independentes que evoluem a cada geração é aderente à ideia de decisão *a posteriori*. Cada indivíduo representa uma solução para o problema, o que permite que se resolva problemas com um ou mais objetivos. Os principais desenvolvimentos utilizando metaheurísticas estão concentrados em três conceitos: dominância das soluções, indicadores de qualidade da população e na decomposição do problema em subproblemas. Neste trabalho se utiliza o conceito de dominância proposto por Deb (2001) que define que a solução x_1 domina x_2 ($x_1 \preceq x_2$), se duas condições são verdadeiras: a solução x_1 não é pior do que a solução x_2 em nenhum dos M objetivos e; a solução x_1 é melhor do que a solução x_2 em pelo menos um dos M objetivos. Se uma destas condições é violada, x_2 é não-dominada por x_1.

Dado um conjunto de soluções **P**, o conjunto de soluções não-dominadas **P'** é dado por aquelas soluções que não são dominadas por nenhuma outra solução de **P**. A representação destas soluções no espaço de objetivos é denominada *Fronteira* (ou *Frente*) *de Pareto*.

Um algoritmo evolutivo clássico para otimização multiobjetivo terá as etapas de avaliação de indivíduos, substituição da população e seleção dos indivíduos realizadas de forma essencialmente diferentes ao problema SOP devido à influência da dominância para comparar soluções e, portanto, ordená-las. Os dois principais algoritmos evolutivos baseados em dominância usa-

dos em MOPs são: *Non-dominated Sort Genetic Algorithm-II* (NSGA-II), proposto por Deb et al (2002) e *Strength Pareto Evolutionary Algorithm 2* (SPEA2), por Zitzler et al (2001).

Diversos problemas da vida real possuem mais de três objetivos, o que representa grandes desafios mesmo com a evolução dos MOEAs pois os algoritmos propostos não eram capazes de diferenciar as soluções, de forma que a população tendia a ficar majoritariamente não dominada, dificultando a acomodação de novos indivíduos. Para tratar esta classe de problemas, chamada *Manyobjective*, Deb & Jain (2014) propõem o algoritmo *Non dominated Sort Evolutionary Algorithm - III* (NSGA-III), que utiliza pontos de referência espalhados ao longo do espaço de solução para orientar a evolução. No mesmo ano, Jain & Deb (2014) apresentam uma modificação na dominância proposta no NSGA-III para que o algoritmo trate problemas com restrição de forma que os indivíduos que não violam restrições são organizados em frentes de Pareto de menor *rank* do que os que violam. O tamanho da violação influencia nessa ordenação.

2.2 Otimização Bilevel

O estudo de técnicas e métodos de otimização em mais de um nível hierárquico é motivado pela observação de que diversos problemas reais, em diferentes áreas, seguem uma estrutura hierárquica de decisão com diferentes atores. Em Talbi (2013) é definido que as decisões são tomadas sem cooperação entre os agentes, que controlam subconjuntos das variáveis, e cada decisão tomada em um nível impacta diretamente nos espaços de busca e de solução dos demais. Em Sadigh et al (2012) é apresentado o problema do "fabricante X varejista", como exemplo de otimização hierárquica. Nele, ambos os atores desejam maximizar seus lucros. O fabricante pode ser representado como um líder, pois sua decisão sobre o preço do produto é a primeira a ser feita. O varejista pode ser encarado como um seguidor, pois irá reagir ao preço estabelecido pelo líder através da compra do produto, em maior ou menor quantidade, e do preço que atribuirá para revenda. Para maximizar seu lucro, o fabricante deseja cobrar o preço mais alto possível desde que não implique em diminuição da quantidade de produto comprada pelo varejista. Dessa forma, o fabricante tentará antecipar o comportamento do varejista para determinar o valor de venda de seu produto.

Matematicamente, Deb & Sinha (2009) propõem que um BOP pode ser formulado como um problema de otimização (seguidor) que faz parte do conjunto de restrições de outro problema de otimização (líder), da seguinte forma:

$$\min_{(x_u,\ x_l)} \quad F(x) = (F_1(x), \cdots, F_M(x)) \tag{2}$$

sujeito a:

$$\begin{aligned} x_l \quad \in \quad & \underset{(x_l)}{argmin} \quad f(x) = (f_1(x), \cdots, f_N(x)) \\ sujeito \quad & a: \\ & g(x) \geq 0, \qquad h(x) = 0 \\ G(x) \geq 0, \quad & H(x) = 0 \end{aligned}$$

$F(x)$ é a função multiobjetivo composta pelos *M* diferentes objetivos do problema líder. *G(x)* e *H(x)* representam, respectivamente, as restrições de desigualdade e igualdade também do líder. De forma equivalente, para o problema seguidor, $f(x)$, N, *g(x)* e *h(x)* representam, respectivamente, a função objetivo, o número de objetivos e as restrições de desigualdade e igualdade. x_u e x_l representam as variáveis do líder e do seguidor.

Em Talbi (2013), são apresentadas técnicas que podem ser utilizadas para resolver problemas BOP, como a simplificação do BOP para apenas um nível, a conversão para MOP, co-evolução dos problemas ou resolução sequencial. Nesta última, avaliada neste trabalho, o problema líder gera uma solução ou população de soluções (xu, xl), orientado por uma função objetivo $F(xu, xl)$. Estas soluções são passadas ao problema seguidor que atua apenas sobre as variáveis de decisão xl (as variáveis xu, definidas para o problema líder são tratadas como constantes) para, através da atuação de uma metaheurística, promover melhorias na população. Após a conclusão da otimização do seguidor, a nova população $(xu, xl*)$ retorna indivíduos ao problema líder, representando a população inicial de um novo ciclo iterativo. Este processo se repete até que um critério de parada para o problema é obtido. Quando um ou ambos os níveis do BOP são multiobjetivo a complexidade aumenta significativamente para a solução e esta classe de problemas é chamada *Multi-Objective Bilevel Otimization Problems* (MOBOP).

3. MODELO EVOLUTIVO

O modelo evolutivo utilizado, Programação Genética Linear Orientada à Gramática com Inspiração Quântica (PGLOGIQ), foi proposto em Pereira et al (n.d), possui quatro objetivos e tem como entidades: Linguagem Específica de Domínio (*Domain Specific Language* – DSL), cujo conceito foi apresentado por Deursen & Klint (2002), Gene quântico (GQ), Indivíduo quântico (IQ), Gene clássico (GC), Indivíduo clássico (IC) e o Operador quântico (OpQ).

A DSL representa a tradução das tarefas mais importantes da programação de petróleo em instruções que podem ser usadas pelo algoritmo para criar programas que, efetivamente, representam uma solução de programação. Este processo de criação das instruções garante que nenhuma tarefa será topologicamente inviável. Para representar adequadamente a programação de petróleo, a DSL proposta contempla quatro instruções diferentes para carga da unidade de destilação, duas instruções para transferência de petróleo entre a tancagem do terminal e da refinaria, duas instruções para descarregamento de navios petroleiros na tancagem do terminal e, por último, uma instrução que significa não fazer nenhuma movimentação.

GQ representa a superposição de todas as instruções que podem ser criadas de acordo com o espaço de busca definido pela gramática. De acordo com Dias & Pacheco (2013), sua implementação equivale a uma árvore composta por diferentes vetores de probabilidades acumuladas. Cada GQ tem a mesma estrutura, mas a distribuição de probabilidades será diferente dependendo do processo evolutivo. IQ é uma lista de GQs.

GC é o resultado de todas as observações necessárias nas distribuições de probabilidades do GQ para definir a instrução principal e todos os seus argumentos. Por exemplo, se a primeira observação do GQ for a instrução de carga de unidade de destilação com volume parcial da movimentação e apenas um tanque de carga, serão necessárias duas novas observações: uma definirá o tanque de carga e a outra o percentual do volume máximo a ser movimentado. IC é a disposição linear de todos os GCs e, portanto, tem o mesmo tamanho de IQ.

O processo evolutivo ocorre pela atuação do operador quântico (OpQ) que atualiza a distribuição de probabilidades dos IQs correspondentes aos melhores ICs avaliados. Esta atualização é um incremento na probabilidade de que ocorram as mesmas observações que resultaram nos bons indivíduos. O algoritmo manipula três populações: População Quântica, constituída de IQs, uma População clássica (PC), constituída de ICs e uma População Clássica Auxiliar (PCA) do processo evolutivo, também composta de ICs.

Na PGLOGIQ o programador estabelece, hierarquicamente, a importância dos objetivos e

as soluções são avaliadas e ordenadas de acordo com esta hierarquia.

Este trabalho avalia o desempenho do algoritmo PGLOGIQ em uma estrutura *Bilevel* de resolução sequencial para resolver o problema de programação de petróleo. Para ordenação dos indivíduos foram avaliados tanto a proposta original, de priorização dos indivíduos, como PGLOGIQ modificada, apresentada em Pereira et al (2018), que utiliza os conceitos de dominância em problemas com restrições apresentado no NSGA-III de Jain & Deb (2014). O Algoritmo 1 apresenta a abordagem iterativa do problema, onde o Seguidor ignora as restrições e o Líder as ativa. Este processo segue por n ciclos, até que um critério de parada é atingido.

Algoritmo 1 Abordagem Líder-Seguidor (Bilevel) para ativação das restrições

1: iteração do *loop LL*=0
2: Inicialização das populações:
3: Criar os *NI* indivíduos da população quântica inicial do Líder(PQL_0) e os indivíduos da população quântica inicial do Seguidor(PQS_0)
4: Criar os *NI* indivíduos da população clássica inicial do Líder (PCL_0) a partir da observação de PQL_0 e os indivíduos da população clássica inicial do Seguidor (PCS_0) a partir da observação de PQS_0
5: Construir *schedule* de cada indivíduo da PCL_0 e de cada indivíduo da PCS_0
6: Medir aptidão do *schedule* de cada indivíduo da PCL_0 e da PCS_0 de acordo com a avaliação da função multiobjetivo
7: $LL \leftarrow LL + 1$
8: **while** $LL \leq nLoop$ **do**
9: Executar PGLOGIQ (Hierarquia ou Dominância) do problema Seguidor (sem restrições) por gS gerações.
10: Substituir membros da população quântica do problema Líder por membros da população quântica do Seguidor. Se o número de indivíduos aceitos no Seguidor é superior a 50% da população, copia todos para o Líder. Se for inferior, mantém 50% da população do Líder e copia os 50% melhores indivíduos do Seguidor para o Líder.
11: Executar PGLOGIQ (Priorização ou Dominância) do problema Líder (com restrições) por gL gerações
12: **if** $LL < nLoop$ **then**
13: Substituir membros da população quântica do problema Seguidor por membros da população quântica do Líder. Se o número de indivíduos aceitos no Líder é superior a 50% da população, copia todos para o Seguidor. Se for inferior, mantém 50% da população do Seguidor e copia os 50% melhores indivíduos do Líder para o Seguidor.
14: **end if**
15: $LL \leftarrow LL + 1$
16: **end while**
17: Apresenta o melhor indivíduo da população clássica Líder

4. RESULTADOS

Para avaliar os resultados deste modelo foram utilizados os dados de cinco cenários de programação de uma refinaria real, chamados de CEN01, CEN02, CEN03, CEN04 e CEN05. A configuração topológica desta refinaria é apresentada na Fig. 1. O conjunto de informações iniciais contempla: topologia da planta, capacidades máximas e lastro de cada tanque de petróleo

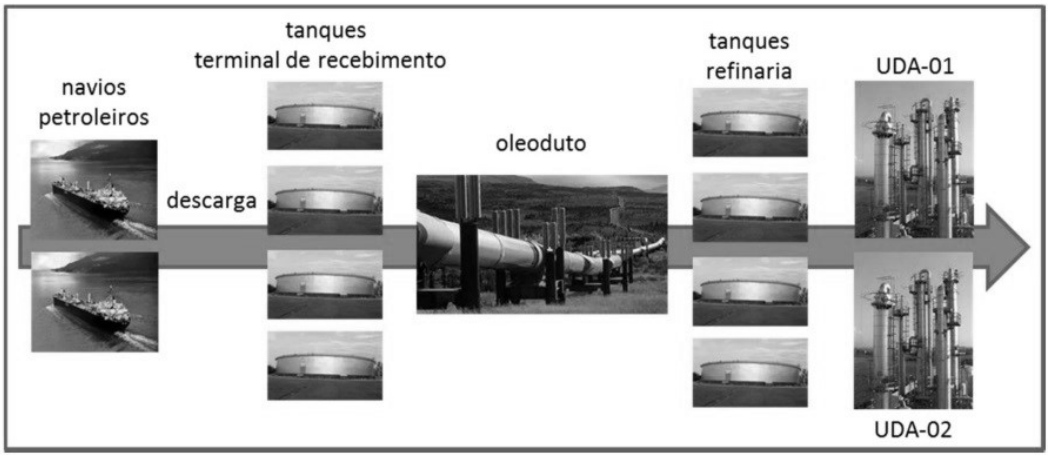

Figura 1- Esquemático da topologia usada como caso de estudo.

do terminal e da refinaria, volume e composição de cada um destes mesmos tanques, inventário do oleoduto que conecta o terminal à refinaria, vazão de carga programada das unidades de destilação e previsão de recebimento de navios petroleiros (incluindo janela de operação, volume e composição de petróleo transportado).

CEN01 e CEN02 são os casos mais simples, com menores horizontes de programação (192 h). O primeiro possui um nível de inventário mais confortável e apenas um navio a ser descarregado, enquanto CEN02 possui dois. CEN03 traz um pouco mais de complexidade pelo aumento de seu horizonte (216 h) e, apesar de ter apenas um navio, o volume de descarregamento é equivalente aos dois navios de CEN02. Esta característica torna necessário acomodar o mesmo volume em um horizonte menor de tempo (dentro da janela de operação contratada para o navio). CEN04 possui um inventário inicial de tancagem significativamente maior, o que demanda uma coordenação de movimentações mais desafiadora. CEN04 e CEN05 possuem maior horizonte (240 h), mas são contrários na questão do inventário pois em CEN05 o inventário está abaixo do confortável e esta característica pode comprometer a operação das unidades de destilação, caso as movimentações não sejam eficientes, apesar de um grande volume de óleo a ser descarregado em dois navios.

Além de restrições em função da topologia da planta e das associadas à capacidades mínimas e máximas dos equipamentos, também são consideradas restrições de teores de acidez e enxofre das misturas de petróleo que podem ser processadas nas unidades de destilação.

Os quatro objetivos do problema são: minimizar o tempo de parada das unidades de destilação, minimizar o atraso no descarregamento dos navios, minimizar o tempo de parada do oleoduto e minimizar o número de trocas de tanque das movimentações. Os objetivos de parada de CDU e atraso de navio são mais importantes do que os outros dois e, para eles, ainda que o ideal seja o valor zero, é aceita uma tolerância. No caso de parada da CDU corresponde a 2% do tempo de horizonte do cenário pois compreende-se que, dada a variabilidade do ambiente operacional, esta redução não significará uma redução real. No caso de descarregamento de navio, é aceita uma hora de atraso, apenas para fins de arredondamento.

Algoritmos evolutivos são métodos não determinísticos e, por isso, rodadas diferentes podem convergir para soluções diferentes. Dessa forma, em cada avaliação são rodadas 50 réplicas e é avaliado o percentual de vezes em que a população final possui, pelo menos, um indivíduo que atenda aos critérios de aceitação dos objetivos críticos. Este valor é a métrica utilizada para

Tabela 1- Percentual de corridas com soluções de programação válidas

Modelo	CEN01	CEN02	CEN03	CEN04	CEN05
PGLOGIQ (Priorização base)	98	100	100	76	36
PGLOGIQCNS	100	100	100	94	76
PGLOGIQ Bilevel (10C-S300-L700)	98	100	84	40	4
PGLOGIQ Bilevel (10C-S500-L500)	92	100	62	22	4
PGLOGIQCNS Bilevel (10C-S300-L700)	98	100	98	58	6
PGLOGIQCNS Bilevel (10C-S500-L500)	98	100	92	44	6

Tabela 2- Percentual de corridas com soluções válidas para variações do PGLOGIQ Bilevel

Caso	CEN01	CEN02	CEN03	CEN04	CEN05
PGLOGIQ Bilevel (10C-S1000-L1000-P0.5)	100	100	100	96	54
PGLOGIQ Bilevel (10C-S600-L1400-P0.5)	100	100	98	98	64
PGLOGIQ Bilevel (1C-S3000-L7000)	98	100	100	90	46

comparar os casos e tratado como "percentual de soluções aceitas" ao longo do texto.

A Tabela 1 apresenta os resultados do percentual de soluções aceitas nos diferentes modelos. PGLOGIQ (Priorização base) se refere ao modelo, baseado na priorização dos objetivos feita pelo programador. PGLOGIQCNS representa o algoritmo proposto por Pereira et al (2018), que utiliza o conceito de dominância dos indivíduos influenciada pela violação das restrições. PGLOGIQ *Bilevel* representa o algoritmo PGLOGIQ aplicado em dois níveis de decisão. No Seguidor (que evolui por 300 ou 500 gerações, dependendo do caso), as restrições de qualidade do problema são ignoradas. As melhores soluções são transferidas para o Lider (que evolui por 700 ou 500 gerações) que passa a considerá-las no processo. São executados 10 ciclos iterativos de transferência entre os indivíduos. PGLOGIQCNS *Bilevel* representa a mesma estrutura de ciclos e migração de indivíduos de PGLOGIQ *Bilevel*, mas o algoritmo evolutivo de cada nível é o PGLOGIQCNS.

Os resultados evidenciam o pior desempenho da abordagem Bilevel em qualquer configuração estudada, seja comparado ao PGLOGIQ ou ao PGLOGIQCNS. No entanto, algumas observações podem ser feitas diante deles. Os casos em que houve maior número de gerações no Líder (700 em cada ciclo) tem desempenho melhor do que seu par com 500 gerações, em ambos os algoritmos evolutivos. Este comportamento pode ser explicado porque essa configuração se assemelha mais aos modelos de um nível, nos quais as restrições do problema são consideradas desde a primeira geração. Dessa forma, é necessário avaliar se a abordagem Bilevel em si não é eficiente para tratar o problema ou se seu baixo desempenho pode ser associado a uma proposta inadequada de separar os níveis com base na ativação de restrições. Para tal, foram feitas algumas avaliações variando configurações da PGLOGIQ *Bilevel*.

Na Tab 2, as configurações (10C-S1000-L1000-P0.5) e (10C-S600-L1400-P0.5) represen-

tam a avaliação do impacto de permitir maior evolução em cada ciclo, mantendo-se os 10 ciclos originais. Para preservar o número total de avaliações (custo computacional), a população foi reduzida pela metade. A comparação destes resultados com a tabela 1 mostra o ganho obtido em permitir que a evolução ocorra por mais gerações em cada ciclo, pese a redução da população.

A configuração (1C-S3000-L7000) representa a comparação mais simples entre o PGLOGIQ e a contribuição da estrutura *Bilevel* pois possui apenas 1 ciclo de troca entre as populações. Após 3.000 gerações irrestritas, os melhores indivíduos da população Seguidor são transferidos para o Líder que evolui por outras 7.000 gerações com as restrições. O melhor indivíduo desta população é apresentado ao final. A comparação de PGLOGIQ com PGLOGIQ *Bilevel* de 1 ciclo, nas tabelas 1 e 2 mostra desempenho melhor da abordagem Bilevel nos dois cenários mais complexos, empate em CEN02 e CEN03 e piora em CEN01. No entanto, deve ser ressaltado que esta perda é pouco significativa comparado com o benefício atingido em CEN04 e CEN05.

5. CONCLUSÕES

Este trabalho propõe um algoritmo evolutivo em dois níveis hierárquicos de decisão (*Bilevel*) para tratar o problema da programação de petróleo em refinaria. Os resultados obtidos são comparados com dois algoritmos de apenas um nível de decisão: PGLOGIQ com priorização pré-definida dos objetivos e PGLOGIQCNS com abordagem multiobjetivo para classificar as soluções. São avaliadas diferentes configurações do algoritmo proposto, chamado PGLOGIQ *Bilevel*, com o objetivo de encontrar seu melhor desempenho, considerando questões como tratamento dos objetivos, evolução em cada estágio do problema e tamanho da população. De maneira geral, pode-se observar pelos resultados apresentados na seção 4., que os modelos *Bilevel* terão melhor desempenho se for permitida maior evolução para cada etapa, mantendo o número total de avaliações. No entanto, nenhuma configuração é superior aos resultados obtidos na PGLOGIQCNS. Desta forma, pode-se considerar que existe benefício em utilizar uma abordagem *Bilevel* para resolver o problema de programação de petróleo, mas há um longo caminho de desenvolvimento para obter o melhor desempenho da abordagem.

Referências

Bodington, C.E. (1995), “Planning, scheduling and control integration in the process industries.”, 5th ed., USA: McGraw-Hill.

Cruz, D.D.S. (2007), ‘Refinery Scheduling Using Genetic Algorithms: A Study for Crude Oil Scheduling Case.”, Dissertação de Mestrado. COPPE/UFRJ, Rio de Janeiro.

Deb, K. (2001), “Multi-objective Optimization using Evolutionary Algorithms”, John Wiley & Sons, 1st edition, England.

Deb, K., Pratap, A., AgarwalL, S., Meyarivan, T. (2002),“A fast andelitist multiobjective genetic algorithm: NSGA-II.”, IEEE Transactionson Evolutionary Computation, 6:182–197, 2002

Deb, K., Sinha, A. (2009), “An evolutionary approach for bilevel multi-objective problems.’, Applied Soft Computing, 35:17–24, 2009

Deb, K., Jain, H. (2014) “An Evolutionary Many-Objective Optimization Algorithm Using Reference-point Based Non-dominated Sorting Approach, Part I: Solving Problems with Box Constraints”, IEEE Transactionson Evolutionary Computation 18, 577–601.

Dias, D.M., Pacheco, M.A.C. (2013), “Quantum-Inspired Linear Genetic Programming as a Knowledge Management System.”, Comput. J. 56, 9, 1043–1062.

Deursen, A.V., Klint, P. (2002), “Domain Specific Language Design Requires Feature Descriptions.” Journal of Computing and Information Technology 10, n.1, 1–17.

Jain, H., Deb, K. (2014), “An evolutionary many-objective optimization algorithm using reference-point based nondominated sorting approach, Part II: Handling constraints and extending to an adaptive approach”, IEEE Transactionson Evolutionary Computation 4, 602–622.

Magalhães, M.V.O. (2004), "Refinery Scheduling.", Tese de Doutorado. Imperial College, Londres.
Masood, A. and Mei, Y. and Chen, G and Zhang, M. (2016), "Many-Objective Genetic Programming for Job-Shop Scheduling, IEEE Congress on Evolutionary Computation, 63, 219-216
Moro, L.F.L., Pinto, J.M. (2004), "Mixed-Integer programming approach for short-term crude oil scheduling.", Industrial and engineering chemistry research, vol 43, 1 (2004) 85-94.
Moro, L.F.L., (2000), "Técnicas de Otimização Inteira Mista para o Planejamento e Programação de Produção em Refinarias de Petróleo.", Tese de Doutorado. Universidade de São Paulo, São Paulo.
Mouret, S., Grossmann, I. and Pestiaux, P. (2011), " A new Lagrangian decomposition approach applied to the integration of refinery planning and crude-oil scheduling", Computers and Chemical Engineering 35, 2750–2766.
Oliveira, F., Almeida, M. R. and Hamacher, S. (2008), "Genetic Algorithms Applied to Scheduling and Optimization of Refinery Operations.", Proceedings of the VI ALIO/EURO Workshop on Applied Combinatorial Optimization 6 , 182–197.
Pereira, C.S, Dias,D. M., Vellasco, M.B.R., Viana, F.H.F., Martí, L. (2018) "Crude Oil Refinery Scheduling: Addressing a Real-World Multiobjective Problem through Genetic Programming and Dominance-based Approaches." GECCO 2018, Proceedings of the Genetic and Evolutionary Computation Conference Companion, 1821-1828.
Sadigh, A. N., Mozafari, M., Karimi, B., (2012), "Manufacturer-retailer supply chain coordination: A bi-level programming approach.", Advances in Engineering Software, 45:144–152, 2012
Shah, N.K., Ierapetritou, M.G. (2015), "Lagrangian decomposition approach to scheduling large-scale refinery operations.", Computers and Chemical Engineering 79 (2015), 1–29.
Sinha, A.K., (2009), "Multi-Agent Based Petroleum Supply Chain Coordination: A Co-Evolutionary Particle Swarm Optimization Approach", Industrial Engineering, 1349-1354.
Talbi, E.G. (2013), "Metaheuristics for Bi-level Optimization", Springer,Heidelberg, 1st edition, 2013
Wu, F., Chu, N., Chu, C. and Zhou, M. (2009), "Short-Term Schedulability Analysis of Multiple Distiller Crude Oil Operations in Refinery With Oil.", IEEE Transactions on Systems, Man and Cybernetics 39 (2009), 1–16.
Xu, J., Zhang, S., Zhang, J., Wang, S. and Xu, Q., (2017)., "Simultaneous scheduling of front-end crude transfer and refinery processing.", Computers and Chemical Engineering 96, 212–236.
Zitzler, E.; Laumanns, M.; Thiele, L. (2001), "SPEA2: Improving the strength pareto evolutionary algorithm.", Evolutionary Methods for Design Optimization and Control with Applications to Industrial Problems,1:2–21, 2001

CRUDE OIL SCHEDULING USING BILEVEL OPTIMIZATION IN A QUANTUM-INSPIRED GENETIC PROGRAMMING MODEL

Abstract. *Crude oil refinery scheduling is a high complex problem wich encompass a set of decisions to optimize resource alocation, task sequencing and the time base definition of this tasks, do not violating different types of constraints and aiming to attend the multiples objectives. It has a combinatorial nature and, despite of its complexity, lacks of optimization tools to support the decision process. In this study, we propose the use of a Bilevel optimization algorithm to tackle the problem, i.e., the methodology splits the solution search in two decision levels. The first one, named Follower, ignores the problem constraints and, the other level, named Leader, takes them in consideration. The proposed algorithm is based on a quantum-inspired genetic programming model. The multiples objectives are tackled considering a priori and a posteriori approaches. The results are compared with two single-level models, one based on prioritization of objectives and one that uses multiobjective techniques, such as the dominance of solutions. Potential benefit of Bilevel is observed when compared to prioritization approach. From the results it is considered that the development of the research should be continued.*

Keywords: *Bilevel optimization, Multiobjective optimization, Genetic programming, Quantum-inspired genetic programming, crude oil refinery scheduling*