4
Modelo de Solução

Para construir o plano de drenagem de um determinado campo de petróleo são necessárias algumas informações prévias como, por exemplo, a composição geológica do campo. A partir dessas informações é possível identificar as regiões com maior possibilidade de acúmulo de hidrocarbonetos e com isso traçar uma alternativa de desenvolvimento com poços produtores e injetores localizados estrategicamente. Todavia, para que seja calculado o valor presente líquido da alternativa de desenvolvimento de um campo é necessário que seja estabelecido um cronograma de abertura os poços [16].

Para definir o cronograma de ativação dos poços é necessário estabelecer as características do campo a ser explotado, o número, a localização e os controles de produção e injeção dos poços e o tempo total de concessão para explotação do campo. Além disso, para cada poço, devem ser consideradas restrições de ordem técnica e operacional. As restrições técnicas dizem respeito aos tempos mínimos de intervenção e de retorno financeiro de cada poço e à disponibilidade da sonda de intervenção. A restrição operacional, por sua vez, está relacionada à capacidade máxima da plataforma de produção. Estabelecer essas restrições evitam a obtenção de soluções inviáveis e, consequentemente, contribuem para a diminuição do espaço de busca do problema de otimização.

O modelo computacional proposto neste trabalho busca identificar um cronograma de abertura de poços de uma determinada alternativa de produção que maximize o VPL dessa alternativa. Para isso são levadas em consideração todas as entradas e restrições mencionadas anteriormente. O modelo de solução é composto pelos Módulos Otimizador, Simulador e Avaliador, cujas interações estão ilustradas na Figura 4.1.

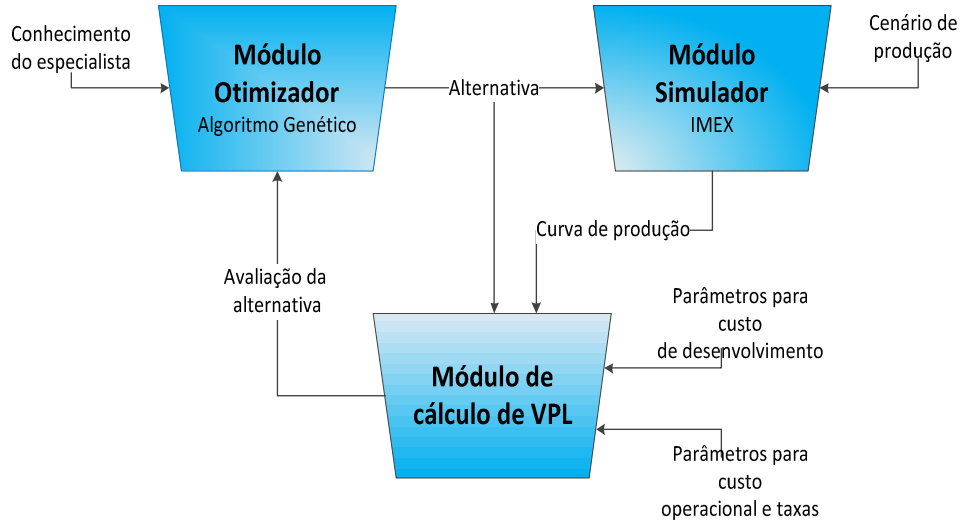

Figura 4.1: Módulos que compõem o modelo de solução

O Módulo Otimizador é composto de um algoritmo genético que tem a função de determinar o cronograma de abertura dos poços da alternativa. Essa alternativa com seu cronograma é então submetida ao Módulo Simulador, que utiliza um simulador de reservatórios para fornecer as previsões de produção de óleo, gás e água. Estas, por sua vez, são utilizadas pelo Módulo Avaliador para se obter o VPL da alternativa. Finalmente, o VPL calculado é enviado ao Módulo Otimizador que o utiliza como valor associado à avaliação da alternativa (cromossomo). As seções seguintes descrevem, em detalhes, cada módulo separadamente.

4.1. Módulo Otimizador

O Módulo Otimizador é composto de um algoritmo genético [29] [30], que busca encontrar o melhor cronograma de ativação de poços, a partir da evolução de uma população de indivíduos. Cada etapa deste módulo é descrita a seguir.

4.1.1. Restrições

Para definir o cronograma de ativação dos poços é necessário conhecer as características do campo a ser explotado, o número, a localização e os controles de produção e injeção dos poços e o tempo total de concessão para explotação do

campo. Além disso, para cada poço, devem ser consideradas restrições de ordem técnica e operacional. As restrições técnicas dizem respeito aos tempos mínimos de construção dos poços e de interligação dos poços com a plataforma de produção, disponibilidade da sonda de intervenção para fazer tal interligação, e tempo mínimo de retorno financeiro de cada poço. A restrição operacional, por sua vez, está relacionada à capacidade máxima da plataforma de produção e condições de operação dos poços. Estabelecer essas restrições evita a obtenção de soluções inviáveis e, consequentemente, contribuem para a diminuição do espaço de busca do problema de otimização.

A data de abertura do primeiro poço está condicionada à disponibilidade da plataforma. Já o intervalo mínimo de abertura entre poços é limitado pelos tempos mínimos de construção dos poços e interligação dos mesmos com a plataforma de produção.O modelo utiliza a informação do tempo mínimo de retorno financeiro dos poços para evitar a abertura de poços em datas muito próximas ao fim da concessão do campo. Assim, poços cuja data de abertura sejam posteriores ao seu tempo mínimo de retorno financeiro, não são abertos.

A fim de não ultrapassar a capacidade máxima de operação da plataforma de produção, os cronogramas otimizados que levam a produções superiores a essa capacidade, têm sua avaliação penalizada. O cronograma penalizado é excluído e, essa penalização se justifica pelo fato da produção diária da alternativa de produção só ser conhecida após a simulação da mesma. Dessa forma, os cronogramas otimizados sempre obedecem às restrições impostas pelo problema em questão.

4.1.2. Representação da Solução

A aplicação de Algoritmos Genéticos à resolução de problemas de otimização está condicionada à obtenção de uma representação adequada para a solução do problema. Neste módulo, as soluções são representadas por cromossomos compostos por dois segmentos distintos, conforme ilustra a Figura 4.2.

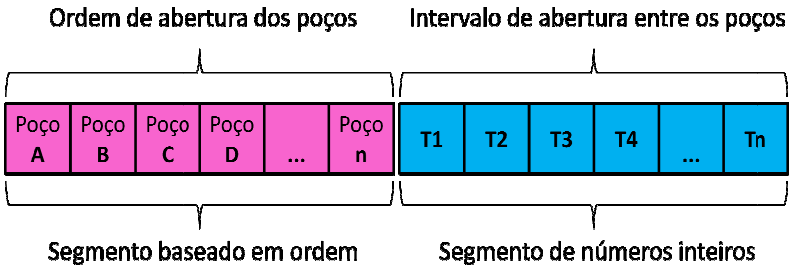

Figura 4.2: Representação do cromossomo

O primeiro segmento utiliza representação baseada em ordem e o número de genes é igual ao número de poços a serem ativados. Este segmento tem a função de definir a ordem de abertura dos poços do campo. Já o segundo segmento utiliza representação baseada em números inteiros e é responsável por definir o intervalo de abertura entre os poços. Vale destacar que esse tipo de representação possibilita evoluir separadamente a ordem de abertura e o intervalo de abertura, aplicando-se operadores genéticos apropriados a cada segmento do cromossomo.

4.1.3. Decodificação

Além da representação é necessário estabelecer uma regra de decodificação que permita construir a solução real a partir do cromossomo, para que possa ser efetivamente avaliada. Assim, segundo o modelo proposto, a ordem de ativação dos poços é dada pela ordem estabelecida no primeiro segmento do cromossomo. Já o intervalo entre as datas de abertura desses poços é determinado pelos genes do segundo segmento, onde: o primeiro gene do segundo segmento determina o intervalo de abertura do primeiro poço a ser aberto; o segundo gene determina o intervalo de abertura entre o primeiro e o segundo poço a ser aberto; e assim sucessivamente.

Na Figura 4.3 é apresentado um exemplo de um cromossomo e de sua respectiva decodificação. A decodificação acontece da seguinte forma: o poço "C" será o primeiro a ser aberto, e sua abertura acontecerá 90 dias a partir do início da concessão do campo, ou seja, no dia 01/04/2010, haja vista que a concessão do campo se inicia em janeiro de 2010. 660 dias após a abertura do poço "C" será aberto o poço "A", 120 dias após a abertura do poço "A" será aberto o poço "E", e assim sucessivamente. Isso é feito até que todos os poços sejam abertos ou se chegue à data limite para explotação do campo.

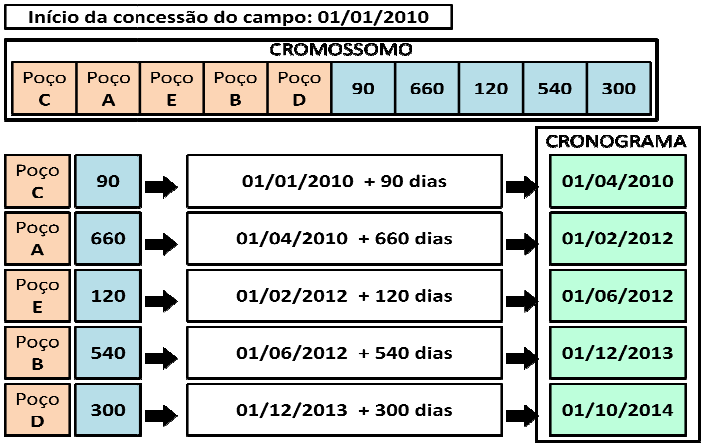

Figura 4.3: Exemplo de decodificação do cromossomo

4.1.4. Geração da População Inicial

A geração da população inicial de um algoritmo genético é fundamental para a eficiência do processo de otimização. No modelo de solução proposto a população inicial é gerada aleatoriamente e, assim, busca-se gerar indivíduos com ampla diversidade genética.

Entretanto, partindo-se de uma população inicial de indivíduos mais promissores, há uma chance maior de se encontrar soluções melhores com uma quantidade menor de ciclos de evolução (gerações) do algoritmo genético. Assim, este modelo permite incorporar sementes iniciais à população inicial.

4.1.5. Operadores Genéticos

Como mencionado anteriormente, os operadores genéticos são responsáveis pela modificação dos indivíduos a fim de gerar novas soluções para o problema em questão. Como este modelo apresenta um cromossomo com dois segmentos distintos, cada segmento sofre ação de operadores específicos, tanto de mutação quanto de cruzamento. Os operadores utilizados a cada geração são escolhidos aleatoriamente, através de uma roleta. A seguir são descritos os operadores aplicados a cada segmento do cromossomo.

4.1.5.1. Cruzamento

No primeiro segmento foram utilizados três operadores de cruzamento distintos, denominados: *PMXCrossover, OXCrossover* e *CXCrossover* [34]. O operador *PMXCrossover (Partially-Matched Crossover)* gera descendentes ao escolher uma subsequência da ordem de um dos pais e preservando a ordem do maior número possível de genes do outro pai.

O *PMXCrossover* estabelece dois pontos de corte aleatórios sobre os indivíduos pais e as subcadeias de genes encontradas entre esses cortes serão herdadas integralmente pelos indivíduos filhos. Essas subcadeias também determinam um mapeamento de relacionamento entre os genes dos pais e os genes fora da subcadeia deverão ser mudados seguindo este mapeamento. De acordo com o exemplo da Figura 4.4, são herdados integralmente pelos indivíduos filhos os genes CD e AE e como há um mapeamento de relacionamento entre os genes C e A e os genes D e E, ao se adicionar os genes restantes ao indivíduo filho1, o gene C do indivíduo pai2 passa ser A e o gene D passa a ser E. O mesmo acontece com o indivíduo filho2, onde o A passa a ser C e o E passa a ser D. Isso faz com que o processo de cruzamento não gere soluções inválidas.

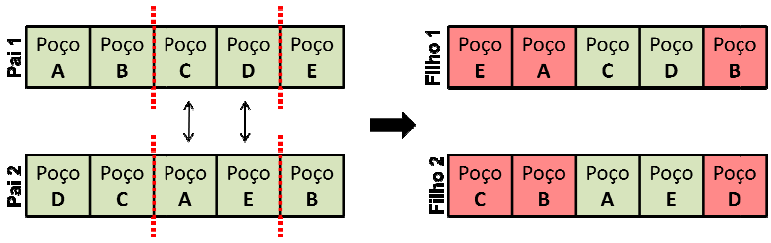

Figura 4.4: Operador de cruzamento *PMXCrossover*

Já o operador *OXCrossover (Order Crossover)* cria descendentes ao escolher uma subsequência de um dos pais e mantendo a ordem relativa dos genes do outro pai. Este operador estabelece dois pontos de corte aleatórios sobre os indivíduos pais, onde as subcadeias de genes encontradas entre estes cortes são herdadas integralmente pelos indivíduos filhos. A partir do último corte em cada cromossomo, o método faz uma busca cíclica no cromossomo do outro indivíduo pai pelos genes que não estão na subcadeia herdada, preenchendo assim o cromossomo, como mostra a Figura 4.5.

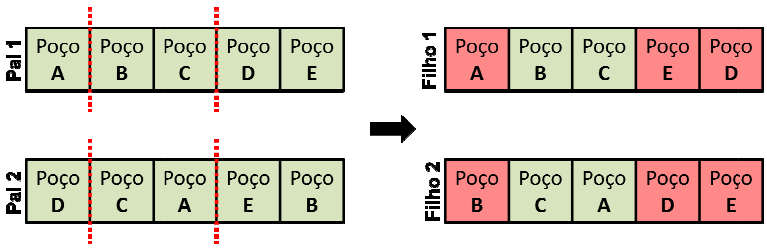

Figura 4.5: Operador de cruzamento *OXCrossover*

O operador *CXCrossover (Cicle Crossover)* atua sobre um subconjunto de genes que formam um "ciclo" de posições, em ambos os pais. No exemplo da Figura 4.6, pode-se identificar o ciclo A-D-D-E-E-A, cujos genes são copiados para um determinado filho e os genes faltantes são copiados do outro pai.

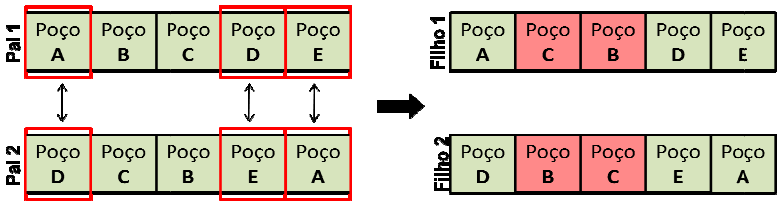

Figura 4.6: Operador de cruzamento *CXCrossover*

O segundo segmento sofre a ação de três operadores de cruzamento distintos: *SinglePointCrossover, TwoPointCrosover* e *IntegerUniformCrossover*.

O operador *SinglePointCrossover* efetua o cruzamento com um único ponto de corte, onde a primeira parte do cromossomo do Filho 1 é herdada do Pai 1 e a segunda parte do cromossomo é herdada do Pai 2, e o mesmo acontece com o Filho 2 (Figura 4.7). O operador *TwoPoinCrossover* efetua o cruzamento de forma semelhante ao operador *SinglePointCrossover*, contudo, com dois pontos de corte no segmento (Figura 4.8). O operador *IntergerUniformCrossover* utiliza um padrão (máscara) criado aleatoriamente para a geração dos filhos. Assim, onde houver um (0) na máscara, o gene é copiado do ancestral 1, e onde houver um (1), o gene é copiado do ancestral 2, e para gerar o segundo descendente, os ancestrais são trocados de posição (Figura 4.9).

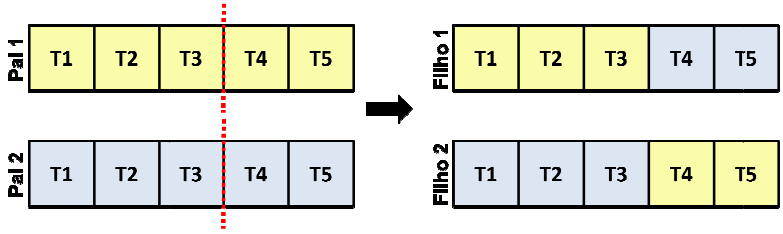

Figura 4.7: Operador de cruzamento *SinglePointCrossover*

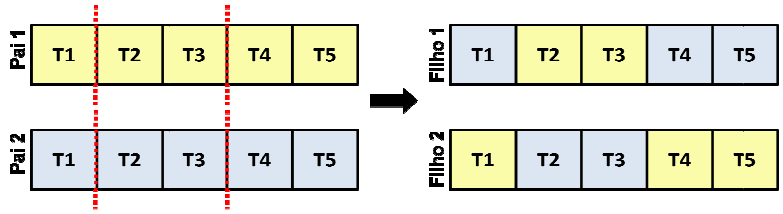

Figura 4.8: Operador de cruzamento *TwoPointCrossover*

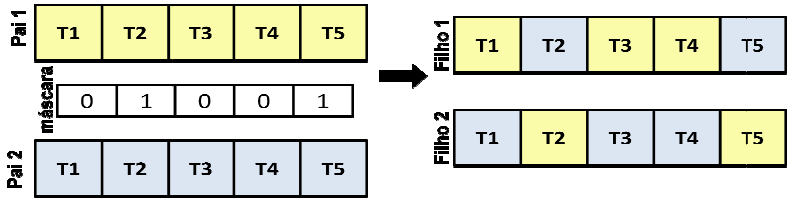

Figura 4.9: Operador de cruzamento *IntergerUniformCrossover*

4.1.5.2. Mutação

O primeiro segmento, baseado em ordem, sofre a ação de quatro operadores de mutação distintos, denominados: *PIMutation, SwapMutation, RotateLeftMutation e RotateRightMutation.*

O operador *PIMutation (Policy Iteration Mutation)* (Figura 4.10) inverte a ordem de um determinado intervalo de genes, cujos limites são escolhidos aleatoriamente, e o operador *SwapMutation* (Figura 4.11) sorteia dois genes do mesmo segmento e faz a troca dos seus valores. Já o operador *RotateLeftMutation* (Figura 4.12) aplica a mutação rotacionando os genes para a esquerda, enquanto o operador *RotateRightMutation* (Figura 4.13) aplica a mutação rotacionando os genes para a direita. A escolha do operador de mutação a ser aplicado a cada geração é feita aleatoriamente.

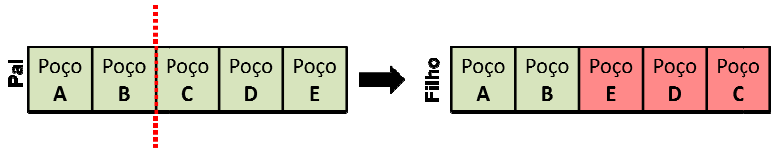

Figura 4.10: Operador de mutação *PIMutation*

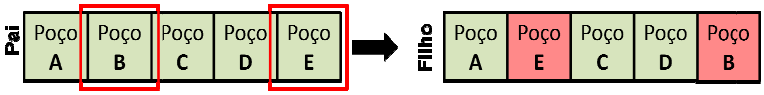

Figura 4.11: Operador de mutação *SwapMutation*

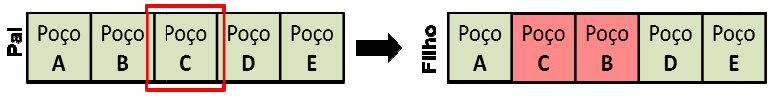

Figura 4.12: Operador de mutação *RotateLeftMutation*

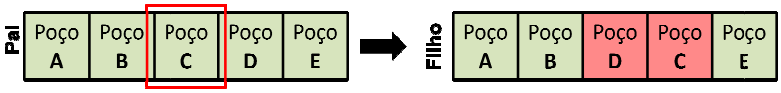

Figura 4.13: Operador de mutação *RotateRightMutation*

O segundo segmento, que utiliza genes do tipo inteiro, sofre a ação do operador de mutação uniforme denominado *IntegerUniformMutation* (Figura 4.14). Este operador realiza a mutação baseada em uma máscara aleatória. Nessa operação os valores dos genes selecionados são substituídos por um valor aleatório entre os limites máximo e mínimo do gene.

Figura 4.14: Operador de mutação *IntegerUniformMutation*

4.2. Módulo Simulador

O Módulo Simulador utiliza o simulador de reservatórios IMEX [35] para fornecer os perfis de produção de óleo, gás e água referentes à alternativa gerada pelo Módulo Otimizador. A otimização se inicia com todos os poços fechados e ao longo do processo de otimização as datas de abertura são alteradas de acordo com os cronogramas produzidos pelo Módulo Otimizador. Após a simulação, o simulador fornece um arquivo de saída contendo dados sobre a produção de óleo, gás e água. Estes dados são utilizados pelo Módulo Avaliador para determinar o valor de avaliação da alternativa.

4.3. Módulo Avaliador

O Módulo Avaliador usa os dados de produção para determinar o VPL da alternativa, calculando toda receita obtida e descontando os custos de operação e desenvolvimento do projeto. De acordo com [12], o valor presente representa uma precificação no dia de hoje para algum objeto ou evento que ocorrerá somente no futuro. Para isto, torna-se necessário ter uma boa previsão do comportamento deste evento futuro no momento atual. O cálculo do VPL é descrito a seguir pelas Equações 1-10.

$$VPL = VP - D \tag{1}$$

Onde:

VP: valor presente do projeto;

D: custo de desenvolvimento do projeto.

$$VP = (VPr - VPcop) * (1 - I) \tag{2}$$

Onde:

VPr: valor presente da receita;

$VPcop$: valor presente do custo operacional;

I: alíquota de impostos.

$$VPr = \sum^{T} \left[\frac{R(t)}{1 + tma^{\left(\frac{d(t)}{365}\right)}} \right] \tag{3}$$

Onde:

$R(t)$: receita no tempo t;

T: tempo total de produção;

tma: taxa mínima de atratividade;

$d(t)$: dia no tempo t.

$$R(t) = \left[\left(qo(t) * po(t) \right) + \left(qg(t) * pg(t) \right) \right] * \left(d(t) - d(t-1) \right) \tag{4}$$

Onde:

qo: vazão de produção de óleo(m^3/d);

po: preço de venda do óleo no tempo *t* (US$/bbl);

qg: vazão de produção de gás(1000m^3/d);

pg: preço de venda do gás no tempo *t* (US$/1000 m^3).

$$VPcop = \sum_{t=1}^{T} \left[\frac{Cop(t)}{1 + tma^{\left(\frac{d(t)}{365}\right)}} \right] \quad (5)$$

Onde:

Cop: custo operacional no tempo *t*.

$$Cop = M + Ry * R(t) + f + Cp + Ci \quad (6)$$

Onde:

M : custo de manutenção;

Ry : royalties;

f : custos fixos;

Cp : custos de produção;

Ci : custos de injeção.

$$M = m * np * \frac{[d(t) - d(t-1)]}{365} \quad (7)$$

Onde:

m: custo de manutenção por poço por ano;

np: número de poços abertos.

$$Cp = [cpo * qo(t) + cpw * qw(t) + cpg * qg(t)] * [d(t) - d(t-1)] \quad (8)$$

Onde:

cpo: custo de produção de óleo (US$/m3);

qo: vazão de produção de óleo (m3/d);

cpw: custo de produção de água (US$/m3);

qw: vazão de produção de água (m3/d);

cpg: custo de produção de gás (US$/1000m3);

qg: vazão de produção de gás (1000m3/d).

$$Ci = [ciw * qwi(t)] * [d(t) - d(t-1)] \tag{9}$$

Onde:

ciw: custo de injeção de água (US$/m3);

qwi: vazão de injeção de água (m3/d).

$$D = \sum^{N} \left[\frac{p}{(1+tma)^{\frac{da(t)}{365}}} \right] \tag{10}$$

Onde:

N: número total de poços

p: custo do poço mais o custo da árvore de natal;

$da(t)$: dia de abertura do poço no tempo t.

Os parâmetros: alíquota de impostos, tempo total de produção, taxa mínima de atratividade, preço de venda do óleo e do gás, custos de manutenção por poço, royalties, custos fixos, custos de produção e de injeção, custos de manutenção por poço e o custo de cada poço com árvore de natal, devem ser informados pelo especialista. Os parâmetros referentes às vazões diárias de produção e de injeção são frutos da simulação do reservatório. A equação de VPL considera custos fixos que, dentre eles, podem ser considerados os custos de aquisição de plataforma de produção. Contudo, a equação não leva em consideração custos de manutenção da plataforma, que podem variar ao longo do tempo.

4.4. Detalhes de Implementação

Para a implementação do modelo de solução proposto foi construído um sistema no ambiente MS Visual Studio 2008, usando a linguagem orientada a objetos C#. O sistema, denominado OCTOPUS 2.0, é um avanço em relação ao sistema OCTOPUS (versões 1.x) que se encontra implantado em alguns dos ativos da Petrobras. O sistema OCTOPUS é o resultado de uma parceria de pesquisa e desenvolvimento, entre a Petrobras e o Laboratório ICA, que teve início em 2000. Trata-se de um sistema desenvolvido especificamente para a solução do problema de otimização de planos de drenagem. A partir da descrição de um reservatório (modelo de simulação), o sistema busca encontrar, por meio de um algoritmo genético, uma configuração de poços que maximize o VPL do projeto. No processo de busca pela melhor configuração, são levados em consideração a quantidade, o tipo (injetor ou produtor), a trajetória (vertical, horizontal ou direcional) e a localização dos poços.

O sistema OCTOPUS foi inspirado em [12] e [36] e teve a sua versão preliminar disponível para testes no CENPES (Centro de Pesquisas e Desenvolvimento da Petrobras) em 2007. Em 2008 foi disponibilizada a primeira versão do sistema para uso nos ativos da empresa. A partir de então, o sistema vem sendo aperfeiçoado e modificado (versões 1.x) de acordo com as necessidades. Em [13] são apresentados alguns resultados importantes obtidos com a aplicação do OCTOPUS na otimização de planos de drenagem em casos reais da Petrobras.

Ao longo desses anos, com o amadurecimento do sistema OCTOPUS e também da própria equipe do ICA envolvida com pesquisas na área de E&P, surgiu naturalmente a ideia de transformar o OCTOPUS em um sistema mais abrangente, que pudesse apoiar especialistas da área na tomada de decisões associadas às mais diversas etapas do processo de exploração e produção de petróleo. Assim, o sistema OCTOPUS foi reestruturado a fim de possibilitar a inclusão de novas funcionalidades.

Em 2011 iniciou-se o desenvolvimento do OCTOPUS 2.0, que representa uma mudança de conceito em relação às versões 1.x. A partir da versão 2.0, o OCTOPUS deixa de ser um sistema específico para a otimização de planos de

drenagem e se torna um ambiente integrado para a gerência de reservatórios. Segundo esse novo conceito, o sistema permite o acoplamento de novas funcionalidades que se mostrem úteis para as atividades relacionadas à gerência de reservatórios. Uma dessas funcionalidades é a otimização do cronograma de ativação de poços, proposto neste trabalho, uma vez que o sistema OCTOPUS considera que todos os poços são abertos simultaneamente.

O modelo de solução proposto neste trabalho, segue a arquitetura modular do sistema OCTOPUS 2.0. No Módulo Simulador, descrito anteriormente na seção 4.2, a comunicação entre o algoritmo otimizador e o simulador de reservatórios se dá por meio de arquivos de texto. Assim há comunicação nos dois sentidos, escrita de arquivos na linguagem do simulador e leitura de alguns arquivos fornecidos pelo simulador. Nos arquivos enviados para o simulador estão contidas todas as características do reservatório, localização e trajetória dos poços, controles de produção e o cronograma de abertura dos poços.

Ao sistema OCTOPUS está acoplado o componente GACOM [37] que é desenvolvido e mantido pelo Laboratório ICA. Tal componente é utilizado pelo Módulo Otimizador, descrito na seção 4.1, uma vez que o GACOM fornece todos os métodos necessários para a construção de modelos de soluções baseados em Algoritmos Genéticos. O GACOM é constituído de vários módulos que podem ser reutilizados e agrupados conforme as necessidades do problema para o qual se deseja construir um modelo de solução. Além disso, essa organização modular permite também a prototipação fácil e rápida de novas soluções. Apesar de ser um componente com código fechado, o GACOM disponibiliza um conjunto de interfaces que possibilita a sua extensão por meio do acoplamento de módulos específicos que sejam necessários para o desenvolvimento de um determinado modelo de solução.