Bruno Nocera Zanette

Aplicação de métodos clássicos de busca local no problema de planejamento do despacho hidrelétrico

Dissertação apresentada como requisito parcial à obtenção do grau de Mestre em Informática no Programa de Pós-Graduação em Informática, setor de Ciências Exatas, da Universidade Federal do Paraná.

Área de concentração: *Ciência da Computação*.

Orientador: Fabiano Silva.

Curitiba PR
2017

Resumo

O planejamento do despacho hidrelétrico é uma questão que não só afeta o Brasil financeira e ambientalmente, como também é um grande problema computacional ainda não resolvido de forma eficiente. Esse planejamento constitui-se das ações operacionais que devem ser executadas mensalmente por cada usina hidrelétrica por um longo período, e o objetivo é atender a demanda de energia elétrica e manter os reservatórios de água das usinas cheios, de forma a reduzir o uso de outras fontes mais caras e poluentes. Essa questão é classificada como um problema de otimização não-linear com restrições de larga escala. Neste estudo foi desenvolvido um otimizador para o problema de despacho hidrelétrico, com base num simulador de despacho hidrelétrico e técnicas clássicas de busca local, capaz de encontrar planos melhores que o inicial num curto espaço de tempo. Este otimizador local pode ser usado em conjunto com algoritmos de busca mais refinados, provendo bons pontos iniciais para os mesmos.

Palavras-chave: algoritmos de busca local, monte carlo, geração de energia hidrelétrica.

Abstract

The planning of hydroelectric power plants is a matter that does not only affects Brazil financially and environmentally, but it is also a big computational problem that is not efficiently solved yet. This planning is composed by the operational actions that must be executed monthly by each hydroelectric power plant for a long period, and the objective is to meet the demand for electric power and to maintain the water reservoirs of the hydroelectric power plants filled, in order to reduce the usage of more expensive and poluent power sources. This question is classified as a large scale nonlinear optimization problem with restrictions. In this study was developed an optimizer for the problem of hydroelectric power plants planning, based on a simulator of hydroelectric power plant and classic local search techniques, capable of finding plans better than the initial one in a short period of time. This local optimizer can be used along with more refined search algorithms, providing good start points to them.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Acrônimos

DINF	Departamento de Informática
PPGINF	Programa de Pós-Graduação em Informática
UFPR	Universidade Federal do Paraná
ONS	Operador Nacional do Sistema Elétrico
SIN	Sistema Interligado Nacional
MME	Ministério de Minas e Energia (Brasil)

Capítulo 1

Introdução

A energia elétrica é um recurso essencial para o desenvolvimento e a sobrevivência de qualquer sociedade atual e, portanto, é de suma importância que seu fornecimento seja irrestrito e de baixo custo. No Brasil, o órgão responsável por coordenar e controlar a geração e transmissão de energia elétrica no Sistema Interligado Nacional (SIN) é o Operador Nacional do Sistema Elétrico (ONS). É do ONS a responsabilidade de criar um plano de geração de energia que garanta o funcionamento do SIN, o qual se constitui das ações operacionais mensais - quantidade de água vertida e turbinada - de cada usina hidrelétrica. Esse planejamento engloba as ações operacionais mensais para um período de 5 (cinco) anos - totalizando 120 ações - para todas as usinas hidrelétricas pertencentes ao SIN - que ultrapassa a casa da centena - o que resulta em mais de 12000 ações a serem planejadas.

Posteriormente esse plano deverá ser usado pelos agentes responsáveis pela gerência dessas usinas, os quais irão detalhar essas ações operacionais mensais em partes diárias e horárias para que enfim possam ser propriamente executadas. Neste trabalho foram estudados alguns métodos para buscar planos que satisfaçam a demanda de energia da melhor forma possível, utilizando um simulador de despacho hidrelétrico e algoritmos de busca local clássicos da literatura de inteligência artificial.

1.1 Sistema hidrelétrico brasileiro

Afim de fazer esse planejamento deve-se primeiro conhecer o funcionamento do sistema hidrelétrico brasileiro, o qual faz parte do SIN. O SIN é dividido em 5 (cinco) subsistemas: SE/CO (Sudeste/Centro-Oeste), S (Sul), NE (Nordeste), N (Norte) e Itaipu, como ilustrado na figura 1.1. De acordo com o diagrama esquemático das usinas hidrelétricas do SIN [ONS, 2015], atualmente são 163 (cento e sessenta e três) usinas hidrelétricas ativas, distribuídas entre 12 (doze) bacias hidrográficas e gerenciadas por 57 (cinquenta e sete) agentes diferentes.

O SIN utiliza diversas fontes diferentes de energia em conjunto para garantir o fornecimento ininterrupto de energia, para o caso de alguma falhar ou não conseguir gerar toda a

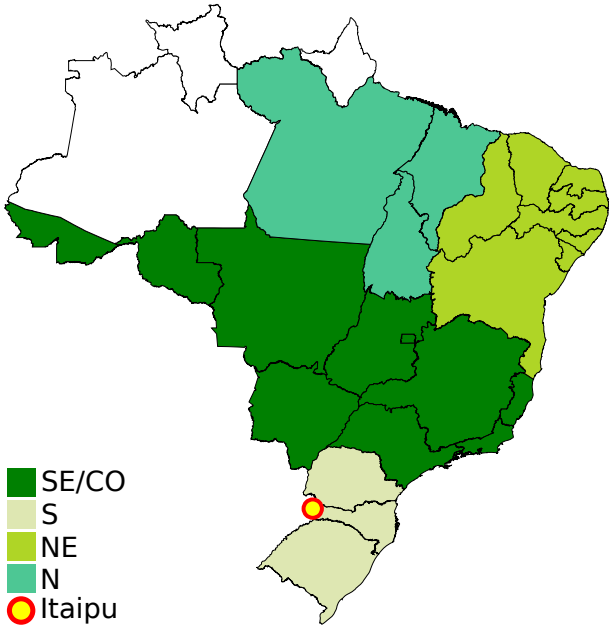


Figura 1.1: Mapa dos subsistemas do SIN

energia necessária. As maiores fontes de energia no Brasil são térmica, eólica, e a hidráulica, sendo essa última a mais barata e, graças à vasta rede hidrográfica presente no Brasil, origem de 78,7% de toda a energia produzida. A segunda maior fonte de geração de energia elétrica do país é a térmica, responsável por 17,2% da capacidade instalada no Brasil, tendo por combustível gás, biomassa, óleo diesel, carvão mineral e nuclear. A figura 1.2 exibe essas informações de forma mais detalhada.

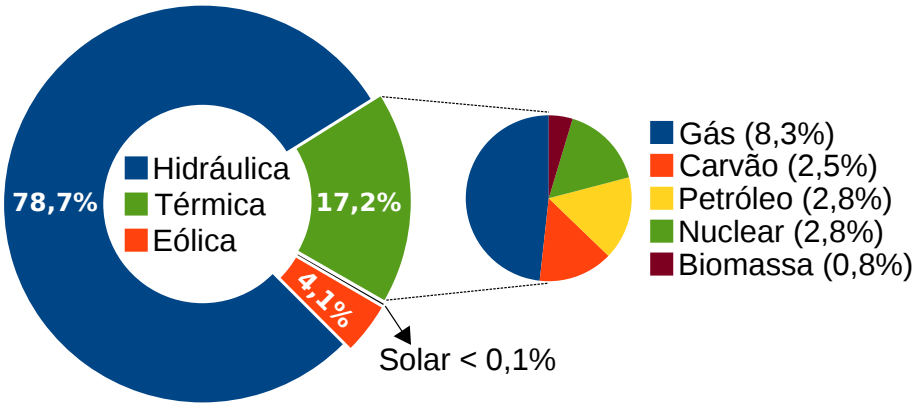


Figura 1.2: Matriz de produção de energia elétrica
Fonte: [MME, 2016]

A possibilidade de estocar os combustíveis, e assim não depender de fatores externos para poder produzir energia, torna as usinas termelétricas bastante confiáveis, pois há poucos fatores que podem interromper a produção, e dinâmicas, no sentido de poderem aumentar e reduzir a produção a qualquer momento. Por outro lado, as usinas hidrelétricas são completamente dependentes do nível de água dos rios nos quais estão instaladas, o que limita a produção e aumenta o risco de interrupção da produção.

A energia térmica seria a melhor fonte de energia elétrica se não fosse o fato de ser mais poluente e cara que a energia hidráulica, o que explica as porcentagens de produção de cada uma. Os combustíveis usados para gerar energia térmica possuem um preço elevado, diferente da água que está disponível livremente na natureza, além de que, durante o processo de produção, liberam elementos tóxicos no ar. Isso, aliado à limitação da produção das usinas hidrelétricas de acordo com a quantidade de água nos rios, faz com que um bom planejamento da produção seja essencial para garantir o menor custo da energia. Em outras palavras, é essencial manter a alta produção das usinas hidrelétricas durante todo o tempo, reduzindo a necessidade de produção de outras fontes mais caras, mas não tão alta a ponto de exceder a demanda.

1.2 Funcionamento das usinas hidrelétricas

Para que possamos discutir esse planejamento é necessário primeiro entendermos o funcionamento básico dos principais componentes de uma usina hidrelétrica. Existem dois tipos principais de usinas: usinas com reservatório e usinas fio d'água.

As usinas fio d'água não possuem reservatórios de água e utilizam turbinas que aproveitam a velocidade do rio para gerar energia, como descrito no capítulo 3 de [ANEEL, 2008].

> Essas usinas fio d'água reduzem as áreas de alagamento e não formam reservatórios para estocar a água ou seja, a ausência de reservatório diminui a capacidade de armazenamento de água, única maneira de poupar energia elétrica para os períodos de seca.

Ainda de acordo com [ANEEL, 2008], as usinas com reservatório são compostas por:

> basicamente, por barragem, sistema de captação e adução de água, casa de força e vertedouro, que funcionam em conjunto e de maneira integrada. A barragem tem por objetivo interromper o curso normal do rio e permitir a formação do reservatório. Além de estocar a água, esses reservatórios têm outras funções: permitem a formação do desnível necessário para a configuração da energia hidráulica, a captação da água em volume adequado e a regularização da vazão dos rios em períodos de chuva ou estiagem.

Nesse tipo de usina a energia é gerada pelo turbinamento de uma certa quantidade de água, que é a ação de passar a água por entre as turbinas, fazendo-as girar e transformando a energia potencial do volume de água armazenado em energia elétrica. Para que seja possível turbinar a água pelo conduto forçado, que nada mais é que uma tubulação onde a água escoa sob uma pressão diferente da atmosférica, o volume do reservatório deve estar sempre abaixo de um volume máximo e acima de um limite mínimo. Após passar pelo conduto forçado a casa de máquinas o volume de água turbinado vai para o canal de fuga para então seguir o fluxo normal do rio. A figura 1.3 ilustra esses componentes.

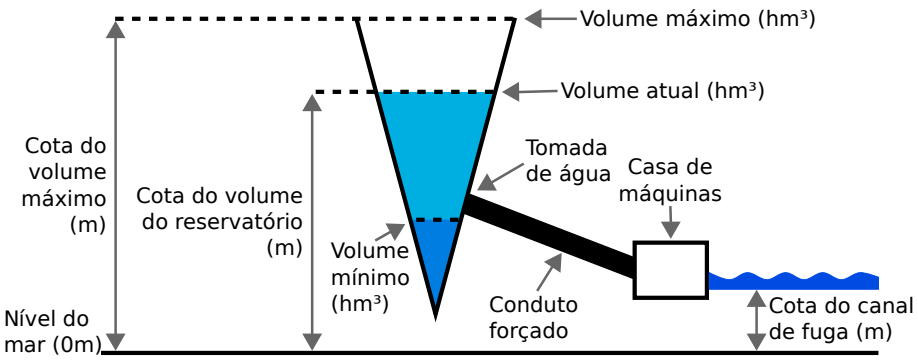


Figura 1.3: Componentes de uma usina hidrelétrica com reservatório

O volume de água contido no reservatório é influenciado não somente pela quantidade de água vertida e turbinada pela própria usina mas também pelas usinas a montante do rio, que alimentam o reservatório. Logo, para respeitar os limites do reservatório todas as ações operacionais devem ser planejadas em conjunto ou situações inesperadas podem acontecer. Por exemplo, se uma usina represar muita água ao mesmo tempo em que as outras usinas a montante vertem muita água, pode acabar faltando água para essas usinas. Sabendo que a geração de energia elétrica depende do cumprimento dessas restrições, é de suma importância planejar as ações operacionais de tal forma que essas situações jamais aconteçam.

Um fator que influencia diretamente no volume dos reservatórios, e consequentemente no planejamento das ações, são as variações sazonais. Todo ano presenciamos novos fenômenos meteorológicos que alteram o clima de diversas regiões do Brasil, o que acaba modificando a vazão dos rios. Todos esses dados são monitoradas por várias instituições e quase sempre são distribuídos públicamente. Porém, esses dados pertencem ao passado e podem não representar muito bem o que acontecerá no futuro. Portanto, afim de planejar ações operacionais futuras com um certo grau de confiança é indicado utilizar séries sintéticas de vazões, geradas por modelos estocásticos capazes de modelar séries que reproduzem as estatísticas históricas e preservam as correlações espaciais entre as usinas [Detzel et al., 2014]. Neste estudo foram usadas séries geradas pelo modelo CARMA [Detzel et al., 2014] para o projeto PHOENIX [Bessa et al., 2013].

Outro detalhe importante é que cada uma dessas usinas possui, por diversos motivos [Xavier et al., 2005], um potencial energético diferente, como também pode ser visto no diagrama esquemático das usinas hidrelétricas do SIN. Essas diferenças são um dos fatores críticos para o planejamento de produção, pois influenciam diretamente no modo como o volume de água deve ser utilizado por cada usina.

1.3 Objetivo do trabalho

Nesse capítulo foram resumidas as principais características e desafios do planejamento de geração de energia hidrelétrica no Brasil. Foi demonstrado que um planejamento mal feito pode

resultar na elevação dos custos e até a interrupção de parte do sistema, e que esse planejamento deve levar em conta diversos detalhes para evitar que esses problemas aconteçam.

O objetivo desse trabalho é implementar um otimizador de planos de geração de energia hidrelétrica com base em técnicas de busca local clássicas da inteligência artificial capazes de buscar planos melhores baseados numa função *fitness*. Um simulador do sistema hidrelétrico foi usado para validar os planos e calcular o volume e a quantidade de energia gerada por cada usina.

No capítulo 2 são descritos de forma mais detalhada os principais aspectos envolvidos no processo de geração de energia por uma usina hidrelétrica. No capítulo 3 são estudadas algumas técnicas e algoritmos de busca local capazes de resolver o problema da otimização do planejamento. No capítulo 4 são explicados os diversos componentes da solução encontrada. No capítulo 5 são apresentados e discutidos os resultados obtidos com a execução do programa implementado durante o estudo. Por fim a conclusão do trabalho apresenta as contribuições e trabalhos futuros.

Capítulo 2

Problema do planejamento da geração de energia hidrelétrica

O plano de geração consiste no planejamento de médio/longo prazo das ações operacionais mensais de uma usina hidrelétrica, as quais ditam os resultados mensais do sistema, que nesse caso é a quantidade de energia gerada e o volume dos reservatórios das usinas. O cálculo da energia gerada por cada usina envolve funções não-lineares e variáveis que incluem o volume do reservatório, a vazão de água vertida e turbinada e outras exclusivas de cada usina. Por sua vez, o cálculo do volume do reservatório depende diretamente das ações operacionais das usinas à montante. Além disso, é imprescindível que o plano seja factível, o que significa que as ações operacionais devem ser tais que as restrições operativas impostas pelo sistema sejam respeitadas durante todo o período avaliado.

Ou seja, tanto a validação do plano quanto os resultados mensais do sistema envolvem variáveis que só são instanciadas durante a simulação do sistema hidrelétrico. A consequência disso é que, a princípio, não há outra forma de abordar esse problema sem usar um simulador do sistema hidrelétrico que calcule a quantidade de energia produzida por um plano e o nível dos reservatórios de água das usinas, o que dificulta a resolução do mesmo por torná-lo, juntamente com a grande quantidade de variáveis a serem trabalhadas, caro computacionalmente. Essa característica também restringe o uso de métodos matemáticos e puramente heurísticos, pois é difícil detectar e prever o comportamento dos resultados desses cálculos. Por isso neste trabalho foram usados métodos de buscas estocásticos que não dependem diretamente do conhecimento prévio do comportamento do problema.

Nas próximas seções são abordados os principais aspectos contidos na elaboração e execução de um plano de geração de energia hidrelétrica. Na seção 2.1 é explicado como é feita a simulação de um plano, resumindo como essas ações, juntamente com as séries de vazões, afetam o volume do reservatório das usinas hidrelétricas. Esse cálculo do volume do reservatório é detalhado na seção 2.2 e as restrições operativas do sistema - e como estas são aplicadas na simulação - na seção 2.3. Na seção 2.4 é abordado de forma bastante resumida o cálculo da quantidade de energia gerada por uma usina hidrelétrica dada as ações operacionais executadas.

Na seção 2.5 é apresentado de forma breve como os resultados de cada plano foram avaliados. Na seção 2.6 é discutido o problema da transmissão. Por fim, na seção 2.7 são apresentados trabalhos correlatos que nortearam este trabalho.

2.1 Simulação e planejamento das ações operacionais

A simulação de um sistema hidrelétrico se resume na simulação das ações operacionais mensais de cada usina com o objetivo de estimar a quantidade de energia (E) que será gerada em cada mês e como se comportará o volume dos reservatórios (Vol) dado as séries de vazões de cada usina (Vz). Essas ações operacionais são a quantidade total de água que deve ser vertida (Qvt) e turbinada (Qc) num dado mês. O conjunto das ações operacionais de todas as usinas para todos os períodos formam o plano de geração. Diferente de todas as outras variáveis que compõem o sistema hidrelétrico, como as especificações técnicas das usinas e as séries de vazões, que não podem ser controladas pelo ONS, essas ações operacionais podem ser modificadas e influenciam diretamente no resultado da operação do sistema.

Por exemplo, um certo conjunto de ações operacionais pode fazer com que o sistema não gere muita energia mas mantenha os reservatórios sempre cheios, prevenindo a falta de água. Outro pode resultar numa quantidade muito grande de energia gerada nos primeiros meses mas que nos meses seguintes não consiga manter a produção devido ao baixo volume de água nos reservatórios. São incontáveis as possíveis combinações dessas ações operacionais e para cada uma delas há um resultado diferente.

Computacionalmente esse planejamento consiste em atribuir valores para as variáveis Qvt e Qc para cada mês, o que resulta em 120 (cento e vinte) valores reais para cada usina, para um plano de 5 anos, e tudo isso obedecendo as restrições operativas impostas pelo sistema, como descrito na seção 2.3. Portanto, no total são mais de 12000 (doze mil) variáveis que devem ser ajustadas, uma vez que a quantidade de usinas ultrapassa a casa das centenas. Porém, o real desafio não é apenas atribuir valores à essas variáveis de forma que o plano seja apenas válido, e sim encontrar um conjunto de valores que produzam o melhor resultado possível de acordo com as diretivas de quem está coordenando o sistema. Ou seja, caso o desejo do SIN seja produzir o máximo possível de energia, a melhor combinação será aquela que produza mais energia do que todas as outras combinações. Entretanto, devido a quase infinita gama de combinações, essa afirmação não pode ser feita pois não é possível testar toda elas. Sendo assim, a qualidade de um plano será diretamente proporcional à quão próximo desse objetivo o resultado for. Essa questão é abordada mais detalhadamente na seção 2.5.

Os valores dessas variáveis são definidos em arquivos que são passados como parâmetros para o simulador. Essa prática facilita a simulação de vários cenários diferentes, que é algo bastante comum se pensarmos que a previsão de chuvas é atualizada constantemente e que usinas podem ser construídas, modificadas, ou até mesmo destruídas. No total são três arquivos de entrada principais: um descritor das características das usinas, como o valor inicial da variável

Vol, um com o plano que deve ser simulado, contendo os valores de Qvt e Qc para cada usina para todos os períodos, e outro com a previsão das vazões (Vz) para cada usina para todos os períodos. Com exceção da variável Vol, todas as outras não são alteradas durante a simulação.

2.1.1 Séries de vazões

Antes de seguir com as explicações sobre a simulação do despacho hidrelétrico, é importante detalhar como as séries de vazões são representadas numericamente. Como já foi dito, cada usina possui uma série de vazão própria, entretanto, é importante notar que a vazão é referente ao rio em que cada usina está instalada e que várias usinas podem estar instaladas num mesmo rio. A vazão de um rio pode mudar ao longo do curso d'água e assim a vazão do mesmo nos pontos em que cada usina está instalada é diferente.

A figura 2.1 mostra dois cenários fictícios distintos. No cenário A o fluxo do rio não é alterado por nenhuma interferência e assim a vazão no ponto 3 é igual a soma das vazões dos pontos 1 e 2 com o incremento natural da vazão, exemplificado pelas chuvas. Já no cenário B, as barragens represaram toda a vazão, representando uma interferência, restando ao ponto 3 apenas a água provida pela chuva neste ponto. Esses exemplos comprovam como os valores das vazões em cada ponto do rio podem variar de acordo com as ações operacionais de cada usina.

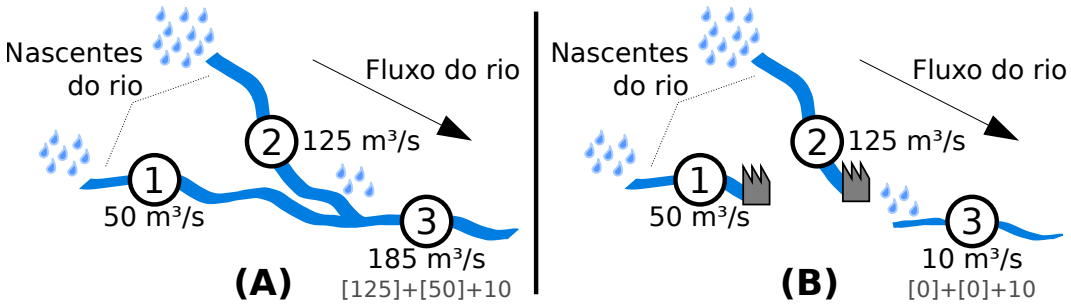


Figura 2.1: Exemplo da alteração no fluxo dos rios.

Uma vez que é impossível prever as ações operacionais de cada usina no momento de calcular as vazões, os valores são dados como a vazão incremental natural do rio no ponto em que cada usina está instalada, como mostrado no cenário A, que representa o estado original de escoamento no rio. Para calcular a vazão total afluente (VzT) ao reservatório de cada usina (s) e cada período (p) após a execução das ações operacionais das usinas à montante (Mon) é usada a equação 2.1.

$$VzT[s,p] = Vz[s,p] + (\sum_{m=1}^{Mon}(Qvt[m,p] + Qc[m,p]) - \sum_{m=1}^{Mon}(Vz[m,p])) \tag{2.1}$$

2.1.2 Exemplo de simulação

Para melhor explicar como a simulação do sistema hidrelétrico funciona, suponha um sistema com apenas 3 (três) usinas, cujas informações estão na tabela 2.1. Nessa tabela estão expostos valores imaginários para os principais parâmetros da simulação (Vol e Vz) e para as variáveis Qvt e Qc dessas usinas para um certo mês. Nos arquivos de entradas usados neste trabalho os valores de Qvt, Qc e Vz estão representados em m^3/s e o de Vol em hm^3 ($1000000\,m^3$). Porém, para facilitar a compreensão dos cálculos, serão usados os valores convertidos para hm^3, supondo que o mês simulado possui 30 (trinta) dias ($1m^3/s * (30 * 86400s)/1000000 = 2,592hm^3$), pois todos os cálculos serão feitos nessa medida. A figura 2.2 descreve como essas usinas estão dispostas fisicamente.

Usina	**Qvt** hm^3	**Qc** hm^3	**Vz** hm^3	**Vol** hm^3
1	100	150	50	1000
2	250	300	125	1250
3	0	310	185	500

Tabela 2.1: Exemplo de parâmetros e ações operacionais das usinas

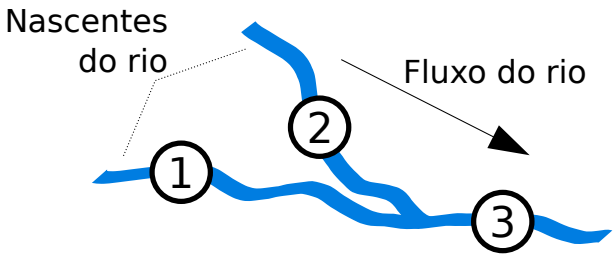


Figura 2.2: Exemplo de disposição física das usinas

A simulação começa a partir das usinas 1 e 2, as mais à montante no trecho do rio do exemplo. O primeiro passo é calcular a vazão total afluente, que no caso são os mesmos valores da vazão incremental natural do rio pois não há nenhuma interferência à montante dos locais onde essas usinas estão instaladas. O próximo passo é calcular o volume do reservatório de acordo como está descrito na seção 2.2. Para isso, soma-se a Vol a vazão incremental natural, resultando nos valores de 1050 hm^3 (1000 + 50) e 1375 hm^3 (1250 + 125) para as usinas 1 e 2, respectivamente. O passo seguinte é executar as ações operacionais, ou seja, verter e turbinar as vazões planejadas, o que reduz o volume do reservatório da usina 1 para 800 hm^3 (1050 − (100 + 150)) e da usina 2 para 825 hm^3 (1375 − (250 + 300)).

Após simular as operações das usinas mais a montante é a vez das usinas subsequentes no rio, que no caso é apenas a usina 3. O primeiro passo é calcular a vazão total afluente, somando o total de água turbinada e vertida pelas usinas à montante ((100 + 150) + (250 + 300) = 800), decrementando o somatório das vazões (50+125 = 175) e acrescendo a vazão incremental natural do rio no ponto em que a usina 3 está instalada, o que resulta em 810 hm^3 (185 + 800 − 175).

O próximo passo é calcular o valor da variável Vol da usina 3, somando-se ao volume a vazão total afluente, o que resulta em 1310 hm^3. Finalmente, as ações operacionais da usina 3 são executadas, turbinando 310 hm^3 e reduzindo Vol para 1000 hm^3.

Independente da quantidade de usinas existentes no sistema, a simulação segue esses mesmos passos para cada uma delas até o último período planejado. Para cada um desses períodos a quantidade de energia gerada por essas usinas é computada para que se possa mensurar a quantidade de energia elétrica produzida por cada subsistema e, consequentemente, o quanto a demanda de cada subsistema foi atendida, como é demonstrado na seção 2.5

2.2 Cálculo do volume do reservatório

O volume dos reservatórios das usinas são atualizados a cada passo da simulação, de acordo com os valores de VzT, calculados pela equação 2.1, e das variáveis Qvt e Qc da usina em questão (s) num dado período (p), de acordo com a equação 2.2.

$$Vol[s,p] = Vol[s,p-1] - (Qvt[s,p] + Qc[s,p]) + VzT[s,p] \tag{2.2}$$

Essa fórmula descreve matematicamente o comportamento do sistema fluvial, resumida na seção 2.1. Todo o volume de água liberado pelas ações operacionais das usinas à montante deságua no reservatório da próxima usina à montante, que por sua vez executa outras ações operacionais e assim por diante para todas as usinas. Quando não há nenhuma usina na sequência do rio o volume de água segue o caminho até o mar. Os reservatórios também são abastecidos pelas vazões dos rios, sendo essa a única fonte externa de água do sistema dentro da simulação.

2.3 Restrições operativas do sistema

Como mencionado na seção 1.2 da introdução, o SIN impõe algumas restrições à operação do sistema hidrelétrico brasileiro afim de mantê-lo em funcionamento de forma segura. Essas restrições são aplicadas para cada usina (s) e cada período (p), mais especificamente, nas variáveis Qvt, Qc - volume de água vertida e turbina - e Vol - volume do reservatório. Todas essas variáveis possuem limites que devem ser respeitados durante toda a operação conforme descrito na equação 2.3.

$$\begin{aligned} min\_Qvt[s] &<= Qvt[s,p] \\ 0 &<= Qc[s,p] <= max\_Qc[s] \\ min\_Vol[s] &<= Vol[s] <= max\_Vol[s] \end{aligned} \tag{2.3}$$

Cada usina possui limites (min_Qvt, max_Qc, min_Vol, max_Vol) diferentes os quais são especificados no arquivo de descrição das mesmas e são fixos durante toda a operação. Quando um desses limites é extrapolado a simulação deve ser interrompida e o plano é considerado inválido.

Afim de esclarecer como esses limites são aplicados na simulação, voltemos ao exemplo da seção 2.1. Desta vez, porém, suponhamos que além dos valores das variáveis Qvt, Qc, Vz e Vol definidos na tabela 2.1, também foram definidos os limites min_A, max_A, min_Vol e max_Vol para essas mesmas usinas, os quais estão exibidos na tabela 2.2.

Usina	**min_Qvt** m^3/s	**max_Qc** m^3/s	**min_Vol** hm^3	**max_Vol** hm^3
1	100	450	100	2000
2	75	300	125	1300
3	10	500	200	1000

Tabela 2.2: Exemplo de restrições das usinas

A primeira coisa que podemos notar é que todas as ações operacionais são possíveis de serem executadas, já que estão dentro dos limites estipulados, mesmo que a usina 1 esteja vertendo exatamente o mínimo de água permitido e a usina 2 turbinando o máximo. Esses limites podem ser verificados antes mesmo da simulação começar, pois tanto os valores dos limites quanto os das variáveis Qvt e Qc são independentes de outras variáveis e não são alterados durante a simulação.

Porém, esse plano de execução não é válido, pois o volume de água do reservatório da usina 3 é superior ao limite máximo ($1000 > 800$) ao final da simulação desse período. Dessa forma, a simulação seria abortada ao constatar essa obstrução das regras. Outro ponto importante é que o volume do reservatório da usina 2 não excede o limite, mesmo que ao somar a quantidade de chuva o valor resultante seja maior ($1375 > 1300$). Isso porque o limite só é aplicado após as ações operacionais serem executadas, já que o que está sendo avaliado é o período como um todo. Como já explicado, as ações operacionais são executadas ao longo de todo esse período, e portanto o valor final da variável Vol da usina 2 é $825hm^3$, que é menor que o limite máximo.

2.4 Cálculo da quantidade de energia gerada

O cálculo da quantidade de energia elétrica (E) produzida por um usina (s) num dado período (p) é dado pela multiplicação da vazão turbinada (Qc), produtividade da usina (P), definido no arquivo de descrição das usinas, e da altura da queda de água líquida (Al), como demonstrado na equação 2.4.

$$E[s, p] = Al * P * Qc[s, p] \tag{2.4}$$

O valor de *Al* é dado pelo cálculo da diferença entre a altura bruta da queda e a perda hidráulica, que corresponde à perda de carga durante a queda. Por sua vez, o cálculo da altura bruta é dado por polinômios do quarto grau, cujos parâmetros incluem algumas variáveis definidas no arquivo de descrição das usinas, a diferença entre o volumes atual e o antigo do reservatório e a quantidade de água que está sendo turbinada e vertida. Esses polinômios determinam a cota do reservatório em função do volume de água armazenado no mesmo e a cota do canal de fuga do reservatório em função da vazão total na usina hidrelétrica, ambos demonstrados na figura 1.3. Para mais informações sobre o cálculo da energia elétrica gerada por uma hidrelétrica eu sugiro a leitura do relatório técnico sobre usinas hidrelétricas elaborado pela EPE [EPE, 2008], no qual o mesmo é apresentado de maneira mais detalhada.

2.5 Qualidade do plano

A qualidade do plano é avaliada por uma função que calcula o quão perto de um objetivo pré-definido o resultado da simulação de um plano está. O valor resultante dessa função define o quão bom é o plano, sem que seja necessário armazenar ou processar mais nenhuma outra informação sobre o mesmo. Neste trabalho os objetivos são atender a demanda energética e manter os reservatórios de água cheios. Porém, como demonstrado na seção 4.4, o uso de duas variáveis diferentes no cálculo do *fitness* pode resultar em planos melhores, de acordo com a premissa estipulada, mas que não são os desejados. Por exemplo, um plano que resulte em reservatórios mais cheios mas que produza menos energia pode ser considerado um plano melhor do que um que mantenha os reservatórios num nível aceitável e produza mais energia. Para contornar esse problema será discutido o uso de penalidades e pesos para cada variável para moldar o comportamento da função.

A função *fitness* é usada pelo algoritmo que busca por planos melhores, o qual tenta encontrar planos que resultem em valores cada vez menores quando avaliados pela mesma. Durante essas buscas nenhum outro fator é levado em conta além desses valores. Por isso a consistência nos resultados é crucial, pois, caso essas funções resultem em ora valores maiores e ora valores menores para planos melhores, a busca teria que ser modificada para cada caso ou estaria por vezes buscando por planos piores. A diferença em usar cálculos e artifícios diferentes é que com cada um se estará buscando por um objetivo ligeiramente diferente.

Na seção 4.4 é apresentada a função criada durante este estudo e como as penalidades e os pesos afetam os resultados.

2.6 Rede de transmissão

Por fim, como nem sempre é possível, em um sistema predominantemente hidrelétrico, sem mencionar as questões de custo variável das usinas hidrelétricas, que todos os subsistemas

consigam suprir a própria demanda, é economicamente interessante que haja uma rede de transmissão de energia, para que se possa transferir a energia gerada em um ponto para outro. Fisicamente essa rede é formada de cabos e subestações de energia capazes de transmitir uma certa quantidade máxima de energia, as quais restringem as ações operacionais a serem executadas.

Essa rede pode ser representada por um grafo direcionado no qual as arestas representam as linhas de transmissão e os nós são os pontos de transmissão. O termo ponto foi utilizado intencionalmente pois pode-se considerar que essa transmissão é feita desde um subsistema para outro até de uma residência para outra, dependendo do grau de detalhamento desejado.

Neste trabalho foram utilizadas as informações fornecidas pelo projeto PHOENIX [Bessa et al., 2013], ilustradas na figura 2.3, considerando que a transmissão ocorre de um subsistema para outro. Também foi considerado que uma carga de energia pode ser transmitida livremente pelo grafo e que não há diferenças de custos entre os diversos caminhos. Ou seja, é possível transmitir a energia gerada por Itaipu até o subsistema Norte e o caminho escolhido não afetará o resultado do plano de geração. Essa suposição é falsa por duas razões: há tarifas de transmissão para cada linha de transmissão e perdas de carga que dependem da extensão do percurso percorrido, porém, não foi possível implementar esses custos em tempo hábil.

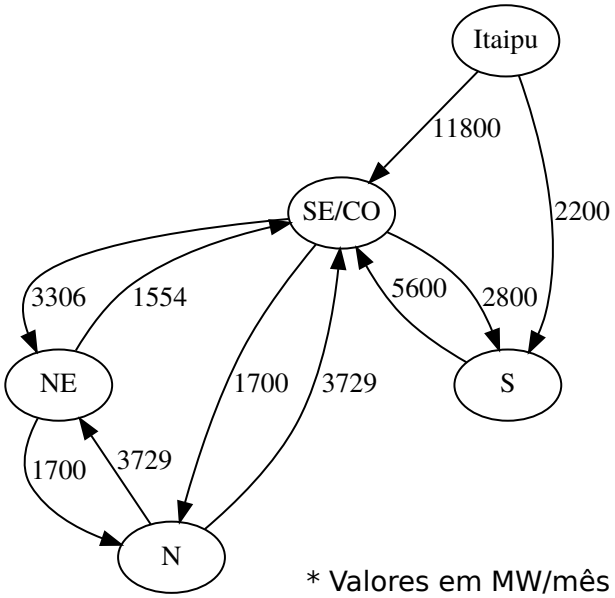


Figura 2.3: Grafo de transmissao

Com a adição da rede de transmissão ao sistema o cálculo de energia disponível (Ed) em cada subsistema (ss) em um certo período (p) deixa de ser apenas o que o mesmo produziu (E) e incorpora o quanto de energia foi recebida (Er) e transmitida (Et), conforme demonstrado na equação 2.5.

$$Ed[ss, p] = E[ss, p] + Er[ss, p] - Et[ss, p] \tag{2.5}$$

2.7 Trabalhos correlatos

Em sua pesquisa, [Andriolo, 2014] apresenta uma modelagem do problema de otimização da geração de energia elétrica via Algoritmos Genéticos, na qual a função *fitness* é composta por funções objetivo que procuram utilizar os reservatórios da melhor forma possível, como minimizar as perdas por vertimento em épocas chuvosas, custos de produção, economizar água em períodos de seca, atender à demanda por barra e os limites de intercâmbio entre subsistemas da rede. Os algoritmos foram implementados na plataforma Matlab 7.10.0 (R2010a) e os testes foram realizados em um computador Intel(R) Core(TM) i7-3770 CPU @ 3.40GHz, RAM de 16,0 GB com SO Windows 7, utilizando um sistema de 33 barras e 7 usinas hidrelétricas. O tempo de execução do teste com limitação de transmissão demorou 5,12 horas e os resultados foram satisfatórios, de acordo com o autor.

No trabalho [Bessa et al., 2013] é apresentado o modelo PHOENIX, que modela as usinas hidrelétricas individualmente com a inclusão de usinas térmicas e a análise de risco. Esse modelo foi usado como base para a implementação deste trabalho. Três métodos distintos de otimização foram avaliados: Otimização Pseudo-Booleana (PBO); Otimização por Enxame de Partículas (PSO); Otimização por Subida de Encosta (SE). De acordo com os autores os mais promissores foram o Enxame de Partículas e a Subida de Encosta. Os testes foram realizados com 111 usinas hidrelétricas e 32 usinas térmicas, mas não foram informados os tempos de execução de cada um.

2.8 Considerações

Nas seções desse capítulo foram apresentados os principais cálculos que compõem a simulação do despacho hidrelétrico e foi detalhado como as séries de vazões são representadas numericamente. Esses cálculos são utilizados pelo algoritmo simulador, detalhado na seção 4.1. Além disso, foram expostos os desafios do planejamento da geração de energia hidrelétrica, que consiste na otimização de milhares de variáveis, as ações operacionais mensais de todas usinas, de uma função não-linear com restrições. No capítulo 4 é apresentada a solução proposta por este trabalho para esse problema, baseada nas técnicas clássicas de busca local: subida de encosta, algoritmo têmpera simulada e métodos de Monte Carlo.

No capítulo 3 é feito um estudo das principais técnicas de busca local afim de verificar como as mesmas podem ser utilizadas para solucionar o problema proposto e no capítulo 5 são apresentados os resultados obtidos com a solução proposta neste trabalho.

Capítulo 3

Revisão bibliográfica

Existem diversos algoritmos diferentes que podem ser usados para buscar soluções para o problema descrito no capítulo 2. Entre eles estão os métodos de subida de encosta, têmpera simulada e algoritmos genéticos. Outro método que pode ser usado é o de Monte Carlo que, mesmo não sendo originalmente criado para esse fim, é capaz de encontrar soluções como é descrito na seção 3.2. Na seção 3.1 é estudado o funcionamento desses algoritmos, para que nos próximos possam ser adaptados para o problema proposto.

3.1 Algoritmos de busca local

Problemas de busca são bastante comuns na área de inteligência artificial, e se resumem em buscar um estado ou um sub-conjunto de estados dentro de um conjunto de estados, que aumente a qualidade da solução do problema de acordo com a função *fitness* - nome dado a função que avalia a qualidade do resultado, descrita na seção 2.5. Na maioria das vezes o caminho trilhado até a solução é irrelevante. Dependendo do problema esse conjunto pode ser grande demais para que seja possível procurar por todas as regiões do mesmo afim de encontrar a melhor solução existente. Por isso que, em alguns casos, deseja-se buscar em apenas uma sub-região desse espaço com o simples intuito de encontrar uma boa solução ou uma solução melhor que uma solução inicial. Os algoritmos utilizados nesses casos são denominados algoritmos de busca local.

Os algoritmos destinados à realizar essas buscas são categorizados de acordo com as garantias de resultados proporcionadas pelos mesmos. São completos aqueles que garantem que o objetivo será encontrado, caso exista; não-completos os que não garantem que o objetivo será encontrado; e ótimos aqueles que sempre encontram a melhor solução existente.

Os conceitos de máximo/mínimo local e global, que são utilizados nas subseções a seguir, vêm da análise do comportamento da função objetivo e dos valores que a mesma assume para cada estado válido. O máximo/mínimo global é o ponto em que a função assume o maior/menor valor dentre todos os pertencentes ao conjunto imagem da mesma. Já o máximo/mínimo local é

aquele em que a função assume o maior/menor valor dentre os pontos vizinhos. A figura 3.1 expõe esses e alguns outros comportamentos para a função objetivo.

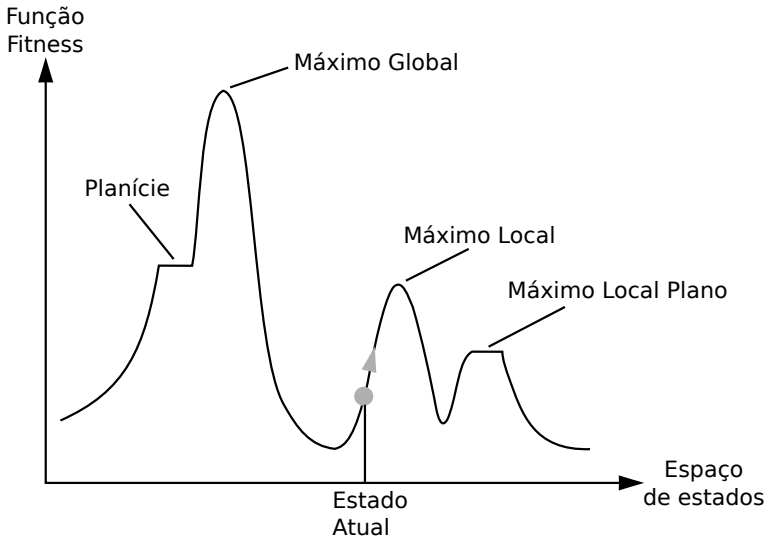


Figura 3.1: Análise do comportamento de uma função objetivo

3.1.1 Subida de encosta e Têmpera simulada

O algoritmo subida de encosta é talvez o mais simples e conhecido dentre os algoritmos de busca local. Definido como uma iteração que continuamente se move na direção do valor mais alto e termina ao chegar num ponto onde nenhum de seus vizinhos possuam valores maiores que o valor corrente, o mesmo é considerado não-completo pois nem sempre encontra solução pelo fato de poder ficar preso num máximo local. Durante toda a execução desse algoritmo apenas um estado é armazenado na memória por vez, sendo substituído toda vez que um estado considerado melhor é encontrado.

O algoritmo têmpera simulada alia aleatoriedade ao algoritmo subida de encosta para contornar o problema de ficar preso em máximos locais, e assim evoluir para um algoritmo de busca completo, pois sempre encontra uma solução. Assim como o algoritmo de subida de encosta, esse também armazena apenas um estado por vez. Esse método é baseado no processo metalúrgico de temperar, ou enrijecer, um metal ou vidro que é obtido esquentando o material até uma alta temperatura e depois esfriando-o gradualmente, fazendo com que o mesmo atinja um estado cristalino de baixa energia [Russell e Norvig, 1995]. Usando o exemplo dado por esse mesmo livro para exemplificar esse método:

> Imagine a tarefa de colocar uma bola de ping-pong na fenda mais funda de uma superfície acidentada. Se nós apenas deixarmos a bola rolar, ela irá parar em um mínimo local. Se nós agitarmos a superfície, nós podemos devolver a bola para fora desse mínimo local. O truque é agitar com força suficiente para apenas devolver a bola para fora de mínimos locais, mas não forte o bastante para desalojá-la do

> mínimo global. A solução têmpera simulada é começar agitando bastante (por exemplo, numa alta temperatura) e, em seguida, reduzir gradualmente a intensidade de agitação (por exemplo, reduzindo a temperatura).

Em outras palavras, o análogo à etapa de esquentar é permitir que o algoritmo realize pulos, com uma certa probabilidade disso acontecer, em direção a um ponto aleatório, mesmo que este seja pior do que o corrente de acordo com a função objetivo. Essa decisão a princípio pode parecer equivocada, mas é fundamental para evitar que o algoritmo fique preso em algum ponto, pois não restringe o caminho que o mesmo deve seguir. Ou seja, mesmo que o ponto corrente seja um máximo local, há uma chance de o algoritmo continuar a busca a partir de um ponto mais baixo que esse, o que pode levá-lo à um ponto ainda mais alto. O tamanho e a chance desses pulos acontecerem são decrementadas ao longo da execução do algoritmo - inspirado pela etapa de esfriar o material - pois considera-se que a cada passo o objetivo está mais próximo e portanto não precisa mais de tanto calor.

Resumidamente, o algoritmo têmpera simulada descreve uma maneira simples e eficaz de restringir a área de busca, facilitando o descobrimento de novas soluções. Devido à sua simplicidade, esse método pode ser utilizado em diversas situações, inclusive no problema estudado neste trabalho. Nesse caso, as variações no plano são cada vez menores conforme o mesmo se aproxime do objetivo, representando a etapa de esfriamento.

3.1.2 Busca em feixe local e algoritmos genéticos

Os algoritmos de busca em feixe local e genéticos, diferentemente dos já citados, armazenam mais de um estado ao mesmo tempo, e se beneficiam disso para gerarem novos estados de forma mais inteligente. A esse conjunto de estados é dado o nome de *população*, e um estado é chamado de *organismo* ou *indivíduo*.

O início do algoritmo de busca em feixe local se dá na geração de k indivíduos aleatórios, e a cada passo todos os sucessores desses indivíduos são gerados e avaliados. De todos esses sucessores são selecionados os k melhores, e assim o processo é reiniciado. Na variante estocástica desse mesmo algoritmo, ao invés de simplesmente selecionar os melhores indivíduos são selecionados k indivíduos ao acaso, com uma probabilidade maior de selecionar indivíduos melhores avaliados. Essa variação tem o intuito de aliviar o problema da busca se focar num espaço de busca muito pequeno pela falta de diversidade da população.

Os algoritmos genéticos, por sua vez, são uma variação do algoritmo de busca em feixe local estocástico e se inspiram na reprodução sexuada, e não na assexuada como é no caso do algoritmo original. Dessa forma, os novos indivíduos são gerados a partir da combinação de dois indivíduos. Essa combinação é feita em 3 (três) etapas principais: seleção, cruzamento e mutação. Na primeira etapa são selecionados os pares de indivíduos. Na segunda etapa os estados de cada um deles são divididos numa posição aleatória e então trocados, gerando um terceiro estado. Na última etapa esse novo indivíduo sofre uma mutação numa posição aleatória,

se tornando um indivíduo com características diferentes dos que o geraram. A principal vantagem oferecida por esse método vem da etapa de cruzamento, que é a possibilidade de combinar blocos de estados, os quais podem ser um pedaço do objetivo da busca.

De acordo com [Russell e Norvig, 1995], os algoritmos genéticos causaram um grande impacto em problemas de otimização, mas ainda não se sabe em quais condições os mesmos são bem sucedidos. Além disso, acredita-se que seja necessário ter uma representação cuidadosamente projetada do problema para se obter sucesso, o que demanda um bom conhecimento do problema. Por essas razões esses algoritmos não foram usados neste trabalho, mas têm potencial para obterem ótimos resultados.

3.2 Métodos de Monte Carlo

Os métodos de Monte Carlo são processos estocásticos, nos quais suas conclusões são obtidas a partir da observação de eventos aleatórios e podem ser usados para resolver tanto problemas determinísticos quanto probabilísticos [Hammersley, 2013]. Suas principais aplicações variam desde cálculos simples, como a probabilidade de uma face de uma moeda cair para cima, até a criação de uma inteligência artificial capaz de vencer um campeão mundial no jogo Go [Bouzy e Cazenave, 2001] [Silver et al., 2016].

A abordagem mais simples para problemas probabilísticos, com base nesses métodos, é observar números aleatórios que simulam o evento real e inferir o resultado a partir do comportamento desses números [Hammersley, 2013]. Por exemplo, para calcular a probabilidade de cair um certo número num lançamento de um dado, basta lançar esse dado repetidas vezes e anotar os números sorteados. O resultado será dado pela divisão da quantidade de vezes que esse número foi sorteado pelo total de lançamentos. Note que a quantidade de faces não foi especificada de propósito, pois esse método não depende dessa informação, sendo válido para qualquer dado, desde que este seja perfeito, ou seja, que todas as faces tenham a mesma probabilidade de serem sorteadas.

Porém, é importante esclarecer que essa classe de algoritmos não deve ser utilizada exclusivamente como uma caixa-preta [Andrieu et al., 2003], onde nenhum conhecimento prévio é aplicado, como no exemplo da moeda. Ao incorporar um conhecimento prévio do domínio do problema ao projeto da solução é possível obter resultados mais significativos, que sem esses requisitos tornariam a resolução difícil ou até mesmo impossível.

A resolução de problemas determinísticos comprovam a necessidade da aplicação desse conhecimento. Uma boa maneira de exemplificar isso é utilizar esse método para determinar o valor de π. Para isso, suponha uma circunferência de raio 1, $r = 1$, inscrita num quadrado com lados de tamanho 2, que, portanto, possui uma área de 4, $A_{quad} = 4$. Sabendo que a área de uma circunferência (A_{circ}) é dada pela equação $A_{circ} = \pi * r^2$, ao substituir o valor de r por 1, temos que a área dessa circunferência é $A_{circ} = \pi$. Dessa forma, basta calcular a área dessa circunferência para obter o valor de π. Para isso iremos sortear coordenadas ao acaso dentro desse quadrado, as

quais terão uma chance de pertencer ou não a área da circunferência, como demonstrado na figura 3.2. Após o sorteio de várias coordenadas é possível perceber que a proporção de coordenadas que pertencem a área da circunferência dentre todos os pontos sorteados, P_{circ} tende a 0.78537. Em outras palavras, aproximadamente 78.537% das coordenadas sorteadas dentro da área do quadrado pertencem a área da circunferência. Sabendo que nesse caso a área do círculo também pode ser calculada como uma proporção da área do quadrado, temos que, com $A_{quad} = 4$ e $P_{circ} = 0.78537$, $A_{circ} \approx P_{circ} * A_{quad} = 0,78537 * 4 = 3.1415 \approx \pi$.

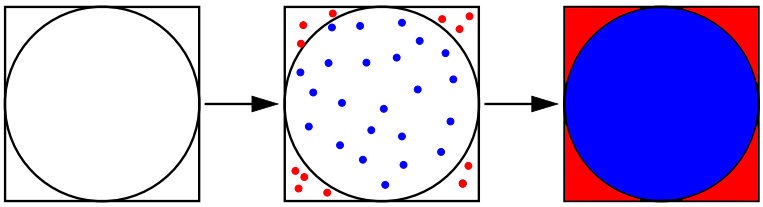


Figura 3.2: Cálculo da área da circunferência a partir da observação de pontos aleatórios

Nesse exemplo é possível perceber tanto onde o conhecimento prévio foi exigido quanto onde a aleatoriedade foi aplicada. Para formular e simplificar o problema de modo que seja possível resolvê-lo dessa forma exige um bom conhecimento prévio do assunto, pois sem conhecer as fórmulas e o comportamento das figuras geométricas utilizadas seria impossível encontrar a solução. Já a aleatoriedade foi utilizada para calcular a área do círculo com base apenas na observação e interpretação de eventos aleatórios.

3.2.1 Precisão dos resultados

Uma característica muito importante dos métodos de Monte Carlo é que a precisão dos resultados depende diretamente da quantidade de eventos aleatórios observados. Quanto mais eventos forem observados maior será a precisão. Nos exemplos previamente citados, sem perda de generalidade, poderia ter sido realizado apenas um lançamento do dado ou sorteado apenas uma coordenada, mas nesses casos os resultados seriam completamente diferentes - e errados - dos experimentos com mais observações. Se o número observado no lançamento do dado fosse igual ao número de interesse, poderia-se imaginar que a probabilidade de obtê-lo é máxima, da mesma forma que seria nula caso fosse diferente. Da mesma forma a área da circunferência poderia ser 0 ou 1, dependendo da coordenada sorteada. Porém, na medida que a quantidade de observações aumenta essa variação diminui e o resultado se aproxima cada vez mais do valor correto. A tabela 3.1 demonstra a evolução do cálculo do valor de π, com base no método descrito previamente, de acordo com a quantidade de observações feitas.

Tal comportamento pode ser explicado pois na medida que mais observações são feitas mais o espaço de busca é explorado, gerando uma informação que até então o algoritmo não possuía. No fundo, é essa informação que faz com que esses métodos funcionem. Ao observar um grande número de eventos o algoritmo passa a conhecer o espaço de busca bem o bastante

Observações	Proporção	Valor de π	Erro (%)
10^1	0.900000	3.600000	0.145916
10^2	0.820000	3.280000	0.044056
10^3	0.794000	3.176000	0.010952
10^4	0.789200	3.156800	0.004841
10^5	0.783830	3.135320	0.001997
10^6	0.785359	3.141436	0.000050
10^7	0.785319	3.141275	0.000101
10^8	0.785386	3.141543	0.000016
10^9	0.785402	3.141608	0.000005

Tabela 3.1: Evolução do cálculo do valor de π com base nos métodos de Monte Carlo

para inferir, com uma certa confiança, algo sobre ele. Porém, como também já foi abordado, não basta apenas ter essa informação, é preciso saber usá-la.

3.2.2 Capacidade de realizar buscas locais

Outra peculiaridade desses métodos é a capacidade de realizar buscas. Previamente foi demonstrado como a observação de eventos aleatórios pode ser utilizada para deduzir outros fatos. No entanto, essas observações também podem ser vistas como uma espécie de busca aleatória. Computacionalmente esse algoritmo é ineficiente pois no pior caso pode acabar fazendo mais comparações do que o próprio tamanho do espaço de busca. Por outro lado, em casos onde o objetivo da busca não é um elemento específico e sim uma região dentro desse espaço, e o espaço de busca é muito grande para que possa ter seu comportamento estudado, esse método pode ser uma alternativa bastante simples e funcional.

Imagine a situação de termos que encontrar, de olhos vendados, uma área quente numa superfície gelada apenas usando o tato. O método mais convencional de fazer essa busca seria deslizar a mão por essa superfície numa trajetória padronizada até encontrar a região de interesse, como demonstrado no lado esquerdo da figura 3.3. Porém, de maneira quase intuitiva, se simplesmente tocássemos em regiões aleatórias dessa superfície até encontrar essa região, como demonstrado no lado direito da figura, conseguiríamos, em muitas vezes, encontrar essa região mais rapidamente. A explicação para isso é que a aleatoriedade das comparações nos permite testar uma área maior do que apenas deslizando a mão, na qual essa área é lentamente revelada a cada pequeno movimento.

Ainda assim, é importante frisar que, mesmo nesses casos especiais, esse método de busca completamente aleatório continua não sendo muito eficiente, e também não garante que o objetivo da busca será alcançado. Outros métodos, como enxame de partículas, também utilizam a aleatoriedade como base para busca, mas se municiam de outros artifícios para se tornarem mais eficientes e confiáveis.

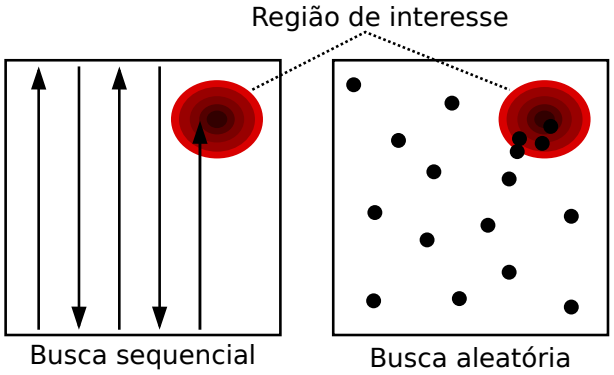


Figura 3.3: Diferentes métodos de busca de região

Um artifício que poderia ser usado para aumentar a eficiência na busca pela região mais quente seria focar numa região de busca ao encontrar uma região ligeiramente mais quente do que as outras testadas, uma vez que essa região quente também irá esquentar as regiões vizinhas. Dessa forma, a busca passa a não ser mais completamente aleatória e a cada ponto mais quente encontrado a região de busca é reduzida até que a região de interesse seja encontrada, aumentando consideravelmente a probabilidade de sucesso.

Essas características fazem dos métodos de Monte Carlo uma boa alternativa para a solução do problema proposto neste trabalho, especialmente se utilizados junto com o algoritmo têmpera simulada, do modo como explicado no parágrafo anterior. Esses métodos são simples, não necessitam de conhecimento prévio do problema e são capazes de obterem bons resultados, se bem utilizados.

3.3 Considerações

Nesse capítulo foram apresentados alguns algoritmos de busca local que foram usados como base para a construção do otimizador de planos de geração de energia hidrelétrica, além de conceitos importantes para a compreensão da solução que é apresentada no capítulo 4. Nesse capítulo é demonstrado como essas técnicas foram adaptadas e combinadas afim de solucionar o problema proposto.

Algumas características dos algoritmos subida de encosta e têmpera simulada foram utilizadas pelo algoritmo otimizador que é apresentado na seção 4.6. O método de Monte Carlo foi utilizado para a construção do buscador local, que é detalhado na seção 4.5.

Capítulo 4

Solução

A solução apresentada neste trabalho é um buscador local de planos baseado no método de Monte Carlo que é controlado por um algoritmo otimizador baseado nos métodos têmpera simulada e subida de encosta. Um simulador do sistema hidrelétrico brasileiro foi usado para validar e calcular os resultados mensais de cada plano. Esse simulador também faz pequenas correções no plano caso o mesmo seja considerado inválido durante a simulação. Para avaliar os planos foi implementada uma função objetivo baseada na demanda total de energia e no somatório dos volumes dos reservatórios das usinas de cada subsistema.

Este trabalho conta com um simulador simplista completamente funcional, um simulador da rede de transmissão de energia, um gerador de planos, um buscador de planos capaz de encontrar planos melhores que o inicial e um otimizador dos parâmetros do algoritmo de busca. Além disso, foram embutidos no simulador dois métodos que geram planos durante a simulação de acordo com heurísticas pré-definidas. Os planos e resultados obtidos com o uso desses métodos podem ser usados como bons pontos de partida para outros algoritmos mais refinados. Esses componentes, destacados em cinza no fluxograma da solução proposta apresentado na figura 4.1, são explicados nas próximas subseções. Durante essas explicações são utilizados os conceitos explicadas nas seções 2.1, 2.3, 2.4 e 2.6.

4.1 Simulador

O simulador foi implementado de maneira simples segundo o modelo de despacho hidrotérmico sugerido pelo projeto PHOENIX [Bessa et al., 2013]. Para isso foi usado um grafo direcionado em forma de árvore, como exemplificado na figura 4.2, onde os nodos representam as usinas hidrelétricas e as arestas o fluxo das vazões dos rios. Esse grafo é gerado a partir das informações de cada usina, lidas pelo programa durante sua inicialização, como os identificadores de cada usina e das usinas que estão na sequência do rio e a qual subsistema cada uma pertence. Durante esse processo também é lida a quantidade de energia demandada por cada subsistema em cada período.

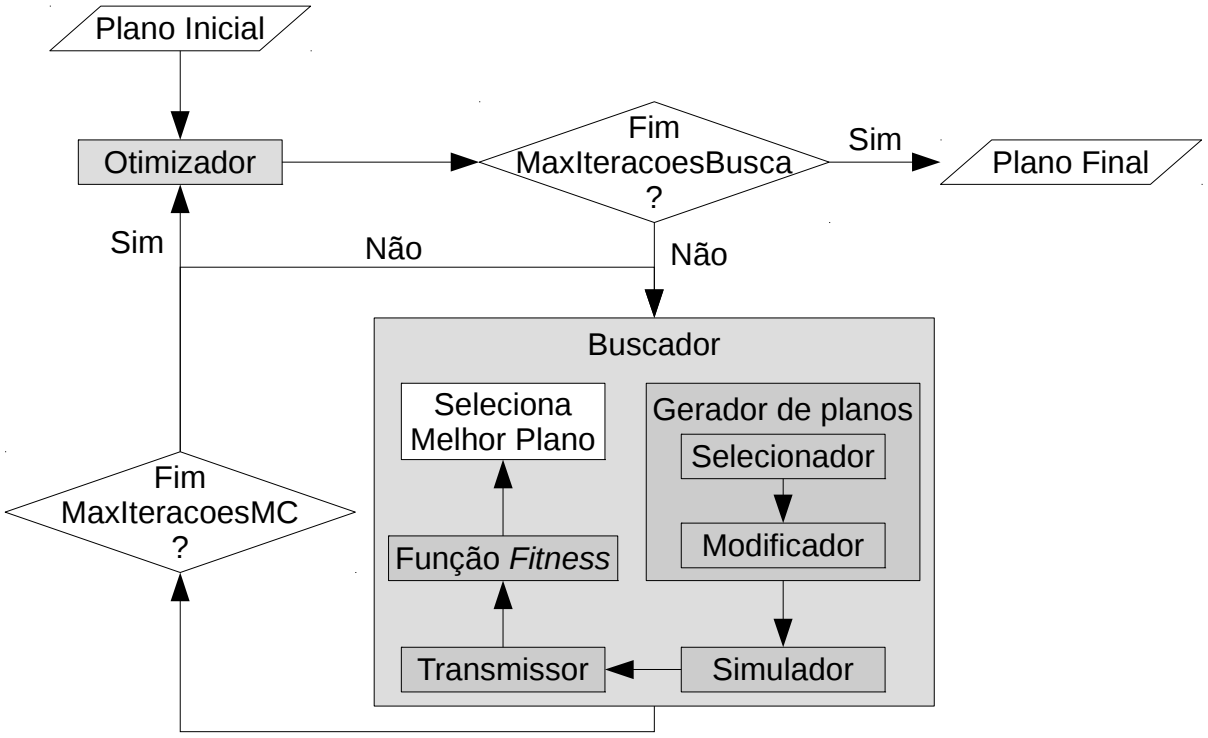


Figura 4.1: Fluxograma da solução proposta

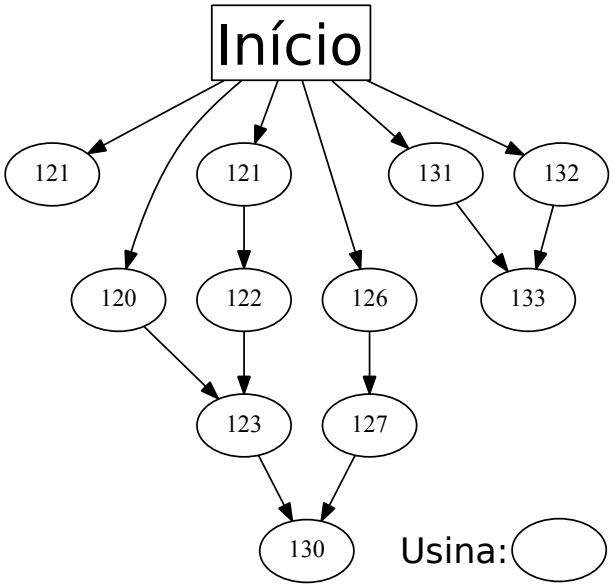


Figura 4.2: Exemplo de grafo de usinas

Os passos implementados são os mesmos explicados nas seções 2.2 e 2.4. A simulação é iniciada a partir dos nodos do topo da árvore, que representam as estações mais perto da nascente do rio. Esses passos são repetidos para todos os nodos da árvore e para cada período simulado. Entretanto, afim de reduzir a quantidade de acessos a memória e o tempo de execução do algoritmo, a ordem desses passos foi alterada para que todos os períodos de uma usina sejam simulados antes de iniciar os cálculos da próxima. Essa decisão não afeta os resultados da

simulação e faz com que as informações e as ações da usina precisem ser carregadas apenas uma vez. O algoritmo 1 exibe uma versão simplificada destes passos.

Na linha 3 do algoritmo do simulador (1) é feito o cálculo da vazão total, como demonstrado na seção 2.1.1, considerando que a função $calculaVazaoMontante$ retorna o somatório das vazões vertidas e turbinadas pelas usinas à montante da usina simulada. O cálculo do volume do reservatório da usina simulada após a execução das ações operacionais, explicado na seção 2.2, e a verificação dos limites mínimo e máximo do reservatório são feitas, respectivamente, nas linhas 4 e 5. A função $verificaVolume$ também faz correções nas ações operacionais do período simulado, se necessário, como descrito na seção 4.1.2. Na linha 6 é feito o cálculo da energia elétrica gerada pela usina simulada no período em questão, como explicado na seção 2.4. A última função executada dentro do laço principal do algoritmo é $verificaRestricoes$, que verifica o atendimento de todas as restrições operativas expostas na seção 2.3. Caso alguma das restrições não sejam respeitadas, a simulação é interrompida e o algoritmo retorna um erro. Após a simulação de todas as usinas e de todos os períodos o algoritmo retorna os resultados da simulação, tais como as séries de volume e de energia elétrica gerada.

Algoritmo 1 Algoritmo do simulador

Require: $usinasSimuladas$, $periodoInicial$, $periodoFinal$, Vol

1: **for all** $s \leftarrow emOrdemTopologica(usinasSimuladas)$ **do**
2: **for all** $p \leftarrow periodoInicial$ **to** $periodoFinal$ **do**
3: $VzT[s,p] \leftarrow calculaVazaoMontante(s,p,plano) + Vz[s,p]$
4: $Vol[s,p] = Vol[s,p-1] + emHectometroCubico(VzT[s,p],p)$
5: $verificaVolume(s,p,Vol[s,p],plano)$
6: $E[s,p] \leftarrow calculaEnergiaGerada(s,volumePrevio)$
7: **if not** $verificaRestricoes(s,p)$ **then**
8: **return** $erro$
9: **end if**
10: **end for**
11: **end for**
12: **return** $resultados$

4.1.1 Gerador de planos gulosos e conservadores

Com base na implementação do simulador, foram implementados dois métodos que geram planos em tempo de execução com base nos valores da variável Vol - volume do reservatório. Esses métodos foram denominados de guloso e conservador, inspirados pela heurística usada em cada um. Isso foi feito para comprovar o funcionamento do simulador como um todo, pelo fato delas terem um resultado único e previamente conhecido, e para servir como ponto de partida para a busca de planos melhores, no caso de não haver um plano inicial. Os planos resultantes desses métodos são considerados como ruins pois foram gerados a partir de

heurísticas bastante simples, que ignoram diversos aspectos considerados importantes para um bom planejamento, como o planejamento de todas as usinas como um todo. Sendo assim, de certa forma, esses planos também podem servir como uma avaliação inicial dos demais algoritmos, pois os mesmos, na teoria, devem ser capazes de encontrar planos melhores que esses para que sejam considerados minimamente eficazes. Ambos os métodos calculam os valores de Qvt e Qc - volume de água vertida e turbinada - de acordo unicamente com o valor corrente de Vol de cada usina e ignoram as futuras ações operacionais e as séries de vazões dos rios.

O método guloso consome toda água disponível nos reservatórios, mantendo-os sempre no nível mínimo. Para isso é turbinada toda a água possível de acordo com a capacidade de cada usina e então vertido o excedente de água do reservatório. O intuito desse método é gerar o máximo de energia possível em cada período. A figura 4.3 apresenta o resultado obtido pela execução desse método, e nela é possível observar que o reservatório da usina se mantém no limite mínimo durante quase toda a simulação. Perto dos períodos 7 e 40 o reservatório é cheio além do mínimo pois o limite de turbinamento desta usina foi alcançado e um excedente de água não pode ser turbinada.

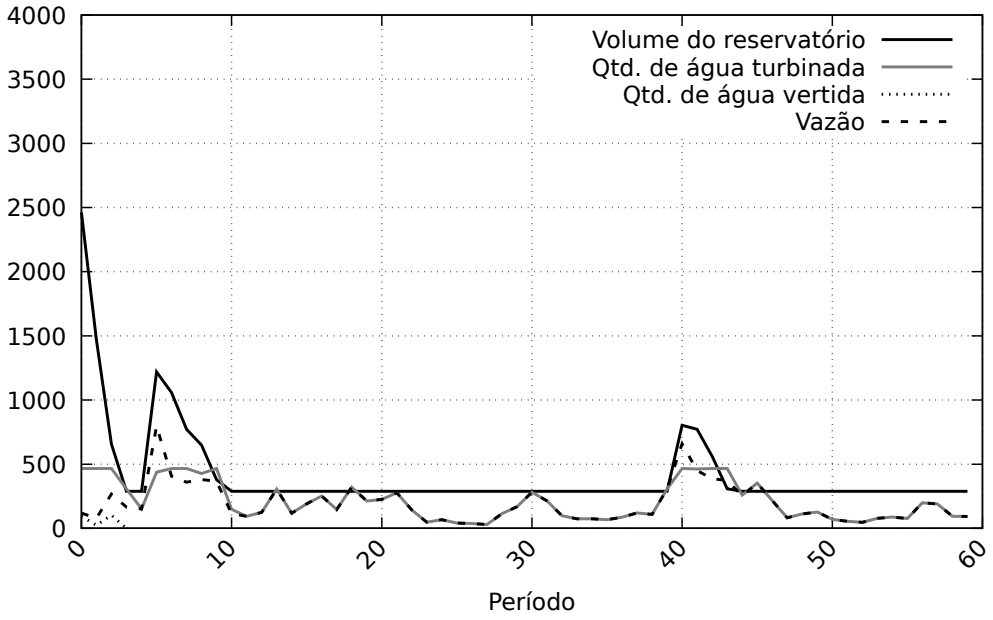


Figura 4.3: Resultados da simulação com um plano guloso

O método conservador, por outro lado, consome o mínimo de água possível afim de manter os reservatórios sempre cheios. A figura 4.4 apresenta o resultado desse método para a mesma usina usada no exemplo anterior. Nela é possível observar que o reservatório da usina chega ao seu limite máximo rapidamente. Após o reservatório chegar ao seu limite, toda a água provinda da afluência é turbinada, gerando energia. Próximo dos períodos 5 e 40 é possível observar que uma quantidade de água é vertida pois neste período a quantidade de água afluente supera a quantidade de água que a usina é capaz de turbinar.

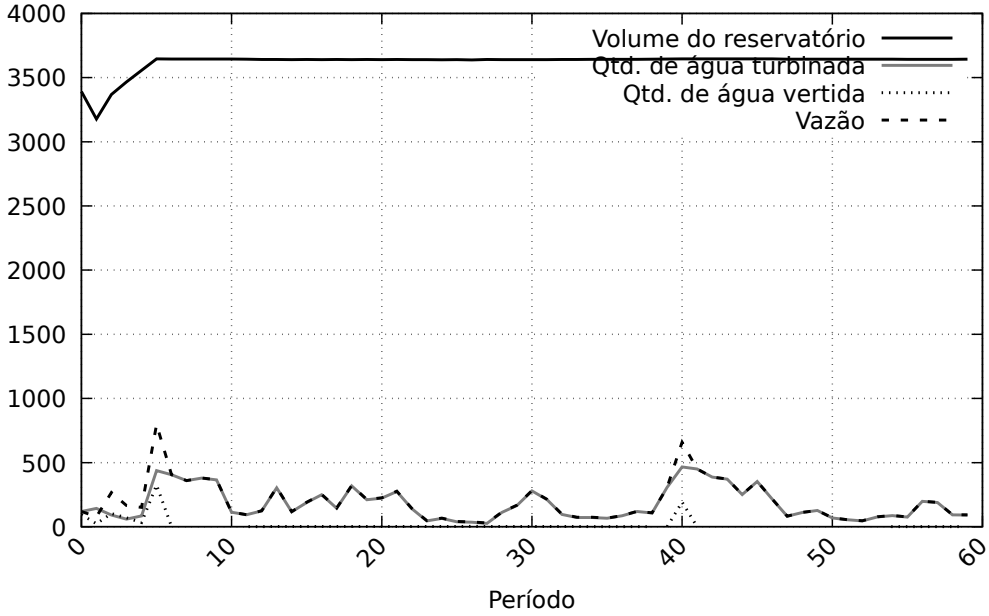


Figura 4.4: Resultados da simulação com um plano conservador

4.1.2 Correções no plano em tempo de execução

Durante os testes preliminares foi percebido que muitos planos estavam sendo considerados inválidos pelo simulador por extrapolarem o volume mínimo de água do reservatório, o que estava prejudicando os resultados do programa como um todo. Investigando o problema mais a fundo foi notado que a grande maioria desses casos poderiam ser corrigidos com pequenas alterações nas ações operacionais a serem executadas por essas usinas. Por isso, fugindo um pouco do propósito original do simulador e se aproximando um pouco do gerador de planos gulosos e conservadores, foi implementada uma função que corrige as ações, se necessário e se possível, de forma que o plano passe a obedecer as restrições operativas do sistema. Essa função simplesmente reduz a quantidade de água vertida e turbinada afim de deixar o volume de água existente no reservatório no nível mínimo. Para isso, primeiramente é reduzida a quantidade de água vertida, que não afeta o quanto de energia elétrica será produzida pela usina, e posteriormente, caso o volume continue abaixo do permitido, a quantidade de água turbinada.

4.2 Gerador de planos

O gerador de planos é talvez a parte mais simples da solução. O processo se divide em duas fases: seleção e modificação. Na fase de seleção são selecionadas as usinas que terão suas ações operacionais modificadas e na fase de modificação as ações operacionais das usinas selecionadas são modificadas de fato. Essas duas fases são detalhadas nas subseções a seguir.

4.2.1 Fase de seleção

A fase de seleção, detalhada no algoritmo 2, tem como parâmetros um conjunto de usinas, $usinasBuscadas$, e a quantidade de usinas a serem modificadas, $qtdUsinasModificadas$, que pode ou não ser igual ao total de usinas contidas em $usinasBuscadas$. O algoritmo de seleção retorna um subconjunto de $usinasBuscadas$ contendo $qtdUsinasModificadas$ usinas selecionadas aleatoriamente. É importante destacar que $usinasBuscadas$ pode ser diferente do conjunto total de usinas, o que permite estreitar o foco da busca em um subconjunto de usinas sem ter que alterar os dados de entrada.

O primeiro passo do algoritmo de seleção (2) é verificar se o valor do parâmetro $qtdUsinasModificadas$ é válido, ou seja, maior que zero e menor que a quantidade de usinas contidas no conjunto $usinasBuscadas$. Caso o valor não seja válido, o algoritmo retorna o conjunto $usinasBuscadas$. Nas linhas 2 a 5 é feito o preenchimento da variável $indices$, que inicialmente é um conjunto vazio. É importante esclarecer que esse conjunto não deve permitir a repetição de valores, ou o algoritmo não irá funcionar corretamente. O laço termina quando a quantidade de índices contidos na variável $indices$ for igual a quantidade de usinas que se deseja modificar ($qtdUsinasModificadas$). Foi decidido usar os índices do conjunto $usinasBuscadas$ para selecionar as usinas porque os identificadores das usinas não são sequenciais, o que impossibilita a seleção aleatória sem a validação do identificador, que é algo caro computacionalmente. O último passo, definido entre as linhas 6 e 9, é preencher a variável $subset$ com as usinas que estão armazenadas nos indíces do conjunto $usinasBuscadas$ selecionados no passo anterior. O algoritmo retorna a variável $subset$.

Algoritmo 2 Algoritmo do selecionador de usinas

Require: $usinasBuscadas$, $qtdUsinasModificadas$

1: **if** $qtdUsinasModificadas > 0$ **and** $qtdUsinasModificadas < qtde(usinasBuscadas)$ **then**
2: $indices \leftarrow cnjVazio$
3: **while** $qtde(indices) < qtdUsinasModificadas$ **do**
4: $indices \leftarrow RANDOM(seed)\%qtde(usinasBuscadas)$
5: **end while**
6: $subset \leftarrow cnjVazio$
7: **for all** $i \leftarrow indices$ **do**
8: $subset \leftarrow subset + usinasBuscadas[i]$
9: **end for**
10: **else**
11: **return** $usinasBuscadas$
12: **end if**
13: **return** $subset$

4.2.2 Fase de modificação

A fase de modificação, detalhada no algoritmo 3, recebe como parâmetro um conjunto de usinas qualquer e modifica os valores das variáveis do plano de cada uma delas afim de gerar um novo plano. Os planos das usinas restantes são copiados se necessário. Essas alterações são feitas somando ao valor original um percentual aleatório, negativo ou positivo, do valor limitado por uma taxa de variação máxima, $maxPercentage$. Além disso, o algoritmo também pode receber o parâmetro $idSubsistemaBuscado$, que determina o subsistema no qual a busca deve focar e uma variável de controle $usePeriodoCritico$ que habilita ou não o uso do período crítico. Caso seja determinado um subsistema e o uso do período crítico seja habilitado, o algoritmo identifica os pontos mínimos da série de volume do subsistema, resultante da simulação do plano original, escolhe um aleatoriamente e modifica apenas as ações entre alguns períodos precedentes ao período selecionado até o próprio período crítico. O objetivo disso é evitar modificar partes do plano que já estejam boas e priorizar a correção daquelas que estejam piores. Caso contrário, as ações de todos os períodos são modificadas.

O algoritmo 3 exibe os passos executados durante a modificação do plano. A inicialização das variáveis é feita entre as linhas 1 e 9. Nesse trecho destaca-se o uso da função $getPeriodoCritico$ para identificar e selecionar um período crítico e a opção de modificar as ações a partir de 9 períodos antes do período crítico. Na linha 10 é feita a cópia do plano original para um novo plano que será modificado. O algoritmo então calcula os sinais e as variações para cada ação operacional, respeitando a taxa de variação máxima especificada pela variável $maxPercentage$, e as soma às ações do plano copiado, gerando novas ações. Esses passos, descritos entre as linhas 13 e 18, são executados para cada usina do conjunto $cnjUsinas$ e para os períodos entre os valores das variáveis $periodoInicial$ e $periodoBuscado$. O algoritmo retorna o plano modificado.

Algoritmo 3 Algoritmo do modificador de planos

Require: $periodoFinal, idSubsistemaBuscado, cnjUsinas$

1: $periodoInicial \leftarrow 0$
2: $periodoBuscado \leftarrow periodoFinal$
3: **if** $idSubsistemaBuscado$ **not** $null$ **and** $usePeriodoCritico$ **then**
4: $\quad periodoCritico \leftarrow getPeriodoCritico(idSubsistemaBuscado)$
5: $\quad$ **if** $periodoCritico$ **not** $null$ **then**
6: $\quad\quad periodoBuscado \leftarrow periodoCritico$
7: $\quad\quad periodoInicial \leftarrow periodoBuscado - 9 >= 0 ? periodoBuscado - 9 : 0$
8: $\quad$ **end if**
9: **end if**
10: $novoPlano \leftarrow copiaDe(originalGPlan)$
11: **for all** $s \leftarrow cnjUsinas$ **do**
12: $\quad$ **for** $p \leftarrow periodoInicial$ **to** $periodoBuscado$ **do**
13: $\quad\quad sinal \leftarrow ((RANDOM(seed)\%100) < 50) \ ? \ 1.0 \ : \ -1.0$
14: $\quad\quad perc \leftarrow (RANDOM(seed) \mod maxPercentage)/100.0$
15: $\quad\quad novoPlano.Qc[s,p] \leftarrow novoPlano.Qc[s,p] * (sinal * perc)$
16: $\quad\quad sinal \leftarrow ((RANDOM(seed)\%100) < 50) \ ? \ 1.0 \ : \ -1.0$
17: $\quad\quad perc \leftarrow (RANDOM(seed) \mod maxPercentage)/100.0$
18: $\quad\quad novoPlano.Qvt[s,p] \leftarrow novoPlano.Qvt[s,p] * (sinal * perc)$
19: $\quad$ **end for**
20: **end for**
21: **return** $novoPlano$

Os parâmetros $cnjUsinas$, $qtdUsinas$ e $maxPercentage$ foram inspirados no algoritmo têmpera simulada, explicado na seção 3.1.1, e são importantes pois definem o quão diferente do plano original o novo plano pode ser. À medida que os valores dessa variáveis são decrementados os planos gerados serão cada vez menos diferentes do original, de forma análoga a etapa de esfriamento do algoritmo têmpera simulada. São essas as variáveis que o algoritmo otimizador, detalhado na seção 4.6, controla para que o buscador local de planos, apresentado na seção 4.5, continue encontrando planos.

4.3 Rede de transmissão

O algoritmo de transmissão de energia utiliza uma abordagem de força bruta para transmitir toda a energia possível dos subsistemas com superávit de energia para aqueles com déficit. Como já explicado na seção 2.6, foi considerado que é possível transmitir energia de um subsistema A para um subsistema C, caso haja um subsistema B que receba de A e transmita para C.

Em cada iteração do algoritmo de transmissão (4) é verificado se o subsistema tem superávit de energia e, em caso positivo, é transferido o máximo de energia possível para os subsistemas que demandam e estão em condição de receber, sempre respeitando o limite máximo de transmissão. O algoritmo termina quando não houver nenhuma transmissão na última iteração, o que significa que não há mais energia sobrando ou faltando em nenhum subsistema. Não foi implementado um cálculo de prioridade de transmissão, para os casos em que um subsistema pode transmitir para mais de um receptor.

O cálculo da energia requerida por cada subsistema leva em conta o déficit de energia de todos os subsistemas para os quais o mesmo pode transmitir, para garantir a propriedade transitiva da transmissão. Para exemplificar, suponhamos que o subsistema B tenha um superávit de $100MW/mes$ e possa transmitir para os subsistemas C e D, que estão com um déficit de energia elétrica de $200MW/mes$ e $50MW/mes$, respectivamente. Se o cálculo considerar apenas o subsistema B, o resultado seria nulo, pois o mesmo não tem déficit de energia elétrica. Porém, ao englobar os subsistemas receptores de B no cálculo, temos que B precisa $150MW/mes$ ($200MW/mes$ -$100MW/mes$ -$50MW/mes$). Dessa forma, A irá transmitir $150MW/mes$ para B, se possível, para que B retransmita essa energia elétrica para C e D posteriormente.

O algoritmo de transmissão (4) é executado para todos os subsistemas e para todos os períodos. Primeiramente é calculada a quantidade de energia requerida pelo subsistema em questão ($superavitEnergia$), passo descrito na linha 5, e então, na linha 6, é verificado se há superávit de energia elétrica. Caso haja, é iniciado um laço para todos os subsistemas receptores de energia elétrica do subsistema em questão. Dentro desse laço são calculados: a quantidade máxima de energia elétrica que pode ser transmitida ($maxDisponivel$), na linha 8, e a quantidade de energia elétrica requerida pelo subsistema receptor ($energiaRequerida$), na linha 9. Logo após esses cálculos, na linha 10, é verificado se é possível e necessário realizar a transmissão e, se for, é realizado o cálculo da quantidade de energia elétrica que será transmitida. Esse cálculo, descrito nas linhas 11 e 12, se resume a obtenção do valor mínimo dentre os valores das variáveis $superavitEnergia$, $maxDisponivel$ e $energiaRequerida$, que representa a quantidade máxima possível e necessária de energia elétrica a ser transmitida. Na linha 13 é decrementado do valor da varíavel $superavitEnergia$ a quantidade de energia elétrica transmitida e na linha 14 é executado a função $transmitirEnergia$ que armazena o valor transmitido nos resultados da simulação do plano. Esse laço é executado enquanto o valor da variável $transmitindo$, atualizada na linha 15, for verdadeiro. O algoritmo não retorna nenhum valor, pois todos os valores calculados são armazenados nos resultados da simulação do plano ao longo da execução do mesmo.

Algoritmo 4 Algoritmo do transmissor de energia elétrica

Require: Resultados da simulação

```
1:  for all p ← todosPeriodos do
2:      repeat
3:          transmitindo ← false
4:          for all ss ← todosSubsistemas do
5:              superavitEnergia ← calculaEnergiaRequerida(ss, p)
6:              if superavitEnergia > 0 then
7:                  for all r ← getSubsistemasReceptores(ss) do
8:                      maxDisponivel ← getTransmissaoMaxima(ss, r) − getEnergiaEnviada(ss, r, p)
9:                      energiaRequerida ← calculaEnergiaRequerida(r, p)
10:                     if superavitEnergia > 0 and energiaRequerida > 0 and maxDisponivel > 0 then
11:                         energia ← min(energiaRequerida, maxDisponivel)
12:                         energia ← min(energia, superavitEnergia)
13:                         superavitEnergia ← superavitEnergia − energia
14:                         transmitirEnergia(ss, r, p, energia)
15:                         transmitindo ← true
16:                     end if
17:                 end for
18:             end if
19:         end for
20:     until transmitindo
21: end for
```

4.4 Função *fitness*

A função *fitness*, por vezes também chamada de função objetivo, é a responsável por avaliar a qualidade da entrada, como já explicado na seção 2.5. A função implementada, detalhada no algoritmo 5, calcula os déficits normalizados de energia elétrica e volume do reservatório para todos os subsistemas (ss) e períodos (p), aqui denominados de *fits* reais. Os *fits* reais que superam o limite máximo estipulado para cada variável ($percMinVolume$ e $percMinEnergia$) são penalizados, sendo elevados a uma potencia previamente definida ($penalidadeVolume$ e $penalidadeEnergia$). Os valores resultantes dessas verificações de penalidade são multiplicados pelo peso dado a cada variável ($pesoVolume$ e $pesoEnergia$), e à esses valores foi dado o nome de *fit*. O valor do *fitness* é o somatório de todos os *fits* dos subsistemas buscados.

Como já abordado na seção 2.5, o uso de duas variáveis no cálculo do *fitness* pode levar a resultados corretos mas não desejados. Justamente por isso, os pesos e penalidades foram pensados, para permitir dar mais relevância para a redução dos déficits de uma certa variável ou para períodos em que o déficit é muito grande. Ou seja, um plano pode ser considerado melhor se a simulação do mesmo resultar em um déficit menor de energia num período crítico do que

outros planos que apenas elevem um pouco os volumes dos reservatórios em vários períodos. Dessa forma espera-se que os resultados dos planos encontrados tenham menos variações bruscas dos *fits*, e consequentemente dos *fits* reais.

Para melhor explicar a atuação desses pesos e penalidades nos resultados da função *fitness*, consideremos os *fits* reais de dois planos distintos descritos na tabela 4.1. Vale lembrar que quanto menor foi o *fit* real, mais perto do objetivo a variável está naquele período. Já na tabela 4.2 estão os *fits* desses mesmos planos, com os *fits* penalizados em destaque. No plano A, os *fits* reais de volume são significativamente menores do que os do plano B, mas o pico nos valores dos *fits* reais de energia é menos acentuado no plano B. Ainda assim, o plano A é considerado melhor do que o B, se olharmos apenas para os *fits* reais. Porém, ao aplicarmos os pesos e penalidades a esses *fits* reais, os valores de energia passam a ter uma relevância muito maior, e com isso o plano B passa a ser melhor do que o A.

Plano	Variável	Fits reais						Somatório	Fitness real
A	Energia	0.000	0.100	0.170	0.350	0.250	0.050	0.920	2.070
	Volume	0.100	0.250	0.350	0.200	0.100	0.150	1.150	
B	Energia	0.050	0.070	0.100	0.200	0.120	0.050	0.590	2.670
	Volume	0.300	0.450	0.480	0.350	0.250	0.250	2.080	

Tabela 4.1: Séries de *fits* reais para dois planos fictícios.

Plano	Variável	Fits						Somatório	Fitness
A	Energia	0.000	0.070	**0.958**	**1.276**	**1.094**	0.035	3.433	3.778
	Volume	0.030	0.075	0.105	0.060	0.030	0.045	0.345	
B	Energia	0.035	0.049	0.070	**1.008**	**0.878**	0.035	2.075	2.699
	Volume	0.090	0.135	0.144	0.105	0.075	0.075	0.624	

Tabela 4.2: Séries de *fits* para os mesmos dois planos
Parâmetros aplicados no cálculo:
$percMinEnergia$ = 0.1 / $penalidadeEnergia$ = 2.0 / $pesoEnergia$ = 0.7
$percMinVolume$ = 0.5 / $penalidadeVolume$ = 2.0 / $pesoVolume$ = 0.3

O algoritmo 5 exibe os passos do cálculo da função *fitness*. Os cálculos dos *fits* e *fits* reais são feitos para todos os subsistemas e para todos os períodos, entretanto, apenas aqueles pertencentes aos subsistemas buscados ($subsistemasBuscados$) são utilizados no cálculo do *fitness*, descrito entre as linhas 32 e 36. Nas linhas 3 a 15 são calculados os *fits* e *fits* reais da variável volume e nas linhas 16 e 27 os da variável energia. Os cálculos são muito similares nas duas variáveis. O primeiro passo é obter o valor total e o objetivo da variável. Caso o objetivo seja nulo, como no caso da demanda de energia elétrica do subsistema Itaipu, por exemplo, são somados um valor fixo nas variáveis do cálculo para evitar divisões por zero. O próximo passo é calcular o *fit* real da variável, dividindo o valor total pelo objetivo e decrementando o valor absoluto do resultado de 1, de modo que o *fit* real represente o déficit absoluto da variável. Por fim, é calculado o *fit*, multiplicando o valor *fit* real pelo peso da variável ($pesoVolume$ e $pesoEnergia$)

e elevando o resultado à penalidade aplicada (*penalidadeVolume* e *penalidadeEnergia*) caso o valor do *fit* real seja maior que o mínimo estipulado (*percMinVolume* e *percMinEnergia*).

Algoritmo 5 Algoritmo da função *Fitness*

```
Require: Resultados da simulação
for all ss ← todosSubsistemas do
    for all p ← todosPeriodos do
        total = volume[ss, p] − getVolumeMinimo(ss)
        total = (total < 0.0) ? 0.0 : total
        objetivo = getVolumeMaximo(ss) − getVolumeMinimo(ss)
        if objetivo == 0.0 then
            total ← total + 1000.0
            objetivo ← 1000.0
        end if
        fitRealVolume[ss, p] ← 1.0 − (total/objetivo)
        if fitRealVolume[ss, p] > percMinVolume then
            fitVolume[ss, p] ← pesoVolume ∗ pow((fitRealVolume[ss, p] +
            1.0), penalidadeVolume)
        else
            fitVolume[ss, p] ← pesoVolume ∗ fitRealVolume[ss, p]
        end if
        total = totalEnergiaGerada[ss, p]
        objetivo = getDemanda[ss, p]
        if objetivo == 0.0 then
            total ← total + 1000.0
            objetivo ← 1000.0
        end if
        fitRealEnergia[ss, p] ← abs(1.0 − abs(total/objetivo))
        if fitRealEnergia[ss, p] > percMinEnergia then
            fitEnergia[ss, p] ← pesoEnergia ∗ pow((fitRealEnergia[ss, p] +
            1.0), penalidadeEnergia)
        else
            fitEnergia[ss, p] ← pesoEnergia ∗ fitRealEnergia[ss, p]
        end if
        fitReal[sistema] ← fitRealVolume[ss, p] + fitRealEnergia[ss, p]
        systemFit[p] ← fitVolume[ss, p] + fitEnergia[ss, p]
    end for
end for
for all ss ← subsistemasBuscados do
    for all p ← todosPeriodos do
        fitness ← fitVolume[ss, p] + fitEnergia[ss, p]
    end for
end for
```

4.5 Buscador

O buscador é o algoritmo responsável por encontrar um plano melhor (*melhorPlano*) tendo como ponto de partida um plano original (*planoOriginal*). A implementação feita neste

trabalho, descrita no algoritmo 6, se baseia nos conceitos do método de Monte Carlo, explicado na seção 3.2, e nos algoritmos já detalhados para resolver essa questão. Em cada iteração do mesmo o plano de entrada é modificado, simulado e avaliado, e ao fim de todas as iterações ($maxIteracoesMC$) o plano que resultar nos melhores resultados, ou seja, tiver o menor *fitness*, é selecionado. Para delimitar o escopo da busca foram usadas as duas variáveis que controlam quais e quantas usinas devem ser buscadas ($usinasBuscadas$ e $qtdUsinasBuscadas$) e outra para restringir o quão diferente do plano de entrada os novos planos poderão ser ($maxPorcentagem$). Esses parâmetros são utilizados na execução do algoritmo gerador de planos, detalhado na seção 4.2.

O algoritmo 6 exibe os passos executados durante a busca local de planos. O mesmo recebe como parâmetros um conjunto de usinas a serem buscadas, um plano, a quantidade de iterações que deve executar e os parâmetros $maxPorcentagem$ e $useCriticalPeriods$, referentes ao gerador de planos explicado na seção 4.2. Nas primeiras linhas do algoritmo é feito a inicialização da variável $idSubsistemaBuscado$, que recebe o identificador do subsistema que mais possui usinas no conjunto de usinas $usinasBuscadas$, caso esse conjunto não seja vazio, ou um valor nulo. A inicialização das demais variáveis utilizadas pelo algoritmo é feita nas linhas subsequentes, de 7 a 10. Na linha 11 é iniciado o laço principal, executado $maxIteracoesMC$ vezes. Em cada iteração são executados, nessa ordem, os algoritmos: selecionador (2), modificador (3), simulador (1), transmissor (4) e função *fitness* (5). Após a execução desses passos, na linha 17, é feita a seleção do melhor plano de acordo com os valores dos *fitness* do até então melhor plano e o último plano simulado. O algoritmo retorna o melhor plano encontrado após todas as iterações.

Para reduzir o tempo de execução do algoritmo, principalmente quando executado em processadores com múltiplos núcleos, o laço principal do algoritmo foi paralelizado utilizando a API OpenMP. Cada *thread* fica encarregada de executar uma parcela das iterações, selecionando o melhor plano encontrado pela mesma. Ao final do laço, é feita uma seleção do melhor plano geral, a qual só pode ser executada por uma *thread* por vez, para evitar conflitos.

Algoritmo 6 Buscador de planos

Require: $usinasBuscadas$, $plano$, $maxPorcentagem$, $useCriticalPeriods$, $maxIteracoesMC$

1: **if** $usinasBuscadas$ **not** $null$ **then**
2: $idSubsistemaBuscado \leftarrow getIdSubsistemaBuscado(usinasBuscadas)$
3: **else**
4: $usinasBuscadas \leftarrow todasUsinas$
5: $idSubsistemaBuscado \leftarrow null$
6: **end if**
7: $seed \leftarrow time(null)$
8: $melhorPlano \leftarrow plano$
9: $threadMelhorPlano \leftarrow plano$
10: $setOpcoesGerador(seed, plano, maxPorcentagem, useCriticalPeriods)$
11: **for all** $i \leftarrow maxIteracoesMC$ **do**
12: $subset \leftarrow selecionador(usinasBuscadas, qtdUsinasBuscadas, \&seed)$
13: $novoPlano \leftarrow modificador(subset, \&periodoInicial)$
14: $resultado \leftarrow simulador(novoPlano, usinasSimuladas, periodoInicial)$
15: $transmissor(resultado)$
16: $funcaoFitness(resultado)$
17: $threadMelhorPlano \leftarrow selecionaMelhorPlano(plan, threadMelhorPlano)$
18: **end for**
19: $melhorPlano \leftarrow selecionaMelhorPlano(threadBestGPlan, melhorPlano)$
20: **return** $melhorPlano$

4.6 Otimizador

Por fim, o algoritmo otimizador é o que mantêm o buscador local de planos ativo, utilizando o conceito de resfriamento da técnica de busca têmpera simulada. A cada iteração que não retornar um plano melhor, ou se a melhora do *fitness* for menor que $minDecrementoFitness$, a variável $maxPorcentagem$, que controla a variação máxima das ações, é decrementada. Quando a variável $maxPorcentagem$ atinge um limite mínimo ($minPorcentagem$), a mesma é reiniciada e a quantidade de usinas buscadas ($qtdUsinasBuscadas$) é reduzida. Cada um desses decrementos reduz o escopo da busca e aumenta a chance do buscador encontrar planos válidos e melhores.

Porém, essa redução do escopo também impacta diretamente no quão melhor os planos encontrados poderão ser, já que os mesmos serão cada vez mais similares ao plano original. Para que o algoritmo não fique indefinidamente encontrando planos infimamente melhores, foi estabelecido um limite máximo de iterações de busca ($maxIteracoesBusca$). Quando esse limite é atingido, ou quando tanto a porcentagem mínima de variação quanto a quantidade de usinas buscadas chegarem ao mínimo, a execução do programa é finalizada.

O algoritmo 7 detalha os passos executados pelo otimizador. Nas primeiras linhas é feito a simulação e a validação do plano inicial para que então, na linha 8, comece de fato a otimização. Nessa linha é iniciado o laço principal do algoritmo que percorre os valores do vetor das quantidades de usinas buscadas ($vetorQtdUsinasBuscadas$) e logo na linha seguinte, já dentro do laço, a variável $maxPorcentagem$ é iniciada com o valor do parâmetro $maxPorcentagemInicial$. É então iniciado, na linha 10, um segundo laço que termina quando o valor da variável $maxPorcentagem$ for menor do que o valor do parâmetro $minPorcentagem$ ou quando o algoritmo executar a quantidade máxima de iterações, determinada pelo parâmetro $maxIteracoesBusca$. Dentro desse laço, na linha 15, o algoritmo buscador (6) é executado e depois é feito o cálculo do decremento do *fitness* entre os dois último planos retornados pelo buscador. Na linha 17 é verificado se o decremento do *fitness* é menor que o valor do parâmetro $minDecrementoFitness$ e, caso seja, o valor da variável $maxPorcentagem$ de acordo com o parâmetro $decrementoPorcentagem$. O algoritmo retorna o último plano encontrado pelo buscador.

Algoritmo 7 Otimizador de planos

Require: $plano$, $maxPorcentagemInicial$, $minDecrementoFitness$, $decrementoPorcentagem$, $minPorcentagem$, $vetorQtdUsinasBuscadas$, $maxIteracoesBusca$

1: $resultado \leftarrow simulador(plano, todasUsinas, 0)$
2: $transmissor(resultado)$
3: $funcaoFitness(resultado)$
4: **if not** $valida(plano)$ **then**
5: **return** $erro$
6: **end if**
7: $iteracao \leftarrow 0$
8: **for all** $qtdUsinasBuscadas \leftarrow vetorQtdUsinasBuscadas$ **do**
9: $maxPorcentagem \leftarrow maxPorcentagemInicial$
10: **repeat**
11: **if** $iteracao > maxIteracoesBusca$ **then**
12: **return** $plano$
13: **end if**
14: $ultimoFitness \leftarrow plano.fitness$
15: $plano \leftarrow buscador(plano, qtdUsinasBuscadas, maxPorcentagem)$
16: $decrementoFitness \leftarrow ultimoFitness - plano.fitness$
17: **if** $decrementoFitness < minDecrementoFitness$ **then**
18: $maxPorcentagem \leftarrow maxPorcentagem * decrementoPorcentagem$
19: **end if**
20: $iteracao \leftarrow iteracao + 1$
21: **until** $maxPorcentagem > minPorcentagem$
22: **end for**
23: **return** $plano$

4.7 Considerações

Nesse capítulo foram apresentados os componentes da solução proposta neste trabalho para o problema de planejamento de despacho hidrelétrico: um simulador do despacho hidrelétrico, um gerador de planos composto por um selecionador e um modificador, um simulador da rede de transmissão de energia elétrica, uma função fitness, um buscador local baseado nos métodos de Monte Carlo e um otimizador dos parâmetros de busca inspirado no algoritmo têmpera simulado. O aspecto mais importante dessa proposta é a junção dos principais conceitos do algoritmo têmpera simulada e dos métodos de Monte Carlo na elaboração do buscador local e do otimizador, que juntos formam o modelo de otimização implementado neste trabalho.

No capítulo 5 são apresentados os resultados do simulador de despacho hidrelétrico, afim de constatar a fidelidade dos mesmos com o que foi explicado no capítulo 2. No capítulo 5 também são demonstrados e discutidos os resultados obtidos com a execução do modelo de otimização proposto. Com o intuito de comprovar a eficácia e a eficiência do modelo foram realizados testes com diferentes conjuntos de parâmetros e diversos planos iniciais.

Capítulo 5

Resultados

Neste capítulo são apresentados e discutidos os resultados no modelo proposto no capítulo 4. Todos os algoritmos foram implementados na linguagem $C++$ e os testes foram executados em um nodo de um servidor com 4 *sockets* Intel Xeon E5-4627 v2 @ 3.30GHz com 8 núcleos por socket, 256GB de memória RAM e sistema operacional Debian 8.8. A avaliação dos resultados foi dividida em duas partes, a primeira para garantir a correção das simulações (seção 5.1) e a outra para analisar a eficácia do modelo de otimização (seção 5.2).

5.1 Resultados da simulação

Para avaliar a correção da simulação é preciso verificar principalmente se a simulação do funcionamento do sistema hidrelétrico corresponde com o que foi descrito nas seções 2.1 e 2.2, e se todas as restrições operativas estipuladas na seção 2.3 estão sendo cumpridas. Para isso usaremos como exemplo as usinas Barra Grande, Campos Novos, Machadinho e Itá. Essas usinas foram escolhidas por pertencerem ao subsistema Sul, o mesmo abordado na pesquisa de [Andriolo, 2014], e por representarem características importantes do sistema hidrelétrico, como a junção de rios e os dois tipos principais de usinas: usinas com reservatório e usinas fio d'água. A figura 5.1 exibe a disposição física dessas usinas, na tabela 5.1 estão as principais informações das mesmas e na tabela 5.2 estão os valores das ações e das vazões no período 41 que são utilizados para explicar os gráficos da figura 5.2, que exibem as vazões e volumes resultantes da execução das ações operacionais do plano inicial para essas usinas. Assim como nos exemplos apresentados na seção 2.1.2, foram usados os valores convertidos para hm^3.

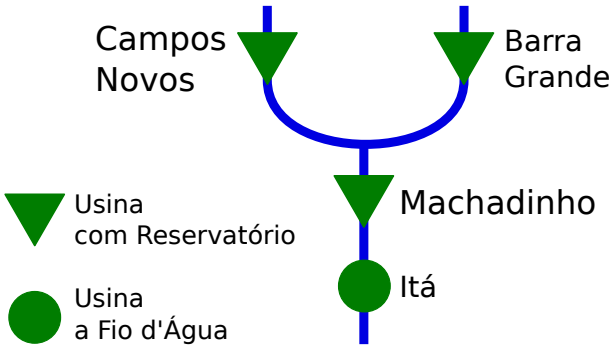


Figura 5.1: Usinas de exemplo.

Usina	**min_Qvt** m^3/s	**max_Qc** m^3/s	**min_Vol** hm^3	**max_Vol** hm^3
Barra Grande	516,00	516,00	2711,80	4904,40
Campos Novos	558,00	558,00	1320,00	1477,00
Machadinho	1311,00	1311,00	2283,00	3340,00
Itá	1590,00	1590,00	5100,00	5100,00

Tabela 5.1: Restrições das usinas Barra Grande, Campos Novos, Machadinho e Itá

Usina	**Qvt** hm^3	**Qc** hm^3	**Vz** hm^3	**Vol** hm^3
Barra Grande	773,20	1380,28	3193,14	3863.99
Campos Novos	3773,82	1492,54	5304,31	1320,02
Machadinho	7187,68	3421,44	12119,80	2831,79
Itá	12643,6	4014,99	18169,30	5100,00

Tabela 5.2: Dados das vazões e volume dos reservatórios das usinas Barra Grande, Campos Novos, Machadinho e Itá no período 41

Nos gráficos da figura 5.2 é possível visualizar algumas informações bastante interessantes. Uma delas é o efetivo cumprimento das restrições operativas. No gráfico 5.2(b) o volume do reservatório da usina Barra Grande atinge o mínimo no período 35 e máximo no 41, e para isso a usina reduz ao máximo a vazão vertida e vazão turbinada afim de economizar água e logo após tem um pico de vertimento para não exceder o volume, como demonstrado no gráfico 5.2(a). Já na usina Itá, por não possuir um reservatório, é necessário turbinar toda a vazão disponível. Porém, em alguns períodos a vazão turbinada alcança o limite máximo, obrigando o vertimento do restante da vazão, como é possível ver nos períodos 7 e 41 do gráfico 5.2(g). Essas mesmas situações também ocorrem nas outras usinas.

Outro ponto interessante é perceber como as séries de vazões funcionam num cenário real como exemplificado nas seções 2.1.1 e 2.1.2. Uma boa amostra é o pico de vazão que acontece em todas as usina no período 41. Usando a usina Machadinho como exemplo, a vazão incremental natural da mesma nesse período, de acordo com as séries sintéticas de

vazões usadas, é de 12119,80 hm^3, no entanto a vazão total foi de 11042,20 hm^3, pois 1039,66 hm^3 ($(773,20 + 1380,28) - 3193,14$) foram represados pela usina Barra Grande, e 37,95 hm^3 ($(3773,82 + 1492,54) - 5304,31$) foram represados pela usina Campos Novos.

5.1.1 Reflexo das alterações das ações operacionais nos resultados da simulação

Na seção anterior (5.1) foram demonstrados os resultados da simulação do plano inicial e agora é discutido como as alterações nesse plano se refletem nesses resultados. As mesmas usinas dos exemplos anteriores são usadas. Entretanto, dessa vez os gráficos da figura 5.3 apresentam as diferenças entre o plano inicial e o plano final, o melhor encontrado durante a execução do programa utilizando apenas as usinas do subsistema Sul. Vale ressaltar que os resultados da simulação do plano inicial não variam de entre as execuções pois o plano é o mesmo.

Começando pelas usinas mais à montante (Barra Grande e Campos Novos), o primeiro fato que vale ser notado nos gráficos 5.3(a) e 5.3(c) é que não há diferenças nas somas das vazões dessas usinas, novamente por serem as usinas mais à montante nos rios onde estão instaladas. Por outro lado, as alterações das ações operacionais dessas usinas repercutem diretamente na usina Machadinho, a próxima no fluxo do rio, como exibido no gráfico 5.3(e). Por exemplo, no período 17 as usinas Barra Grande e Campo Novos passaram a turbinaram mais água, o que possibilitou a usina Machadinho a também turbinar muito mais água sem sofrer uma grande redução do volume do reservatório.

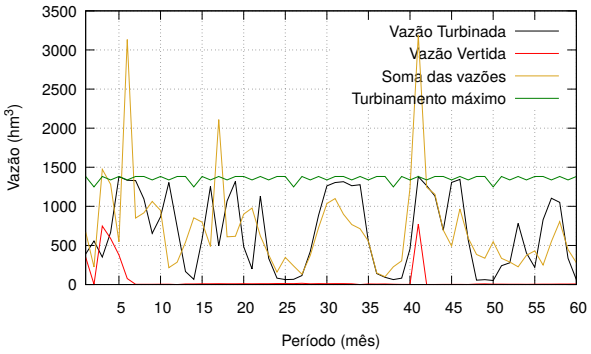
Em relação aos reflexos dessas ações operacionais nos volumes dos reservatórios, chama a atenção a elevação do nível das usinas que possuem reservatório em grande parte dos períodos, como é mostrado nos gráficos 5.3(b), 5.3(d), 5.3(f). Especialmente entre os períodos 5 e 15 era notório o baixo nível dos reservatórios nos resultados da simulação do plano inicial, o que foi corrigido reduzindo a quantidade de água turbinada nos períodos precedentes.

Sobre a usina Itá, um ponto relevante de apontar nos gráficos 5.3(g) e 5.3(h) são as alterações nas ações operacionais necessárias para turbinar toda a vazão disponível. Porém, em alguns períodos a usina foi obrigada a também verter mais água pois a vazão turbinada já atingiu o limite, desperdiçando água, uma vez que não há nenhuma outra usina à jusante. Isso demonstra que o plano final não é perfeito, ainda que tenha melhorado em relação ao plano inicial.

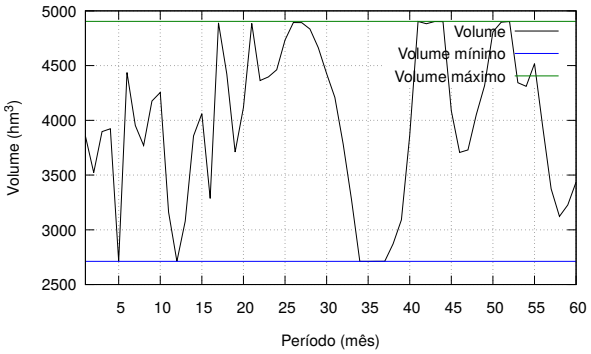
5.1.2 Análise da geração de energia elétrica

Depois da dicussão das séries de volume e vazões resultantes da simulação dos planos inicial e final, resta a análise da quantidade de energia gerada. Novamente são usadas como exemplos as mesmas usinas e resultados das seções anteriores.

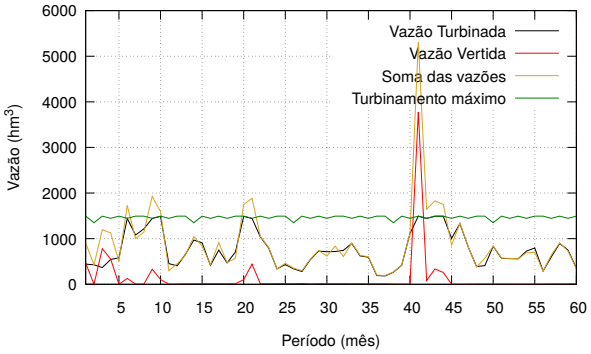
A quantidade de energia gerada por uma usina é diretamente proporcional ao volume do reservatório, vazão turbinada e potência da mesma. Porém, é a variação da vazão turbinada que



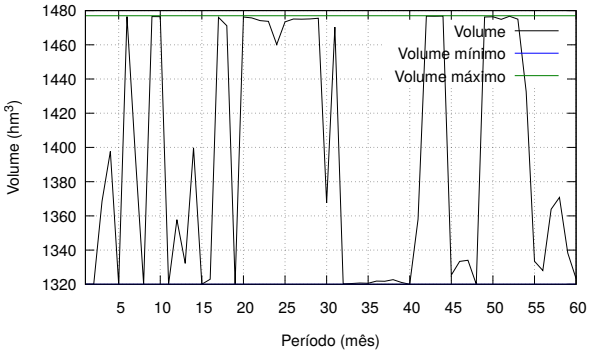
(a) Vazão - Barra Grande



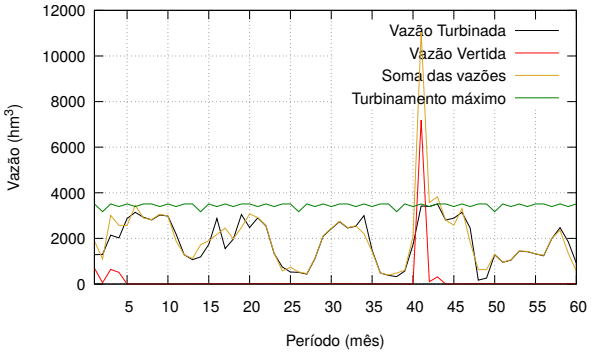
(b) Volume - Barra Grande



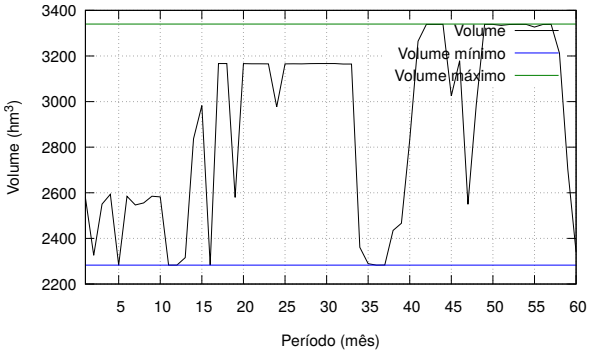
(c) Vazão - Campos Novos



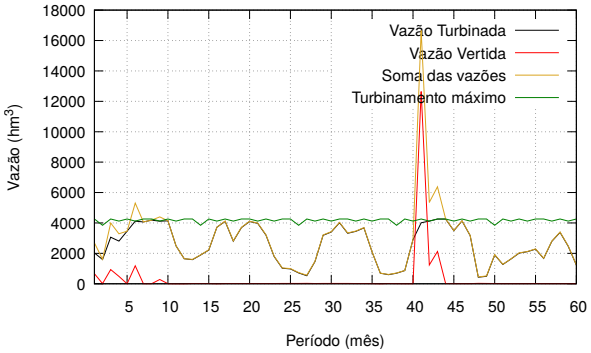
(d) Volume - Campos Novos



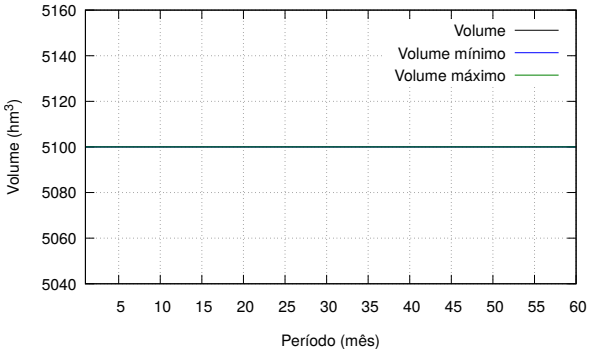
(e) Vazão - Machadinho



(f) Volume - Machadinho

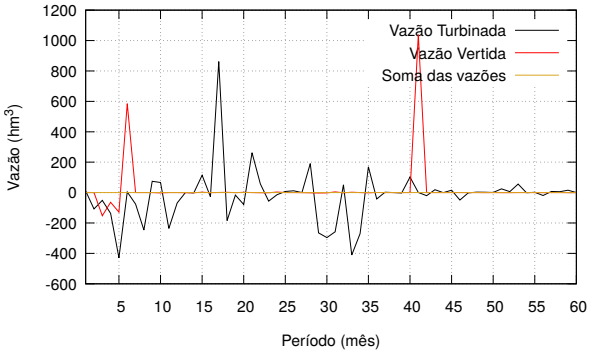


(g) Vazão - Itá

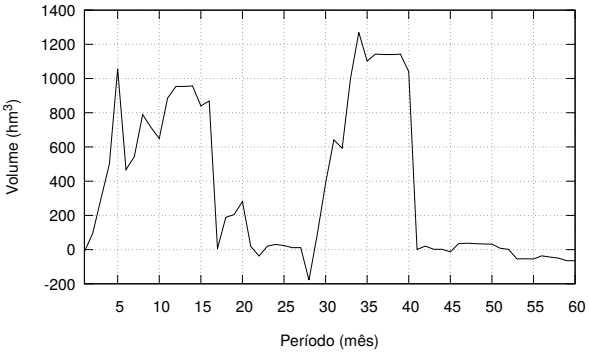


(h) Volume - Itá

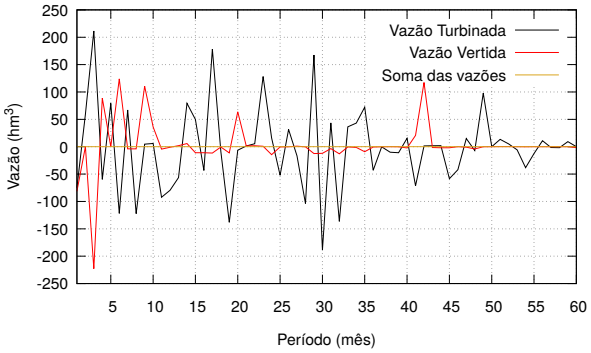
Figura 5.2: Vazões e Volumes das usinas Barra Grande, Campos Novos, Machadinho e Itá resultantes da simulação do plano inicial.



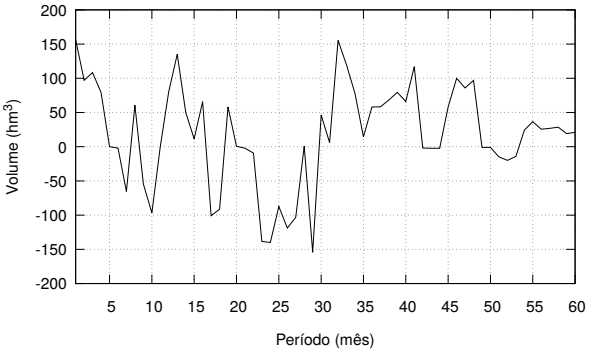
(a) Diferença - Vazão - Barra Grande



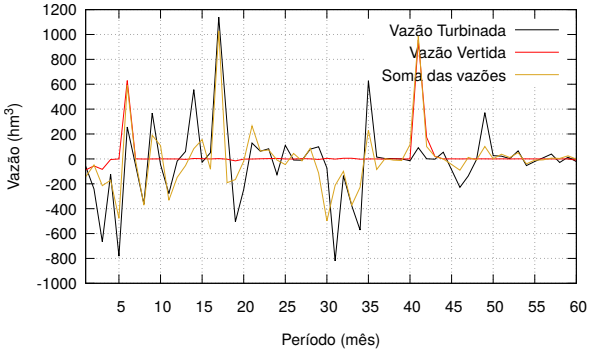
(b) Diferença - Volume - Barra Grande



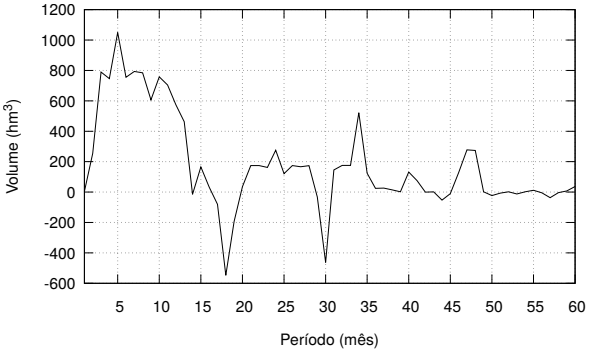
(c) Diferença - Vazão - Campos Novos



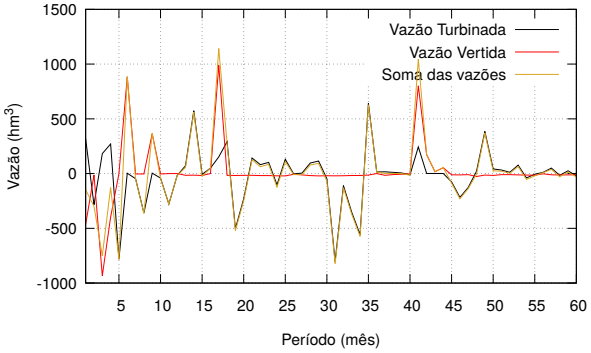
(d) Diferença - Volume - Campos Novos



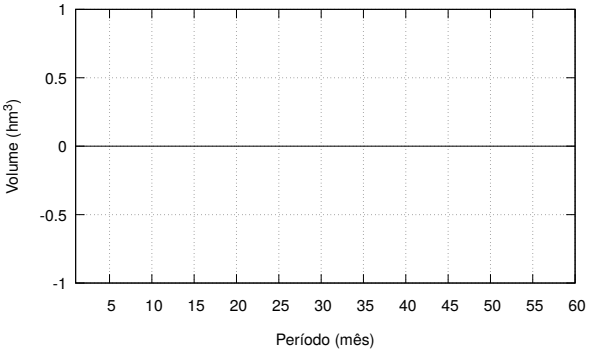
(e) Diferença - Vazão - Machadinho



(f) Diferença - Volume - Machadinho



(g) Diferença - Vazão - Itá



(h) Diferença - Volume - Itá

Figura 5.3: Vazões e Volumes das usinas exemplos resultantes da simulação do plano inicial.

mais impacta o resultado do cálculo. É possível perceber a semelhança das curvas dos gráficos 5.4(a), 5.4(c), 5.4(e) e 5.4(g) com as séries de vazão turbinada dos gráficos 5.2(a), 5.2(c), 5.2(e) e 5.2(g). Ainda assim, manter o volume dos reservatórios sempre elevados é crucial, pois caso contrário não há vazão suficiente para turbinar.

Da mesma forma, é possível ver uma grande semelhança com as diferenças entre os valores de energia gerada entre o plano inicial e final, exibidos nos gráficos 5.4(b), 5.4(d), 5.4(f) e 5.4(h), com as diferenças das séries de vazão turbinada dos gráficos 5.3(a), 5.3(c), 5.3(e) e 5.3(g).

5.2 Resultados do modelo de otimização

Saindo dos resultados das simulações e entrando no âmbito da otimização do plano, o primeiro tópico que precisa ser abordado, antes de discutir os resultados em si, são os parâmetros do algoritmo otimizador e, principalmente, os parâmetros do algoritmo de busca, por ditarem o comportamento da função *fitness*. Para avaliar como esses parâmetros influenciam nos resultados foram elaborados alguns conjuntos de parâmetros, que são apresentados na subseção 5.2.1. Nas demais subseções é discutido o comportamento da função *fitness* com cada um desses conjuntos e apresentado os resultados obtidos.

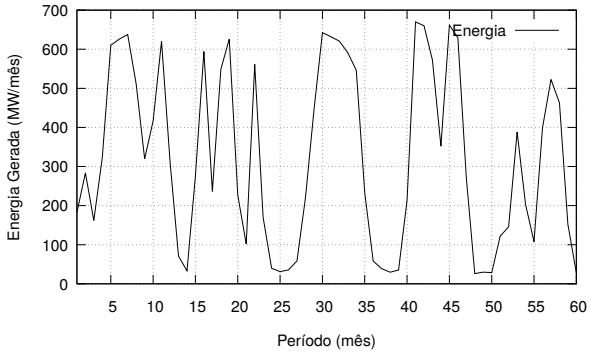
5.2.1 Método de avaliação

Uma vez que os parâmetros do otimizador não interferem diretamente nos resultados da busca, e apenas regem até que ponto o algoritmo de busca vai ser executado, em todos os testes foram utilizados os mesmos parâmetros, descritos na tabela 5.3. Os valores escolhidos foram aqueles que apresentaram os melhores resultados durante os testes preliminares.

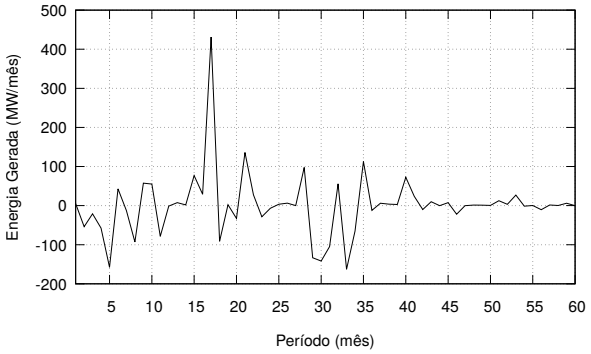
Parâmetro	Valor
maxIteracoesBusca	2500
maxIteracoesMC	256
maxPorcentagemInicial	0.275
decrementoPorcentagem	0.75
minPorcentagem	0.1
minDecrementoFitness	0.001
vetorQtdUsinasBuscadas	0 50 25 10
usePeriodoCritico	Verdadeiro

Tabela 5.3: Parâmetros de execução do algoritmo otimizador.

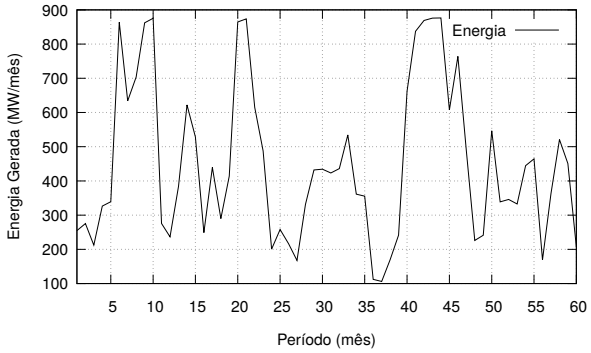
Os conjuntos de parâmetros do algoritmo de busca foram pensados para testar como os parâmetros de pesos e penalidades de cada variável avaliada pela função de *fitness* afetam os resultados das buscas, como explicado na seção 4.4. Com isso em mente, os conjuntos foram elaborados a partir de combinações das seguintes premissas: Com Penalidades (Cp) e Sem penalidades (Sp); Mínimos iguais (Mi) e Mínimos diferentes (Md); Pesos iguais (Pi) e Pesos



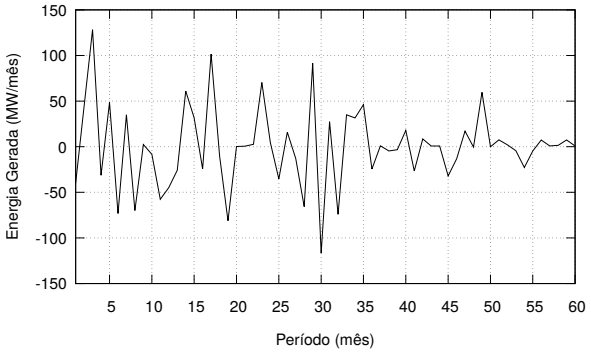
(a) Energia - Barra Grande



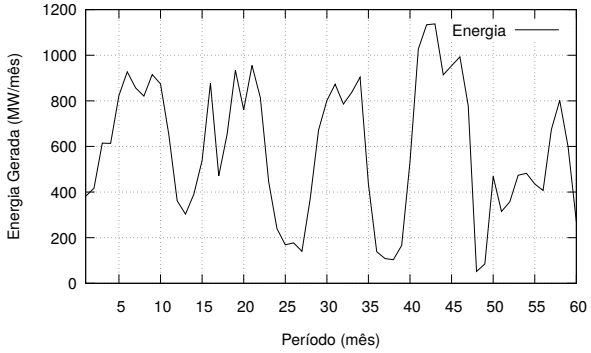
(b) Diferença - Energia - Barra Grande



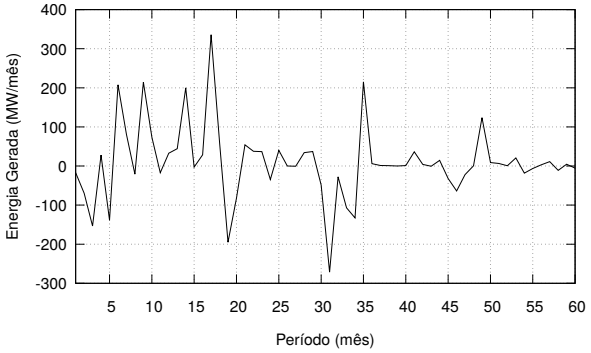
(c) Energia - Campos Novos



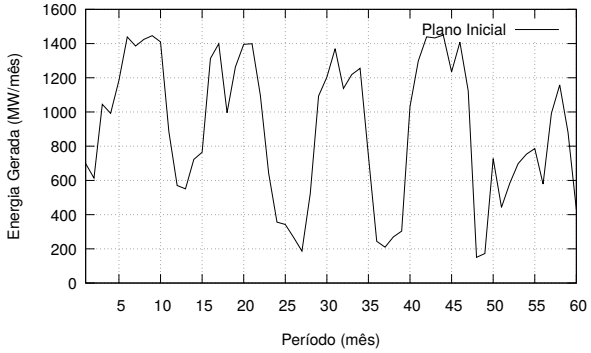
(d) Diferença - Energia - Campos Novos



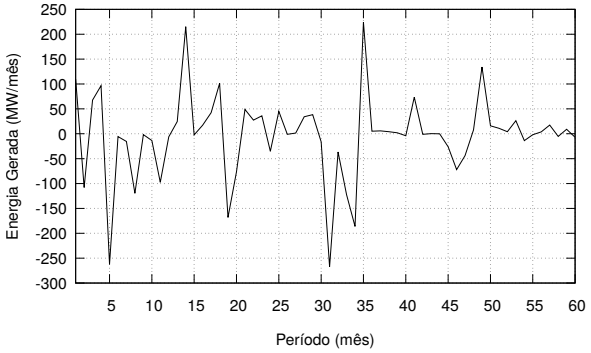
(e) Energia - Machadinho



(f) Diferença - Energia - Machadinho



(g) Energia - Itá



(h) Diferença - Energia - Itá

Figura 5.4: Energia gerada pelas usinas exemplos resultante da simulação do plano inicial.

diferentes (Pd). Os conjuntos resultantes dessas combinações e os valores escolhidos para cada variável estão descritos na tabela 5.4. No conjunto em que a premissa de mínimos diferentes foi aplicada, foi estipulado um mínimo menor para variável de energia de forma a restringir ainda mais o déficit de energia. Já no caso em que os pesos aplicados foram diferentes a variável de energia também foi priorizada, recebendo um peso maior do que a variável de volume.

Para facilitar a interpretação dos resultados, o otimizador foi executado buscando apenas o subsistema Sul e as 21 usinas pertencentes ao mesmo, mas mantendo a transmissão de energia entre os subsistemas. Dessa forma não é necessário exibir os resultados de todos os subsistemas e os resultados da busca ficam mais aparentes, uma vez que com menos variáveis a busca se torna mais fácil e a diferença entre o plano inicial e final é maior. Na seção 5.2.4 são apresentados os testes da busca em todas as 111 usinas.

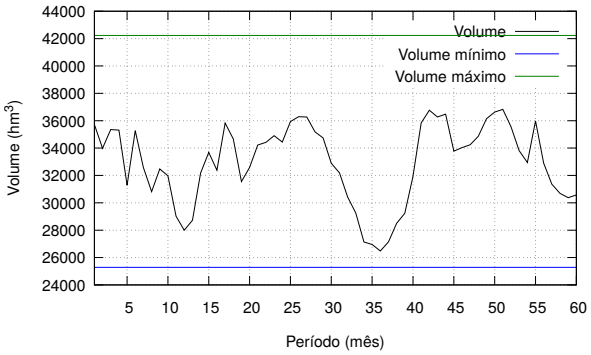
Parâmetro	**CpMdPd**	**CpMiPd**	**CpMdPi**	**CpMiPi**	**SpPd**	**SpPi**
$percMinVolume$	0.50	0.30	0.50	0.30	999.9	999.9
$penalidadeVolume$	2.0	2.0	2.0	2.0	1.0	1.0
$pesoVolume$	0.3	0.3	1.0	1.0	0.3	1.0
$percMinEnergia$	0.10	0.30	0.10	0.30	999.9	999.9
$penalidadeEnergia$	2.0	2.0	2.0	2.0	1.0	1.0
$pesoEnergia$	0.7	0.7	1.0	1.0	0.7	1.0

Tabela 5.4: Conjuntos de parâmetros de busca
CpMdPd = Com penalidades, Mínimos diferentes, Pesos diferentes;
CpMiPd = Com penalidades, Mínimos iguais, Pesos diferentes;
CpMdPi = Com penalidades, Mínimos diferentes, Pesos iguais;
CpMiPi = Com penalidades, Mínimos iguais, Pesos iguais;
SpPd = Sem penalidades, Pesos diferentes;
SpPi = Sem penalidades, Pesos iguais;

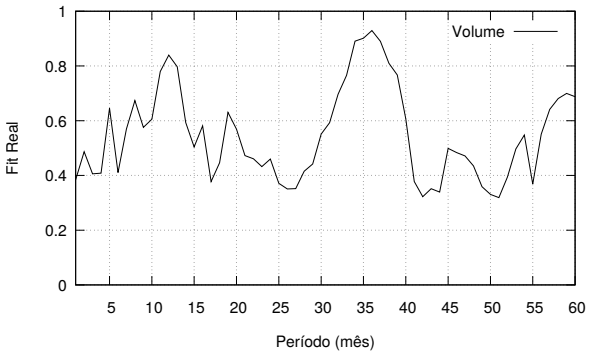
5.2.2 Comportamento da função *fitness* com os diferentes conjuntos de parâmetros

Para estudar o comportamento da função *fitness* é preciso, antes de mais nada, analisar os somatórios das séries de volume e energia das usinas que constituem o subsistema Sul, resultantes da simulação do plano inicial, expostos nos gráficos 5.5(a) e 5.5(c). É com base nesses valores, mais o limite máximo dos reservatórios e a série de demanda de energia elétrica, que os *fits* reais, exibidos nos gráficos 5.5(b) e 5.5(d), são calculados. Vale lembrar, como já mencionado na seção 4.4, que os *fits* reais são valores entre 0 e 1 que indicam o quão distantes do objetivo os valores das variáveis estão, sem penalidades ou pesos.

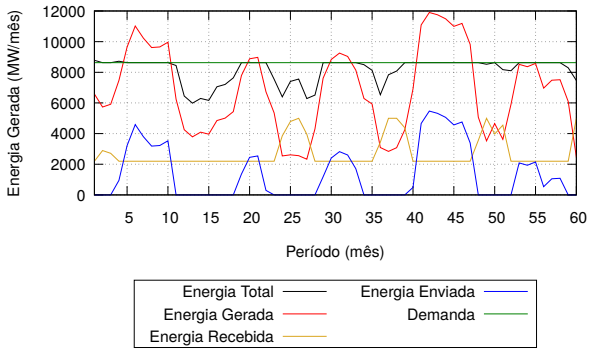
A série de *fits* reais da variável volume é muito semelhante a série dos somatórios dos volumes do subsistema Sul invertida, pois nada mais é que a diferença normalizada entre o limite máximo de volume e o valor do somatório. Por outro lado, a análise da série de *fits* reais da variável energia requer um pouco mais de interpretação, pois também depende



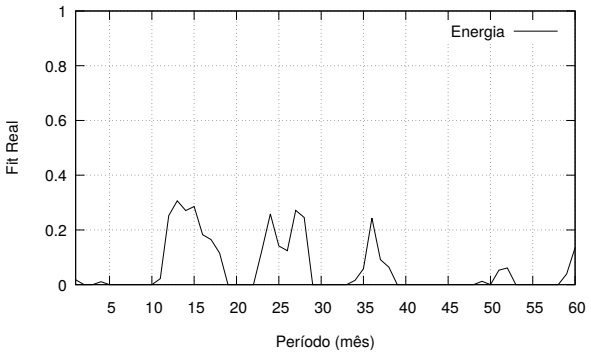
(a) Volume - Subsistema Sul



(b) RealFit Volume - Subsistema Sul



(c) Energia - Subsistema Sul



(d) RealFit Energia - Subsistema Sul

Figura 5.5: Volume e Energia do subsistema Sul resultantes da simulação do plano inicial.

do quanto de energia elétrica foi transmitida e recebida pelo subsistema em questão, como explicado na seção 2.6. No gráfico 5.5(c) é possível perceber que o subsistema recebe em média aproximadamente 2000 MW/mes e em alguns períodos chega a transmitir quase 6000 MW/mes. Nota-se também que nesses períodos a quantidade de energia elétrica gerada pelo subsistema Sul excede a demanda. Porém, esses excessos são compensados pela quantidade de energia elétrica transmitida, implicando no atendimento perfeito da demanda.

Por exemplo, a demanda de energia elétrica é satisfeita integralmente em diversos períodos e por isso os valores dos *fits* reais são nulos nesses períodos. Por outro lado, o somatório dos volumes dos reservatórios das usinas pertencentes ao subsistema Sul quase atingiu o limite mínimo no período 36, o que explica o *fit* real do volume tão alto nesse período.

É importante salientar que o cálculo dos *fits* reais não depende de nenhum outro parâmetro de execução, baseando-se unicamente nos resultados obtidos por cada subsistema. Por esse motivo os gráficos da figura 5.5 representam os resultados da simulação do plano inicial para todos os conjuntos de parâmetros já citados. Portanto, esses valores são usados como referenciais de comparação para os demais resultados das diversas execuções do programa.

No entanto, a busca utiliza o somatório dos *fits* reais com as penalidades e pesos aplicados, como explicado na seção 4.4. Como esses valores são diferentes em cada conjunto de parâmetros, os valores calculados em cada execução são diferentes mesmo para o plano inicial.

A figura 5.6 apresenta os gráficos dos *fits* de energia e volume para cada conjunto de parâmetro para a simulação do plano inicial.

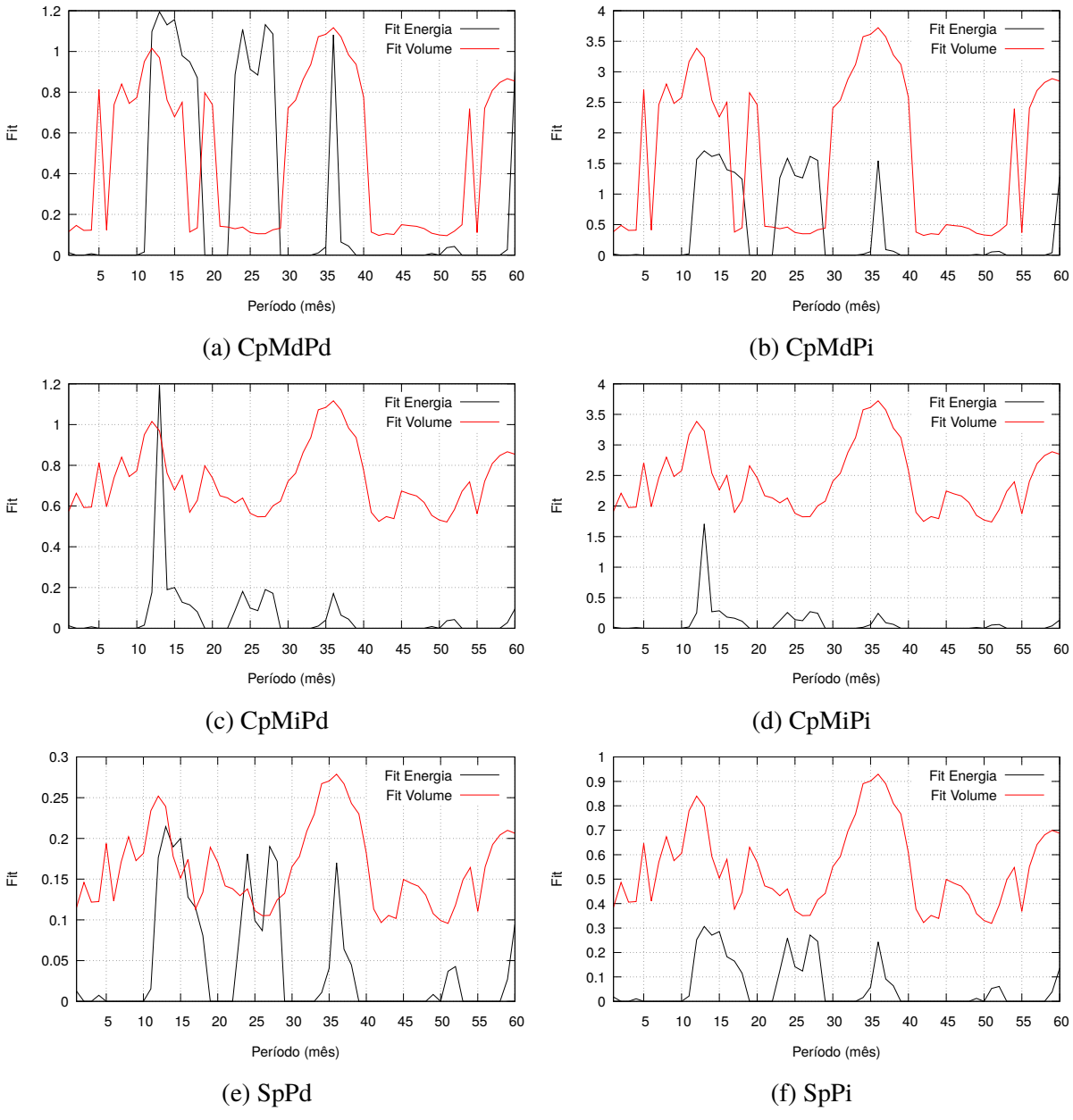
O detalhe mais notável nesses gráficos são as diferentes grandezas entre os valores dos *fits* de energia e volume. No gráfico 5.6(a), por exemplo, os *fits* de energia tem a mesma grandeza do que os de volume, mesmo sendo déficits menores. A explicação para isso é que aconteceram mais penalidades devido ao mínimo estipulado para os *fits* de energia ser menor do que aos dos conjuntos em que os mínimos são iguais ou nulos, além do peso da série de energia ser maior do que a de volume. Já no 5.6(b), a série de *fits* de volume tem uma relevância mais acentuada, pois os pesos aplicados nas duas séries são iguais. Nos gráficos 5.6(c) e 5.6(d) essas diferenças são ainda mais evidentes, com a exceção do período 13 no qual o *fit* real de energia foi penalizado, o que associado ao peso maior do que o dos *fits* de volume fez com que esse valor se sobressaia aos demais no gráfico 5.6(c).

A consequência esperada disso na busca é que, uma pequena melhora nos resultados nos períodos em que foram aplicadas penalidades ao *fit* real, ou em que o peso da variável seja maior, será muito mais significativa do que nos outros períodos. Em outras palavras, na execução do programa com o conjunto de parâmetros CpMiPd (com penalidades, mínimos iguais e pesos diferentes), um plano provavelmente será considerado melhor se a simulação do mesmo resultar um déficit menor de energia no período 13, do que outros que apenas elevem um pouco os volumes dos reservatórios. Dessa forma os planos selecionados como melhores a cada iteração da busca tenderão ser aqueles que reduzam os *fits* mais penalizados e com pesos maiores.

Porém, isso não significa que o restante dos *fits* não são alterados também. Um dos motivos é porque o *fitness*, que é o valor usado para qualificar cada plano, é o somatório de todos os *fits* de todos os subsistemas. Sendo assim, um plano ainda pode ser considerado bom caso o incremento de alguns *fits* sejam compensados pela redução de outros, principalmente se os pesos forem iguais ou não houver penalidades. Essas situações podem ser notadas nos gráficos da figura 5.7. O gráfico 5.7(a), por exemplo, que representa os resultados do conjunto de parâmetros de busca CpMdPd (Com penalidades, Mínimos diferentes, Pesos diferentes), é o que apresenta a redução mais consistente dos *fits* reais de energia dentre todos os outros. Tal situação é previsível uma vez que são os resultados do conjunto de parâmetros que mais prioriza os *fits* de energia. Já nos outros gráficos, os *fits* reais de energia pioraram em relação aos do plano inicial em diversos períodos, em especial no período 40 do gráfico 5.7(c).

5.2.3 Evolução do valor da função *fitness* ao longo das buscas

Finalmente, após a análise dos *fits* e *fits* reais das variáveis buscadas, resta a apresentação da evolução da função *fitness* ao longo da busca. Os gráficos da figura 5.8 exibem a evolução da função *fitness* em cada uma das execuções, e os da figura 5.9 exibem o *fitness* real apenas para comparação, pois não são utilizados pelos algoritmos.



(a) CpMdPd

(b) CpMdPi

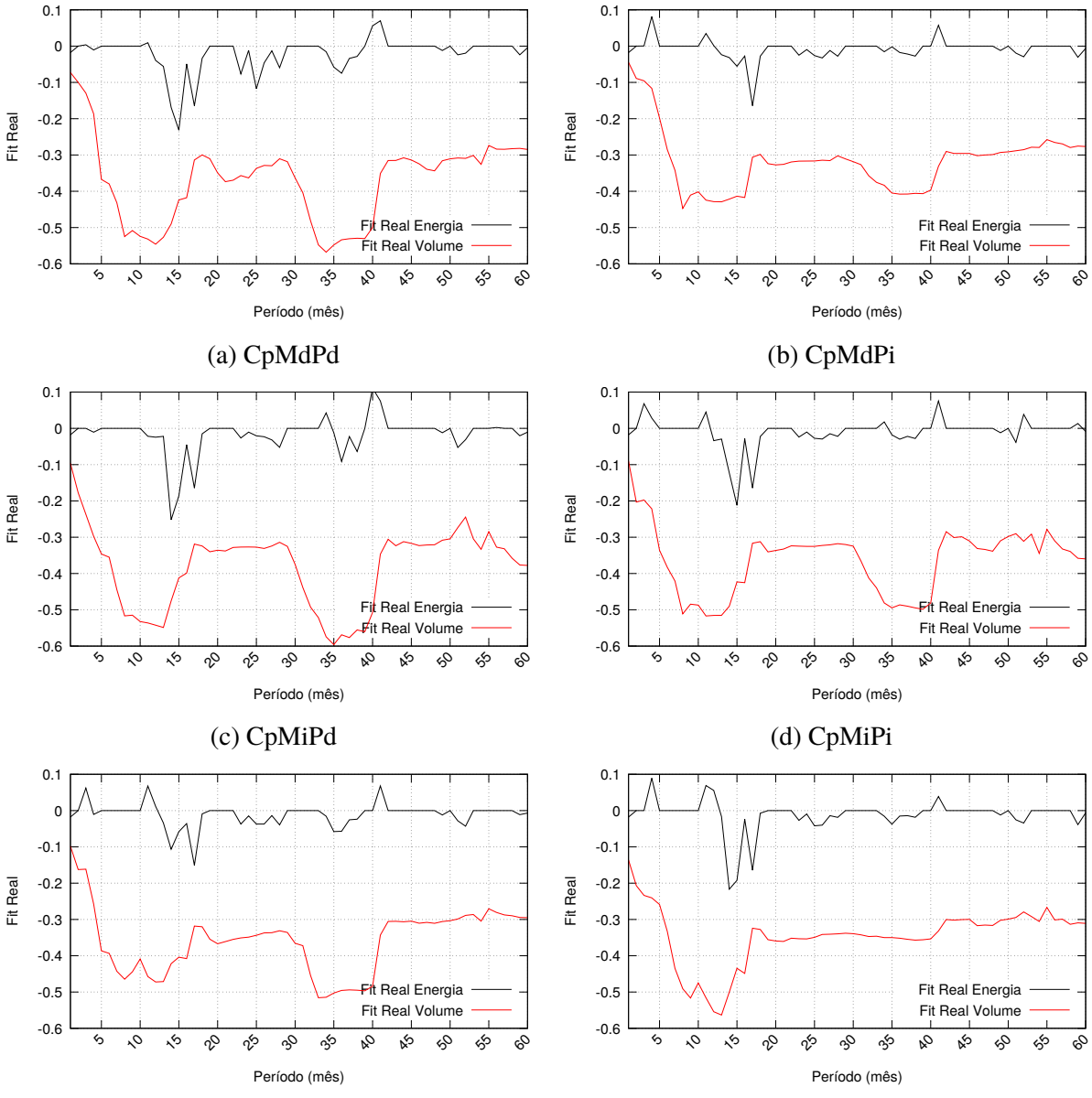
(c) CpMiPd

(d) CpMiPi

(e) SpPd

(f) SpPi

Figura 5.6: Fit de Volume e Energia do subsistema Sul resultantes da simulação do plano inicial.



(a) CpMdPd

(b) CpMdPi

(c) CpMiPd

(d) CpMiPi

(e) SpPd

(f) SpPi

Figura 5.7: Diferença dos *fits* reais de volume e energia do subsistema Sul entre a simulação do plano inicial e final.

Como esperado, em todos os casos houve uma redução significante do valor da função *fitness* nas primeiras iterações da busca e depois os resultados se estabilizam, até que as condições de parada são atingidas. Ainda assim, é interessante notar que, mesmo que as curvas dos gráficos dos *fitness* reais sejam semelhantes, o comportamento da função *fitness* varia consideravelmente de um execução para outra. Nos resultados da execução com o conjunto de parâmetros CpMdPd (Com penalidades, Mínimos diferentes, Pesos diferentes), por exemplo, a participação dos *fits* de energia passam a ser mais relevantes na composição do valor do *fitness* do que os *fits* de volume, diferentemente de todas as outras execuções. Tal comportamento corrobora os resultados que são discutidos na seção 5.2.4.

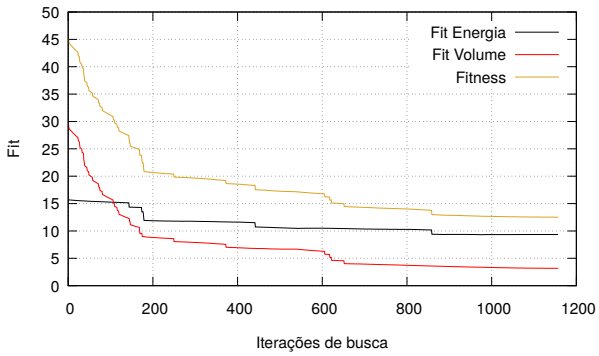
Com relação a quantidade de iterações de busca nas execuções com cada conjunto de parâmetro, não foi possível estabelecer um padrão para cada conjunto. Em nenhuma das execuções dos testes apresentados neste trabalho o limite de iterações foi atingido. Além disso, em todas as execuções o valor da função *fitness* começa a se estabilizar antes de aproximadamente 500 iterações. Os motivos da decisão de usar um limite de iterações grande foram: facilitar a análise dos resultados e evitar que alguma peculiaridade do programa fosse suprimida. Porém, esse limite pode ser muito menor caso se deseje otimizar a relação entre o tempo de execução e a qualidade do plano final.

5.2.4 Busca em todos os planos

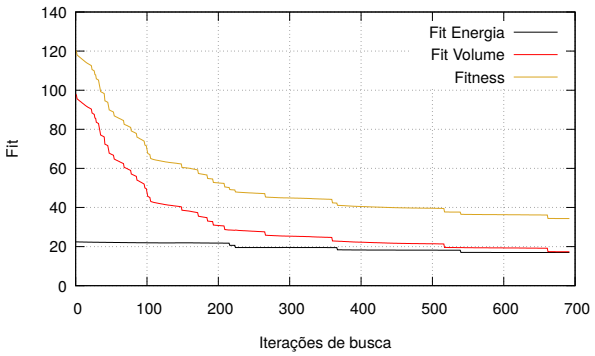
Para comprovar tanto a influência dos parâmetros de busca nos resultados quanto a eficácia da busca com diferentes planos iniciais, mais alguns testes foram feitos, utilizando o mesmo método de avaliação explicado na seção 5.2.1. Porém, dessa vez os testes foram executados para todos os planos iniciais disponíveis e, para estressar ao máximo o algoritmo de busca, também foram feitos os testes buscando em todas as usinas.

As figuras 5.10 e 5.11 mostram a melhora percentual dos somatórios dos *fits* reais de cada variável entre cada plano inicial e o melhor plano encontrado, avaliando apenas o subsistema Sul e todos os subsistemas, respectivamente. Os gráficos da figura 5.10 são bastante parecidos entre si e apresentam uma grande melhora em todas as execuções. O subsistema Sul tem muito déficit de volume dos reservatórios e pouco de energia elétrica, como mostrado na figura 5.5, então é esperado que os resultados de todas as buscas encontrem planos que elevem o nível dos reservatórios e mantenham o bom resultado elétrico. Além disso, como apenas o subsistema Sul e suas usinas foram considerados durante as buscas, o que reduz drasticamente a quantidade de variáveis buscadas e o valor dos *fits*, a melhora percentual não é tão significativa quanto pode parecer.

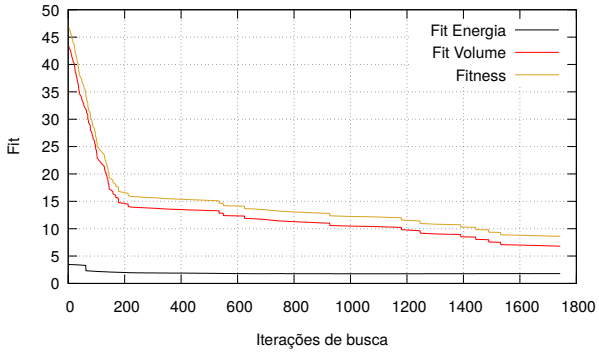
Por outro lado, os gráficos da figura 5.11 representam resultados muito mais interessantes. Ao considerar todas as usinas de todos os subsistemas o escopo da busca aumenta significativamente e, como cada subsistema tem características distintas, a margem para melhoras é maior mas também mais volátil. Em outras palavras, da mesma forma que um plano pode ser considerado



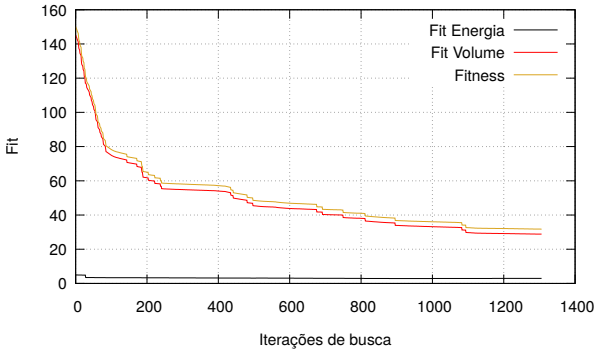
(a) CpMdPd



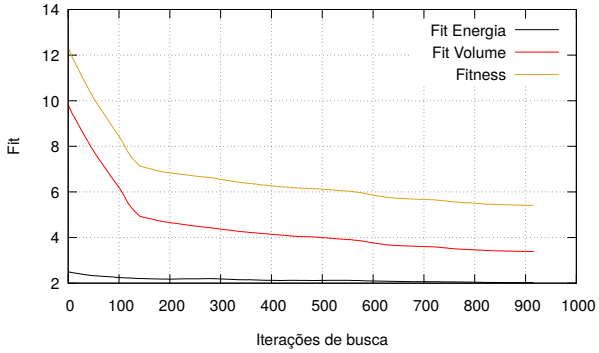
(b) CpMdPi



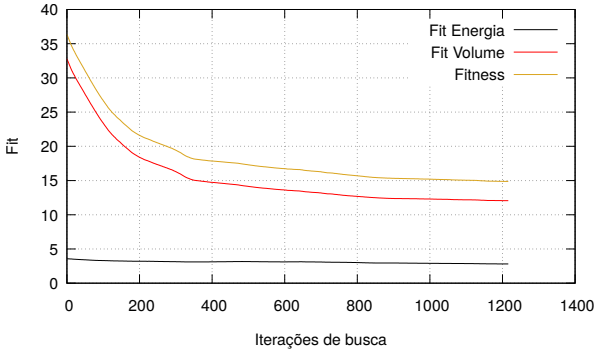
(c) CpMiPd



(d) CpMiPi

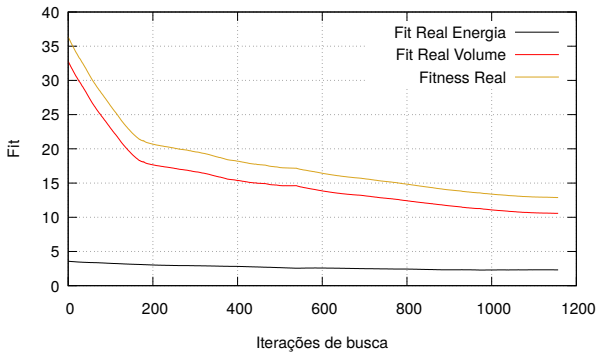


(e) SpPd

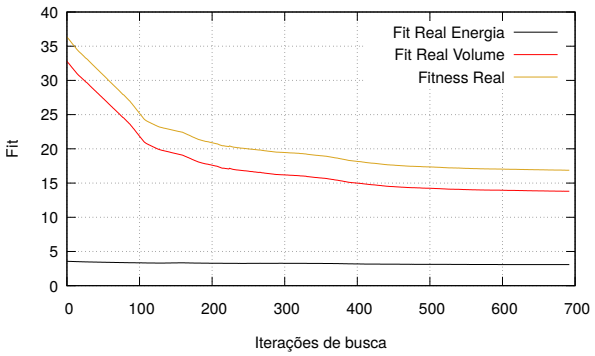


(f) SpPi

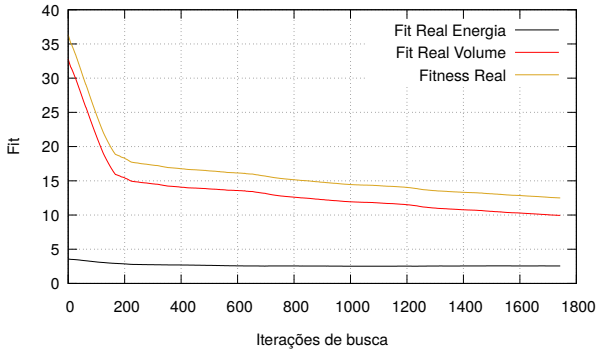
Figura 5.8: Evolução dos *fits*.



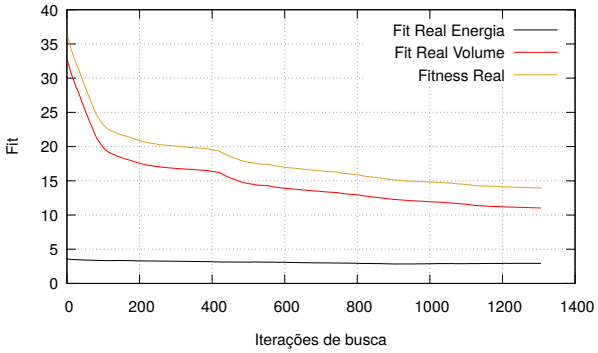
(a) CpMdPd



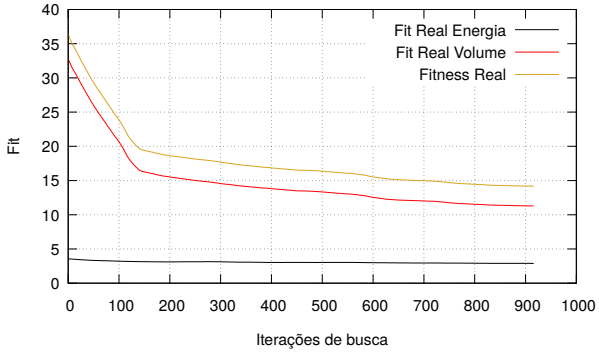
(b) CpMdPi



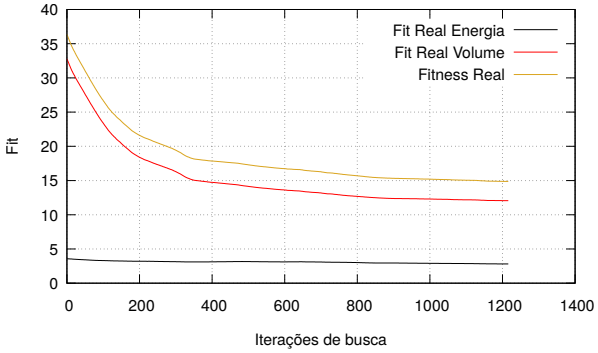
(c) CpMiPd



(d) CpMiPi



(e) SpPd



(f) SpPi

Figura 5.9: Evolução dos *fits* reais.

melhor caso os resultados ruins de uma usina sejam compensados por outros melhores de outra usina, os resultados ruins de um subsistema também podem ser compensados por outros melhores de outros subsistemas. Essa característica do algoritmo de busca implementado neste trabalho abre margem para resultados bastante diversificados quando o escopo da busca é muito amplo, o que pode ser restringido ou não dependendo dos parâmetros usados.

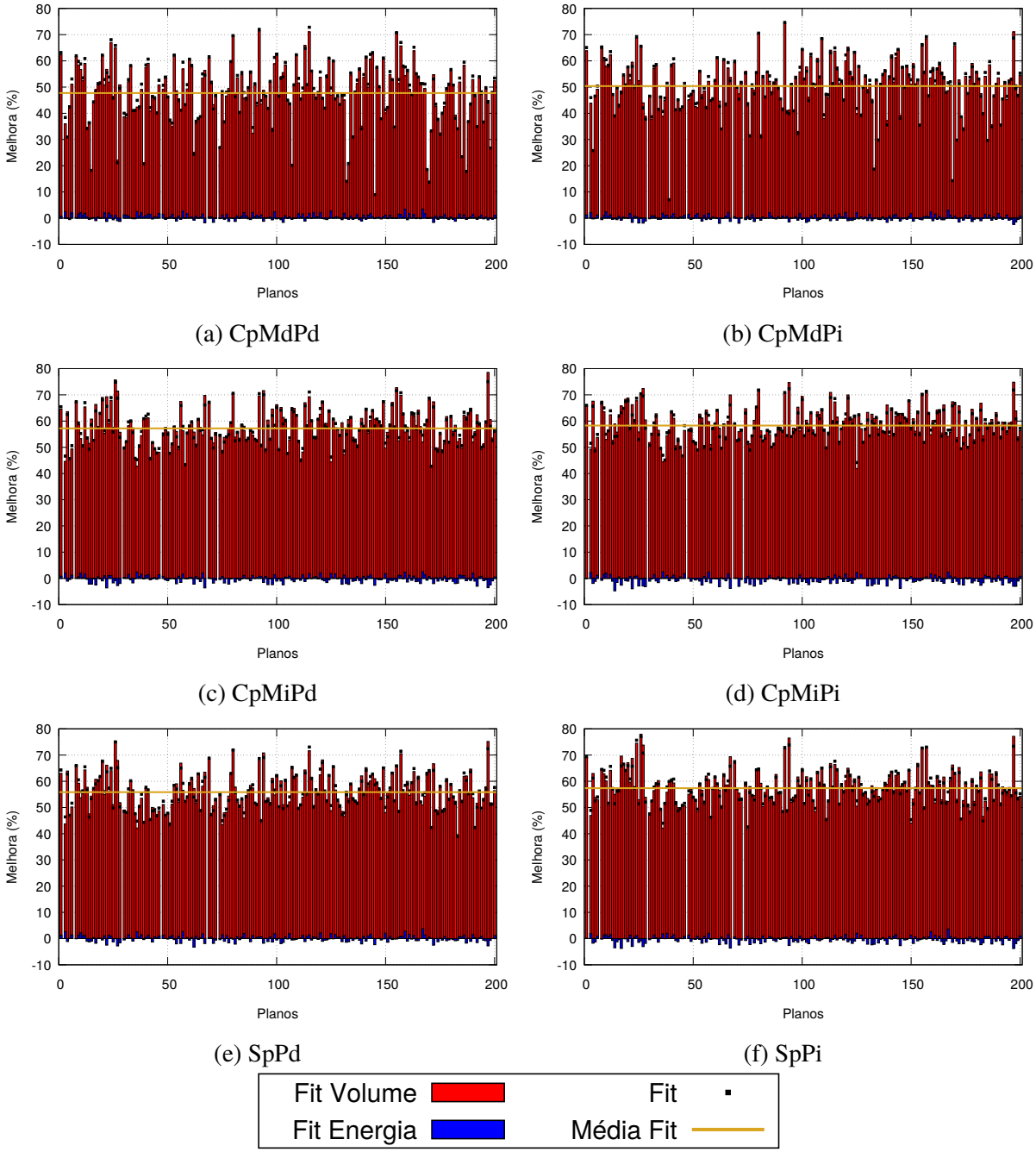
No gráfico 5.11(a), por exemplo, os *fits* reais de energia elétrica não variaram muito e os *fits* reais de volume tiveram uma pequena melhora, o que vai de encontro com as premissas do conjunto de parâmetros CpMdPd (Com penalidades, Mínimos diferentes, Pesos diferentes). Já no gráfico 5.11(f), que exibe os resultados das buscas com o conjunto de parâmetros SpPi (Sem penalidades, Pesos iguais), o percentual médio de melhora foi bastante significativo mas houve uma grande piora nos *fits* reais de energia, compensadas pelos *fits* reais de volume. Tal resultado é compreensível dado as características do conjunto de parâmetros SpPi (Sem penalidades, Pesos iguais), mas contradiz um dos principais objetivos do trabalho, que é aumentar a produção de energia hidrelétrica afim de reduzir a geração de outras fontes mais caras.

Já os resultados das buscas com os conjuntos de parâmetros CpMiPd (Com penalidades, Mínimos diferentes, Pesos iguais) e CpMiPi (Com penalidades, Mínimos iguais, Pesos iguais) exibidos respectivamente nos gráficos 5.11(c) e 5.11(d), são mais balanceados, com uma pequena piora nos *fits* reais de energia em prol de uma grande melhora nos *fits* reais de volume. Realisticamente os planos encontrados nessas buscas não são tão bons quanto os das buscas com o conjunto de parâmetros CpMdPd (Com penalidades, Mínimos diferentes, Pesos diferentes), mas podem servir como bons pontos de partida para outros algoritmos de buscas mais refinadas.

5.3 Tempo de execução

Por fim, para verificar o desempenho do programa ao ser executado em múltiplas *threads*, foram realizadas duas sequências de testes, uma buscando apenas as usinas do subsistema Sul e outra buscando em todas as usinas. O tempo médio de cada teste foi calculado com base nos tempos de execução de 10 iterações de busca seguidas, executadas após um pré-aquecimento com 5 iterações de busca. Os testes foram realizados no mesmo equipamento descrito no início deste capítulo. As tabelas 5.5 e 5.6 exibem os resultados da execução da busca apenas no subsistema Sul e da busca em todas as usinas, respectivamente.

Os resultados obtidos foram considerados bons, ainda mais quando considerados os tempos diminutos de cada iteração sequencial. Porém os mesmos ainda podem ser melhorados, principalmente na busca apenas no subsistema Sul. O *speedup* do programa ao ser executado em 32 *threads* foi de apenas 11 vezes, com uma eficiência de 35%. Uma provavél explicação para isso é que a concorrência pelo acesso a variável $melhorPlano$ é grande, devido ao tempo diminuto para gerar e simular cada plano, o que faz com que cada *thread* termine de processar a carga de trabalho muito mais rapidamente. Já nos testes das buscas em todas as usinas os resultados foram melhores, com uma eficiência de quase 60% na execução com 32 *threads*. Entretanto,

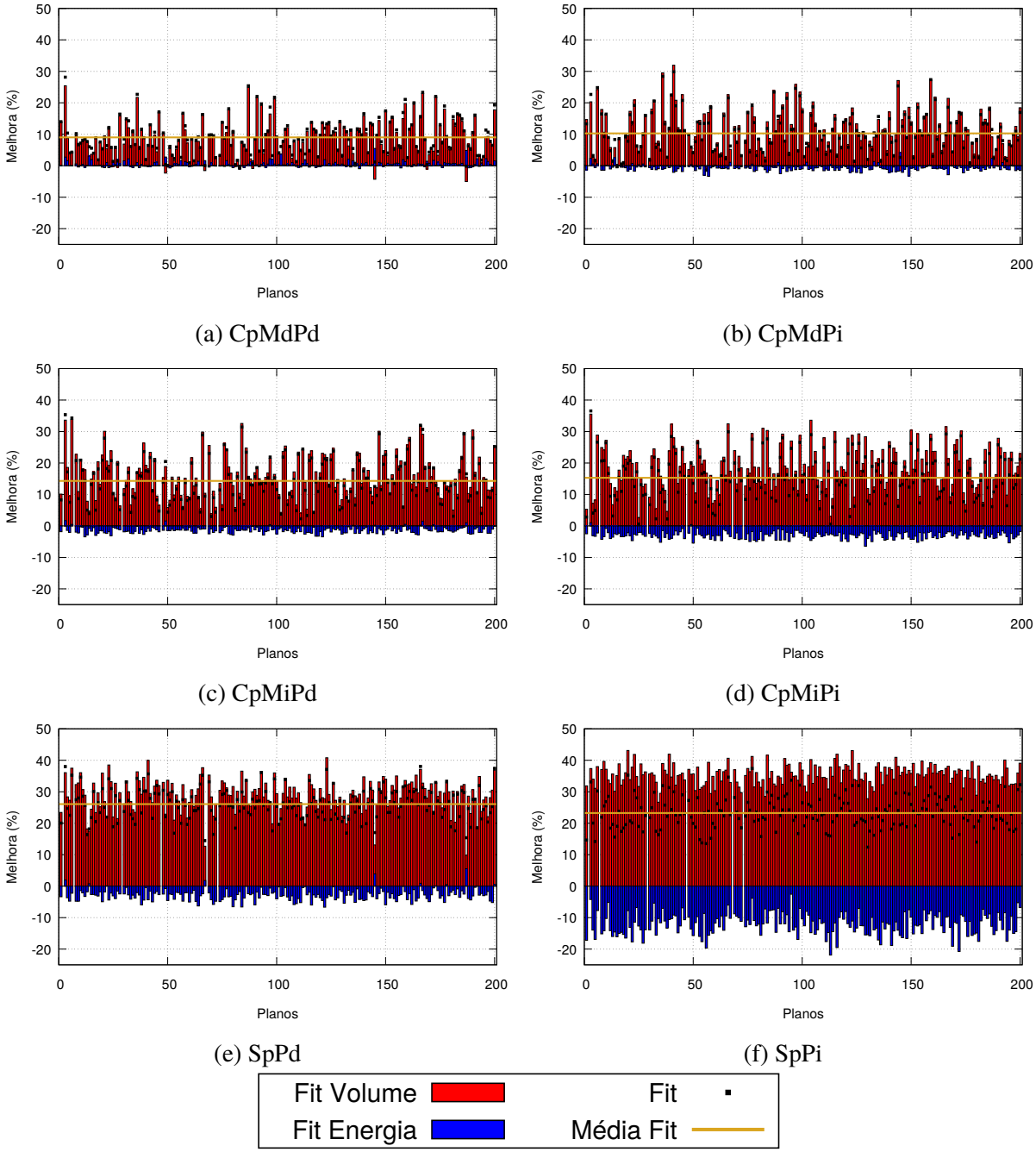


(a) CpMdPd (b) CpMdPi

(c) CpMiPd (d) CpMiPi

(e) SpPd (f) SpPi

Figura 5.10: Melhora percentual dos *fits* reais de volume e energia



(a) CpMdPd (b) CpMdPi

(c) CpMiPd (d) CpMiPi

(e) SpPd (f) SpPi

Figura 5.11: Melhora percentual dos *fits* reais de volume e energia

levando em conta que as *threads* não possuem dependências entre si, esse valor foi considerado baixo. Nesse caso, uma possível explicação é a ociosidade das *threads* que terminam o trabalho antes das outras, devido a necessidade de sincronização exigida pelo algoritmo otimizador.

Uma solução plausível para isso seria usar uma implementação baseada no modelo mestre/escravo, de forma a reduzir a competição das *threads* por um recurso e diminuir o impacto da sincronização, uma vez que o mestre, além de selecionar o melhor plano, também poderia ser responsável pela otimização da busca.

Threads	Tempo médio (s)	*Speedup*	Eficiência
001	0.29150	1.00000	1.00000
002	0.23083	1.26285	0.63142
004	0.13825	2.10857	0.52714
006	0.10056	2.89876	0.48313
008	0.07954	3.66486	0.45811
012	0.05618	5.18829	0.43236
016	0.04368	6.67321	0.41708
024	0.03186	9.15081	0.38128
032	0.02549	11.43402	0.35731

Tabela 5.5: Tempo médio de execução de uma iteração de busca apenas no subsistema Sul.
Tempo médio para gerar o plano: 0,000125s
Tempo médio para simular o plano: 0,001201s

Threads	Tempo médio (s)	*Speedup*	Eficiência
001	1.38986	1.00000	1.00000
002	0.85108	1.63306	0.81653
004	0.47338	2.93606	0.73402
006	0.33192	4.18729	0.69788
008	0.25870	5.37256	0.67157
012	0.18173	7.64780	0.63732
016	0.14283	9.73118	0.60820
024	0.10845	12.81560	0.53398
032	0.07345	18.92209	0.59132

Tabela 5.6: Tempo médio de execução de uma iteração de busca em todas as usinas.
Tempo médio para gerar o plano: 0.001866s
Tempo médio para simular o plano: 0.003610s

5.4 Considerações

Nesse capítulo foi demonstrado que o simulador funciona de acordo com o que foi explicado no capítulo 2 e que o buscador local de planos, mesmo sendo bastante simples, foi capaz de encontrar planos significativamente melhores em todos os testes realizados, com todos os conjuntos de parâmetros e com todos os planos iniciais, num curto período de tempo.

Foi discutido como os parâmetros de busca afetam os resultados, mostrando que a escolha de certos valores pode fazer com que o programa encontre planos considerados bons pela função *fitness* mas que realisticamente são ruins.

O desempenho da execução paralela do algoritmo de busca local foi considerado bom, principalmente pelo fato dos tempos das iterações sequenciais já serem bastante pequenos. Sendo assim, qualquer tempo gasto no controle das iterações tem um grande impacto na eficiência da execução paralela. Ainda assim, foi proposto a implementação baseada no modelo mestre/escravo que pode solucionar os problemas de concorrência por recursos e reduzir ainda mais o tempo de execução de cada iteração de busca.

Capítulo 6

Conclusão

Este trabalho apresentou uma implementação funcional de um simulador de despacho hidrelétrico capaz de calcular os volumes finais dos reservatórios das usinas para todos os períodos simulados, calcular a quantidade de energia elétrica gerada por cada usina em todos os períodos, simular a transmissão de energia entre os subsistemas e verificar o atendimento às restrições operativas.

Foi proposto um método de busca local, inspirado no método de Monte Carlo, capaz de encontrar planos melhores que o plano inicial num curto período de tempo, que utiliza o simulador para obter os resultados operacionais de cada plano gerado. Também foi proposto um método para otimizar os parâmetros de busca após as iterações que não retornam planos melhores, de modo a manter o algoritmo de busca local ativo por mais algumas iterações.

Apesar da simplicidade do método proposto, os resultados obtidos foram considerados muito bons. O método de otimização implementado foi capaz de encontrar planos significativamente melhores em todos os testes realizados, com todos os conjuntos de parâmetros e com todos os planos iniciais, num curto período de tempo.

O tempo médio de execução de uma iteração de busca, que gera e avalia 256 planos, em todas as 111 usinas utilizando 32 *threads* foi de aproximadamente $0.07s$, com uma eficiência de quase 60%, um resultado bastante satisfatório considerando que os tempos das iterações sequenciais já serem bastante pequenos. Ainda assim, foi proposto a implementação baseada no modelo mestre/escravo que pode solucionar os problemas de concorrência por recursos e reduzir ainda mais o tempo de execução de cada iteração de busca.

Outro quesito deste trabalho que pode ser aprimorado é o próprio modelo de otimização proposto. O modelo apresentado neste trabalho, elaborado a partir da junção dos principais conceitos do algoritmo têmpera simulada e dos métodos de Monte Carlo, se provou eficiente na tarefa de encontrar rapidamente um plano melhor, entretanto, pela grande quantidade de variáveis do problema e por ser um método puramente aleatório, é muito provável que alguns dos valores gerados sejam ruins.

Como trabalhos futuros sugere-se o uso de métodos mais elaborados de busca no intuito de encontrar planos ainda melhores. Além disso, o método de busca implementado neste trabalho

pode gerar uma grande quantidade de dados, pois gera e avalia milhares de planos distintos, que podem ser usados por algoritmos de reconhecimento de padrões com o propósito de detectar quais alterações são mais ou menos relevantes.

Referências Bibliográficas

[Andrieu et al., 2003] Andrieu, C., de Freitas, N., Doucet, A. e Jordan, M. I. (2003). An introduction to mcmc for machine learning. *Machine Learning*, 50(1):5–43.

[Andriolo, 2014] Andriolo, R. F. (2014). Acoplamento elétrico energético no planejamento da operação em médio prazo com restrição de transmissão. Dissertação de Mestrado.

[ANEEL, 2008] ANEEL (2008). *Atlas de Energia Elétrica do Brasil : arquivos de pesquisa e geoprocessamento*. ANEEL.

[Bessa et al., 2013] Bessa, M. R., Vallejos, C. A., Detzel, D. H., Mine, M., Marcílio, D. C., Oening, A. P., Matioli, L. C., Haas, P., Fernandes, T. S., Silva, F. et al. (2013). Otimização do despacho hidrotérmico mediante algoritmos híbridos com computação de alto desempenho: modelo phoenixvii. Em *Congresso de Inovação Tecnológica em Energia Elétrica CITENEL. Anais... Rio de Janeiro: Inovação Tecnológica em Energia Elétrica*.

[Bouzy e Cazenave, 2001] Bouzy, B. e Cazenave, T. (2001). Computer go: An ai oriented survey. *Artificial Intelligence*, 132(1):39 – 103.

[Detzel et al., 2014] Detzel, D. H. M., Mine, M. R. M., Bessa, M. R. e Bloot, M. (2014). Cenários sintéticos de vazões para grandes sistemas hídricos através de modelos contemporâneos e amostragem. *RBRH: revista brasileira de recursos hídricos*, 19(1):17–28.

[EPE, 2008] EPE (2008). Considerações sobre repotenciação e modernização de usinas hidrelétricas. Relatório técnico, Empresa de Pesquisa Energética (EPE).

[Hammersley, 2013] Hammersley, J. (2013). *Monte carlo methods*. Springer Science & Business Media.

[MME, 2016] MME (2016). Boletim de monitoramento do sistema elétrico - abril - 2016. `http://www.mme.gov.br/web/guest/secretarias/energia-eletrica/publicacoes/boletim-de-monitoramento-do-sistema-eletrico`. [Online; acessado em 23-Junho-2016].

[ONS, 2015] ONS (2015). Mapa de integração eletroenergética. `http://www.ons.org.br/conheca_sistema/mapas_sin.aspx`. [Online; acessado em 23-Junho-2016].

[Russell e Norvig, 1995] Russell, S. e Norvig, P. (1995). A modern approach. *Artificial Intelligence. Prentice-Hall, Egnlewood Cliffs*, 25:27.

[Silver et al., 2016] Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M. et al. (2016). Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484–489.

[Xavier et al., 2005] Xavier, L., Diniz, A., Costa, F. e Maceira, M. (2005). Aprimoramento da modelagem da função de produção energética das usinas hidroelétricas no modelo decomp: metodologia e resultados. *XVIII SNPTEEE, Curitiba*.