Universidade de Brasília – UnB
Faculdade UnB Gama – FGA
Engenharia de Software

Computação Bioinspirada para Recomendação de Contratos de Energia Elétrica

Autor: Daniel Porto de Souza
Orientador: Prof. Dr. Renato Coral Sampaio

Brasília, DF
2024

Daniel Porto de Souza

Computação Bioinspirada para Recomendação de Contratos de Energia Elétrica

Monografia submetida ao curso de graduação em Engenharia de Software da Universidade de Brasília, como requisito parcial para obtenção do Título de Bacharel em Engenharia de Software.

Universidade de Brasília – UnB

Faculdade UnB Gama – FGA

Orientador: Prof. Dr. Renato Coral Sampaio

Brasília, DF

2024

Daniel Porto de Souza
Computação Bioinspirada para Recomendação de Contratos de Energia Elétrica/ Daniel Porto de Souza. – Brasília, DF, 2024-
63 p. : il. (algumas color.) ; 30 cm.

Orientador: Prof. Dr. Renato Coral Sampaio

Trabalho de Conclusão de Curso – Universidade de Brasília – UnB
Faculdade UnB Gama – FGA , 2024.

1. Particle Swarm Optimization. 2. Busca Global. I. Prof. Dr. Renato Coral Sampaio. II. Universidade de Brasília. III. Faculdade UnB Gama. IV. Computação Bioinspirada para Recomendação de Contratos de Energia Elétrica

CDU 02:141:005.6

Daniel Porto de Souza

Computação Bioinspirada para Recomendação de Contratos de Energia Elétrica

Monografia submetida ao curso de graduação em Engenharia de Software da Universidade de Brasília, como requisito parcial para obtenção do Título de Bacharel em Engenharia de Software.

Trabalho aprovado. Brasília, DF, 12 de Setembro de 2024 – Data da aprovação do trabalho:

Prof. Dr. Renato Coral Sampaio
Orientador

Prof. Dr. Bruno Cesar Ribas
Convidado 1

Profa. Dra. Loana Nunes Velasco
Convidado 2

Brasília, DF
2024

Este trabalho é dedicado à memória de minha irmã.

Resumo

Os gastos com energia elétrica são despesas recorrentes. As grandes instituições despendem montantes milionários todos os anos com custos de energia em contratos que, muitas vezes, são desfavoráveis. O projeto MEPA tem um papel fundamental nesse contexto. Sendo oriundo da parceria entre o Ministério da Educação e a Universidade de Brasília, se trata de uma plataforma para recomendações de contratos de energia elétrica. O sistema propõe o auxílio na gerência das faturas de energia elétrica, sugerindo adequações nos contratos conforme o histórico de consumo da instituição. O presente trabalho tem o objetivo de avaliar a viabilidade e implementar a utilização de algoritmos de busca global bioinspirados para o aprimoramento do sistema de recomendações de contratos de energia implementado na plataforma. Foi realizada uma prova de conceito aplicando dois algoritmos bioinspirados em conjunto com o modelo estatístico utilizado atualmente para a comparação entre eles. Dessa forma, foi possível entender melhor o funcionamento e as limitações da abordagem atual, bem como a obtenção de resultados satisfatórios no que tange à viabilidade, tanto do algoritmo *particle swarm optimization* quanto do algoritmo genético. Sendo assim, foi feita a implementação da integração dos algoritmos bioinspirados à plataforma.

Palavras-chave: *Particle Swarm Optimization*. Algoritmo genético. Busca global.

Abstract

Energy expenses are recurring costs. Large institutions spend millions annually on energy in contracts that are often unfavorable. The MEPA project plays a fundamental role in this context. Originating from a partnership between the Ministry of Education and the University of Brasília, it's a platform for recommending electricity contracts. The system assists in managing electricity bills, suggesting contract adjustments based on the institution's consumption history. This work aims to evaluate the feasibility and implement the use of bio-inspired global search algorithms to enhance the energy contract recommendation system implemented on the platform. A proof of concept was carried out, applying two bio-inspired algorithms alongside the statistical model currently in use for comparison. This allowed for a better understanding of the current approach's operation and limitations, as well as obtaining satisfactory results regarding the feasibility of both the *particle swarm optimization* algorithm and the genetic algorithm. Consequently, the integration of bio-inspired algorithms into the platform was implemented.

Key-words: Particle Swarm Optimization. Genetic Algorithm. Global Search.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

kV	Quilovolt
kW	Quilowat
kWh	Quilowat-hora
CLI	*Command Line Interface*
UC	Unidade consumidora
PSO	*Particle Swarm Optimization*
UML	*Unified Model Language*
GA	*Genetic Algorithm*
MEPA	Monitoramento de Energia em Plataforma Aberta
TUSD	Tarifa de Utilização do Sistema de Distribuição
TE	Tarifa de Energia Elétrica

Sumário

1 Introdução

A eficiência no gasto público tem sido um ponto central nas discussões sobre a sustentabilidade das finanças e na promoção do desenvolvimento econômico. A qualidade desse tipo de gasto refere-se não apenas à quantidade de recursos alocados, mas, também, à eficácia com que são utilizados. Para (TRIDAPALLI; FERNANDES; MACHADO, 2011) os governos da atualidade estão sendo obrigados a fazer mais com menos já que a sociedade tem exigido mais transparência, justiça e equidade nos contratos públicos em um contexto onde o acesso à informação é crescente.

Além disso, o "fazer mais com menos" é de suma importância em cenários de crise, onde o repasse público é incerto e os recursos tendem a ficar cada vez mais limitados. Esse contexto pode ser observado nas instituições de ensino superior do Brasil, onde, nos últimos anos, acumulam-se repetidos cortes e contingenciamentos (BORGES, 2024).

Sendo assim, faz-se necessário a análise das potenciais melhorias existentes no que tange às áreas cujos gastos apresentam impactos consideráveis no orçamento das universidades. Gastos recorrentes como os de energia elétrica podem ser uma parcela significativa das despesas operacionais de muitas instituições.

Todos os anos, as universidades públicas brasileiras despendem de montantes milionários em contratos de energia elétrica. De acordo com dados da ANDIFES(Associação Nacional dos Dirigentes das Instituições Federais de Ensino Superior), somente no ano de 2023 esse valor foi de 516,5 milhões de reais (ANDIFES, 2024).

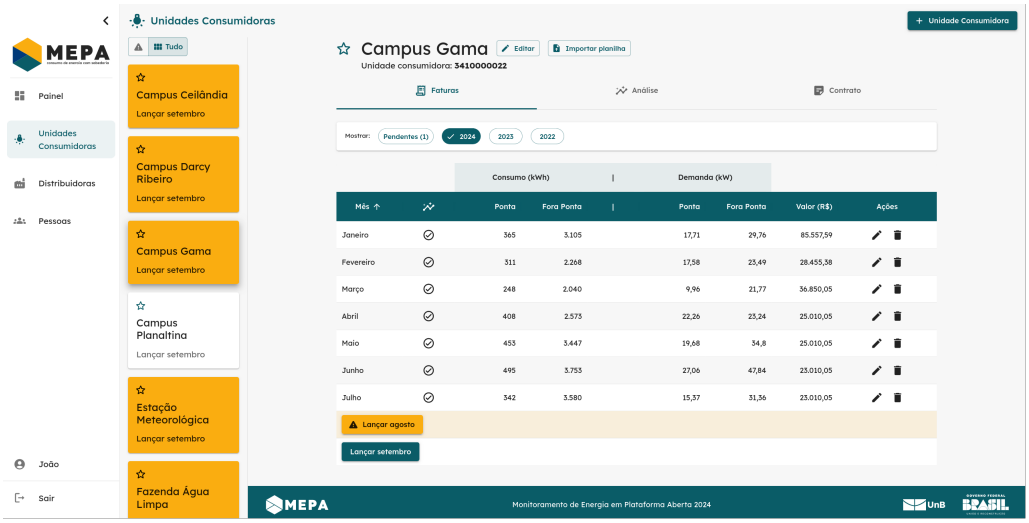
Acontece que os contratos de energia elétrica firmados entre as instituições e as distribuidoras podem apresentar espaço para economia, sendo de responsabilidade da gestão a identificação de tais otimizações. Reduzir esse tipo de gasto sem comprometer a qualidade dos serviços prestados é um desafio que exige um conhecimento aprofundado sobre as regras contratuais, bem como sobre os perfis e padrões de consumo da instituição.

1.1 Contextualização

Com o objetivo de auxiliar as instituições de ensino superior na gestão e adequação dos contratos de energia elétrica, o projeto Monitoramento de Energia em Plataforma Aberta (MEPA) se trata de uma plataforma *open source* criada por meio de uma parceria entre o Ministério da Educação e a Universidade de Brasília (LAPPIS-UNB, 2024).

A plataforma oferece um sistema de recomendação que, com base no histórico de consumo registrado, sugere ajustes nos contratos e gera relatórios sobre o consumo, visando à economia de recursos. A Figura 1 mostra uma página da plataforma.

Figura 1 – Monitoramento de Energia em Plataforma Aberta (MEPA)



Fonte: Site do MEPA.

Para tanto, o módulo de recomendações utiliza um modelo estatístico simples, pesquisando 11 percentis do histórico de consumo e comparando entre as possibilidades de tarifação. Com a metodologia atual, já é possível gerar uma grande economia aos usuários da plataforma.

Mesmo assim, e por a recomendação se tratar de um problema de otimização, busca-se aprimorar a funcionalidade com abordagens mais sofisticadas a fim de gerar ainda mais economia e vantagens ao usuário.

1.2 Justificativa

Na literatura, encontram-se diversas aplicações da computação bioinspirada no contexto de energia elétrica. Os problemas de otimização nessa área são amplamente explorados por essa abordagem, que costuma gerar resultados muito favoráveis.

Portanto, visando a evolução do sistema para fornecer uma funcionalidade de recomendação que proporcione ainda mais economia ao usuário, as abordagens de busca global e algoritmos bioinspirados devem ser analisadas e sua viabilidade devidamente estudada.

Além disso, explorar o algoritmo já implementado é uma excelente oportunidade para aprofundar o conhecimento sobre o funcionamento do sistema, consolidando de forma definitiva sua performance e eficácia.

Muitas vezes, entender as limitações de uma funcionalidade já mitiga a necessidade de substituição. Testar abordagens emergentes sugere evolução constante, tanto dos sistemas em questão, quanto das próprias tecnologias estudadas.

1.3 Objetivos

1.3.1 Objetivo Geral

Este trabalho tem como objetivo geral, a avaliação de viabilidade da aplicação de algoritmos de busca global bioinspirados, bem como a sua implementação no sistema de recomendações de contratos de energia elétrica do MEPA.

1.3.2 Objetivos Específicos

Tendo o objetivo geral, pode-se listar os específicos:

- Modelar e implementar o problema de otimização para a utilização dos algoritmos estudados;
- Organizar a estrutura de experimentação;
- Aproveitar o máximo dos dados disponíveis;
- Utilizar o Método atual em conjunto aos estudados a fim de comparação;
- Realizar a implementação das abordagens viáveis.

1.4 Organização da Monografia

O presente trabalho está organizado da seguinte forma:

- **Capítulo 2 - Fundamentação Teórica:** aborda os conceitos de entendimento necessários para o correto embasamento do presente trabalho;
- **Capítulo 3 - Metodologia:** especifica as escolhas metodológicas referentes à condução do trabalho;
- **Capítulo 4 - Prova de Conceito:** mostra detalhadamente como se deu a prova de conceito a cerca da viabilidade dos algoritmos estudados, bem como os seus resultados;
- **Capítulo 5 - Implementação:** apresenta a implementação realizada das abordagens viáveis;
- **Capítulo 6 - Considerações Finais:** oportunidades futuras e conclusão do trabalho.

2 Fundamentação Teórica

Ao longo deste capítulo, serão abordados conceitos fundamentais necessários para o correto embasamento do presente trabalho. Dessa forma, serão apresentados os aspectos intrínsecos do domínio estudado, abordando os detalhes referentes ao contexto e às normas vigentes sobre contratos de energia, as características do sistema de recomendação atual e seus aspectos estatísticos, e, por fim, os conceitos sobre algoritmos de busca global.

2.1 Contratos de Energia

Empresas e instituições que possuem alta demanda de energia elétrica podem ser classificadas como grandes consumidores. Nesse sentido, sua relação com as distribuidoras de energia precisa ser regida por contratos, nos quais os consumidores assumem o compromisso de pagar pela demanda contratada (SANTOS, 2020), que reflete a estimativa de energia a ser utilizada em um determinado período.

Expressa em quilowatts (kW), a demanda contratada é o valor firmado em contrato referente à demanda de potência ativa a ser obrigatória e continuamente disponibilizada pela distribuidora, no ponto de entrega, conforme período de vigência também fixado em contrato, e que deve ser integralmente paga, seja ou não utilizada durante o período de faturamento.

Tendo isso em vista, os termos tidos em tais contrato precisam estar de acordo com uma série de regras e normas, as quais tem o objetivo de classificar as unidades consumidoras e seu tipo de conexão e definir as bases de cálculos para os valores cobrados.

Sendo uma forma de se assegurar o suprimento das necessidades dos consumidores pelo sistema elétrico sem o risco de sobrecarregar a rede e interromper o abastecimento, as unidades consumidoras são encaixadas em grupos. A (ANEEL, 2021) estabelece as regras de prestação do serviço público de distribuição de energia elétrica no Brasil, e fixa os grupos A e B.

O Grupo A é composto por unidades consumidoras conectadas em tensão maior ou igual a 2,3 quilovolts (kV), ou atendidas pelo sistema de distribuição subterrâneo em tensão menor que 2,3 kV. Essas unidades são tipicamente grandes consumidores, como indústrias e grandes instituições. Já o grupo B é formado por unidades consumidoras com conexão em tensão menor que 2,3 kV, sendo geralmente composto por pequenos consumidores, como residências e pequenos negócios.

O presente trabalho focará especificamente no Grupo A, uma vez que o MEPA é exclusivamente direcionado para grandes consumidores (LAPPIS-UNB, 2024). Tal grupo

é composto por subgrupos, definidos pela faixa da tensão de conexão. São estes:

- **Subgrupo A1:** tensão de conexão maior ou igual a 230 kV;
- **Subgrupo A2:** tensão de conexão de 88 kV a 138 kV;
- **Subgrupo A3:** tensão de conexão igual a 69 kV;
- **Subgrupo A3a:** tensão de conexão de 30 kV a 44 kV;
- **Subgrupo A4:** tensão de conexão de 2,3 kV a 25 kV;
- **Subgrupo AS:** tensão de conexão menor que 2,3 kV, a partir de sistema subterrâneo de distribuição.

Além disso, a (ANEEL, 2021) também estabelece os chamados postos tarifários e as modalidades tarifárias. O posto tarifário se trata de um período em horas para aplicação das tarifas de forma diferenciada ao longo do dia, considerando a seguinte divisão:

- **Posto tarifário ponta:** período composto por três horas diárias consecutivas definidas pela distribuidora, considerando a curva de carga de seu sistema elétrico, aprovado pela ANEEL para toda a área de concessão ou permissão;
- **Posto tarifário fora de ponta:** período composto pelo conjunto das horas diárias consecutivas e complementares àquelas definidas nos postos ponta.

As modalidades tarifárias são conjuntos de tarifas aplicáveis às componentes de consumo de energia elétrica e demanda que compõem o valor total da fatura. Ou seja,

$$V_t = V_c + V_d, \tag{2.1}$$

onde V_t é o valor total da fatura, V_c é o valor do consumo e V_d é o valor da demanda. Tais componentes serão destrinchadas mais à frente.

Para tanto, as tarifas são o valor monetário estabelecido pela ANEEL, fixado em R$ (reais) por unidade de energia elétrica ativa ou da demanda de potência ativa, sendo:

- **Tarifa de Energia (TE):** valor monetário unitário determinado pela ANEEL, em R$/MWh, utilizado para efetuar o faturamento mensal referente ao consumo de energia;
- **Tarifa de Uso do Sistema de Distribuição (TUSD):** valor monetário unitário determinado pela ANEEL, em R$/MWh e em R$/kW, utilizado para efetuar o faturamento mensal de usuários do sistema de distribuição de energia elétrica pelo uso do sistema;

- **Tarifa de Uso do Sistema de Distribuição para Geração ($\mathbf{TUSD_g}$):** tarifa aplicada à energia gerada e injetada na rede pelas unidades consumidoras.

Os valores das tarifas podem ser diferentes conforme o posto e a modalidade tarifária. No entanto, no que se refere à componente do consumo de energia elétrica, a aplicação das tarifas independe da modalidade tarifária, e pode ser descrita por

$$V_c = C_p \cdot \frac{TUSD_p + TE_p}{1000} + C_{fp} \cdot \frac{TUSD_{fp} + TE_{fp}}{1000} \tag{2.2}$$

onde V_c é o valor do consumo em reais, C é o consumo medido em kWh, $TUSD$ é a tarifa de uso do sistema de distribuição em R$/MWh e TE é a tarifa de energia em R$/MWh. Os índices p e fp referem-se, portanto, aos postos ponta e fora de ponta respectivamente.

Já para a componente da demanda, a aplicação das tarifas muda conforme a modalidade tarifária. No grupo A, existem as modalidades tarifárias verde e azul, as quais serão descritas a seguir.

2.1.1 Modalidade tarifária horária verde

Nessa modalidade tarifária, não há segmentação por posto tarifário para se obter o valor da demanda. Em outras palavras, a tarifa que incide no componente é a mesma para todos os períodos do dia.

O valor da demanda é dado por

$$V_d = TUSD \cdot D + 3 \cdot TUSD \cdot (U_p + U_{fp}) \tag{2.3}$$

onde V_d é o valor da demanda em reais, $TUSD$ é a tarifa de uso do sistema de distribuição em R$/kW, D é a demanda contratada em kW e, por fim, U_p e U_{fp} são, respectivamente, os valores de ultrapassagem nos postos ponta e fora ponta.

Para os valores de ultrapassagem, que representam o quanto a demanda medida extrapolou a contratada, tem-se

$$U_p = \begin{cases} 0 & \text{se } DM_p - D \leq 0, \\ DM_p - D & \text{se } DM_p - D \geq 0, \end{cases} \tag{2.4}$$

$$U_{fp} = \begin{cases} 0 & \text{se } DM_{fp} - D \leq 0, \\ DM_{fp} - D & \text{se } DM_{fp} - D \geq 0, \end{cases} \tag{2.5}$$

onde DM_p e DM_{fp} são, respectivamente, as demandas medida nos postos ponta e fora ponta. A medição das demandas é realizado pela distribuidora de 15 em 15 minutos, mede-se a potência utilizada, e a média desses valores no mês de referência define a demanda medida mensal.

2.1.2 Modalidade tarifária horária azul

Na modalidade tarifária azul, além do consumo, a demanda também é segmentada pelo posto tarifário. Ou seja, o contrato deve abranger uma demanda contratada para cada posto, e, por conseguinte, a tarifa que incide no componente em posto ponta pode ser diferente da que incide em posto fora de ponta.

Sendo assim, o valor da demanda na modalidade azul é dado por

$$V_d = D_p \cdot TUSD_p + 3 \cdot TUSD_p \cdot U_p + D_{fp} \cdot TUSD_{fp} + 3 \cdot TUSD_{fp} \cdot U_{fp} \tag{2.6}$$

onde V_d é o valor da demanda em reais, $TUSD$ é a tarifa de uso do sistema de distribuição em R$/kW, D é a demanda contratada em kW e, por fim, U_p e U_{fp} são, respectivamente, os valores de ultrapassagem nos postos ponta e fora ponta. Portanto, os índices p e fp referem-se, aos postos ponta e fora de ponta respectivamente.

Com a segmentação das demandas contratadas, o cálculo da ultrapassagem também deve ser feito referente ao posto. Sendo assim, as ultrapassagens ficam sendo

$$U_p = \begin{cases} 0 & \text{se } DM_p - D_p \leq 0, \\ DM_p - D_p & \text{se } DM_p - D_p \geq 0, \end{cases} \tag{2.7}$$

$$U_{fp} = \begin{cases} 0 & \text{se } DM_{fp} - D_{fp} \leq 0, \\ DM_{fp} - D_{fp} & \text{se } DM_{fp} - D_{fp} \geq 0. \end{cases} \tag{2.8}$$

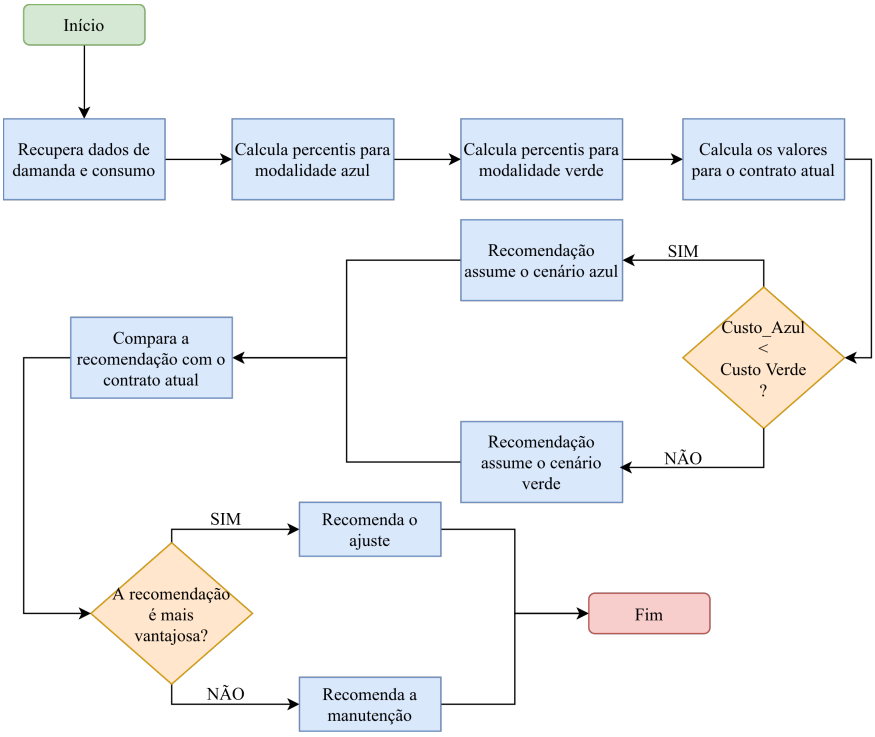
2.2 Sistema de Recomendação

Como citado anteriormente, o MEPA é um projeto *open source* desenvolvido pela UnB em parceria com o Ministério da Educação. Seu objetivo é auxiliar as instituições de ensino superior na gestão e avaliação da adequação dos contratos de energia elétrica, a partir do registro das faturas mensais de suas instalações, gerando relatórios de recomendações de ajustes nos contratos visando à economia de recursos (LAPPIS-UNB, 2024).

Dessa forma, a plataforma conta com um módulo focado em analisar a situação da instituição e oferecer recomendações de ajustes do contrato de energia quando um cenário mais vantajoso, conforme dados prévios, for detectado.

O sistema utiliza uma abordagem estatística simples, realizando uma análise quantitativa de alguns percentis de demanda medida nos meses registrados para cada modalidade tarifária. Dessa forma, é possível comparar os valores de demanda obtidos em cada cenário e indicar o mais vantajoso. A Figura 2 apresenta o fluxograma de funcionamento do sistema de recomendação.

Figura 2 – Fluxograma de funcionamento da recomendação



Fonte: Autor.

O fluxo se inicia com a solicitação de uma recomendação no sistema. Primeiramente, são recuperadas da base de dados as informações referentes ao contrato atual e ao histórico mensal de consumo e demanda registrados.

Com esses dados, inicia-se o cálculo dos percentis. Percentil é uma medida separatriz que divide o conjunto de dados ordenados (rol) em cem partes iguais (MARTINS, 2005). Dessa forma, pode-se dizer que o percentil 10, ou simplesmente P10, é um valor maior do que 10% e menor do que 90% das observações contidas no conjunto de dados.

Assim, o sistema de recomendações do MEPA calcula os percentis P10, P20, P30, P40, P50, P60, P70, P80, P90, P95 e P98 do conjunto de demandas medidas que compõem o histórico analisado, tanto no posto ponta quanto no posto fora de ponta. É importante ressaltar que, mesmo para unidades consumidoras cujo contrato atual está na modalidade verde, as faturas de energia elétrica geralmente informam as demandas medidas em ambos os postos. Isso permite utilizar essas medições para simular os cenários de ambas as modalidades.

Dessa forma, são analisados 11 cenários em cada modalidade tarifária, totalizando 22 cenários. Nessa etapa, cada percentil precisa ter um valor de demanda total a fim de se identificar o representante de cada modalidade. Assim, cada percentil calculará o valor de demanda conforme a Equação 2.3 para a modalidade verde e a Equação 2.6 para a modalidade azul, usando o percentil como a demanda contratada.

Finalmente, com todos os percentis tendo um valor de demanda total atribuído, o sistema procede à realização do resumo. O resumo consiste na consolidação da recomendação mais vantajosa para cada modalidade. Dessa forma, ambas as modalidades terão em seu resumo todos os dados referentes ao percentil que apresentou o menor valor de demanda total, junto aos valores de consumo. Com isso, o resumo também inclui o custo total mensal, que é a soma do valor de consumo com o valor de demanda, vide Equação 2.1, bem como o custo total final, que é a soma dos custos totais mensais calculados.

Com os dois resumos prontos, o sistema pode selecionar qual das propostas é a mais vantajosa. Assim, a proposta que obtiver o menor valor de custo total será escolhida como a recomendação.

Após o cálculo dos percentis para as duas modalidades, o sistema também gera os valores referentes ao contrato atual. Posteriormente, realiza-se a comparação entre a recomendação gerada e o valor atual. Caso seja identificada economia na recomendação gerada, o sistema recomendará o ajuste do contrato para o cenário gerado. Caso contrário, será recomendado manter o contrato atual.

2.3 Algoritmos de busca global

Na solução de problemas mais complexos, que não são possíveis de serem resolvidos através da computação tradicional, tende-se a recorrer a algoritmos que, dentro do conjunto de possíveis soluções, devem realizar buscas pelas melhores. Dessa forma, são necessárias abordagens inteligentes para identificar as soluções adequadas (KAR, 2016).

Em geral, os algoritmos de busca global funcionam a partir de um espaço de busca, que é o conjunto inicial que contém todas as soluções do problema expresso pela função objetivo. O algoritmo seleciona repetidamente candidatos, divide e manipula o conjunto e, através de filtros, elimina os subconjuntos de soluções não viáveis (SMITH, 1987). Essas operações são realizadas até que nenhum conjunto reste para ser dividido.

No contexto da otimização, uma função objetivo é uma função matemática que quantifica a qualidade ou o custo de uma solução dentro do espaço de busca. O objetivo de um algoritmo de otimização, portanto, é encontrar os valores das variáveis de decisão que minimizam ou maximizam essa função (NOCEDAL; WRIGHT, 1999).

Nesse contexto, tem-se a computação bioinspirada, que é um método de pesquisa voltado para a solução de problemas por meio de modelos computacionais baseados nos princípios biológicos e naturais. De acordo com (BINITHA; SATHYA et al., 2012), a natureza é um um exemplo perfeito quando se trata de otimização pois, ao analisar de perto, ela está sempre seguindo soluções muito eficientes nos mais variados cenários e fenômenos.

Então, faz todo o sentido se inspirar na natureza visto que os problemas, emergentes ou não, no campo da computação, podem ter muito em comum com problemas que a natureza já resolveu a muito tempo (BINITHA; SATHYA et al., 2012).

2.3.1 *Particle Swarm Optimization*

Ao se estudar o comportamento de animais sociais como aves e insetos, foram desenvolvidos alguns modelos computacionais denominados de inteligência de enxame. A palavra "enxame", nesse contexto, refere-se a um grupo de indivíduos livres que se comunicam entre si, direta ou indiretamente, atuando em seus ambientes locais (ENGELBRECHT, 2007). A interação entre os indivíduos que compõem o enxame resulta em estratégias de buscas coletivas e distributivas.

Proposto por (KENNEDY; EBERHART, 1995), a otimização por enxame de partículas, *particle swarm optimization* (PSO) em inglês, é um modelo computacional de inteligência de enxame baseado no comportamento social de bandos de pássaros em busca de alimento. De acordo com (BINITHA; SATHYA et al., 2012), o PSO possui um mecanismo de busca único e é amplamente utilizado em várias áreas da engenharia em problemas de otimização devido à sua simplicidade, eficiência e fácil implementação.

2.3.1.1 Descrição do PSO

Essa seção abordará, com base em (SUN; LAI; WU, 2016), a descrição funcional do PSO. O algoritmo utiliza uma população de M partículas de volumes desprezíveis, sendo que cada uma delas é descrita por três vetores N-dimensionais, onde N representa o número de variáveis contidas no domínio do problema.

Na n-ésima iteração, para cada partícula i $(1 \leq i \leq M)$, os vetores serão:

- **O vetor da posição atual:** $X_{i,n} = (X_{i,n}^1, X_{i,n}^2, \ldots, X_{i,n}^N)$;
- **O vetor da velocidade:** $V_{i,n} = (V_{i,n}^1, V_{i,n}^2, \ldots, V_{i,n}^N)$, que representa o incremento da posição atual;
- **O vetor da melhor posição individual:** $P_{i,n} = (P_{i,n}^1, P_{i,n}^2, \ldots, P_{i,n}^N)$.

Inicialmente, pode-se definir aleatoriamente, com distribuição uniforme, as componentes de $X_{i,0}$ utilizando o espaço de busca em $[X_{min}^j, X_{max}^j]$ $(1 \leq j \leq N)$, sendo X_{min}^j e X_{max}^j, respectivamente, os limites inferior e superior para a posição das partículas na dimensão j.

A inicialização do vetor de velocidade $V_{i,0}$ pode ser feita de forma similar ao vetor de posição. Seleciona-se aleatoriamente a componente j em $[-V_{max}^j, V_{max}^j]$ $(1 \leq j \leq N)$,

onde V^j_{max} é o limite superior de velocidade. Já a melhor posição inicial $P_{i,0}$ pode ser definida como sendo igual a $X_{i,0}$.

Além disso, é considerada a melhor partícula globalmente, aquela cuja melhor posição individual retorna o melhor resultado para o problema. Denota-se $P_{g,n}$ essa posição, sendo g o índice da melhor partícula globalmente. Além disso, tal posição é armazenada no vetor de melhor posição global $G_n = (G^1_n, G^2_n, \ldots, G^N_n)$.

Dessa forma, a atualização do vetor de melhor posição individual de cada partícula $P_{i,n}$ é dada por

$$P_{i,n} = \begin{cases} X_{i,n} & \text{se } f(X_{i,n}) < f(P_{i,n-1}), \\ P_{i,n-1} & \text{se } f(X_{i,n}) \geq f(P_{i,n-1}). \end{cases} \tag{2.9}$$

A melhor posição individual da partícula, na iteração n, passa a ser a posição atual caso ela retorne um valor melhor para a função objetivo $f(x)$ do que o valor retornado pela melhor posição individual na iteração anterior. Caso o contrário, a melhor posição individual continua sendo a anterior.

Portanto, o vetor da melhor posição global é encontrado por

$$G_n = P_{g,n}, \tag{2.10}$$

onde $g = argmin[f(P_{i,n})]$ com $1 \leq i \leq M$. Ou seja, a melhor posição global na iteração n será a posição da partícula que melhor minimizar o resultado da função objetivo.

Agora, na iteração $n+1$, cada partícula terá as propriedades de suas componentes atualizadas conforme

$$V^j_{i,n+1} = V^j_{i,n} + c_1 r^j_{i,n}(P^j_{i,n} - X^j_{i,n}) + c_2 R^j_{i,n}(G^j_n - X^j_{i,n}) \tag{2.11}$$

$$X^j_{i,n+1} = X^j_{i,n} + V^j_{i,n+1} \tag{2.12}$$

para $(1 \leq i \leq M)$, $(1 \leq j \leq N)$ e sendo $c1$ e $c2$ coeficientes de aceleração. Os parâmetros $r^j_{i,n}$ e $R^j_{i,n}$ são duas sequências de números aleatórios distribuídos uniformemente entre 0 e 1.

Segundo (SUN; LAI; WU, 2016), essa descrição define o chamado PSO original e que testes mostraram um desempenho longe do satisfatório em problemas mais amplamente utilizados.

Por isso, surgiram duas versões revisadas do PSO. Proposto por (SHI; EBERHART, 1998), o chamado PSO-In inclui um fator multiplicativo w representando o peso inercial. Assim, a velocidade das partículas passa a ser ajustada conforme a fórmula

$$V^j_{i,n+1} = wV^j_{i,n} + c_1 r^j_{i,n}(P^j_{i,n} - X^j_{i,n}) + c_2 R^j_{i,n}(G^j_n - X^j_{i,n}) \tag{2.13}$$

onde w é geralmente menor do que 1 a fim de acelerar a convergência das partículas.

Uma outra revisão proposta por (CLERC, 1999), chamada PSO-Co ajusta a velocidade da partículas utilizando um fator χ na forma

$$V_{i,n+1}^{j} = \chi[V_{i,n}^{j} + c_1 r_{i,n}^{j}(P_{i,n}^{j} - X_{i,n}^{j}) + c_2 R_{i,n}^{j}(G_{n}^{j} - X_{i,n}^{j})] \tag{2.14}$$

sendo χ determinado pela equação

$$\chi = \frac{2}{|2 - \varphi - \sqrt{\varphi^2 - 4\varphi}|} \tag{2.15}$$

com $\varphi = c1 + c2$, onde $c1$ e $c2$ sejam definidos de forma a fazer com que φ seja maior do que 4.

Posto isso, (SUN; LAI; WU, 2016) conclui afirmando que, juntas, essas duas versões do PSO compõem o chamado PSO canônico e que apresentam uma melhor performance no geral. Existem muitas outras revisões do PSO mas todas são baseadas no PSO canônico.

2.3.2 Algoritmo Genético

Os algoritmos evolucionários são os mais estabelecidos dentre os algoritmos bioinspirados. Baseiam-se na evolução natural, a qual, conforme a teoria estabelece, é responsável pelo design de todos os seres vivos, bem como por suas estratégias de interação (BINITHA; SATHYA et al., 2012).

Sendo possivelmente o primeiro algoritmo implementado para simular sistemas genéticos, o algoritmo genético, *genetic algorithm* (GA) em inglês, modela a evolução genética utilizando a seleção, recombinação ou reprodução e a mutação como os principais operadores impulsionadores (ENGELBRECHT, 2007).

2.3.2.1 Descrição do GA

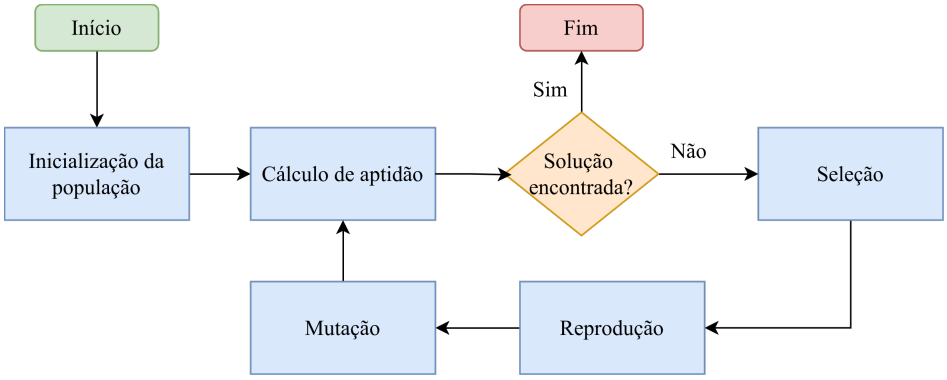
A presente seção irá descrever o algoritmo genético utilizando como base os trabalhos de (ENGELBRECHT, 2007) e (MIRANDA, 2007). A Figura 3 mostra o fluxograma referente ao GA.

Ao iniciar o algoritmo, uma população de n indivíduos é gerada de forma aleatória. O indivíduo é chamado de cromossomo e se trata de uma cadeia de valores, binários, inteiros ou reais, a qual representa uma possível solução para o problema.

Tendo a população definida, se dão inicio as iterações realizando o cálculo de aptidão dos indivíduos, o qual se trata, basicamente, do cálculo da função objetivo e checagem de condições de parada que, de acordo com (ENGELBRECHT, 2007), podem estar nas seguintes situações:

- **Finaliza quando nenhuma melhoria for detectada:** quando as soluções individuais da população utilizada não possuem uma melhoria significativa em relação a gerações passadas;

Figura 3 – Fluxograma do GA

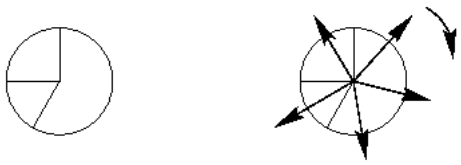


Fonte: Adaptado de (MIRANDA, 2007).

- **Finaliza quando as mudanças estagnam:** quando, dentre um número de gerações consecutivas, a média de mudanças genotípicas é muito baixa;
- **Finaliza ao detectar uma solução aceitável:** quando alguma solução individual x satisfaça $f(x) \leq |f(x) - e|$, sendo f a função objetivo e e o limite de erro.

Prosseguindo o fluxo do algoritmo, a operação de seleção escolhe os indivíduos que avançarão para a próxima operação. A probabilidade de um indivíduo ser selecionado é proporcional à sua aptidão. O método utilizado é chamado de amostragem universal estocástica e, conforme (MIRANDA, 2007), pode ser ilustrado na Figura 4.

Figura 4 – Amostragem estocástica universal



Fonte: (MIRANDA, 2007)

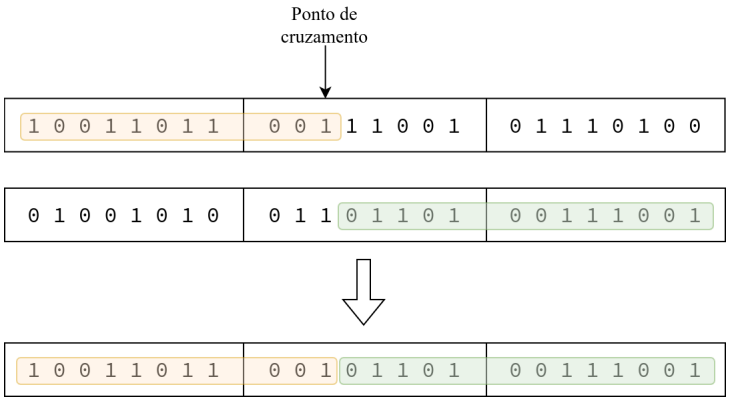
Considerando um círculo dividido em n regiões (tamanho da população), em que cada região possui uma área proporcional à aptidão do indivíduo, sobrepondo-o com uma "roleta" contendo n cursores igualmente espaçados e girando-a, a posição final dos cursores indica os indivíduos selecionados. Uma consequência desse método é que a seleção pode conter várias cópias do mesmo indivíduo, enquanto outros podem simplesmente desaparecer.

Com os indivíduos selecionados, pode-se prosseguir para a reprodução. Nessa operação, a lista de indivíduos selecionados é embaralhada aleatoriamente, gerando uma nova lista chamada lista de parceiros.

Realiza-se, então, um cruzamento, denominado *cross-over*, entre os indivíduos selecionados e os indivíduos ocupantes da mesma posição na lista de parceiros. Cada par é

particionado em um ponto sorteado aleatoriamente e um novo cromossomo é gerado ao permutar a metade inicial de um cromossomo com a metade do outro.

Figura 5 – Reprodução no GA



Fonte: Adaptado de (MIRANDA, 2007).

Com a nova população gerada, antes de se reiniciar o ciclo, pode-se executar a operação de mutação. De acordo com (ENGELBRECHT, 2007), a operação de mutação não é crítica na implementação do GA, sendo uma estratégia de melhoria do algoritmo ao adicionar um fator de diversidade a mais.

Dessa forma, como posto por (MIRANDA, 2007), a mutação se dá ao alterar aleatoriamente um gene (componente) de indivíduos também escolhidos aleatoriamente. Sendo assim, alguns indivíduos da nova população podem ter um de seus genes alterado aleatoriamente.

3 Metodologia

Este capítulo apresenta a metodologia utilizada para a realização deste trabalho. De acordo com (GÓMEZ, 2000), a metodologia refere-se ao início de um processo de pensamento cujo objetivo é a produção de um conhecimento novo. Sendo assim, de forma situacional, os insumos e resultados deste trabalho visam explorar uma situação específica, produzindo conhecimento para aprimorar a resolução da problemática, o que se enquadra, de acordo com (ENGEL, 2000), nas características da pesquisa-ação.

Dessa maneira, baseando-se nas fases da pesquisa-ação, a metodologia aplicada à produção deste trabalho abrange as etapas de entendimento inicial, levantamento teórico, prova de conceito e implementação.

3.1 Entendimento Inicial

Nessa etapa, buscou-se o entendimento da problemática com o intuito de compreender o domínio do problema. Para isso, a utilização do MEPA foi fundamental.

Ao realizar uma exploração minuciosa do funcionamento do sistema, bem como a leitura de sua documentação (já que se tratar de um projeto *open source*), foi possível identificar as necessidades e o objetivo da plataforma, o qual consiste em auxiliar na economia de recursos das instituições de ensino superior que utilizam a plataforma.

Além disso, de forma mais aprofundada, foi realizado um estudo detalhado sobre o sistema de recomendação atual. Por meio do repositório aberto, foi possível explorar a arquitetura da solução e examinar minuciosamente as abordagens adotadas, o que aumentou consideravelmente o entendimento sobre as necessidades e potenciais formas de ação.

3.2 Levantamento Teórico

Com o objetivo de estabelecer um referencial teórico sólido, buscou-se por fontes relevantes relacionadas ao domínio do problema, abrangendo aspectos das normas e legislações vigentes, a classe de algoritmos de busca global, e a computação bioinspirada, que se destaca por seu grande potencial em aprimorar a solução atual.

A fundamentação teórica foi construída com foco nas características específicas do contexto em questão, aprofundando-se de maneira fundamentada nas nuances da computação bioinspirada e nas normas que regulamentam a distribuição de energia no Brasil.

Essas normas, utilizadas como regras de negócio pela plataforma, desempenham um papel central no desenvolvimento e implementação da solução.

3.3 Prova de Conceito

Optou-se pela realização de uma prova de conceito, visto que um dos principais objetivos deste trabalho é avaliar a viabilidade da aplicação de um novo método à solução de recomendações do MEPA. Nesse contexto, a prova de conceito se revela a metodologia ideal. Por contar com um alto componente prático e experimental, ela se alinha às características da pesquisa-ação, especialmente na fase de avaliação do plano de intervenção.

3.4 Implementação

Baseando-se nos resultados da prova de conceito, foi realizada a implementação das abordagens que se mostraram viáveis. Nesse sentido, a implementação foi feita visando aprimorar aspectos como o reuso de código, a modularização e a simplicidade. Isso garantirá que os novos módulos de otimização sejam compatíveis com a arquitetura existente da plataforma e possam ser facilmente mantidos e expandidos.

3.5 Ferramentas

Essa seção apresenta as ferramentas com as quais esse trabalho foi elaborado.

- ***Python:*** linguagem de programação alto nível famosa pela sua simplicidade. Amplamente utilizada na área de *data science*, permite o desenvolvimento em múltiplos paradigmas. Foi utilizado por conta da sua gama de bibliotecas voltadas para manipulação de dados e também por ser a linguagem utilizada na plataforma abordada;
- ***Jupyter:*** ambiente virtual de código aberto que proporciona a codificação e experimentação por meio de documentos interativos chamados *notebooks*. Sua utilização se deu devido a facilidade que entrega na execução parcial e simples de blocos de código, bem como por oferecer uma estrutura completa para a realização da experimentação;
- ***Pandas:*** biblioteca *Python* especializada em manipulação e análise de dados. Foi utilizada por conta da gama de funcionalidades e estruturas de dados;
- ***Scikit-opt:*** biblioteca *Python* especializada em computação bioinspirada. Foi utilizada pois possui a implementação dos algoritmos do PSO e do GA estudados nesse trabalho. Além disso, é bem atualizada e bem recomendada.

- ***Matplotlib:*** biblioteca *Python* para visualização de dados. Foi utilizada por conta de sua simplicidade e pela qualidade das visualizações geradas;
- **Excel:** plataforma de planilhas eletrônicas desenvolvida pela Microsoft e muito consagrada no mercado. Foi utilizada pois os dados disponibilizados para as testagens estão registrados nessas planilhas.
- ***Docker:*** plataforma de código aberto para *conteinerização*. Foi utilizada por conta da facilidade de empacotamento de soluções e por oferecer a imagem *jupyter/scipy-notebook* contendo a maioria dos recursos necessários para este trabalho.

4 Prova de Conceito

A presente prova de conceito consiste em uma experimentação computacional para analisar quantitativamente o desempenho dos algoritmos bioinspirados PSO e GA no contexto de recomendação de ajuste de contratos de energia elétrica.

Utilizando o *Jupyter Notebook*, que é uma ferramenta popular no ambiente de ciência de dados e desenvolvimento de software, foram desenvolvidas rotinas para integrar os recursos do módulo de recomendação do MEPA aos algoritmos bioinspirados estudados. Com isso, foi feita a utilização de uma base de dados de contas de energia de unidades consumidoras (UC) internas de 12 universidades federais brasileiras a fim de gerar dados comparativos entre os desempenhos das abordagens em diferentes cenários.

4.1 Objetivos

Esta prova de conceito tem como objetivo, de forma geral, validar a viabilidade da aplicação de algoritmos bioinspirados de busca global na recomendação de ajustes de contratos de energia elétrica, baseando-se no histórico de faturas registrado.

Busca-se avaliar quantitativamente o desempenho dos algoritmos estudados e sua precisão na recomendação de contratos que proporcionem a maior taxa de economia ao usuário, bem como observar as vantagens e desvantagens dessas abordagens.

Além de colaborar para essa observação, esta prova de conceito também proporciona um estudo analítico sobre a abordagem atualmente utilizada no sistema de recomendações do MEPA, aumentando, assim, o entendimento sobre o que já está sendo oferecido, bem como ampliando a visão para maiores incrementos e estudos futuros.

4.2 Configuração do ambiente

Para garantir a praticidade, foi utilizada uma imagem Docker que contempla a maioria dos requisitos necessários para a experimentação. Dessa forma, a imagem *jupyter/scipy-notebook*[1] possui muitos recursos facilitadores para a realização de experimentos no *Jupyter Notebook*.

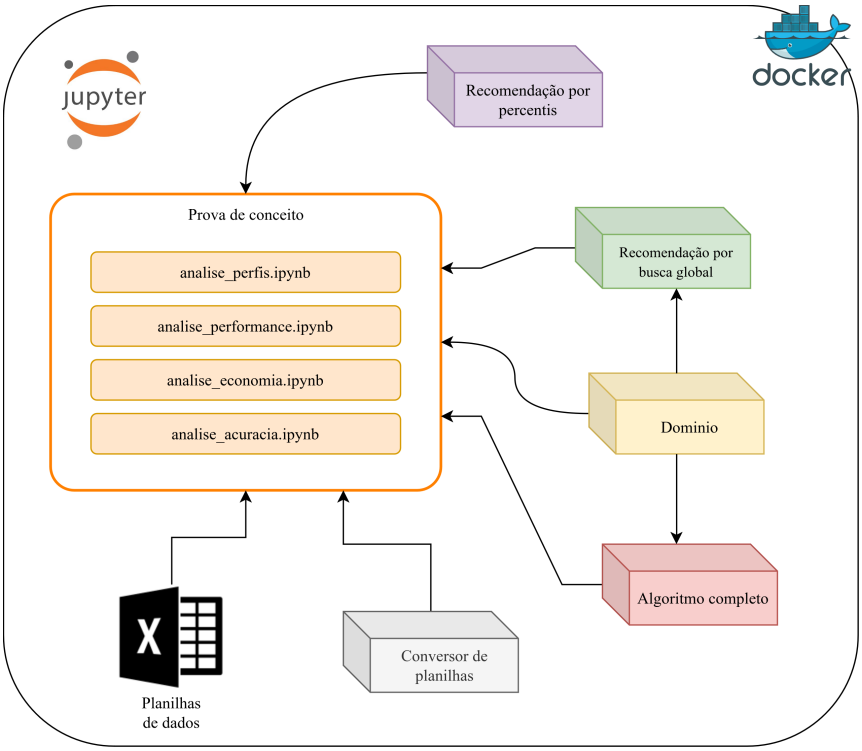
Toda a prova de conceito foi realizada via *Jupyter Notebook* acessado via navegador. Para simplificar a configuração do ambiente, foi criado um repositório no GitHub[2] contendo todos os passos necessários para rodar o *Jupyter* com as rotinas utilizadas.

[1] Imagem Docker: <https://hub.docker.com/r/jupyter/scipy-notebook>
[2] Repositório da prova de conceito: <https://github.com/DanielPortods/Recomendacao>

4.3 Arquitetura

A arquitetura elaborada para a experimentação utiliza o *Jupyter Notebook* como ponto de entrada, sendo o elemento central onde foram codificadas todas as rotinas necessárias. Foi utilizada a linguagem de programação *Python* devido à sua ampla gama de bibliotecas voltadas à computação bioinspirada, bem como por ser utilizada na codificação do MEPA. A Figura 6 ilustra a arquitetura desta prova de conceito.

Figura 6 – Diagrama de arquitetura



Fonte: Autor.

Basicamente, todos os recursos desenvolvidos são importados nos arquivos que integram a prova de conceito. Tais arquivos utilizam a extensão ".ipynb", formato utilizado pelo *Jupyter Notebook*, e podem conter texto, código de programação, gráficos e visualizações interativas, tornando-o uma escolha poderosa para documentar, compartilhar e executar análises e modelos de forma reproduzível.

Foram utilizados os recursos de orientação a objetos disponibilizados pelo *Python*, bem como alguns padrões de projeto cabíveis para melhor estruturar a experimentação. Para compreender a estrutura e o funcionamento da prova de conceito, segue uma breve descrição dos módulos ilustrados no diagrama da Figura 6.

- **Recomendação por percentis:** se trata do sistema de recomendação atualmente em uso no MEPA. É responsável por analisar os dados históricos das faturas de

energia e sugerir a recomendação mais vantajosa das analisadas. Esse módulo foi extraído diretamente do repositório do projeto[3];

- **Recomendação por busca global:** gera a recomendação utilizando as implementações do PSO e do GA da biblioteca de algoritmos bioinspirados *scikit-opt*;
- **Algoritmo completo:** implementa um algoritmo exaustivo, que percorre todas as possíveis combinações de contratos. Ele avalia cada uma dessas combinações para determinar qual delas oferece a melhor otimização de custos e, assim, retorna a solução mais eficiente;
- **Domínio:** define as funções objetivo e outros parâmetros específicos do problema de otimização. Ele opera como um *singleton*, o que faz dele globalmente único, sendo acessível pelos módulos de busca global e do algoritmo completo;
- **Conversor de planilhas:** oferece uma classe para converter dados de planilhas Excel em estruturas de dados mais flexíveis, como dicionários e *dataframes*.

4.4 Tratamento dos Dados

As informações utilizadas nesta prova de conceito são dados reais obtidos das faturas de 12 universidades brasileiras, referentes ao período entre 2019 e 2022. Tais dados estão dispostos em planilhas do Excel no formato *xlsx*, sendo que cada planilha corresponde a uma universidade e cada aba da planilha a uma unidade consumidora interna da instituição.

A Figura 7 mostra a estrutura das planilhas de dados. Como as planilhas possuem colunas complexas, com células mescladas, e linhas de cabeçalhos compostos com comentários explicativos, foi necessário um esforço adicional para extrair as linhas e colunas desejadas.

Uma estratégia seria utilizar algum serviço de planilhas eletrônicas, como o Google Sheets, para fazer o *download* da planilha convertida no formato *csv* (*comma-separated-values*). No entanto, essa abordagem apresenta problemas devido à má qualidade do documento gerado e à alta carga de atividade manual que seria necessária.

Assim, optou-se por utilizar as funcionalidades de leitura de arquivos *xlsx* da biblioteca *Pandas* para tratar as planilhas e convertê-las nas estruturas de dados utilizadas pelas rotinas. Com essa biblioteca, foi possível implementar a classe do conversor de planilhas, que permite extrair de forma simplificada os dados de cada planilha, separados por unidades consumidoras, detectar os detalhes do contrato atual de cada uma e verificar inconsistências nos dados.

[3] MEPA: <https://gitlab.com/lappis-unb/projetos-energia/mec-energia/mec-energia-api>

Figura 7 – Planilha de dados de faturas

	A	B	C	D	E	F	G	H	I
1	Informações Gerais								
2	Nome da distribuidora			Grupo/Subgrupo			Tipo de tarifa		
3	ENEL CE			A4			Verde		
4	Dados da Fatura de Energia								
5	Ano	Mês	Consumo Ponta (kWh)	Consumo Fora Ponta (kWh)	Demanda Ponta Medida (kW)	Demanda Fora Ponta Medid (kW)	Demanda Contratada Ponta (kW)	Demanda Contratada For Ponta (kW)	Valor da fatura (R$)
6	2019	Janeiro	1232	19079	57,12	100,8	139		R$ 11.886,79
7	2019	Fevereiro	1077	17637	38,64	95,34	139		R$ 11.146,33
8	2019	Março	1120	17950	28,98	109,62	139		R$ 11.035,06
9	2019	Abril	2354	26068	87,36	144,06	139		R$ 16.719,29
10	2019	Maio	2163	22733	67,2	152,46	139		R$ 16.365,01
11	2019	Junho	2400	24845	86,94	1337,76	139		R$ 18.436,28
12	2019	Julho	2316	22334	79,8	137,76	139		R$ 16.124,34
13	2019	Agosto	1213	14889	37,8	80,64	139		R$ 11.795,41
14	2019	Setembro	2975	26607	83,16	137,76	139		R$ 20.849,81
15	2019	Outubro	3297	28142	110,46	140,7	139		R$ 20.682,17
16	2019	Novembro	2913	27156	94,08	164,32	139		R$ 20.543,41
17	2019	Dezembro	2124	22918	70,59	130,93	139		R$ 16.854,42
18	2020	Janeiro	2316	26620	47	105	139		R$ 17.905,33
19	2020	Fevereiro	942	14305	31,08	86,94	139		R$ 10.882,87
20	2020	Março	1345	17622	86,94	154,56	139		R$ 14.075,11
21	2020	Abril	1451	15948	75,18	154,98	139		R$ 12.447,87
22	2020	Maio	894	12255	23,1	52,08	139		R$ 9.283,01
23	2020	Junho	898	10199	19,74	47,88	139		R$ 8.550,40
24	2020	Julho	971	11605	23,52	34,86	139		R$ 8.814,11
25	2020	Agosto	963	10166	19,32	36,12	139		R$ 8.482,50
26	2020	Setembro	887	9500	19,74	35,28	139		R$ 7.877,94
27	2020	Outubro	1074	12237	25,62	44,52	139		R$ 9.681,63
28	2020	Novembro	1138	13544	27,72	85,68	139		R$ 10.692,63
29	2020	Dezembro	1250	14743	40,88	82,39	139		R$ 11.217,33
30	2021	Janeiro	947	11984	43	72	139		R$ 10.474,40
31	2021	Fevereiro	996	13717	24,36	70,56	139		R$ 10.593,53
32	2021	Março	1049	14011	47,88	101,22	139		R$ 10.699,28
33	2021	Abril	1139	12552	30,24	43,26	139		R$ 10.357,78

Fonte: Autor.

4.5 Modelagem

A utilização dos algoritmos de busca global tem como requisito a modelagem do problema de otimização a fim de se definir uma função objetivo capaz de abstrair matematicamente o domínio estudado. Para tanto, é interessante começar pela definição do espaço de busca.

O espaço de busca é, basicamente, o conjunto de soluções possíveis, vide Seção 2.3, onde os algoritmos de busca global irão procurar pelas melhores soluções. Dessa forma, ao se analisar o problema, nota-se que, na verdade, tratam-se de dois espaços de busca: um para a modalidade tarifária verde e outro para a modalidade tarifária azul, os quais possuem, respectivamente, uma e duas dimensões.

Na modalidade azul, as duas dimensões referem-se aos valores de demanda contratada nos postos ponta e fora de ponta, com os espaços de busca representados, respectivamente, pelos intervalos $[min_{dp}, max_{dp}]$ e $[min_{dfp}, max_{dfp}]$, onde min e max são os valores mínimos e máximos registrados no histórico para as demandas medidas nos respectivos postos.

Já na modalidade verde, a variável de decisão diz respeito à demanda única que deve ser contratada, e o espaço de busca é representado pelo intervalo $[min_d, max_d]$, onde min_d e max_d são, respectivamente, a menor e a maior demanda medida no histórico de registros, seja em posto ponta ou fora de ponta.

Posto isso, originada da Equação 2.3, a função

$$Fv(D) = TUSD \cdot \sum_{i=1}^{n} D + 3 \cdot (U_p + U_{fp}) \tag{4.1}$$

descreve a função objetivo da modalidade verde para os n meses analisados no histórico.

Semelhantemente, originada da Equação 2.6, a função

$$Fa(D) = TUSD_p \cdot \sum_{i=1}^{n} (D_0 + 3 \cdot U_p) + TUSD_{fp} \cdot \sum_{i=1}^{n} (D_1 + 3 \cdot U_{fp}) \tag{4.2}$$

representa a função objetivo da modalidade azul para os n meses analisados no histórico.

Tendo as funções objetivo devidamente modeladas, o próximo passo é implementá-las em *Python* para que possam ser utilizadas pelos algoritmos de busca global nas minimizações que definem o aspecto da otimização. Sendo assim, inicialmente foram implementadas as funções no Trecho de Código 1 .

Trecho de Código 1 – Funções objetivo

```
1  def green_objective_func(self, x, current = False) -> .0:
2      exceeded_value = self.calc_exceeded_demand_values(GREEN, x, current)
3      value = len(self.hist) * x[0] + exceeded_value
4      return self.green.na_tusd_in_reais_per_kw * value + \
5             self.consumption_cost_on_green
6
7  def blue_objective_func(self, x, current = False) -> .0:
8      exceeded_values = self.calc_exceeded_demand_values(BLUE, x, current)
9      value_i = len(self.hist) * x[0] + exceeded_values[0]
10     value_ii = len(self.hist) * x[1] + exceeded_values[1]
11
12     return (self.blue.peak_tusd_in_reais_per_kw * value_i) + \
13            (self.blue.off_peak_tusd_in_reais_per_kw * value_ii) + \
14            self.consumption_cost_on_blue
```

Fonte: Autor.

A variável x que as funções recebem como parâmetro, na verdade, se trata de um vetor que vai conter uma posição na modalidade verde e duas na azul. Esse vetor será oferecido pelos algoritmos de busca global durante sua utilização. A função *calc_exceeded_demand_values()* retorna o valor de ultrapassagem conforme as equações descritas nas seções 2.1.1 e 2.1.2.

As funções objetivo retornam o valor total das contas de energia do período analisado. Dessa forma, conforme a Equação 2.1, devem conter a componente do valor do consumo representada no código pelos itens *consumption_cost_on_green* e *consumption_cost_on_blue*, os quais foram calculados previamente conforme a Equação 2.2 para todo o período analisado.

É importante notar nas Linhas 4, 12 e 13 do Trecho de Código 1 a aplicação das tarifas $TUSD$. Essas tarifas são componentes dos objetos *green* e *blue*, os quais são únicos

globalmente. Ou seja, todas as análises utilizam as mesmas tarifas, que foram obtidas diretamente do site da Companhia Paranaense de Energia (COPEL)[4].

Os métodos *green_objective_func()* e *blue_objective_func()*, funções objetivo para a modalidade verde e azul, respectivamente, foram implementados usando a estratégia de utilizar as operações da biblioteca *Pandas*, visto que o histórico se trata de um *Dataframe*, um tipo interno dessa mesma biblioteca. Essa estratégia foi seguida visando uma implementação mais simples e legível e na expectativa de uma boa performance.

Entretanto, já nas primeiras rodadas de testes, observou-se um tempo de execução elevado da busca global. Sendo assim, a fim de comparação, foram implementadas outras duas funções objetivo, agora seguindo uma estratégia diretamente iterativa em vez de utilizar as operações da biblioteca *Pandas*. O Trecho de Código 2 mostra tal implementação.

Trecho de Código 2 – Funções objetivo usando iterações

```
def green_objective_func_it(self, demands) -> .0 :
    demand_value = 0
    demand = round(demands[0])
    for bill in self.hist.values:
        exceeded_value = self._exceeded_demand(bill[2], demand) + \
                         self._exceeded_demand(bill[3], demand)
        demand_value += demand + 3 * exceeded_value

    return float(self.green.na_tusd_in_reais_per_kw) * demand_value + \
           self.consumption_cost_on_green

def blue_objective_func_it(self, demands) -> .0 :
    peak_demand_value = 0
    off_peak_demand_value = 0
    peak_demand, off_peak_demand = round(demands[0]), round(demands[1])
    for bill in self.hist.values:
        peak_exceeded = self._exceeded_demand(bill[2], peak_demand)
        peak_demand_value += peak_demand + 3 * peak_exceeded

        off_peak_exceeded=self._exceeded_demand(bill[3],off_peak_demand)
        off_peak_demand_value += off_peak_demand + 3 * off_peak_exceeded

    return (float(self.blue.peak_tusd_in_reais_per_kw) * \
           peak_demand_value) + \
           (float(self.blue.off_peak_tusd_in_reais_per_kw) * \
           off_peak_demand_value) + \
           self.consumption_cost_on_blue
```

Fonte: Autor.

O parâmetro *demands* tem a mesma função que o parâmetro x na função original,

[4] Tarifas: <https://www.copel.com/site/copel-distribuicao/tarifas-de-energia-eletrica/>

mudou-se apenas a nomenclatura para maior clareza. O método *_exceeded_demand()* é uma versão iterativa do *calc_exceeded_demand_values()* da versão original. Essa abordagem alternativa percorre cada linha do histórico, acessando as células contendo as demandas medidas e acumulando os valores calculados nas variáveis internas.

Além disso, como o cálculo é feito individualmente por fatura, a obtenção da ultrapassagem pode ser simplificada. Dessa forma, conforme mostrado no Trecho de Código 3, foi implementado o método *_exceeded_demand()* que realiza um cálculo mais simples apenas retornando a diferença, caso seja positiva, entre a demanda medida e a contratada.

Trecho de Código 3 – Método de cálculo de ultrapassagem versão iterativo

```
1 def _exceeded_demand (self, meassured_demand, contracted_demand):
2         return max(.0, meassured_demand - contracted_demand)
```

Fonte: Autor.

4.6 Análise de esforço computacional

A primeira análise desta prova de conceito consiste em um *benchmark* comparativo do tempo de execução entre as abordagens implementadas. Dessa forma, nessa etapa, serão comparados os tempos médios de execução da recomendação por percentis, dos algoritmos globais, em ambas as versões iterativa e não iterativa apresentadas na Seção 4.5, bem como a execução do algoritmo completo.

O algoritmo completo realiza uma varredura de todas as combinações possíveis dentro do espaço de busca, garantindo que todas as soluções sejam avaliadas. Para isso, ele utiliza as funções objetivo mostradas no Trecho 2 em seus laços de repetição, conforme mostram as Linhas 3 e 11 do Trecho de Código 4.

Trecho de Código 4 – Laços de repetição do algoritmo completo

```
1  green_v, green_d = .0, 0
2  for v in range(int(round(self.g_lb)), int(round(self.g_ub))+1):
3      value = Dominio().green_objective_func_it([v])
4      if value <= green_v or green_v == .0:
5          green_v, green_d = value, v
6
7  blue_v = .0
8  blue_d = (0, 0)
9  for v_i in range(int(round(self.p_lb)), int(round(self.p_ub))+1):
10     for v_ii in range(int(self.o_lb), int(self.o_ub)+1):
11         value = Dominio().blue_objective_func_it([v_i, v_ii])
12         if value <= blue_v or blue_v == .0:
13             blue_v = value
14             blue_d = (v_i, v_ii)
```

Fonte: Autor.

É possível observar que, na modalidade azul, o comportamento do algoritmo é quadrático, o que significa que o tempo de execução cresce proporcionalmente ao quadrado do número de combinações, tornando-o mais custoso em termos de tempo de processamento. Além disso, é possível observar a definição dos espaços de busca nas Linhas 2, 9 e 10 com as variáveis definidas previamente na rotina conforme o explicado na Seção 4.5.

Posto isso, a presente análise de esforço computacional é fundamental, pois fornece uma visão clara de como cada método se comporta em termos de tempo de execução, o que é crucial para a escolha da estratégia mais adequada para a otimização dos contratos de energia elétrica.

Para tanto, foram utilizados dados de faturas fornecidas por 130 unidades consumidoras de diferentes universidades federais. É importante salientar que, para o MEPA realizar a geração de uma recomendação, é requisitado, no mínimo, a informação de 6 faturas para análise, sendo 12 o número ideal.

Dessa forma, esse *benchmark* foi realizado para todos os cenários de quantidade de faturas informadas. Ou seja, cada unidade consumidora possui ao menos 12 faturas informadas em ordem cronológica, e todos os algoritmos serão executados para cenários contendo de 6 a 12 faturas. A Tabela 1 mostra, em milissegundos, os tempos médios de execução obtidos.

Tabela 1 – Tempo médio de execução em milissegundos

Histórico	**Percentil**	**PSO**	**GA**	**PSO-it**	**GA-it**	**Completo**
6 meses	7,2	3.460	1.500	130	80	2.710
7 meses	7,4	7.070	1.870	210	100	3.030
8 meses	7,4	6.990	1.860	220	110	3.270
9 meses	7,5	3.820	1.550	170	90	3.510
10 meses	7,3	6.930	1.830	250	120	3.810
11 meses	7,4	6.610	1.860	250	120	4.100
12 meses	7,5	4.010	1.620	200	110	4.340

Fonte: Autor.

É possível observar que as versões iterativas apresentam uma vantagem significativa em termos de tempo de execução em comparação com as versões não iterativas. Em vista disso, optou-se por selecionar as versões iterativas como a abordagem inicial para a busca global nas análises subsequentes.

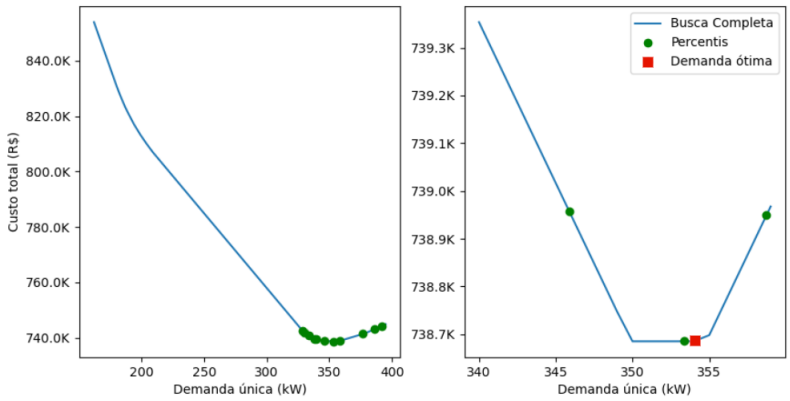
4.7 Análise de perfis

Para uma melhor compreensão dos comportamentos observados nos algoritmos executados e estudados nesta prova de conceito, foi realizado um estudo mais aprofundado

dos perfis dos resultados obtidos, com o objetivo de analisar em maior detalhe as nuances de cada abordagem e identificar potenciais padrões no problema de otimização.

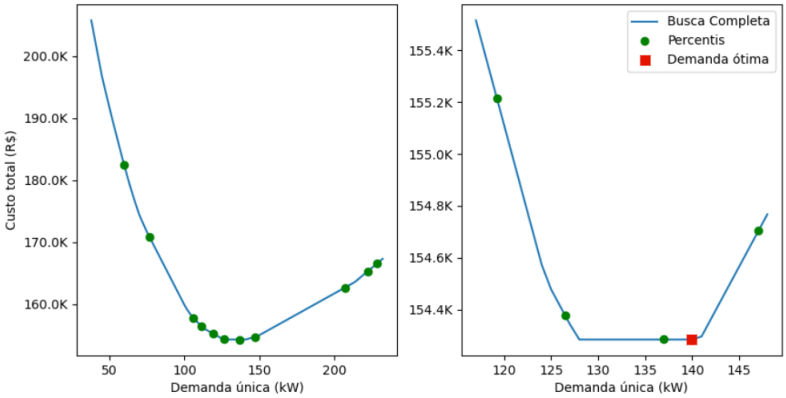
Num primeiro momento, foi registrado o perfil descrito pelo algoritmo completo dentro dos espaços de busca definidos para as modalidades verde e azul. Assim, foi possível observar o comportamento gerado pelo algoritmo do percentil, relacionando os percentis analisados com os custos gerados por eles. Os gráficos ilustrados nas figuras 8 e 9 exemplificam essa visualização para a modalidade verde em duas unidades consumidoras da Universidade Federal do Paraná (UFPR).

Figura 8 – Curva de valor por demanda UC 1936905 da UFPR em modalidade verde



Fonte: Autor.

Figura 9 – Curva de valor por demanda UC 102704724 da UFPR em modalidade verde



Fonte: Autor.

Os gráficos apresentam, em azul, a variação do valor total do contrato de energia elétrica ao longo de 12 meses, em função da demanda contratada. Conforme a demanda aumenta, o valor das faturas também se ajusta, criando uma curva que reflete os custos relacionados à contratação de energia.

Sobre essa curva, os pontos destacados em verde indicam as recomendações consideradas pelo algoritmo de percentil. Cada um desses pontos representa um dos 11 percentis analisados pelo algoritmo, ou seja, diferentes valores de demanda que foram destacados como possíveis sugestões com base no histórico de consumo.

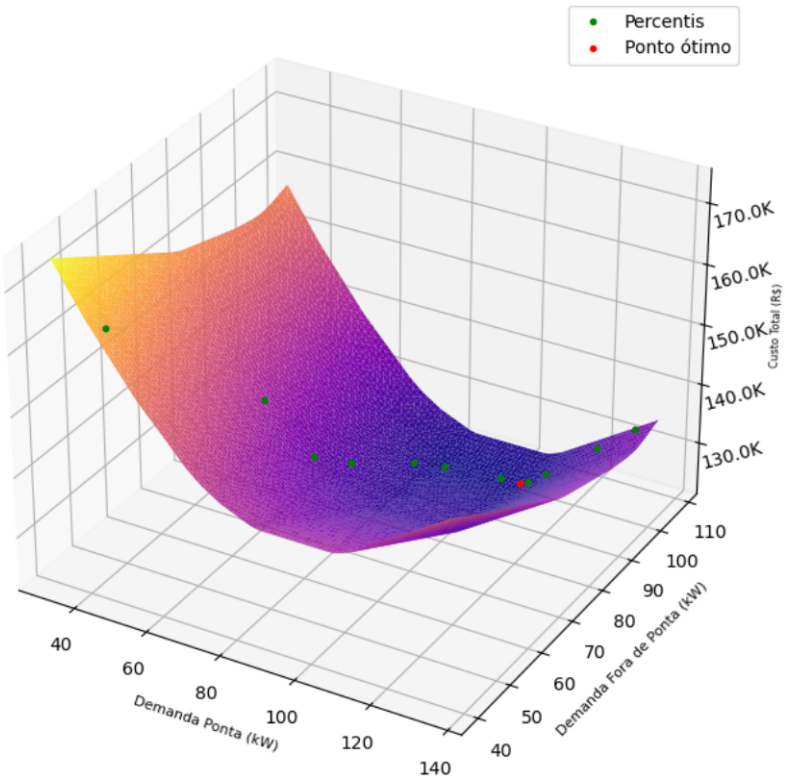
Pode-se notar, nas figuras 8 e 9, que o cenário descrito para a UC 1936905 apresenta um espaço de busca consideravelmente maior do que o descrito para a UC 102704724, os quais são representados, respectivamente, pelos intervalos $[162, 395]$ e $[38, 232]$. Isso indica que a variação da demanda medida no período analisado foi maior na primeira UC.

Nota-se também que os percentis avaliados pelo algoritmo de recomendação foram bastante eficientes em localizar os valores de demanda que mais se aproximam daqueles que minimizam o valor total.

Em ambos os cenários, o gráfico à direita apresenta uma ampliação da visualização no vale da curva. Dessa forma, é possível notar um comportamento constante quando o valor mais otimizado para o custo total é atingido. Isso mostra que pode ocorrer de o valor ótimo ser atingido por um intervalo de demandas cuja diferença influencia de maneira desprezível no custo final. No cenário ilustrado na Figura 9, por exemplo, o menor custo total pode ser atingido por qualquer demanda contida no intervalo $[128, 140]$.

Para a modalidade azul, o mesmo comportamento pode ser observado. A Figura 10, mostra as posições dos percentis analisados pelo algoritmo de recomendação na superfície gerada pela varredura completa do cenário descrito pelo histórico de 12 meses da UC 33261296 da UFPR.

Figura 10 – Superfície de valor por demanda UC 33261296 da UFPR em modalidade azul



Fonte: Autor.

Nesse exemplo, foi detectado que o custo total do período analisado é o mesmo para todas as combinações de demandas ponta e fora de ponta quando pertencentes aos intervalos $[87, 105]$ e $[99, 100]$ respectivamente.

Posto isso, é sabido que obviamente há uma vantagem estratégica ao se contratar uma demanda maior sem prejuízos no custo total. Essa abordagem permite que a unidade consumidora tenha uma margem adicional de segurança no consumo de energia sem incorrer em custos extras, garantindo flexibilidade operacional e evitando penalidades por ultrapassagem de demanda contratada.

Sendo assim, como o custo mínimo pode ser atingidos por diferentes valores de demanda, deve-se entender a demanda ótima, seja única ou não, como sendo a maior demanda do intervalo que gera o menor custo total para o período analisado.

4.8 Análise de acurácia

Nesta seção, será realizada a análise de acurácia dos algoritmos de recomendação estudados. O objetivo é avaliar a exatidão com que o algoritmo baseado em percentis e as versões iterativas do PSO e do GA conseguem identificar as demandas que minimizam os custos totais, tendo os resultados obtidos pelo algoritmo completo como referência conforme o entendimento sobre as demandas consideradas ótimas definido na Seção 4.7.

É importante salientar que, ao longo das experimentações, conforme constatado na Tabela 1, as versões iterativas das abordagens de busca global não apresentaram um tempo de execução melhor que a abordagem atualmente empregada. Tais resultados já eram esperados devido à diferença de complexidade entre os algoritmos comparados. Em termos de tempo de execução, os algoritmos de busca global tendem a ser mais lentos pois realizam iterações mais amplas no espaço de busca, ao contrário da abordagem atual, que acaba se beneficiando ao tratar o problema de maneira mais direta.

Além disso, a análise de esforço computacional apresentada na Seção 4.6 foi conduzida de forma preliminar, sem ajustes ou otimizações específicas para maximizar a performance dos algoritmos de busca global, sendo focada apenas no registro dos tempos de execução para fornecer uma base comparativa inicial. Posto isso, algumas tentativas de melhorias foram realizadas para alcançar níveis aceitáveis de performance.

Buscou-se, assim, reduzir os tempos de execução da busca global mantendo níveis de acurácia maiores que os obtidos pelo algoritmo atual. Para tanto, diferentes execuções dos algoritmos de busca global foram realizadas, modificando seus coeficientes e parâmetros com o intuito de encontrar os valores que melhor otimizassem as rotinas. Focou-se principalmente no coeficiente populacional, por ser utilizado tanto pelo PSO quanto pelo GA.

Tal coeficiente determina a quantidade de partículas no PSO e a quantidade de genes no GA, sendo bastante influente na resolução do problema de otimização. Testaram-se alguns valores para esse coeficiente, e observou-se que o valor 10 apresentou os melhores resultados. Além disso, foi implementada a parada precoce tanto no GA quanto no PSO, definindo-se que, caso as iterações não apresentassem melhorias em 20% das tentativas consecutivas, o algoritmo adotará a solução encontrada como definitiva.

Para além disso, ao analisar os cenários em relação ao tempo de execução, conforme apresentado na Tabela 1 para os tempos do PSO e do GA iterativos, pode-se observar que, utilizando 6, 9 ou 12 meses de histórico, os tempos de execução foram inferiores aos demais. Isso pode indicar que os algoritmos estão ficando limitados em um mínimo local não satisfatória no que tange à demanda recomendada.

Ao relacionar isso com as observações feitas na Seção 4.7, tem-se que os intervalos de demandas que geram o o mesmo custo total, vide gráficos das figuras 8, 9 e 10, foram observados no cenário com histórico de 12 meses, que apresenta um tempo de execução menor.

Posto isso, foi realizado um estudo complementar a fim de entender esse comportamento que consistiu na contagem dos casos em que os intervalos de custo constante acontecem e calcular a porcentagem de unidades consumidoras que incorrem nesse comportamento para cada um dos cenários. A Tabela 2 mostra os resultados obtidos.

Tabela 2 – Taxas de aparição dos intervalos de custo constante

	Meses de histórico						
Modalidade tarifária	**6**	**7**	**8**	**9**	**10**	**11**	**12**
Verde	75%	0%	0%	68%	0%	0%	72%
Azul	90%	0%	0%	85%	0%	0%	86%

Fonte: Autor.

De acordo com os dados expostos pela Tabela 2, percebe-se que os intervalos de custo constante ocorreram justamente nos cenários com menores tempo de execução. Isso indica que a utilização de 6, 9 e 12 meses de histórico na análise pode trazer uma maior complexidade na recomendação de uma demanda ótima.

Posto isso, é válido ressaltar que as funções objetivo utilizadas pelos algoritmos de busca global, são aquelas apresentadas na Seção 4.5, vide código do Trecho 2. Sendo assim, foi proposta uma intervenção nessas funções a fim de se obter melhores resultados com a busca global nos cenários citados. O Trecho 5 mostra o código proposto.

As diferenças entre os códigos dos trechos 2 e 5 são sutis. Nas Linhas 10 e 27, foram apenas subtraídas do resultado final (custo total) as demandas analisadas. Essa estratégia foi proposta para induzir os algoritmos de busca global a selecionarem demandas maiores.

Isso seria plausível, já que tais algoritmos buscam minimizar os resultados das funções.

Trecho de Código 5 – Funções objetivo ajustadas

```
def green_objective_func_it(self, demands) -> .0 :
    demand_value = 0
    demand = round(demands[0])
    for bill in self.hist.values:
        exceeded_value = self._exceeded_demand(bill[2], demand) + \
                         self._exceeded_demand(bill[3], demand)
        demand_value += demand + 3 * exceeded_value

    return float(self.green.na_tusd_in_reais_per_kw) * demand_value + \
           self.consumption_cost_on_green - demand[0]

def blue_objective_func_it(self, demands) -> .0 :
    peak_demand_value = 0
    off_peak_demand_value = 0
    peak_demand, off_peak_demand = round(demands[0]), round(demands[1])
    for bill in self.hist.values:
        peak_exceeded = self._exceeded_demand(bill[2], peak_demand)
        peak_demand_value += peak_demand + 3 * peak_exceeded

        off_peak_exceeded=self._exceeded_demand(bill[3],off_peak_demand)
        off_peak_demand_value += off_peak_demand + 3 * off_peak_exceeded

    return (float(self.blue.peak_tusd_in_reais_per_kw) * \
           peak_demand_value) + \
           (float(self.blue.off_peak_tusd_in_reais_per_kw) * \
           off_peak_demand_value) + \
           self.consumption_cost_on_blue - demand[0] - demand[1]
```

Fonte: Autor.

Assim, ao subtrair as demandas, os algoritmos tenderiam a selecionar as maiores em casos de intervalo de custo constante. É importante deixar claro que essa estratégia insere a necessidade de se somar os valores de demanda novamente ao final do processamento.

Sendo assim, as análises de acurácia consistiram em comparar os resultados obtidos pelos algoritmos estudados com os resultados fornecidos pelo algoritmo completo. Para as demandas recomendadas, foi utilizada a métrica $nRMSE$, *normalized root mean square error* em inglês, uma técnica amplamente utilizada para avaliar a qualidade de modelos estatísticos (OTTO, 2019).

A fórmula da $nRMSE$ é derivada a partir da métrica $RMSE$, que mede o erro quadrático médio, e a normaliza. Tal normalização será feita dividindo o resultado da $RMSE$ pela média dos valores obtidos pelo algoritmo completo.

A fórmula geral do $RMSE$ é:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{4.3}$$

onde y_i são os valores mais otimizados, $\hat{y}_i$ são os valores recomendados e n é o número total de unidades consumidoras.

Para normalizar o $RMSE$, utiliza-se a seguinte fórmula para o $nRMSE$:

$$nRMSE = \frac{\text{RMSE}}{\overline{y}} \tag{4.4}$$

onde $\overline{y}$ é a média dos valores mais otimizados.

Além das demandas recomendadas, as análises de acurácia também avaliaram o potencial de chegar no custo mais otimizado pela recomendação independentemente da demanda que a compõe.

Para isso, foi utilizada a métrica $MAPE$ que calcula o erro percentual absoluto médio entre o custo estimado pela recomendação e o custo mais otimizado. A $MAPE$ é amplamente utilizada na avaliação de modelos de previsão por fornecer uma medida intuitiva da precisão em termos de erro percentual (MARTINS, 2005).

A fórmula utilizada foi:

$$MAPE = \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \tag{4.5}$$

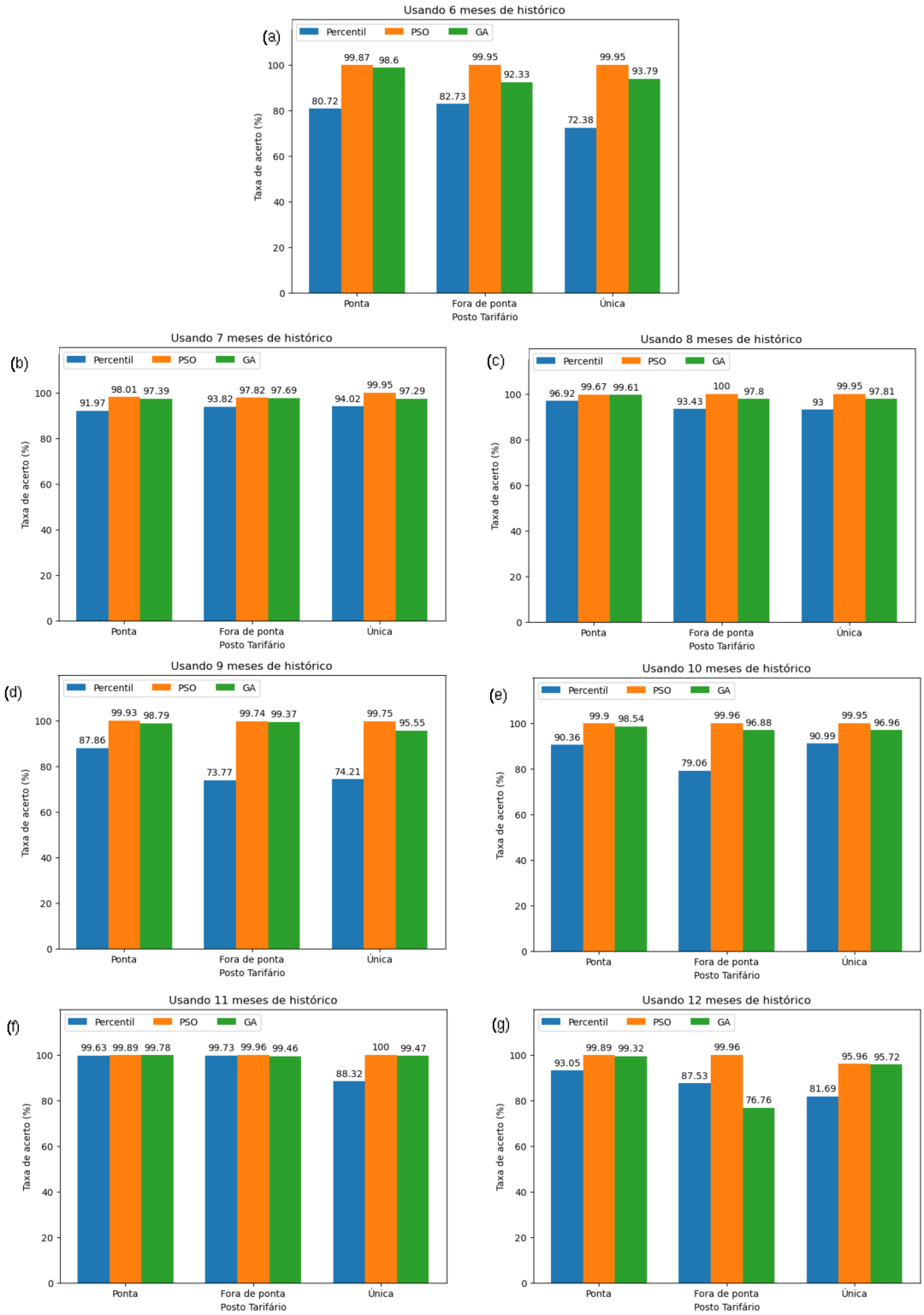
onde y_i representa os valores de custo obtidos pelo algoritmo completo, $\hat{y}_i$ são os valores preditos, n é o número total de unidades consumidoras.

Considerando a natureza estocástica dos algoritmos de busca global, ou seja, que tais abordagens podem produzir resultados diferentes a cada execução de uma mesma análise, suas rotinas foram executadas múltiplas vezes para cada unidade consumidora. Com isso, para determinar o resultado da análise, foi utilizada a moda dessas execuções, ou seja, o valor mais frequentemente obtido pelos algoritmos.

Os cenários analisados seguiram a mesma abordagem utilizada na análise do esforço computacional, vide Seção 4.6. Para isso, foram utilizados históricos de consumo de 130 unidades consumidoras diferentes, selecionando, cronologicamente, de 6 até 12 meses.

Os gráficos apresentados na Figura 11 mostram as taxas de acerto obtidas por cada abordagem em cada posto tarifário. O gráfico (a) refere-se ao cenário com histórico de 6 meses, o gráfico (b) ao de 7 meses, e assim sucessivamente até o cenário com 12 meses, representado no gráfico (g).

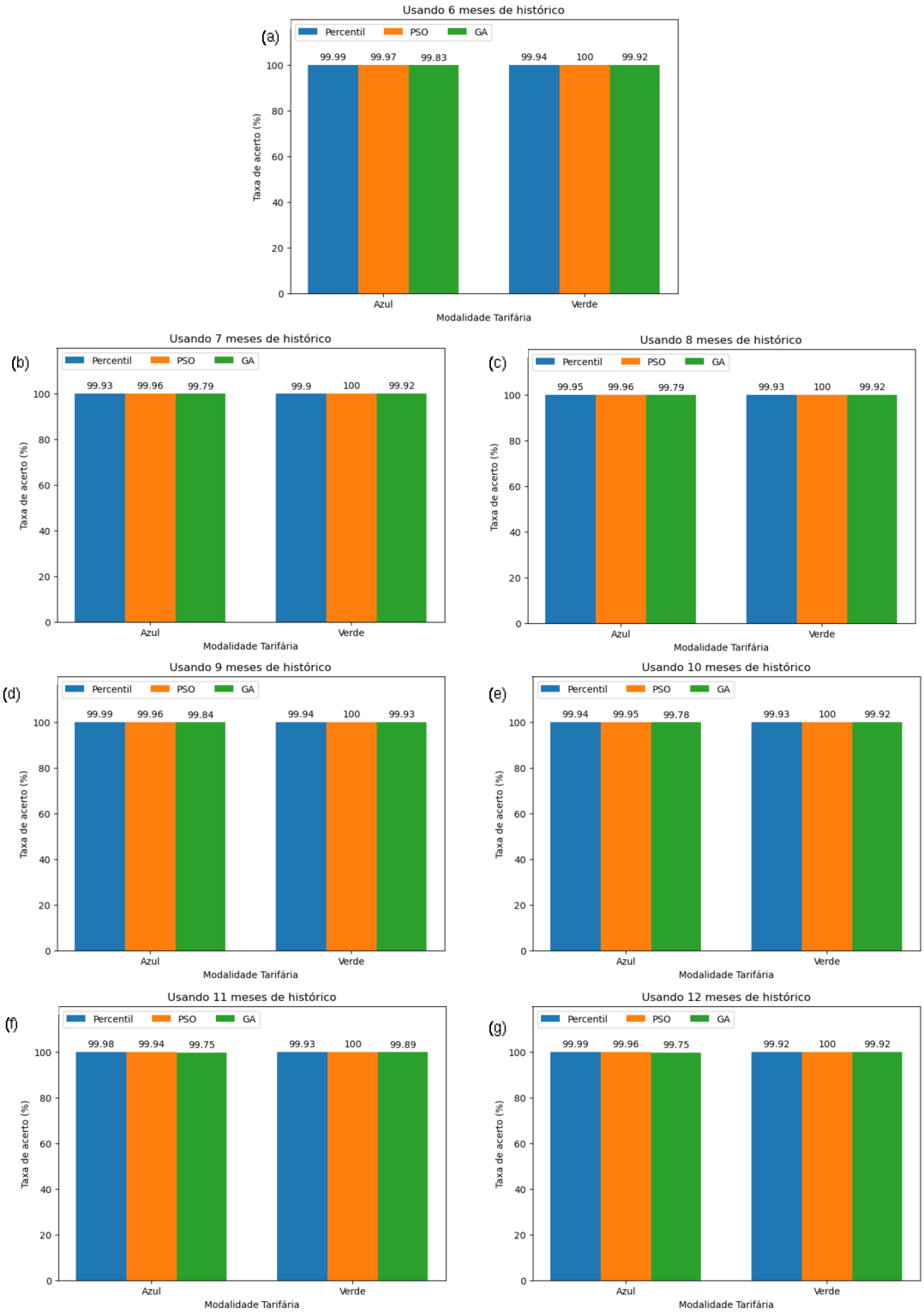
Figura 11 – Taxa de acerto das demandas recomendadas por cada abordagem



Fonte: Autor.

Analisando os gráficos, pode-se observar uma alta taxa de acurácia em recomendar as demandas mais otimizadas por todos os algoritmos. É notável a eficiência do algoritmo baseado em percentis mas, na grande maioria dos cenários, os de busca global levam vantagem.

Figura 12 – Taxa de acerto dos custos obtidos para cada abordagem.



Fonte: Autor.

Agora os gráficos da Figura 12 apresentam uma outra visão. Se trata da acurácia em chegar no custo mais otimizado pela recomendação independentemente da demanda que a compõe. Observa-se que o mesmo comportamento tido nos gráficos da Figura 11 é replicado, com altas taxas de sucesso por todas as abordagens.

Por fim a Tabela 3 mostra o ganho tido em tempo de execução com as modificações feitas em comparação com o tempo do percentil. Dessa forma, pode-se notar que os algoritmos de busca global podem ser até mais rápidos que o atual e, ao mesmo tempo, apresentar uma vantagem de acurácia.

Tabela 3 – Tempos de execução obtidos pelos algoritmos de busca global em milissegundos

	Histórico						
	6 meses	**7 meses**	**8 meses**	**9 meses**	**10 meses**	**11 meses**	**12 meses**
Percentil	7,2	7,4	7,4	7,5	7,3	7,4	7,5
PSO-it	4,75	5,00	5,31	5,90	6,20	6,34	6,79
GA-it	3,98	3,96	4,31	4,95	4,58	5,04	5,39

Fonte: Autor.

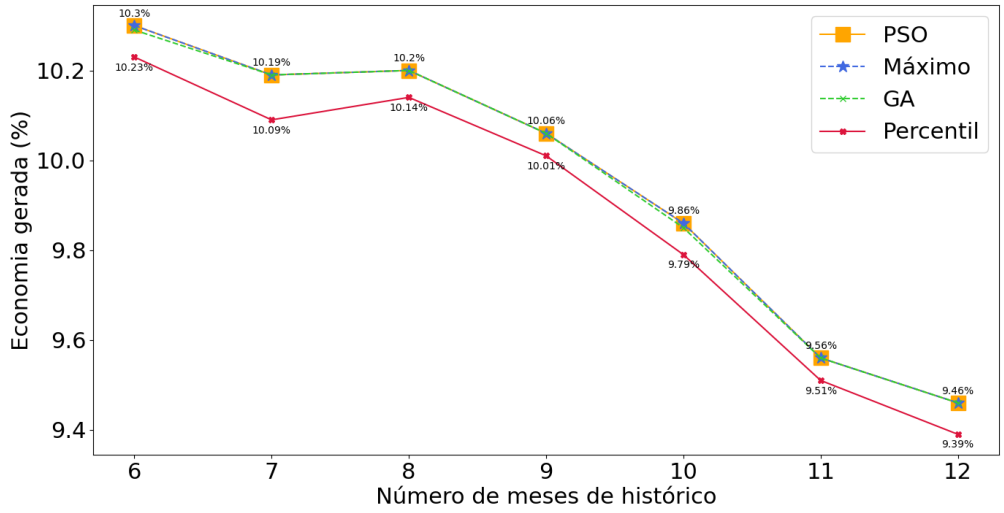
4.9 Análise de economia

Nesta seção, será apresentada a análise de economia gerada por cada uma das abordagens estudadas. O objetivo é comparar o desempenho econômico de cada método de otimização em termos de redução de custos, levando em consideração os diferentes cenários tarifários e históricos de consumo.

Para tanto, o processo consistiu em gerar recomendações pelos algoritmos, calcular o valor real contratado no período analisado e, para cada recomendação gerada, obter a taxa de economia a partir da diferença entre os valores obtidos pelas recomendações e os valores reais calculados. Além disso, também foi executado o algoritmo completo para identificar qual seria o máximo de economia possível de se atingir.

As execuções seguiram a mesma estrutura e utilizaram o mesmo conjunto de dados das análises anteriores. Ou seja, foram utilizados dados de 130 unidades consumidoras com históricos de 6 a 12 meses de forma cronológica. Além disso, para essa análise, foi simulado o ciclo completo da recomendação, no qual são analisadas as recomendações entre as diferentes modalidades tarifárias. O gráfico da Figura 13 apresenta os resultados obtidos.

Figura 13 – Taxa de economia gerada por cada abordagem



Fonte: Autor.

Com o apresentado no gráfico da Figura 13 é possível notar que, em termos de economia, o algoritmo baseado em percentis apresenta uma eficiência muito alta, chegando muito próximo dos valores mais econômicos possíveis de se obter. Ainda assim, também é possível notar uma ligeira vantagem da busca global que praticamente atingem a máxima economia possível.

Porém, ao avaliar a diferença de economia gerada, nota-se que a maior ocorreu no cenário referente ao uso de 7 meses de histórico, no qual a busca global apresenta apenas 0,1% a mais de economia em relação ao método baseado em percentis. A fim de se ter uma ideia do montante economizado a mais, a Tabela 4 apresenta a média em reais para cada cenário.

Tabela 4 – Média de economia a mais gerada pela busca global para cada cenário

	Meses de histórico						
Abordagem	**6**	**7**	**8**	**9**	**10**	**11**	**12**
PSO	R$ 38,43	R$ 126,90	R$ 107.48	R$ 48,17	R$ 165.43	R$ 203.20	R$ 295.61
GA	R$ 39,37	R$ 125,97	R$ 106.99	R$ 48,15	R$ 162.87	R$ 202.83	R$ 293.57

Fonte: Autor.

Ante ao exposto nessa prova de conceito, é possível concluir que as abordagens estudadas apresentam uma margem muito pequena na economia gerada. O algoritmo baseado em percentis demonstra ser altamente eficiente em gerar boas recomendações e possui um ótimo tempo de execução devido à sua natureza determinística.

Com a busca global, entretanto, há um ganho sutil de economia devido à recomendação mais otimizada. O destaque dessa abordagem, porém, está no melhor desempenho

ao lidar com os intervalos de custo constante, possibilitando a recomendação de demandas maiores e provendo, assim, uma vantagem estratégica significativa aos usuários.

A fim de observar quantitativamente essa vantagem, a Tabela 5 mostra a média de ganho em valor de demanda nas abordagens de busca global nos cenários mais complexos que apresentam os intervalos de custo constante.

Tabela 5 – Ganho médio em demanda recomendada pela busca global

	Meses de histórico		
Demanda recomendada	**6**	**9**	**12**
Ponta	4 kW	7 kW	1 kW
Fora de ponta	6 kW	4 kW	4 kW
Única	16 kW	11 kW	10 kW

Fonte: Autor.

Sendo assim, para fins de implementação definitiva no MEPA, é possível observar a viabilidade da aplicação das abordagens baseadas em busca global. Dessa forma, propõe-se a inclusão das mesmas como uma alternativa. Assim, pode-se avaliar qual das metodologias atende melhor às necessidades conforme o contexto, proporcionando maior flexibilidade e abrindo espaço para novas funcionalidades na plataforma.

5 Implementação

O presente capítulo apresenta o processo de implementação das novas abordagens de recomendação no MEPA. Ao integrar os algoritmos bioinspirados PSO e GA ao já existente módulo de recomendações, a implementação objetiva oferecer aos mantenedores do sistema uma maior flexibilidade na escolha da abordagem mais adequada, permitindo a seleção do algoritmo por meio de variáveis de ambiente. Essa evolução possibilita ao MEPA fornecer recomendações mais otimizadas, preservando a eficiência e adaptabilidade necessárias para diferentes cenários de uso. Além disso, foi projetada para facilitar a adição de novos algoritmos bioinspirados, caso seja necessário.

5.1 Requisitos

Para iniciar a implementação, foi necessário levantar alguns requisitos, visto que o MEPA é uma plataforma em desenvolvimento e é preciso garantir que certas regras de negócio sejam seguidas tanto pela abordagem já utilizada quanto pelas novas, baseadas em busca global. Dessa forma, os seguintes requisitos foram levantados:

- **Unidades consumidoras dos subgrupos A2 e A3 são faturadas somente em modalidade azul:** nem todos os subgrupos definidos têm a flexibilidade de escolher a modalidade tarifária de seus contratos. Portanto, as unidades consumidoras dos subgrupos A2 e A3 devem receber recomendações apenas de contratos na modalidade azul;

- **A demanda mínima contratada deve ser de 30 kW:** as contratações estão sujeitas à restrição de que a demanda mínima contratada em qualquer posto de qualquer modalidade tarifária deve ser de 30 kW;

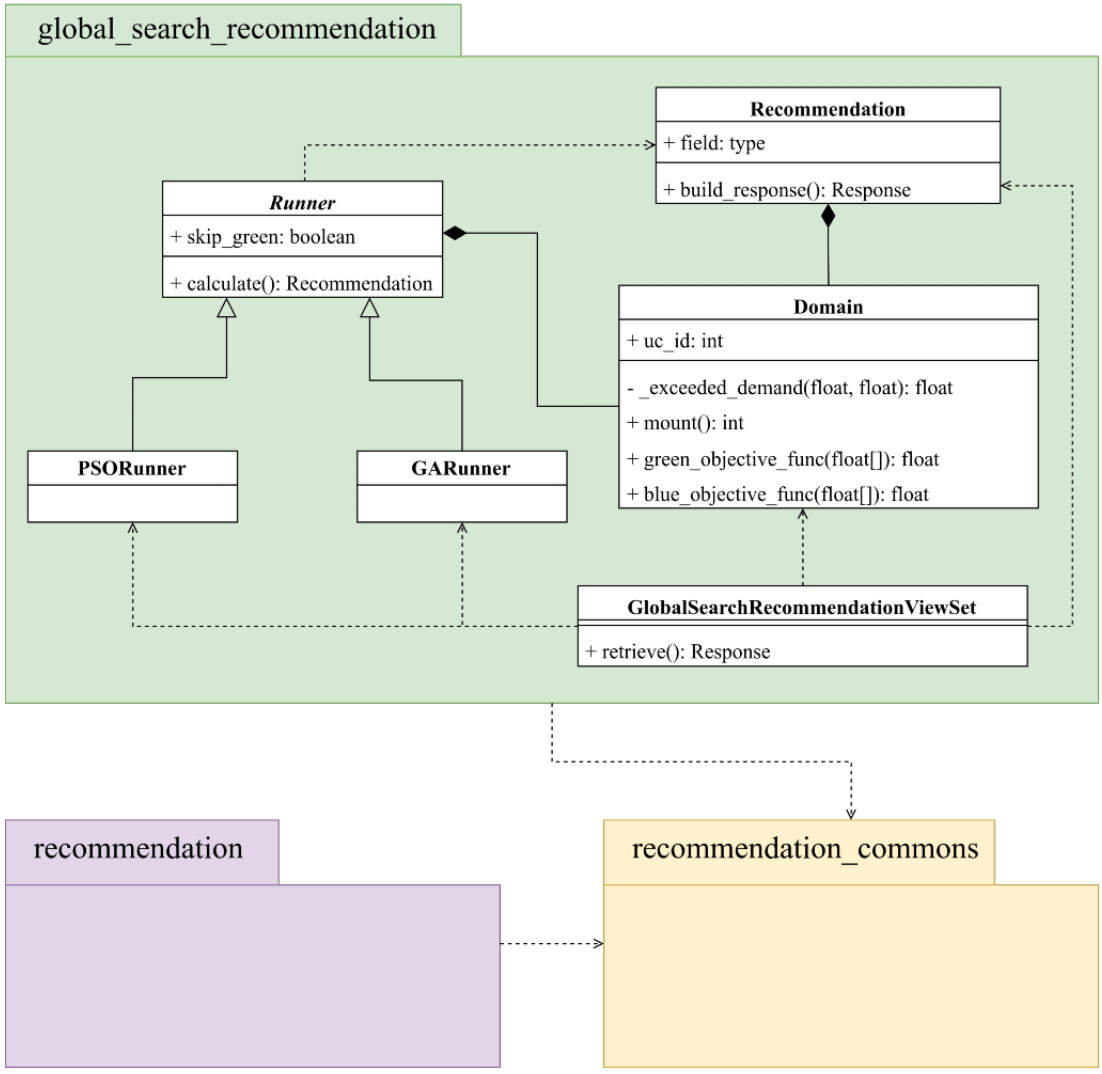
- **Novas regras para a geração de energia:** no primeiro semestre de 2024, a ANEEL atualizou a regulamentação de contratos relacionados à geração de energia. Essas mudanças precisam ser refletidas nos cálculos de recomendação.

Em suma, esses requisitos devem ser contemplados na implementação, garantindo o alinhamento com as regras de negócio da plataforma, a qual deve seguir as normas e regulamentações vigentes referentes às contratações de energia elétrica.

5.2 Arquitetura

Na presente seção, será apresentada a arquitetura da solução implementada para integrar as novas abordagens de recomendação ao MEPA. O objetivo é ilustrar como se dá a interação entre os componentes e como os algoritmos bioinspirados foram incorporados ao sistema, mantendo a modularidade e flexibilidade necessárias para futuras expansões e adaptações. A arquitetura foi desenvolvida para suportar tanto a abordagem já existente quanto as novas. Para tanto, utilizando a *Unified Modeling Language* (UML), foi criado o diagrama de pacotes mostrado na Figura 14.

Figura 14 – Diagrama de pacotes da solução



Fonte: Autor.

O pacote *recommendation* refere-se ao módulo de recomendações baseado em percentis, abordagem utilizada atualmente no MEPA. Como algumas etapas das rotinas de recomendação seguem um modelo semelhante, foi possível reaproveitar diversos recursos comuns entre as abordagens. Assim, foi criado o pacote *recommendation_commons*, que

se tornou uma dependência compartilhada pelos módulos de recomendação.

No pacote *global_search_recommendation*, que representa o módulo de recomendação via busca global, foi implementada a classe *GlobalSearchRecommendationViewSet*. Esta classe possui o maior nível de dependência interna, visto que é responsável por receber a requisição e retornar a recomendação, gerenciando os recursos da recomendação por busca global.

A classe *Recommendation* é uma abstração para a recomendação em si. Ela contém todos os dados necessários e os resultados dos processamentos, e, por meio de seu método *build_response()*, constrói a resposta que será retornada pela aplicação.

A classe *Domain* é responsável por montar o domínio do problema. Ou seja, ela contém os dados necessários para realizar as modelagens iniciais, como os recursos pré-calculados que comporão o contexto da análise a ser feita. Além disso, ela oferece as funções objetivo utilizadas pelos algoritmos.

A classe abstrata *Runner* acopla os recursos necessários para a execução dos algoritmos de busca global. Ela define as regras de execução e oferece a interface do método *calculate()*, que deve ser implementado por qualquer classe que a herde. Esse é o caso das classes *PSORunner* e *GARunner*, que importam os algoritmos correspondentes da biblioteca *scikit-opt* e, em suas implementações do método *calculate()*, os executam e retornam a recomendação gerada.

5.3 Codificação

Nesta seção, serão apresentados trechos notáveis do código desenvolvido para a adição das novas abordagens de recomendação no MEPA. O objetivo é destacar as soluções implementadas para atender aos requisitos levantados previamente, demonstrando como cada um deles foi contemplado na codificação. Vale relembrar que o MEPA possui o código aberto e que a integra desta implementação está no repositório do projeto[1].

Começando com um exemplo da versão final da função objetivo, o Trecho de Código 6 apresenta a implementação referente à modalidade verde. Pode-se notar que essa é a versão iterativa discutida na seção 4.5 e que inclui o ajuste mencionado na seção 4.8.

Trecho de Código 6 – Versão final da função objetivo da modalidade verde

```
1 def green_objective_func(self, demands) -> .0:
2     demand_value = 0
3     demand = round(demands[0])
4     for bill in self.base_consumption_history:
5         peak_exceeded = self._exceeded_demand(bill[2], demand)
```

[1] MEPA: <https://gitlab.com/lappis-unb/projetos-energia/mec-energia/mec-energia-api>

```
6           off_peak_exceeded=self._exceeded_demand(bill[3], demand)
7           demand_value += demand + 3 * (peak_exceeded + off_peak_exceeded)
8
9       total_installed_power = self.consumer_unit.total_installed_power
10      tusd_g = self.green.power_generation_tusd_in_reais_per_kw
11      g_factor = StaticGetters.get_power_generation_factor(
12                  total_installed_power,
13                  tusd_g,
14                  demands)
15
16      power_generation_value = self.consumption_history_length * g_factor
17
18      return float(self.green.na_tusd_in_reais_per_kw) * \
19             demand_value + power_generation_factor + \
20             self.consumption_cost_on_green - demand
```

Fonte: Autor.

Nas linhas de 11 a 14, foi inserido o novo fator *g_factor*, que é adicionado ao custo final retornado pela função. Esse fator está relacionado à nova regra para geração de energia mencionada no levantamento de requisitos e é determinado pelo método *get_power_generation_factor*, conforme apresentado no código do Trecho 7.

Trecho de Código 7 – Método para obtenção do novo fator sobre geração de energia

```
1  @staticmethod
2  def get_power_generation_factor(installed_power_supply,tusd_g,demands):
3      if not installed_power_supply:
4          return 0
5
6      difference = installed_power_supply - decimal.Decimal(max(demands))
7      return float(max(0, difference * tusd_g))
```

Fonte: Autor.

O método do Trecho 7 pertence ao módulo *recommendation_commons* e tem como objetivo aplicar a tarifa $TUSD_g$, vide seção 2.1. De acordo com as diretrizes definidas pela ANEEL no primeiro semestre de 2024, essa tarifa deve incidir sobre a diferença entre a geração e a maior das demandas contratadas, desde que essa diferença seja positiva.

Passando agora para a classe *Runner*, o Trecho de código 8 apresenta o construtor que será herdado por suas subclasses. Os atributos *p_lbound* e *o_lbound*, definidos nas linhas 5 e 9, referem-se, respectivamente, aos limites inferiores dos espaços de busca dos postos ponta e fora de ponta.

Trecho de Código 8 – Método construtor da classe *Runner*

```
1  class Runner(ABC):
2      def __init__(self, domain: Domain) -> None:
3          self.domain = domain
4          self.history = domain.consumption_history
```

```
5           self.p_lbound = max(
6               self.history['peak_measured_demand_in_kw'].min(),
7               NEW_RESOLUTION_MINIMUM_DEMAND
8           )
9           self.o_lbound = max(
10              self.history['off_peak_measured_demand_in_kw'].min(),
11              NEW_RESOLUTION_MINIMUM_DEMAND
12          )
13          self.p_ubound = max(
14              self.history['peak_measured_demand_in_kw'].max(),
15              self.p_lbound + 1
16          )
17          self.o_ubound = max(
18              self.history['off_peak_measured_demand_in_kw'].max(),
19              self.o_lbound + 1
20          )
21          self.g_lb = min(self.p_lbound, self.o_lbound)
22          self.g_ub = max(self.p_ubound, self.o_ubound)
23          subgroup = self.domain.current_contract.subgroup
24          self.skip_green = subgroup in ['A2', 'A3']
```

Fonte: Autor.

Sendo assim, a constante *NEW_RESOLUTION_MINIMUM_DEMAND* é uma variável de ambiente configurável que armazena o valor mínimo de demanda contratada. Dessa forma, os mantenedores do sistema podem modificá-la de maneira simples, caso o valor mude ao longo do tempo.

Além disso, na linha 24, também é possível observar o atributo booleano *skip_green*, sendo definido como verdadeiro caso o subgrupo da unidade consumidora em questão seja A2 ou A3. Assim, o Trecho 9 mostra como ele deve ser utilizado na implementação do método *calculate()* pelas subclasses de *Runner*.

Trecho de Código 9 – Condicional para executar o algoritmo do PSO na modalidade verde

```
1 if not self.skip_green:
2     green = PSO(func=self.domain.green_objective_func, n_dim=1, pop=10,
3                 max_iter=100, w=0.8, c1=0.6, c2=0.6, lb=self.g_lb,
4                 ub=self.g_ub)
5     green.run()
```

Fonte: Autor.

Com isso, todos os requisitos levantados na seção 5.1 foram contemplados pela implementação. É importante salientar que todos eles também foram inseridos no módulo baseado em percentis.

Feita essa implementação, pode-se levantar alguns aspectos relevantes inerentes à codificação. É possível observar que o módulo de recomendação via busca global, mesmo após a implementação do módulo *recommendation_commons*, apresenta uma significativa

vantagem em termos de legibilidade e manutenção do código.

O módulo de recomendação baseado em percentis possui, ao todo, mais de 500 linhas de código e muito disso é código duplicado. Uma base de código mais limpa e concisa simplifica o entendimento do código para novos desenvolvedores e equipe de manutenção, além de reduzir a probabilidade de inserir erros.

6 Considerações Finais

O presente trabalho avaliou a viabilidade de implementar algoritmos de busca global no sistema de recomendações de contratos de energia do MEPA. Motivado pelo grande potencial desses algoritmos para resolver problemas de otimização de tais recomendação, o estudo concentrou-se no *particle swarm optimization* (PSO) e no algoritmo genético (GA).

Embora a prova de conceito realizada não tenha mostrado uma vantagem significativa dos algoritmos em gerar maior economia em comparação com o método atualmente utilizado, a busca global permitiu alcançar os resultados mais otimizados em tempos de execução menores. Ademais, esse esforço de análise foi fundamental para compreender melhor os cenários e os comportamentos das diferentes abordagens.

Por conta da metodologia aplicada, pode-se identificar os intervalos de custo constante existentes na recomendação em certos cenários. Nesses casos, os algoritmos de busca global demonstraram uma vantagem estratégica ao conseguir ampliar a margem dos valores de demanda contratada.

O método baseado em percentis confirmou-se como uma abordagem altamente eficiente, tanto em termos de tempo de execução quanto de economia gerada, aproximando-se dos resultados mais otimizados. Ainda assim, os resultados obtidos demonstram a viabilidade de implementação e o potencial que os algoritmos de busca global possuem para substituir a abordagem atualmente utilizada.

Para trabalhos futuros, uma possibilidade é concentrar-se no ajuste fino dos coeficientes do algoritmo do PSO. O PSO é sensível à escolha dos parâmetros, os quais influenciam diretamente sua eficácia na busca por soluções otimizadas. Realizar um ajuste detalhado desses coeficientes pode melhorar potencialmente sua performance e aumentar as taxas de acurácia.

Além disso, atualmente o MEPA permite que o usuário solicite repetidamente uma recomendação. Considerando que as mudanças de contrato são feitas geralmente de ano em ano e que os dados históricos são modificados comumente apenas mensalmente, é viável considerar o armazenamento das recomendações. Isso permitiria que fossem reutilizadas sem a necessidade de recalculá-las a cada solicitação.

Por fim, espera-se que os algoritmos integrados à plataforma possam ser de grande valia em contextos reais de recomendação de contratos mais otimizados e que ajudem a contribuir para a utilização cada vez mais eficiente dos recursos das universidades e instituições de ensino.

Referências

ANDIFES. *Painel ANDIFES de Informações Orçamentárias*. 2024. Accessed: 2024-08-09. Disponível em: <https://www.andifes.org.br/>. Citado na página 19.

ANEEL, A. N. de E. E. *RESOLUÇÃO NORMATIVA ANEEL Nº 1.000, DE 7 DE DEZEMBRO DE 2021*. 2021. Disponível em: <https://www2.aneel.gov.br/cedoc/ren20211000.pdf>. Citado 2 vezes nas páginas 22 e 23.

BINITHA, S.; SATHYA, S. S. et al. *A survey of bio inspired optimization algorithms*. [S.l.]: Citeseer, 2012. 137–151 p. Citado 3 vezes nas páginas 27, 28 e 30.

BORGES, C. d. S. M. Contingenciamento nas verbas da educação superior: uma análise empírica acerca dos grupos de despesas afetados no período de 2014 a 2021. 2024. Citado na página 19.

CLERC, M. The swarm and the queen: towards a deterministic and adaptive particle swarm optimization. In: IEEE. *Proceedings of the 1999 congress on evolutionary computation-CEC99 (Cat. No. 99TH8406)*. [S.l.], 1999. v. 3, p. 1951–1957. Citado na página 30.

ENGEL, G. I. Pesquisa-ação. *Educar em Revista*, SciELO Brasil, p. 181–191, 2000. Citado na página 33.

ENGELBRECHT, A. *Computational Intelligence: An Introduction*. Wiley, 2007. ISBN 9780470512500. Disponível em: <https://books.google.com.br/books?id=IZosIcgJMjUC>. Citado 3 vezes nas páginas 28, 30 e 32.

GÓMEZ, M. N. G. de. Metodologia da pesquisa no campo da ciência da informação. *Revista de Biblioteconomia de Brasília*, v. 24, n. 3, p. 333–346, 2000. Citado na página 33.

KAR, A. K. Bio inspired computing–a review of algorithms and scope of applications. *Expert Systems with Applications*, Elsevier, v. 59, p. 20–32, 2016. Citado na página 27.

KENNEDY, J.; EBERHART, R. Particle swarm optimization. In: IEEE. *Proceedings of ICNN'95-international conference on neural networks*. [S.l.], 1995. v. 4, p. 1942–1948. Citado na página 28.

LAPPIS-UNB. *MEPA*. 2024. Accessed: 2024-08-22. Disponível em: <https://lappis-unb.gitlab.io/projetos-energia/mec-energia/documentacao/>. Citado 3 vezes nas páginas 19, 22 e 25.

MARTINS, G. de A. *Estatística geral e aplicada*. Atlas, 2005. ISBN 9788522441723. Disponível em: <https://books.google.com.br/books?id=dIROAAAACAAJ>. Citado 2 vezes nas páginas 26 e 49.

MIRANDA, M. N. de. *Algoritmos Genéticos: Fundamentos e Aplicações*. 2007. Citado 3 vezes nas páginas 30, 31 e 32.

NOCEDAL, J.; WRIGHT, S. J. *Numerical optimization*. [S.l.]: Springer, 1999. Citado na página 27.

OTTO, S. A. *How to normalize the RMSE*. 2019. Disponível em: <https://www.marinedatascience.co/blog/2019/01/07/normalizing-the-rmse/>. Citado na página 48.

SANTOS, T. de S. *Contratação de energia elétrica por grandes consumidores no mercado cativo e no mercado livre*. 2020. Citado na página 22.

SHI, Y.; EBERHART, R. A modified particle swarm optimizer. In: IEEE. *1998 IEEE international conference on evolutionary computation proceedings. IEEE world congress on computational intelligence (Cat. No. 98TH8360)*. [S.l.], 1998. p. 69–73. Citado na página 29.

SMITH, D. R. *Structure and design of global search algorithms*. [S.l.], 1987. Citado na página 27.

SUN, J.; LAI, C.; WU, X. *Particle Swarm Optimisation: Classical and Quantum Perspectives*. CRC Press, 2016. (Chapman & Hall/CRC Numerical Analysis and Scientific Computing Series). ISBN 9781439835777. Disponível em: <https://books.google.com.br/books?id=P2HRBQAAQBAJ>. Citado 3 vezes nas páginas 28, 29 e 30.

TRIDAPALLI, J. P.; FERNANDES, E.; MACHADO, W. V. Gestão da cadeia de suprimento do setor público: uma alternativa para controle de gastos correntes no brasil. *Revista de Administração Pública*, SciELO Brasil, v. 45, p. 401–433, 2011. Citado na página 19.