

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE SÃO PAULO
CÂMPUS BIRIGUI

FABRÍCIO MALTA DE OLIVEIRA
LUIZ CHUN ROM HSU

**ALGORÍTIMOS GENÉTICOS APLICADO AO PROBLEMA DE
GRADE HORÁRIA: CASO IFSP BIRIGUI**

Birigui
2022

FABRÍCIO MALTA DE OLIVEIRA
LUIZ CHUN ROM HSU

**ALGORÍTIMOS GENÉTICOS APLICADO AO PROBLEMA DE
GRADE HORÁRIA: CASO IFSP BIRIGUI**

Trabalho de Conclusão de Curso apresentado ao Instituto Federal de Educação, Ciência e Tecnologia de São Paulo Câmpus Birigui, como requisito parcial para conclusão do curso Engenharia da Computação.

Orientador: Eduardo Shiguelo Hoji

Birigui
2022

Malta de Oliveira , Fabrício
Algorítimos Genéticos Aplicado ao problema de
Grade Horária: Caso IFSP Birigui/ Fabrício Malta de
Oliveira , Luiz Chun Rom Hsu, . - Birigui, 2022-
68 p. : il.

Orientador: Eduardo Shigueso Hoji
Trabalho de Conclusão de Curso - Instituto
Federal de Educação, Ciência e Tecnologia de São
Paulo Câmpus Birigui, 2022.

1. Leitura (Ensino superior) 2. Livros
(Investimento). 3. Leitura - Meios auxiliares.
I. Instituto Federal de Educação, Ciência e
Tecnologia. II. Título.

ERRATA

FERREGINO, C. R. A. Tratamentos de neoplasias ósseas apendiculares com reimplantação de enxerto ósseo autólogo autoclavado associado ao plasma rico em plaquetas: estudo crítico na cirurgia de preservação de membro em cães. 2011. 128 f. Tese (Livre-docência) – Faculdade de Medicina, Veterinária e Zootecnia, Universidade de São Paulo, São Paulo, 2011.

| Folha | Linha | Onde se lê | Leia-se |
|-------|-------|---------------|--------------|
| 16 | 10 | auto-conclavo | autoconclavo |

FABRÍCIO MALTA DE OLIVEIRA
LUIZ CHUN ROM HSU

ALGORÍTIMOS GENÉTICOS APLICADO AO PROBLEMA DE GRADE HORÁRIA: CASO IFSP BIRIGUI

Trabalho de Conclusão de Curso apresentado ao Instituto Federal de Educação, Ciência e Tecnologia de São Paulo Câmpus Birigui, como requisito parcial para conclusão do curso Engenharia da Computação.

Orientador: Eduardo Shigueo Hoji

Banca examinadora

Membro 1, titulação e instituição

Membro 2, titulação e instituição

Membro 3, titulação e instituição

Birigui, ____ de _____ de 2022.

Dedico este trabalho a meus pais por todo o apoio e carinho nos momentos difíceis

AGRADECIMENTOS

Aos meus pais, Fulano e Ciclana, por

À minha orientadora, Fulana, por.....

À banca examinadora, composta pelos docentes Fulano e Ciclana, por...

Aos docentes do Departamento de XXXX, por....

Ao pessoal da Biblioteca da XXXX por....

Aos meus colegas de trabalho na XXXX, por...

Às minhas amigas Fulana, Ciclana, Beltrana por...

À XXIX turma de XXXX, por...

“Se vi mais longe foi por estar sobre os ombros de gigantes.”
Isaac Newton

RESUMO

Este projeto consistiu em pesquisar e aplicar meta-heurísticas voltados à solução do problema de geração e organização de grade horária no ambiente acadêmico, com base nos dados atuais do Instituto Federal de Educação, Ciência e Tecnologia de São Paulo, campus Birigui. O problema de geração e organização de grade horária consiste em desenvolver uma solução que se prove eficaz na perspectiva de solucionar problemas de incompatibilidade de horários e disponibilidade de salas. Para a resolução do problema proposto, serão aplicados inicialmente às seguintes meta-heurísticas: algoritmos genéticos, Simulated Annealing e Busca Tabu.

Palavras-chave: Algoritmos Genéticos. Meta-Heurísticas. Organização e Geração De Grade Horária. Otimização Combinatória.

ABSTRACT

Consiste na tradução do resumo para uma língua estrangeira, mantendo-se a mesma formatação e conteúdo. As línguas mais comuns são o inglês, francês e espanhol. Recomenda-se utilizar um serviço especializado de tradução, evitando-se o uso de um tradutor automático.

Keywords: Word 1. Word 2. Word 3. Word 4. Word' 5.

LISTA DE ILUSTRAÇÕES

| | |
|---|----|
| Figura 1 – Fluxograma genérico do algoritmo genético | 20 |
| Figura 2 – Exemplo de cruzamento em um ponto | 23 |
| Figura 3 – Exemplo de cruzamento em dois pontos | 23 |
| Figura 4 – Cruzamento uniforme | 24 |
| Figura 5 – Cruzamento em maioria | 24 |
| Figura 6 – Asc TimeTable | 26 |
| Figura 7 – Chronos - Tela Principal | 27 |
| Figura 8 – GridClass - Tela Principal | 27 |
| Figura 9 – Comparação dos softwares | 28 |
| Figura 10 – Modelo de Formulário de Preferências de Atividades | 30 |
| Figura 11 – Interface do GNU Octave | 31 |
| Figura 12 – Codificação inicial das disciplinas | 32 |
| Figura 13 – Matriz de entrada | 33 |
| Figura 14 – Matriz de solução geral | 33 |
| Figura 15 – Matriz Solução 2 ^a Etapa | 36 |
| Figura 16 – Conjunto de Solução Final | 36 |
| Figura 17 – Modelo Relacional do Banco de Dados | 38 |
| Figura 18 – Visual studio Profissional 2019 | 39 |
| Figura 19 – Divisão das camadas e suas relações | 40 |
| Figura 20 – Função objetivo de cada indivíduo durante a execução. | 42 |
| Figura 21 – Função objetivo de cada indivíduo após a execução. | 43 |
| Figura 22 – Solução obtida após 75 gerações, sendo o 1521º Individuo. | 43 |
| Figura 23 – Tela Principal | 44 |
| Figura 24 – Tela de Cadastro de Curso | 45 |
| Figura 25 – Tela de Listagem Curso | 45 |
| Figura 26 – Tela de Cadastro Disciplina | 46 |
| Figura 27 – Tela de Listagem Disciplina | 46 |
| Figura 28 – Tela de Cadastro Professor | 47 |
| Figura 29 – Tela de Consulta Professor | 48 |
| Figura 30 – Tela de Cadastro Sala | 48 |
| Figura 31 – Tela de Listagem Sala | 48 |
| Figura 32 – Tela de Cadastro Turma | 49 |
| Figura 33 – Tela de listagem Turma | 49 |
| Figura 34 – Tela de Execução do algoritmo | 50 |
| Figura 35 – Script de Execução Principal do Algoritmo Genético | 58 |
| Figura 36 – Script de Codificação das Disciplinas | 59 |

| | |
|---|----|
| Figura 37 – Script CrossOver | 60 |
| Figura 38 – Script Fitness | 60 |
| Figura 39 – Script Geração Inicial Aleatória 1 | 61 |
| Figura 40 – Script Geração Inicial Aleatória 2 | 62 |
| Figura 41 – Script Geração Inicial Aleatória 3 | 63 |
| Figura 42 – Script de Mutação | 63 |
| Figura 43 – Script de Cálculo de Pontuação FPA | 64 |
| Figura 44 – Script de Cálculo de Pontuação Incompatibilidade de Professores | 65 |
| Figura 45 – Script de Cálculo de Pontuação Incompatibilidade de Salas | 66 |

LISTA DE TABELAS

| | |
|--|----|
| Tabela 1 – Trabalhos relacionados, seus respectivos anos e métodos | 25 |
| Tabela 2 – Classes do projeto e suas respectivas funções | 41 |

LISTA DE ABREVIATURAS E SIGLAS

| | |
|-------|--|
| AG | Algoritmos Genéticos |
| BD | Banco de Dados |
| CRUD | Create, Read, Update, Delete |
| C# | C Sharp |
| FPA | Formulário de Preferências de Atividades |
| GH | Grade Horária |
| IDE | Ambiente de Desenvolvimento Integrado |
| MySQL | Michael Widenius's Structured Query Language |
| MVC | Model, View e Controller |
| PSO | Optimização por enxame de partículas |

LISTA DE SÍMBOLOS

| | |
|-----------|----------------------------|
| Γ | Letra grega Gama |
| Λ | Lambda |
| ζ | Letra grega minúscula zeta |
| \in | Pertence |
| Σ | Soma |

SUMÁRIO

| | | |
|----------------|---------------------------------------|-----------|
| 1 | INTRODUÇÃO | 16 |
| 1.1 | Justificativa | 16 |
| 1.2 | Motivação | 17 |
| 2 | FUNDAMENTAÇÃO TEÓRICA | 18 |
| 2.1 | Problema de alocação de grade Horária | 18 |
| 2.2 | Complexidade computacional | 19 |
| 2.3 | Heurísticas | 19 |
| 2.4 | Algoritmos Genéticos | 19 |
| 2.4.1 | Fitness | 21 |
| 2.4.2 | Operadores genéticos | 21 |
| 2.4.2.1 | Seleção | 21 |
| 2.4.2.2 | Mutação | 22 |
| 2.4.2.3 | Recombinação | 22 |
| 2.5 | Trabalhos Relacionados | 24 |
| 2.6 | Softwares semelhantes | 25 |
| 2.6.1 | Asc TimeTables | 26 |
| 2.6.2 | Chronos | 26 |
| 2.6.3 | GridClass | 27 |
| 3 | METODOLOGIA | 29 |
| 3.1 | Primeira etapa | 29 |
| 3.1.1 | GNU OCTAVE | 30 |
| 3.1.2 | Codificação | 31 |
| 3.1.3 | Matriz Solução | 33 |
| 3.1.4 | Execução do Algoritmo | 34 |
| 3.2 | Função Objetivo | 35 |
| 3.3 | Segunda Etapa | 36 |
| 3.3.1 | Matriz de Solução | 36 |
| 3.3.2 | Banco de Dados | 37 |
| 3.3.3 | Visual Studio | 39 |
| 3.3.4 | Prototipação em C# | 39 |
| 3.3.5 | Fluxo de execução | 40 |
| 4 | RESULTADOS | 42 |
| 4.1 | Primeira Etapa | 42 |

| | | |
|------------------------------|--------------------------------|-----------|
| 4.2 | Segunda Etapa | 44 |
| 4.2.1 | Telas Visuais | 44 |
| 4.3 | Discussão | 51 |
| 5 | CONCLUSÃO | 52 |
| REFERÊNCIAS | | 53 |
| GLOSSARIO | | 56 |
| APÊNDICE | | 57 |
| ANEXO | | 67 |
| ÍNDICE | | 68 |

1 INTRODUÇÃO

Atualmente presente em quase todas as instituições de ensino, as grades de horários são de grande importância e essenciais para a rotina do corpo docente, discente e dos funcionários, pois uma vez elaborada, coloca em ordem os horários da instituição durante todo seu período letivo. A solução manual desta questão é usualmente adotada, isso denota a perda excessiva de tempo na resolução e, muitas vezes, ineficiência.

O problema de alocação de horários escolares começou a ser estudado há mais de 50 anos, conforme apresentado em GOTLIEB (1963), que elaborou um algoritmo complexo para organizar, dentre outras variáveis, os períodos e os professores. Com o decorrer dos anos, este problema ganhou reconhecimento na área de otimização combinatória, possuindo diversos trabalhos publicados.

Existe uma grande dificuldade na área de otimização acerca do problema de tabela-horário. Sua complexidade pode ser ainda maior de acordo com suas condições e restrições (SCHAERF, 1999). Com o avanço da complexidade e a elaboração de novos problemas combinacionais, surgiu a necessidade de desenvolver algoritmos mais eficientes, capazes de proporcionar respostas de boa qualidade.

Diante do exposto, os métodos manuais se tornaram extremamente custosos no quesito tempo, abrindo caminho para o surgimento das meta-heurísticas, que são métodos de otimização amplamente utilizadas nos dias atuais, devido à facilidade de implementação e a capacidade de solucionar problemas combinacionais utilizando tempos satisfatórios.

1.1 JUSTIFICATIVA

A elaboração dos horários manualmente se torna uma tarefa difícil e demanda perda excessiva de tempo devido ao grande número de restrições e requisitos, muitas vezes conflitantes entre elas. Aliado a algoritmos inteligentes, novas soluções podem ser geradas com menos esforço aplicado. Neste projeto, inicialmente foi envolvido os cursos da área industrial do IFSP – Campus Birigui, o problema de geração e organização de grade horária, leva em consideração restrições relativas aos horários dos docentes, além da incompatibilidade da atribuição de salas de uso específico ou laboratórios.

Após ser aplicado nos cursos da área da indústria, o algoritmo foi expandido

de modo a suportar todos os cursos do campus.

1.2 MOTIVAÇÃO

Desenvolver um projeto que envolva várias áreas da engenharia de computação, como eletrotécnica, eletrônica, microcontroladores, algoritmos e desenvolvimento de software, explorando todo o conhecimento adquirido durante a graduação.

2 FUNDAMENTAÇÃO TEÓRICA

Esta seção destina-se a dedicar de maneira detalhada o problema de geração de grade horária, explicando o conceito, a complexidade, as técnicas de solução e os algoritmos a serem utilizados no presente trabalho. Também é demonstrado e citado, trabalhos relacionados na mesma área, com interpretação similar, de maneira que sucinta o problema abordado.

2.1 PROBLEMA DE ALOCAÇÃO DE GRADE HORÁRIA

Sabe-se que uma grade horária, quando produzida de forma adequada, afeta diretamente o ambiente letivo, evitando possíveis conflitos de incompatibilidade e até influenciando na organização do meio. Para isso ser alcançado há uma série de restrições que necessitam ser atendidas, sendo que para cada ambiente, há diferentes tipos de particularidades (CISCON; ALVARENGA, 2005).

Schaerf (1999) propõe três classes de problemas nesses tipos de ambientes:

- *School Timetabling*: Seguimento semanal de aulas, evitando que professores e alunos tenham incompatibilidade ou duplicidade de horário;
- *Course Timetabling*: Evitar a simultaneidade de cursos com estudantes em comum, trata do seguimento semanal das aulas de todo um conjunto de cursos em um ambiente de ensino;
- *Examination Timetabling*: Tabelamento voltado para exames escolares, evitar que tenha exames duplicados entre estudantes em comum, também possibilitar o máximo possível de exames a serem aplicados;

Neste trabalho é abordado o problema de *School Timetabling* voltado a resolver incompatibilidade de cursos, turmas e docentes.

Além disto, a literatura procura trabalhar de maneira genérica com problemas deste tipo, sabe-se que cada ambiente de ensino usa um formato diferente de organização de horários, além de possuir restrições únicas, desta maneira torna-se complexo encontrar algum tipo de solução global que atenda todos os casos. Diante do exposto, diversas técnicas de otimização já foram aplicadas para solucionar o problema, conforme apontado em (KOTSKO; STEINER, 2003).

2.2 COMPLEXIDADE COMPUTACIONAL

A complexidade do problema, está atrelada com base na quantidade restrições e variáveis envolvidas na particularidade do assunto abordado, além de aumentarem sua complexidade de maneira exponencial, portanto, uma solução obtida em tempo aceitável acaba sendo inviável de ser obtida, conforme apontado por (ALMEIDA, 2015).

Diante do exposto, é considerado que problemas do gênero de escalonamento de grade horária são categorizados como NP completo de acordo inicialmente com Even, Itai e Shamir (1975) e reforçado Schaerf (1999), devido a sua principal característica de inexistir globalmente um algoritmo que consiga obter uma solução ótima em tempo polinomial.

2.3 HEURÍSTICAS

As heurísticas são caracterizadas como técnicas destinadas a resolver um problema específico, mas não precisam provar que a solução resultante é ótima e, geralmente, produzem resultados de qualidade cada vez mais alta (YANG; JIANG; NGUYEN, 2013).

O termo meta-heurística foi inicialmente formulado por Glover (1986), na qual definiu como um método de busca que combina métodos efetivos na solução de problemas avançados de grande porte.

De forma geral, meta-heurística significa desenvolver uma estratégia de mais alto nível, que busca através do espaço de soluções de um problema complexo, sobre os quais não existe muita informação, mas que, uma vez oferecida uma solução candidata, esta pode ser testada. Entretanto, não apresentam garantias de obter o ótimo global, que é a solução ótima do problema. Um ótimo local ocorre quando é obtida uma solução de boa qualidade com relação ao seu entorno, porém não é a melhor solução possível. Alguns exemplos de meta -heurística são a Busca Tabu (HERTZ, 1992) e os Algoritmos Genéticos (AGs) (WEARE; BURKE; ELLIMAN, 1995).

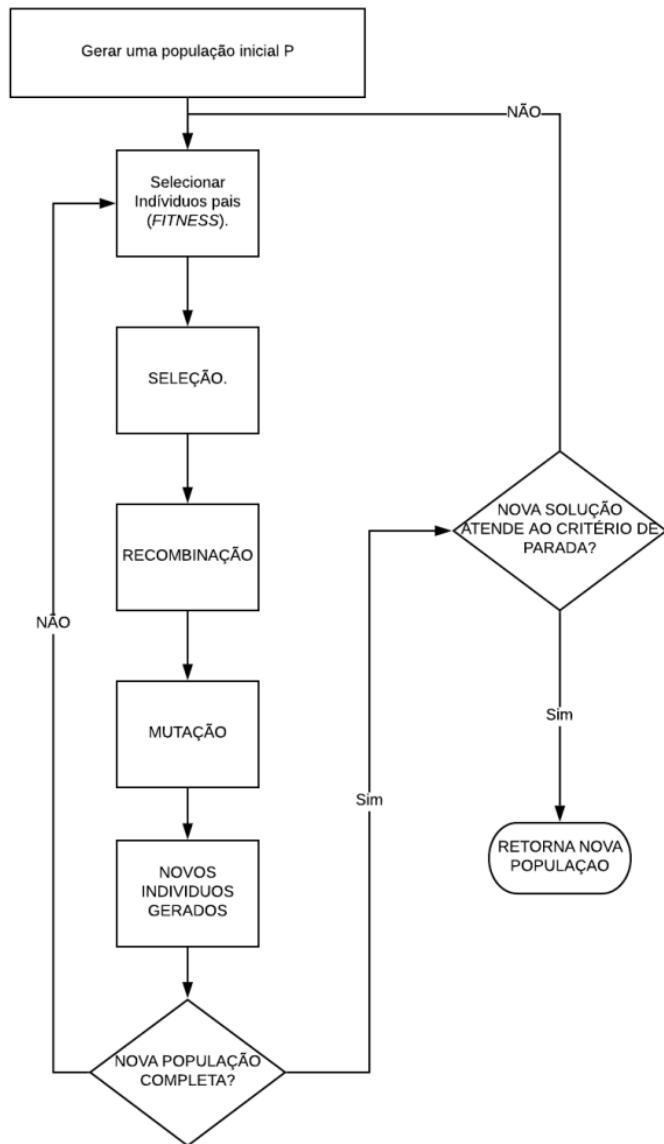
2.4 ALGORITMOS GENÉTICOS

Os Algoritmos genéticos estão entre as mais populares meta-heurísticas aplicadas à resolução de problemas combinatórios, tais algoritmos são métodos de otimização que foram inicialmente propostos por Holland (1992).

Baseiam-se na teoria evolutiva de Darwin, esta estrutura procura simular o

estudo teórico, seguindo as etapas de seleção, recombinação (*crossover*) e mutação, a fim de gerar uma nova população de indivíduos, até que uma proposta de solução atenda ao objetivo do problema ou que o critério de parada seja satisfeito (GOLDBERG, 1989). O fluxograma de um algoritmo genético tradicional é apresentado na Figura 2.

Figura 1 – Fluxograma genérico do algoritmo genético



Fonte: Elaborado pelo autor.

Na Figura 1, observa-se que partindo de uma população inicial P (gerada normalmente de forma aleatória), os indivíduos são selecionados e submetidos ao processo de recombinação e mutação. Na sequência, os indivíduos gerados são colocados em uma nova população e o processo é repetido até que a nova população esteja completa. Ao completar a nova população, ocorre a avaliação da qualidade das soluções propostas. Por fim, o critério de parada é verificado, caso não esteja

satisfeto, uma nova população deve ser formada e o processo de geração de novos indivíduos é reiniciado.

2.4.1 Fitness

Chamada de função objetivo, é a responsável por avaliar cada individuo, conforme a qualidade que apresentam em relação a resposta esperada para o problema. Este índice de aptidão do individuo é transformado em um valor numérico, para que assim seja identificado os indivíduo mais adaptados dentro de uma geração (SILVA, 2014).

2.4.2 Operadores genéticos

A base que estrutura o funcionamento dos algoritmos genéticos, são denominados operadores genéticos. Estes operadores alteram a população de indivíduos através de gerações contínuas, estendendo as iterações até que o critério de parada seja atendido, ou seja, objetivando a escolha de indivíduos mais aptos, necessários para que haja uma diversificação da população e que as características de adaptação adquirida em etapas anteriores seja mantida (GOUVEIA et al., 2021).

Conforme apontado anteriormente na figura 1, temos 3 operações possíveis realizadas nos AG:

2.4.2.1 Seleção

Etapa importante dentro dos AG, determina se o indivíduo participará ou não do processo de reprodução. A etapa de seleção às vezes também é conhecido como o operador de reprodução (GOUVEIA et al., 2021). A taxa de convergência do AG depende do critério de seleção. Existem muitos métodos de seleção para escolha dos melhores cromossomos, dentre estes citamos: técnicas de seleção popularmente conhecidas são: roleta, torneio, elitismo.

A) A seleção da roleta mapeia todas as sequências possíveis em uma roda com uma parte da roleta atribuída a eles de acordo com seu valor de fitness. Esta roda é então girada aleatoriamente para selecionar soluções específicas que participarão da formação da próxima geração. Este método sofre de muitos problemas, como erros introduzidos por sua natureza estocástica (SILVA, 2014).

B) Método de seleção de torneios foi proposta pela primeira vez por Brindle em 1983. Os indivíduos são selecionados de acordo com seus valores de aptidão de uma roleta estocástica em pares. Depois de seleção, os indivíduos com maior valor de

aptidão são adicionados ao grupo da próxima geração. Neste método de seleção, cada indivíduo é comparado com todos os n-1 outros indivíduos se atinge a população final de soluções. Esta técnica funciona relativamente adequada para vários problemas, mas à medida que o tamanho do problema aumenta, a seleção tradicional de roleta tem um desempenho relativamente melhor(KATOCH; CHAUHAN; KUMAR, 2021).

C) Método de seleção elitista foi proposta por Jong (1975) para melhorar o desempenho de seleção de roleta. Garante que o indivíduo elitista em uma geração seja sempre propagado para a próxima geração. Se o indivíduo com o maior valor de aptidão não estiver presente na próxima geração após o procedimento de seleção normal, então a elitista também é incluída na próxima geração automaticamente(LINDEN, 2012).

2.4.2.2 Mutação

Nesta etapa envolve unicamente o indivíduo selecionado, neste caso seu código genético pode sofrer alterações através de diversos fatores, como externos e internos, essas modificações resultantes podem ser benéficas ou não, estas mudanças são consideradas mutações.

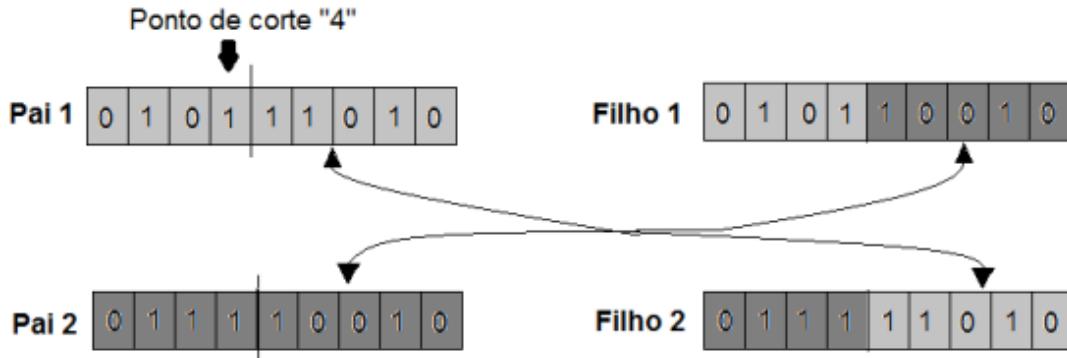
Dessa forma, é possível vincular a genética à teoria da evolução, onde os indivíduos mais bem-sucedidos tendem a buscar parceiros mais atraentes e ainda mais adaptados. Os genes dos bons indivíduos, combinados através de cruzamentos e mutações, tendem a gerar indivíduos ainda mais adequados e, portanto, avanços de evolução natural, obtendo uma maior complexidade e melhor confiabilidade dentro do ambiente de solução no qual os indivíduos estão inseridos (LINDEN, 2012).

2.4.2.3 Recombinação

Também chamado de crossover, é um operador genético usado para combinar a informação genética de dois pais para gerar novos descendentes. É uma maneira de gerar estocasticamente novas soluções a partir de uma população existente e é análoga ao cruzamento que acontece durante a reprodução sexual na biologia.

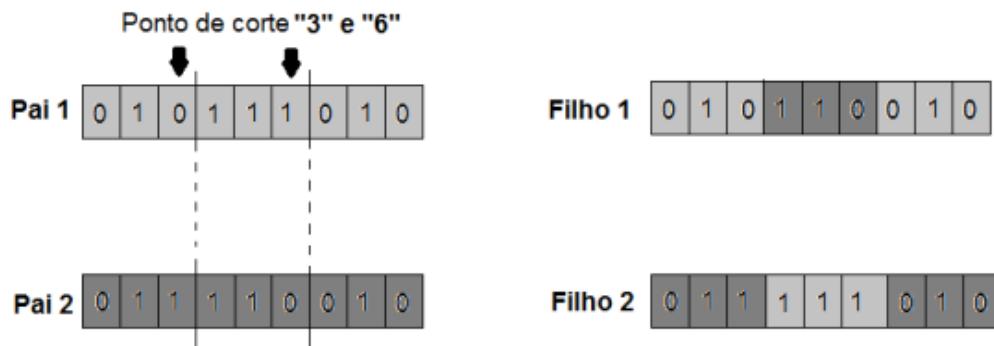
Um ponto – um ponto de cruzamento é escolhido e a partir deste ponto as informações genéticas dos pais serão trocadas gerando assim novos filhos (Figura 2).

Etapa conhecida como , nesta etapa, de maneira biológica as características (*cromossomos*) são selecionados dois indivíduos indivíduos que se cruzam com o objetivo de realizar uma troca de informação (GOUVEIA et al., 2021).

Figura 2 – Exemplo de cruzamento em um ponto

Fonte: (ALMEIDA, 2015)

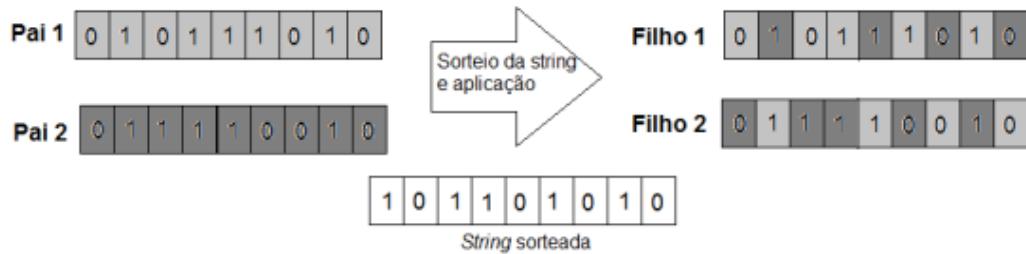
Multiponto – semelhante ao cruzamento de um ponto, porém vários pontos são utilizado para troca informações genéticas (Figura 3).

Figura 3 – Exemplo de cruzamento em dois pontos

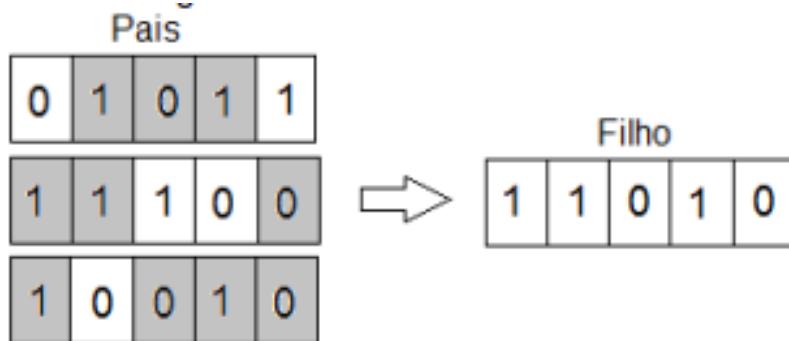
Fonte: (ALMEIDA, 2015)

Uniforme – o cruzamento uniforme é determinado através de um parâmetro, que sorteia sua probabilidade de ser trocado ou não. Assim sendo é sorteado uma vetor com valores de 0 e 1. A troca de informações genéticas se baseia na posição de cada valor do vetor sorteada, se o valor da posição for igual a 1 o filho 1 herda o valor da posição corrente do pai 1 e o filho dois do pai 2, se o valor for igual a 0 o filho 1 recebe o valor do pai 2 da posição corrente e o filho 2 recebe o valor do pai 1 (Figura 4).

Cruzamento em maioria – forma de cruzamento raramente utilizada, pdevido ao fato que a tendênciade que a grade horária converja rapidamente e percorrendo todo o espaço de busca. Seu funcionamento consiste em sortear vários pais e fazer com que cada bit do filho seja igual ao valor da maioria dos pais selecionados, conforme mostrado na Figura 2.4.2.3.

Figura 4 – Cruzamento uniforme

Fonte: (ALMEIDA, 2015)

Figura 5 – Cruzamento em maioria

Fonte: (LINDEN, 2012)

2.5 TRABALHOS RELACIONADOS

Trabalhos relacionados ao problema de grade horário demonstram que os algoritmos podem ter desempenhos satisfatórios comparado com outros algoritmos de busca. No emprego do método AG em trabalhos em aplicações de natureza semelhante, como do Hamawaki et al. (2005) podemos notar obtenção de soluções factíveis melhores que as elaboradas manualmente, além disso atenderam as necessidades da Instituição. As análises foram executadas considerando diversas incompatibilidades e com soluções iniciais infactíveis. Os resultados mostram evidências da importância da técnica de otimização a esse tipo de problema, bem como a validade da abordagem realizada.

Em Lucas (2000) o uso do AG apresentam resultados indesejáveis com relação aos outros algoritmos, porém com adaptações mostra pontos de extrema relevância como por exemplo sua diversidade acerca dos indivíduos gerados.

No trabalho de Babaei, Karimpour e Hadidi (2015) o uso do AG apresentam métodos escalonáveis relação aos outros algoritmos, como abordagens baseadas em métodos de pesquisa, baseados em meta heurística e métodos inteligentes porém com adaptações mostra pontos de extrema relevância que mostram que as abordagens baseados em meta-heurísticas e métodos inteligentes são melhores que os

métodos de busca.

No trabalho de Chen e Shih (2013) foi utilizado a técnica de otimização por enxame de partículas, pelo fato da rápida convergência e ser bastante dinâmico. Com o algoritmo pode ser buscar um espaço de soluções ótimas para a restrições do problema.

Souza et al. (2017) usa o conceito de solvers de programação inteira mista para a solução da heurística chamada de fix-and-optimize. A heurística consiste em fixar uma parte das variáveis, criando subproblemas muito menores, possibilitando sua solução através de métodos exatos e então repetindo o processo de fixação até todas as variáveis serem deixadas livres para a otimização.

Abaixo uma tabela com uma síntese de todos os trabalhos relacionados e seus respectivos ano e métodos utilizados:

Tabela 1 – Trabalhos relacionados, seus respectivos anos e métodos

| Autor | Ano | Técnica |
|-----------------------------------|------|----------------------|
| Hamawaki et al. (2005) | 2005 | Algorítmico Genético |
| Lucas (2000) | 2000 | Algorítmico Genético |
| Babaei, Karimpour e Hadidi (2015) | 2015 | Algoritmo Genético |
| Chen e Shih (2013) | 2013 | PSO |
| Souza et al. (2017) | 2017 | solvers |

Fonte: Elaborado pelo autor.

Observa-se que o problema já vem sido estudado anteriormente com diversos tipos de técnicas. O fato do problema ser bastante complexo possibilita seu uso e estudo de diversas formas e aplicações.

Para que haja melhor entendimento sobre o assunto e técnicas utilizadas em busca de uma solução para o problema, são apresentados os conceitos sobre essas técnicas no APÊNDICE A.

2.6 SOFTWARES SEMELHANTES

As soluções existentes no mercado atualmente são amplas e diversas, nesta seção é exibido soluções semelhantes a estudada no problema proposto e que são de utilização comercial atualmente.

Figura 6 – Asc TimeTable

Fonte: (ASC, 2022)

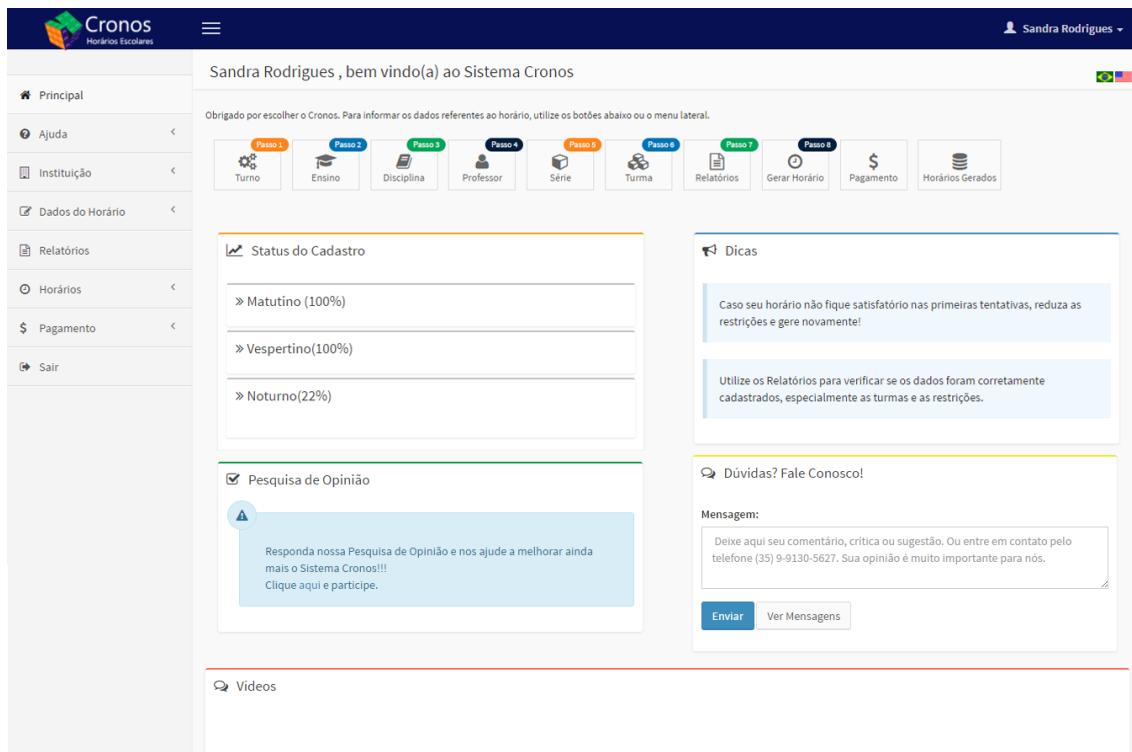
2.6.1 Asc TimeTables

Asc Timetable é um dos softwares mais usados atualmente ele é capaz de encontrar soluções para cursos, focado com relações anuais que condizem com o ensino fundamental e médio, além disso gera horários principalmente para um único turno. Ele inclui a inserção de todas as disciplinas, turmas, sala de aula, professores e seus contratos. Além de permitir a criação de divisões de turmas em grupos, horários quinzenais, a reunião de turmas em uma aula ou ter mais de um professor para uma aula (ASC, 2022).

O programa gera cerca de alguns minutos, o programa gerará o horário completo de acordo com as suas especificações. O software verifica a entrada de dados e o ajuda a remover os erros. Ele também verifica se o horário gerado atende a todas as condições impostas. Ao realizar mudanças manualmente, o programa o informa se houver algum erro.(ASC, 2022).

2.6.2 Chronos

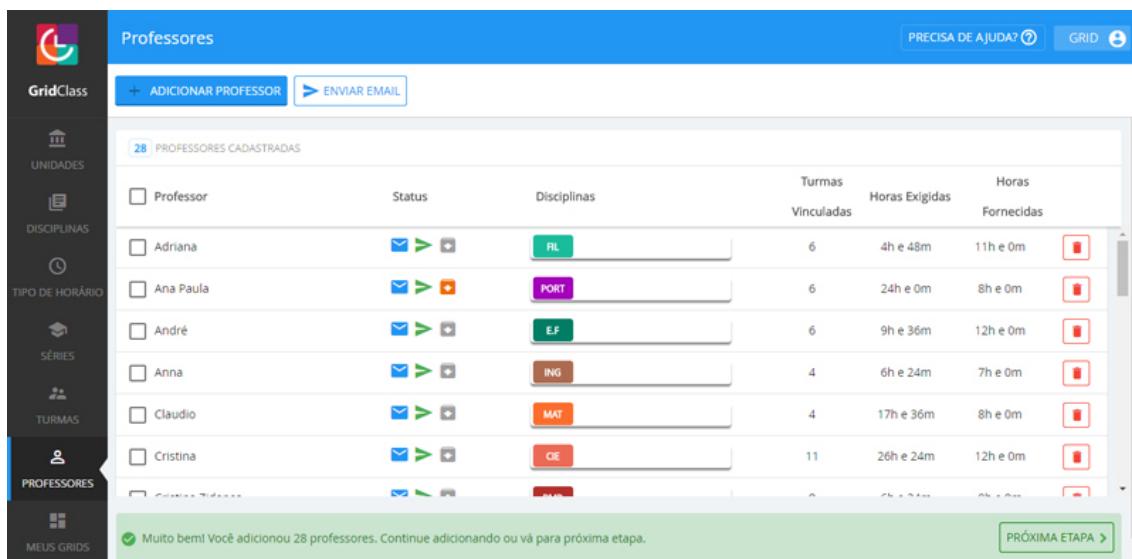
Assim como o Asc TimeTable, o software Cronos também é um programa bem conhecido atualmente. Com ele é possível montar o quadro de horários para os professores e turmas de maneira automática, considerando as principais restrições existentes, sua principal diferença relacionada a outros software são os preços e ser um sistema web com armazenamento em nuvem.

Figura 7 – Chronos - Tela Principal

Fonte: (CHRONOS, 2022)

2.6.3 GridClass

O GridClass se identifica como uma ferramenta para gestores escolares, criada a partir de um algoritmo capaz de relacionar automaticamente todos os dados e encontrar a melhor combinação possível de horários dentro da necessidade da escola.

Figura 8 – GridClass - Tela Principal

Fonte: (GRIDCLASS, 2022)

Todos os 3 softwares oferecem geração de GH automática e tratamentos para

restrições e choques de horários, porém como observado na figura ?? os preços do Asc TimeTable são bem elevados comparados aos outros 2, com o detalhe do Grid-class oferecer um plano anual. Pode observar que apenas o Asc TimeTable disponibiliza publicar o horário para professores e estudantes no dispositivos móveis.

Figura 9 – Comparaçao dos softwares

| Software | Plano 1 | Plano 2 | Plano 3 | Dispositivo Móveis | Armazenamento Online | Turmas Plano Básico |
|---------------|----------------|-------------|--------------|--------------------|----------------------|---------------------|
| Asc TimeTable | 120 dólares | 500 dólares | 1995 dólares | Sim | Não | Ilimitado |
| Chronos | 315R\$ | 559R\$ | 779R\$ | Não | Sim | 10 |
| GridClass | 1.399R\$ Anual | Não possui | Não possui | Não | Sim | 20 |

Fonte: Elaborado pelo autor.

3 METODOLOGIA

O desenvolvimento do trabalho foi dividido em duas etapas, a primeira etapa foi viabilizar a criação de uma codificação que suportasse as características do problema, os códigos desenvolvidos nessa etapa foram desenvolvidos de maneira estruturada, transformando as informações em dados computacionais e executáveis. Durante esta etapa, a solução foi aplicada somente para dois cursos do campus: Tecnologia em Mecatrônica Industrial e Automação Industrial, ambos no período noturno separados por dois horários, antes do intervalo e após. Uma vez comprovado a eficácia dos algoritmos desenvolvidos, a segunda parte foi iniciada.

A segunda etapa correspondeu a uma expansão inicial da solução obtida, de forma que foram desenvolvidas telas visuais interativas para dinamizar a utilização, nesta etapa também foi criado suporte a todos cursos do campus, além da utilização de base de dados, de modo a salvar às informações de codificação inseridas e soluções obtidas. Os códigos desenvolvidos nessa etapa, foram desenvolvidos utilizando modelagem orientada a objeto, utilizando o padrão de arquitetura de software MVC.

3.1 PRIMEIRA ETAPA

Inicialmente, uma proposta de codificação para implementação algorítmica foi feita com o uso de matrizes para a alocação de turmas (linhas) e horários (colunas). Cada docente, disciplina, sala de aula e sua capacidade foram identificados por um valor específico, que por sua vez, foram inseridos na matriz descrita para representar, por exemplo, que o algoritmo atribuiu um horário “X” para um docente “Y” na sala de aula “W”. O algoritmo então partiu de uma solução inicial gerada aleatoriamente.

A análise da qualidade das soluções, disponibilizada pelo método de otimização aplicado, foram definidas a partir de uma função objetivo baseada em penalização. Uma penalidade é atribuída a uma proposta de solução toda vez que uma restrição for violada, por exemplo, a atribuição de um mesmo horário para um docente em salas diferentes. A função objetivo então considera dois níveis de penalidades (média e alta), de acordo com o nível de infactibilidade gerada pela violação da restrição em questão.

A validação da solução obtida, é realizada a partir da comparação com a grade horária gerada e organizada de forma manual, que foi utilizado no primeiro semestre de 2018. Nesta etapa o objetivo foi atingido após método de otimização demonstrar-se capaz de encontrar uma solução de boa qualidade, ou seja, com a minimização da função objetivo (nenhuma ou mínimas violações de restrições) e, além disso, que a

solução fornecida pelos algoritmos for melhor que a solução gerada de forma manual.

Durante a criação de forma manual, cada professor deve possuir um Formulário de Preferências de Atividades (FPA), no qual preenche os dias em que possui disponibilidade para dar aula. Os dias são contidos entre segunda a sábado. Os coordenadores de cada curso analisam todas a FPA's dos professores referentes aos cursos e iniciam as montagens da grade horária.

As prévias dos horários inicialmente são colocadas em um quadro 10 pelos coordenadores, logo são realizadas reuniões para tratar os choques que vão surgindo entre os horários dos professores e por fim cada horários podem é modificado de acordo com a necessidade.

Estas informações são utilizadas na etapa de cálculo de fitness, uma vez que a solução obtida infringiu diretamente a FPA do professor, uma penalidade é calculada e inserida na função objetivo do individuo.

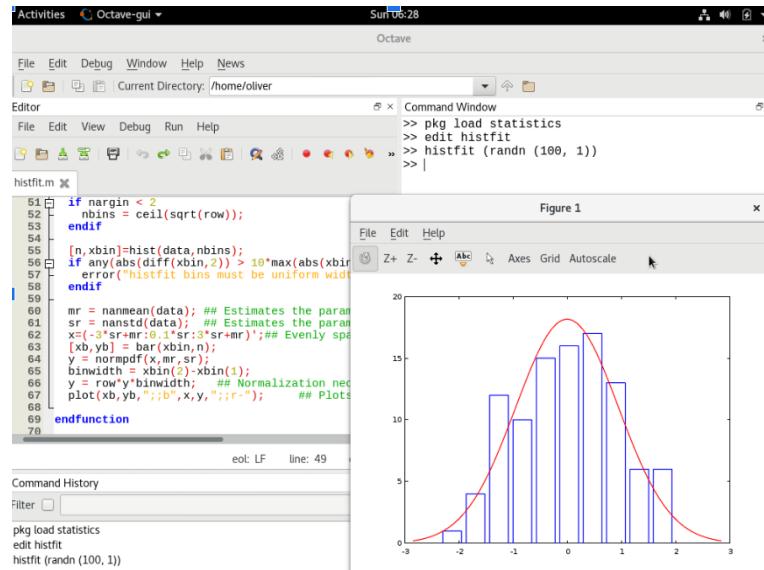
Figura 10 – Modelo de Formulário de Preferências de Atividades

| ANEXO I | | | | | | | |
|--|---------------|-----------|-------|-----------------|--------|------------|--------|
| Instituto Federal de Educação, Ciência e Tecnologia de São Paulo - IFSP Formulário de Preferência de Atividades - FPA (Anexo I - Resolução nº 109 de 4 de novembro de 2015) | | | | | | | |
| Campus: | Ano/Semestre: | | | | | | |
| Identificação do Docente | | | | | | | |
| Docente: | | | | Conhecido como: | | | |
| Área: | | | | e-mail: | | | |
| Prontuário: | Nome: | Telefone: | | Celular: | | | |
| Regime de trabalho: | 20 horas | 40 horas | RDE | Substituto | | Temporário | |
| <i>Selecione seu regime de trabalho.</i> | | | | | | | |
| Disponibilidade de horário para atribuição de componentes curriculares | | | | | | | |
| Matutino | Aula | Segunda | Terça | Quarta | Quinta | Sexta | Sábado |
| 1 | | | | | | | |
| 2 | | | | | | | |
| 3 | | | | | | | |
| 4 | | | | | | | |
| 5 | | | | | | | |
| 6 | | | | | | | |
| Vespertino | Aula | Segunda | Terça | Quarta | Quinta | Sexta | Sábado |
| 1 | | | | | | | |
| 2 | | | | | | | |
| 3 | | | | | | | |
| 4 | | | | | | | |
| 5 | | | | | | | |
| 6 | | | | | | | |
| Noiturno | Aula | Segunda | Terça | Quarta | Quinta | Sexta | Sábado |
| 1 | | | | | | | |
| 2 | | | | | | | |
| 3 | | | | | | | |
| 4 | | | | | | | |
| 5 | | | | | | | |
| 6 | | | | | | | |
| <input type="checkbox"/> Sim, desejo dedicar-me prioritariamente a atividade de ensino. | | | | | | | |

Fonte: (MOREIRA; COLNAGO, 2018)

3.1.1 GNU OCTAVE

Inicialmente a implementação dos algoritmos computacionalmente foi feita através da utilização do software GNU Octave, um software de código aberto que apresenta uma linguagem de programação de alto nível, destinada principalmente a cálculos numéricos, possibilitando então trabalhar com computação matemática para execução de algoritmos. Além disso o Octave ajuda a resolver problemas lineares e não lineares numericamente.

Figura 11 – Interface do GNU Octave

Fonte: (FORGE,)

A figura 11 exibe um exemplo de interface do Octave, na esquerda temos o espaço para a escrita dos algoritmos e funções, a direita acima do gráfico, temos o resultado da execução e chamadas adicionais para novas funções, e posteriormente um gráfico criado em tempo real pelas variáveis registradas no Octave.

Para implementação do método, outro software poderia ser utilizado, porém a escolha pelo GNU Octave consiste no fato de ser um software gratuito, disponível nos computadores dos laboratórios do IFSP.

3.1.2 Codificação

A coleta inicial dos dados foi realizada em conjunto ao coordenador da área da Indústria. Os dados englobam as atribuições das disciplinas aos docentes, as salas de aula de uso específico demandadas pelas disciplinas, assim como a disponibilidade horária dos docentes. Os dados descritos são gerados a cada início de semestre, a partir do preenchimento de uma planilha por parte dos docentes, para a realização da grade horária. Após a coleta dos dados, foi feita uma análise e organização a fim de se realizar a codificação destas informações.

Conforme pode ser observado na figura 12, cada disciplina recebeu um número único, também é registrado o tipo de aula (teórico ou prático), quantidade de aulas semanais e o professor associado. As matérias cujo há necessidade de divisão de turma (devido ao limite de espaço físico dos laboratórios), é associado um número maior que 100, ela também fica atrelada a outra disciplina que há a divisão em questão, nesse caso sendo o número seguinte.

Figura 12 – Codificação inicial das disciplinas

| Número | Disciplinas | Tipo de Aula | Nº de Aulas | Professor |
|---------------|---|---------------------|--------------------|------------------|
| 1 | Eletricidade I - EL1A1 TEÓRICA | T | 2 | PROFESSOR 4 |
| 2 | Tecnologia dos Materiais - TMAA1 TEÓRICA | T | 2 | PROFESSOR 5 |
| 3 | Mecânica Técnica - META1 TEÓRICA | T | 2 | PROFESSOR 1 |
| 4 | Geometria Aplicada - GEAA1 TEORICA | T | 2 | MATEMATICA |
| 5 | Organização, Saúde e Segurança - OSSA1 TEÓRICA | T | 2 | PROFESSOR 2 |
| 6 | Eletrônica Digital - ELDA1 TEÓRICA | T | 2 | PROFESSOR 3 |
| 7 | Eletricidade II - EL2A2 TEÓRICA/PRÁTICA | T/P | 4 | PROFESSOR 7 |
| 8 | Eletrônica Industrial - ELOA2 TEÓRICA/PRÁTICA | T/P | 4 | PROFESSOR 3 |
| 9 | Resistência dos Materiais- RESA2 TEÓRICA | T | 2 | PROFESSOR 8 |
| 10 | Elementos de Máquinas - ELMA2 TEÓRICA | T | 2 | PROFESSOR 8 |
| 11 | Tecnologias de Usinagem - TUSA3 TEÓRICA | T | 2 | PROFESSOR 10 |
| 12 | Microcontroladores - MCRA3 PRÁTICA | P | 2 | PROFESSOR 9 |
| 13 | Mecânica dos Fluidos - MFLA3 TEÓRICA | T | 2 | PROFESSOR 6 |
| 14 | Ensaios Mecânicos - ENSA3 TEÓRICA | T | 2 | PROFESSOR 5 |
| 15 | Planejamento de Projetos de Automação - PPAA3 TEÓRICA | T | 2 | PROFESSOR 6 |
| 16 | Controladores Lógicos Programáveis - CLPA3 PRÁTICA | P | 2 | PROFESSOR 1 |
| 17 | Acionamentos Elétricos - ACEA3 TEÓRICA/PRÁTICA | T/P | 2 | PROFESSOR 11 |
| 18 | Sistemas de Manutenção - SMNA4 TEÓRICA | T | 2 | PROFESSOR 5 |
| 19 | Robótica e CIM - RCMA4 TEÓRICA | T/P | 2 | PROFESSOR 1 |
| 20 | Redes Industriais - RIDA4 PRÁTICA | P | 2 | PROFESSOR 13 |
| 21 | Sistemas Microcontrolados - SMRA4 PRÁTICA | P | 2 | PROFESSOR 12 |
| 22 | Instrumentação Industrial - ITIA4 TEÓRICA | T | 2 | PROFESSOR 13 |
| 23 | Controle de Processos - COPA4 TEÓRICA | T | 2 | PROFESSOR 13 |
| 24 | Energias Alternativas e Desenvolvimento Sustentável - EADA4 TEÓRICA | T | 2 | PROFESSOR 5 |

| DUPLICADAS | | | | |
|-------------------|---|---------------------|--------------------|------------------|
| Número | Disciplinas | Tipo de Aula | Nº de Aulas | Professor |
| 100/101 | Desenho Técnico - DETA1 PRÁTICA | P | 4 | PROFESSOR 2 |
| | Eletrônica Digital - ELDA1 PRÁTICA | P | 4 | PROFESSOR 3 |
| 102/103 | Desenho Assistido por Computador - DACA2 PRÁTICA | P | 4 | PROFESSOR 1 |
| | Pneumática e Hidráulica - PNHA2 PRÁTICA | P | 4 | PROFESSOR 6 |
| 104/105 | Eletricidade I - EL1A1 PRÁTICA | P | 4 | PROFESSOR 4 |
| | Lógica de Programação - LOPA1 PRÁTICA | P | 4 | PROFESSOR 15 |
| 106/107 | Tecnologia de Usinagem - TUSA3 PRÁTICA | P | 4 | PROFESSOR 10 |
| | Eletropneumática e Eletrohidráulica - EPHA3 PRÁTICA | P | 4 | PROFESSOR 1 |
| 108/109 | Eletricidade - Básica ELES1 PRÁTICA | P | 4 | PROFESSOR 7 |
| | Lógica de Programação LOPS1 - Pratica | P | 4 | PROFESSOR 15 |
| 110/111 | Eletrônica Analógica - ELAS3 PRÁTICA | P | 4 | PROFESSOR 12 |
| | Tecnologias de Usinagem - TUSS3 PRÁTICA | P | 4 | PROFESSOR 10 |

Fonte: Do autor

Há também uma matriz de entrada, essa matriz pode ser observada na figura 13, corresponde a associação das disciplinas de cada professor, onde cada disciplina é informada se precisa de sala única para ser ministrada, para estes casos o número da sala é dado, caso contrário é preenchido com o número 1, além de possuir colunas que informam a relação da preferência de disponibilidade de horário nos dias da semana de cada docente, essas informações são importantes como inserção base no código.

As colunas cujo dias da semana estão preenchidos com o número 1 referem-se à disponibilidade que o professor tem naquele dia e em conformidade o número 0 informa a impossibilidade ou não-preferência.

Figura 13 – Matriz de entrada

| Matriz(ix) | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 |
|------------------|------------|------------|------------|------------|------------|------------|------------|------------|------------|------------|------------|------------|------------|------------|------------|------------|---------------|-------------|--------------|--------------|-------------|-----------|
| PROFESSOR NUMERO | nº da sala | Segunda Noite | Terça Noite | Quarta Noite | Quinta Noite | Sexta Noite | Sab Manha |
| PROFESSOR 1 | 1 | 3 | 1 | 16 | 158 | 19 | 159 | 102 | 159 | 106 | 166 | -1 | -1 | -1 | -1 | -1 | 1 | 1 | 1 | 1 | 1 | 0 |
| PROFESSOR 2 | 2 | 5 | 1 | 100 | 1 | 100 | 1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | 1 | 1 | 1 | 0 | 0 | 0 |
| PROFESSOR 3 | 3 | 6 | 1 | 8 | 156 | 101 | 156 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | 1 | 1 | 1 | 1 | 0 | 0 |
| PROFESSOR 4 | 4 | 1 | 1 | 30 | 1 | 39 | 158 | 104 | 156 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | 1 | 1 | 1 | 1 | 1 | 0 |
| PROFESSOR 5 | 5 | 2 | 1 | 14 | 1 | 18 | 1 | 24 | 1 | 27 | 1 | 36 | 1 | -1 | -1 | -1 | 1 | 1 | 1 | 1 | 1 | 1 |
| PROFESSOR 6 | 6 | 13 | 1 | 15 | 1 | 34 | 1 | 103 | 166 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | 1 | 1 | 1 | 1 | 1 | 1 |
| PROFESSOR 7 | 7 | 7 | 158 | 28 | 1 | 108 | 156 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | 1 | 1 | 0 | 1 | 1 | 0 |
| PROFESSOR 8 | 8 | 9 | 1 | 10 | 1 | 26 | 1 | 33 | 1 | 41 | 157 | -1 | -1 | -1 | -1 | -1 | 1 | 0 | 1 | 0 | 1 | 0 |
| PROFESSOR 9 | 9 | 12 | 157 | 38 | 157 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | 0 | 1 | 1 | 1 | 0 | 0 |
| PROFESSOR 10 | 10 | 11 | 1 | 29 | 1 | 35 | 1 | 107 | 165 | 110 | 165 | -1 | -1 | -1 | -1 | -1 | 1 | 0 | 1 | 1 | 1 | 0 |
| PROFESSOR 11 | 11 | 17 | 158 | 25 | 159 | 40 | 157 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | 0 | 1 | 1 | 1 | 1 | 0 |
| PROFESSOR 12 | 12 | 21 | 157 | 31 | 1 | (11) | 156 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | 1 | 1 | 1 | 1 | 1 | 0 |
| PROFESSOR 13 | 13 | 20 | 1 | 22 | 1 | 23 | 1 | 32 | 1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | 1 | 1 | 1 | 1 | 1 | 0 |
| PROFESSOR 14 | 14 | 42 | 1 | 43 | 157 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | 1 | 1 | 1 | 1 | 1 | 0 |
| PROFESSOR 15 | 15 | 49 | 1 | 105 | 157 | 109 | 1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | 0 | 0 | 1 | 1 | 0 | 1 |
| PROFESSOR 16 | 16 | 4 | 1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | 0 | 0 | 0 | 1 | 0 | 0 |
| PROFESSOR 17 | 17 | 44 | 142 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | 0 | 0 | 0 | 0 | 1 | 0 |
| PROFESSOR 18 | 18 | 46 | 158 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | 1 | 0 | 0 | 0 | 0 | 0 |
| PROFESSOR 19 | 19 | 45 | 1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | 0 | 1 | 0 | 0 | 0 | 0 |
| PROFESSOR 20 | 20 | 47 | 1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | 1 | 0 | 0 | 0 | 0 | 0 |
| PROFESSOR 21 | 21 | 48 | 1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | 0 | 1 | 1 | 0 | 0 | 0 |

Fonte: Do autor

3.1.3 Matriz Solução

A codificação da solução utilizada contou com o uso de uma matriz de solução geral para alocação de turmas (linhas) e horários (colunas). A representação foi feita da seguinte forma: a cada duas linhas é representado uma turma, separada por metade do período letivo (nesta etapa somente no noturno), há 18 colunas separadas por blocos de 3, onde cada bloco significa um dia da semana composto por respectivamente: professor, disciplina e sala de aula. As três últimas colunas correspondem ao dia de sábado, e como nem toda turma possui aula reservada para este dia semanal é preenchido então com o número -1.

Figura 14 – Matriz de solução geral

| | SEGUNDA | | | TERÇA | | | QUARTA | | | QUINTA | | | SEXTA | | | SÁBADO | | |
|----|---------|-----|-----|-------|-----|-----|--------|-----|-----|--------|-----|-----|-------|-----|-----|--------|----|-----|
| | P | D | S | P | D | S | P | D | S | P | D | S | P | D | S | P | D | S |
| T1 | 3 | 101 | 156 | 1 | 3 | 1 | 15 | 105 | 157 | 16 | 4 | 1 | 5 | 2 | 1 | -1 | -1 | -1 |
| | 2 | 100 | 1 | 3 | 6 | 1 | 4 | 104 | 156 | 2 | 5 | 1 | 4 | 1 | 1 | -1 | -1 | -1 |
| T2 | 7 | 7 | 158 | 6 | 103 | 166 | 17 | 44 | 142 | 8 | 9 | 1 | 17 | 44 | 142 | -1 | -1 | -1 |
| | 7 | 7 | 158 | 1 | 102 | 159 | 3 | 8 | 156 | 3 | 8 | 156 | 8 | 10 | 1 | -1 | -1 | -1 |
| T3 | 6 | 15 | 1 | 6 | 13 | 1 | 10 | 107 | 165 | 11 | 17 | 158 | 19 | 45 | 1 | -1 | -1 | -1 |
| | 9 | 12 | 157 | 10 | 11 | 1 | 1 | 106 | 166 | 1 | 16 | 158 | 5 | 14 | 1 | -1 | -1 | -1 |
| T4 | 13 | 23 | 1 | 18 | 46 | 158 | 12 | 21 | 157 | 1 | 19 | 159 | 5 | 24 | 1 | -1 | -1 | -1 |
| | 13 | 22 | 1 | 11 | 25 | 159 | 5 | 18 | 1 | 13 | 20 | 1 | 0 | 0 | 0 | -1 | -1 | -1 |
| T5 | 10 | 29 | 1 | 8 | 26 | 1 | 5 | 27 | 1 | 15 | 109 | 1 | 20 | 47 | 1 | 21 | 48 | 1 |
| | 21 | 48 | 1 | 15 | 49 | 1 | 7 | 28 | 1 | 7 | 108 | 156 | 0 | 0 | 0 | 5 | 27 | 1 |
| T6 | 5 | 36 | 1 | 6 | 34 | 1 | 2 | 37 | 159 | 8 | 33 | 1 | 12 | 111 | 156 | 12 | 31 | 1 |
| | 10 | 35 | 1 | 6 | 34 | 1 | 13 | 32 | 1 | 13 | 32 | 1 | 10 | 110 | 165 | 4 | 30 | 1 |
| T7 | 9 | 38 | 157 | 14 | 43 | 157 | 4 | 39 | 158 | 14 | 42 | 1 | 14 | 42 | 1 | 4 | 39 | 158 |
| | 9 | 38 | 157 | 14 | 43 | 157 | 8 | 41 | 157 | 11 | 40 | 157 | 11 | 40 | 157 | 8 | 41 | 157 |

LEGENDA
 P = PROFESSOR D = DISCIPLINA S = SALA DE AULA

Fonte: Do autor

Conforme mostrado no exemplo acima, cada solução gerada então utilizará as mesmas dimensões da matriz acima. A partir dessa matriz é feita também a análise da qualidade da solução gerada, dada por uma função objetivo que analisa os dados obtidos.

Dentro do ambiente de desenvolvimento, é considerado que cada indivíduo da

população é uma solução igual à mostrada na Figura 14.

3.1.4 Execução do Algoritmo

Ao gerar uma população inicial P , conforme a figura 1 são selecionados os indivíduos que passarão pelos processos de recombinação e mutação, no caso, os critérios utilizados trabalham com taxa fixa de chance de mutação, definida em 55%, e com uma chance de 50% de realizar uma recombinação.

A seleção utilizou o modelo de roleta, em que é analisada a qualidade de cada indivíduo dentro da população, formada por 20 indivíduos. Aqueles indivíduos considerados com maior aptidão, ou seja, menor grau de incompatibilidade dada pela função objetivo possuem probabilidades maiores de serem escolhidos para fazer parte da próxima geração. Nesse método, a participação de cada indivíduo na roleta é proporcional à sua aptidão.

Na etapa de mutação foi considerado que o indivíduo selecionado realizará a troca de uma matéria de um dia da semana por outro. A disciplina pode estar presente em um período antes do intervalo ou após, sendo essa matéria escolhida aleatoriamente e o dia também escolhido aleatoriamente, respeitando as restrições de que não pode ser em um dia que não há aula (considerados com valores atribuídos dentro da matriz de solução com -1) e que, caso envolvam matérias duplicadas, deve-se realizar a troca de ambas às matérias que compõem aquele dia com aquela turma.

O processo de recombinação consistiu na troca de informações entre dois indivíduos. Nesse caso, é utilizado o método de crossover em ponto, conforme apontado pela figura 2, ao selecionar dois indivíduos, é realizada uma troca de “disciplinas”, obtendo assim dois novos indivíduos que possuem as características de ambos.

A qualidade de uma solução é mensurada por uma função objetivo de minimização em que as incompatibilidades encontradas recebem uma penalização. As incompatibilidades podem ser obtidas da seguinte forma: quando um professor está ministrando disciplinas no mesmo horário em duas turmas, quando duas turmas estão utilizando um mesmo laboratório ao mesmo tempo e quando um professor leciona no sábado de manhã após ter tido aula no último bloco da sexta. A cada incompatibilidade encontrada, é atribuído um valor de penalidade, e que, para uma solução ser totalmente viável, a função objetivo tem de ser zero, que corresponde justamente quando não há incompatibilidades infactíveis.

3.2 FUNÇÃO OBJETIVO

Para compreender e analisar a qualidade de uma solução gerada, critérios da função objetivo são aplicados, neste caso o objetivo é minimizar a pontuação gerada, mapeando as incompatibilidades obtidas, a análise é feita através de matrizes, onde representa de maneira independente, as incompatibilidades de salas de laboratório de uso exclusivo e de docentes, também é registrado onde ocorreu essa incompatibilidade. A função objetivo é definida como:

Onde:

$$\text{Minimizar } F = p1 \left(\left(\sum_{i=1}^{na} w_i \right) + \left(\sum_{i=1}^{na} x_i \right) + \left(\sum_{i=1}^{np} y_i \right) \right) + \left(\sum_{i=1}^{na} z_i \right) * p2$$

s. a.

$$w_i \in B, x_i \in B, y_i \in B, z_i \in B$$

Sendo:

- na quantidade de aulas a serem atribuídas;
- np quantidade de professores;
- w_i número de aulas atribuídas de forma concomitante na mesma sala de aula;
- x_i número de aulas atribuídas de forma concomitante a um mesmo professor;
- y_i número de conflitos referentes à violação de interjornada dos professores;
- z_i número de aulas atribuídas em horário fora do horário preferencial dos professores;
- $p1$ - peso associado ao grau de incompatibilidade grave;
- $p2$ - peso associado ao grau de incompatibilidade médio.

As variáveis w_i, x_i, y_i, z_i assumem o valor 1 quando uma incompatibilidade é encontrada, caso contrário, assumem o valor 0. As constantes $p1$ e $p2$ representam as penalidades às incompatibilidades encontradas e possuem seu valor fixado em $p1=10.000$ e $p2= 500$. Quando uma solução possui em sua função objetivo ao menos uma incompatibilidade grave, ela é considerada infactível, uma vez que uma incompatibilidade grave implica em uma execução impossível fisicamente.

A convergência do método é atingida quando uma solução com $F = 0$ é encontrada, respeitando o máximo de 1000 gerações. Caso não sejam encontradas soluções sem penalizações, a melhor solução escolhida é a que, além de não possuir

inconsistências graves, possui a menor quantidade de inconsistências médias, visto que essas penalidades são consideradas menores e factíveis do ponto de vista físico.

3.3 SEGUNDA ETAPA

3.3.1 Matriz de Solução

Durante esta etapa, a matriz de solução foi adaptada, de modo a suportar os horários da manhã, tarde e noite, além disso, cada matriz de solução nesta etapa corresponde à turma por vez, tornando-se uma matriz com 15 linhas e 18 colunas, onde cada linha refere-se aos períodos de aula durante o dia, como pode ser observado na figura abaixo.

Figura 15 – Matriz Solução 2^a Etapa

| Curso Turma | | Técnico em Administração Integrado ao Ensino Médio 1º Semestre | | | | | | | | | | | | | | | | | | |
|----------------|----|---|-----|----|-------|-----|----|--------|-----|----|--------|-----|----|-------|-----|----|--------|----|----|---|
| | | SEGUNDA | | | TERÇA | | | QUARTA | | | QUINTA | | | SEXTA | | | SABADO | | | |
| P | D | S | P | D | S | P | D | S | P | D | S | P | D | S | P | D | S | P | D | S |
| M1 | 3 | 101 | 156 | 3 | 6 | 1 | 2 | 5 | 1 | 15 | 105 | 157 | 16 | 4 | 1 | -1 | -1 | -1 | -1 | |
| M2 | 2 | 100 | 1 | 5 | 2 | 1 | 1 | 3 | 1 | 4 | 104 | 156 | 4 | 1 | 1 | -1 | -1 | -1 | -1 | |
| M3 | 7 | 7 | 158 | 7 | 7 | 158 | 3 | 8 | 156 | 6 | 103 | 166 | 17 | 44 | 142 | -1 | -1 | -1 | -1 | |
| M4 | 8 | 10 | 1 | 3 | 8 | 156 | 17 | 44 | 142 | 1 | 102 | 159 | 8 | 9 | 1 | -1 | -1 | -1 | -1 | |
| M5 | 10 | 11 | 1 | 10 | 107 | 165 | 9 | 12 | 157 | 19 | 45 | 1 | 6 | 15 | 1 | -1 | -1 | -1 | -1 | |
| M6 | 5 | 14 | 1 | 1 | 106 | 166 | 6 | 13 | 1 | 11 | 17 | 158 | 1 | 16 | 158 | -1 | -1 | -1 | -1 | |
| T1 | 5 | 18 | 1 | 11 | 25 | 159 | 1 | 19 | 159 | 13 | 20 | 1 | 13 | 23 | 1 | -1 | -1 | -1 | -1 | |
| T2 | 18 | 46 | 158 | 12 | 21 | 157 | 5 | 24 | 1 | 0 | 0 | 0 | 13 | 22 | 1 | -1 | -1 | -1 | -1 | |
| T3 | 8 | 26 | 1 | 5 | 27 | 1 | 15 | 49 | 1 | 0 | 0 | 0 | 5 | 27 | 1 | -1 | -1 | -1 | -1 | |
| T4 | 20 | 47 | 1 | 21 | 48 | 1 | 21 | 48 | 1 | 10 | 29 | 1 | 0 | 0 | 0 | -1 | -1 | -1 | -1 | |
| N1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | |
| N2 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | |
| N3 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | |
| N4 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | |
| N5 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | -1 | |

Fonte: Elaborado pelo autor.

Portanto, cada solução corresponderá há uma matriz tridimensional, onde cada matriz bidimensional inserida, corresponde a uma turma das quais foram selecionada para a obtenção da otimização, resultando assim na figura 16.

Figura 16 – Conjunto de Solução Final

| Curso | | CONJUNTO DE SOLUÇÃO | | | | | | | | | | | | | | | | | | |
|-----------------|---|---------------------|---|---|-------|---|---|--------|---|---|--------|---|---|-------|---|---|--------|---|---|---|
| Curso | | CONJUNTO DE SOLUÇÃO | | | | | | | | | | | | | | | | | | |
| Curso | | CONJUNTO DE SOLUÇÃO | | | | | | | | | | | | | | | | | | |
| Turma | | SEGUNDA | | | TERÇA | | | QUARTA | | | QUINTA | | | SEXTA | | | SABADO | | | |
| P | D | S | P | D | S | P | D | S | P | D | S | P | D | S | P | D | S | P | D | S |
| HORÁRIOS DO DIA | | | | | | | | | | | | | | | | | | | | |
| M1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| M2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| M3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| M4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| M5 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| M6 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| T1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| T2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| T3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| T4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| N1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| N2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| N3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| N4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| N5 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

TURMAS

Fonte: Elaborado pelo autor.

Apesar de utilizada para o desenvolvimento interno do algoritmo, o conjunto de solução não é exibido desta forma para o usuário final. Para estes, é desenvolvido telas independentes de fácil visualização, como filtragem dos horários semanais por turma e professor.

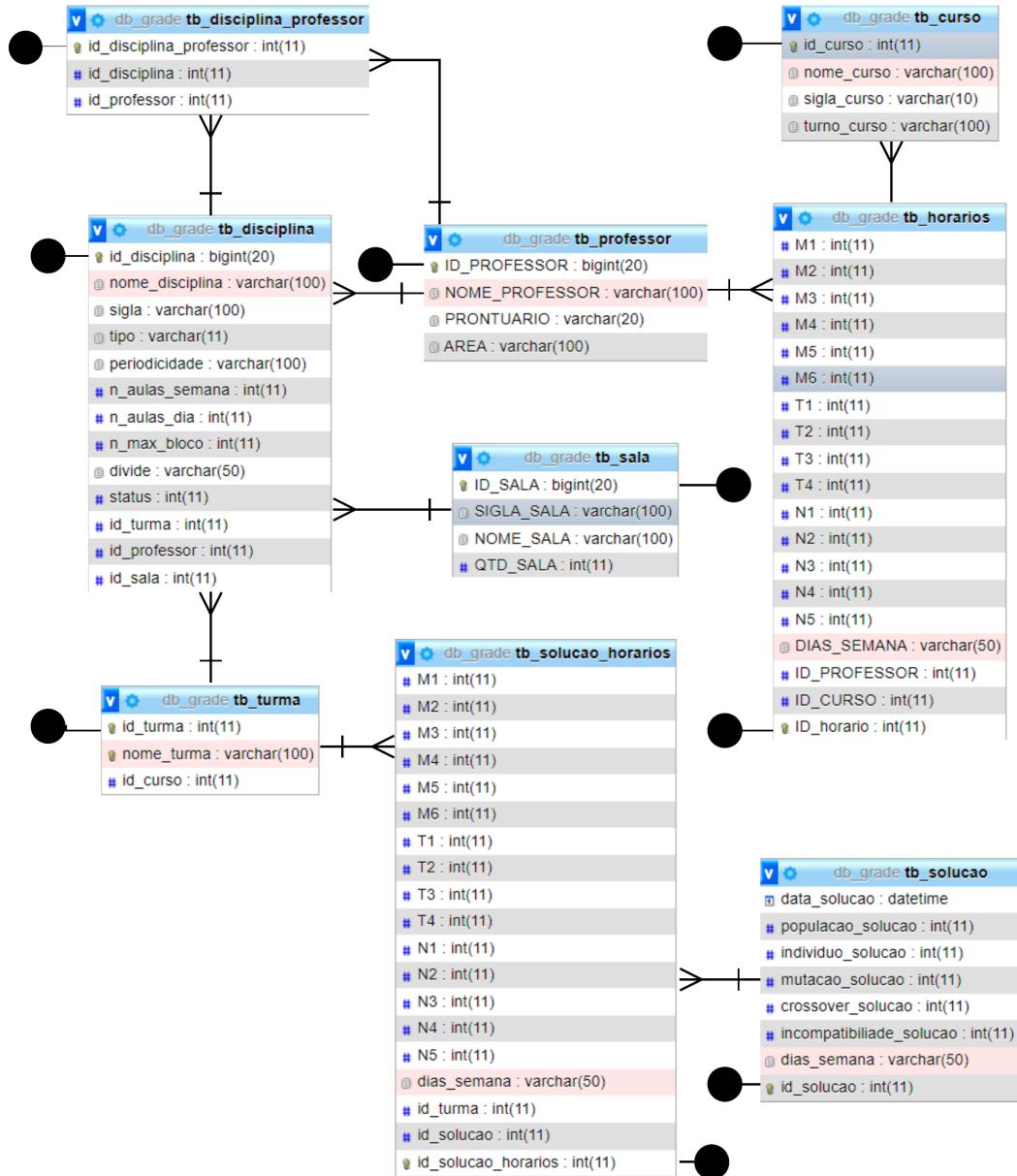
3.3.2 Banco de Dados

O Banco de Dados (BD) é utilizado para armazenar os dados pertinentes da geração dos horários. A Figura 17 mostra o modelo relacional representativo do banco de dados desenvolvido, baseando-se nos dados que a princípio são utilizados no problema. Nela contem todos os tipo de dados de cada tabela e suas relações subjacentes. Todos os dados foram inseridos dentro do BD WAMP.

A seguir é apresentado detalhadamente a utilidade de cada tabela:

- **Disciplina:** A tabela disciplina lista todas as disciplinas presentes de todos os cursos correspondentes, contendo suas siglas, nomes e informações quantitativas, depende diretamente do registro prévio de professor, sala e turma.
- **Professor:** A tabela professor lista todas as professores do campus, servirá para a atribuição de professoras na solução final, além de conter a informações de todas as áreas dos professores como informática, eletrônica, e etc.
- **Solução:** Contem às informações da solução obtida, tais como os operadores genéticos utilizados, e a data que foi obtida a solução.
- **Solução_Horários:** Contem a matriz de solução de cada conjunto de soluções geradas, contendo as informações sobre horários e seus respectivos dias da semana atribuídos a professor, disciplina e sala.
- **Turma:** Contém informações de identidade e nome de cada turma, assim como informações de nome e sigla.
- **Curso:** Contém todos os cursos do campus, e seu principal turno.
- **Horários:** Contem informações de horários(FPA) de cada professor, ou seja, sua disponibilidade semanal. Também contém as informações de disponibilidade dos cursos, registrando assim quando o curso possui ou não aula semanalmente.
- **Sala:** Contem dados de cada sala, usado para atribuição.
- **Disciplina_Professor** Persistência responsável por tratar dos casos em que há uma divisão de professores em paralelo, ou seja, mais de um professor é responsável pela mesma disciplina, contendo assim as disciplinas e seus respectivos professores.

Figura 17 – Modelo Relacional do Banco de Dados



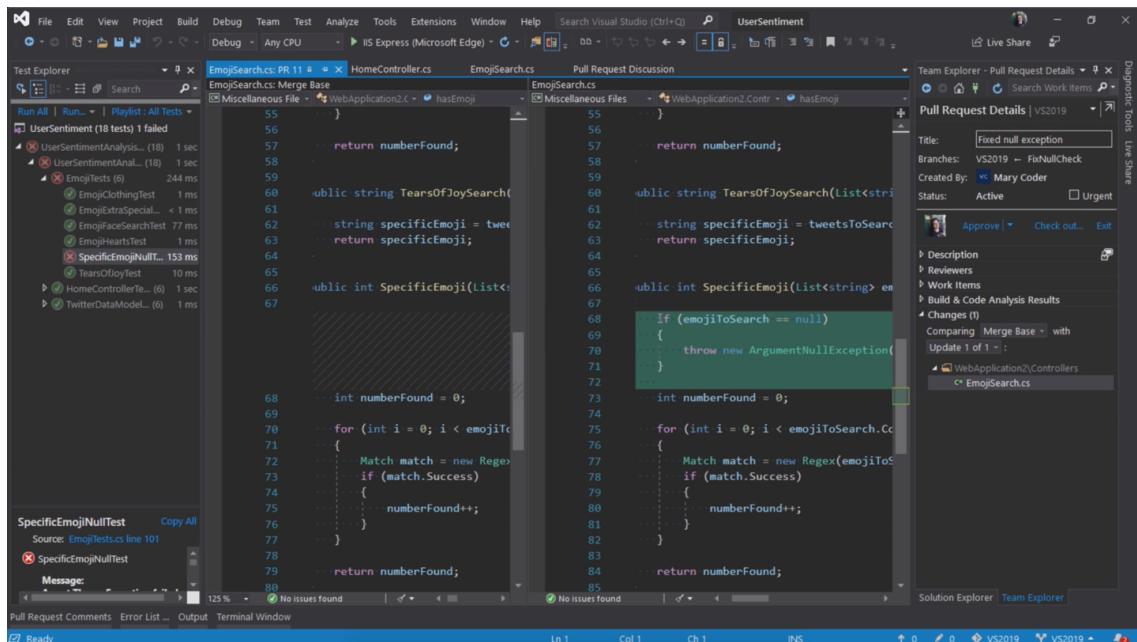
Fonte: Elaborado pelo autor.

Além do exposto, ao observar a figura 17, seguindo o relacionamento entre as tabelas do banco de dados, é possível verificar todos os docentes que são aptos a ministrar determinada disciplina, quais disciplinas possuem vínculo a determinado curso, as restrições que determinados docentes possuem, as áreas de responsabilidade com suas respectivas disciplinas, disciplinas que necessitam de salas específicas e as salas de aula do campus da universidade.

3.3.3 Visual Studio

Criado em 1997, o Microsoft Visual Studio é uma IDE proprietária da Microsoft para desenvolvimento de Software que utiliza frameworks como .NET Framework e às linguagens Visual Basic, C, C++, C# e F#.

Figura 18 – Visual studio Profissional 2019



Fonte: (MICROSOFT, 2019)

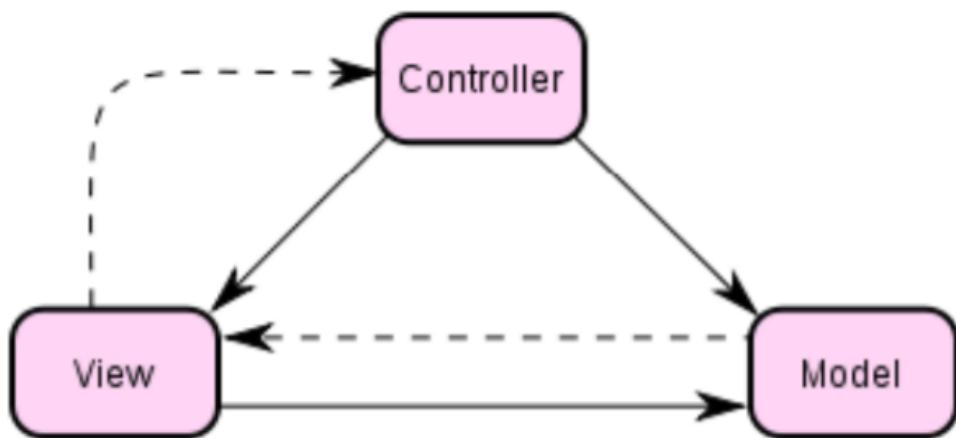
Além de possibilitar o desenvolvimento de interfaces intuitivas de uso, oferece a criação de telas visuais interativas e dinâmicas. No estudo apresentado foi utilizado a versão Community do Visual Studio 2019.

3.3.4 Prototipação em C#

A implementação do projeto foi baseada no padrão de arquitetura MVC(Model, View e Controller) que se baseia em 3 camadas. O modelo MCV surgiu na década de 80 e se tornou popular com aplicações WEB. O MVC tem um simples princípio a cerca de cada camada, as requisições são direcionadas para camada de Controller(Controle), no qual acessa a camada Model(Modelo) para processar a requisição, e por fim exibe os resultados na camada View (Visão). (LUCIANO; ALVES, 2017).

Para comunicação entre o banco de dados MySQL e o protótipo em C#, foi utilizada a biblioteca em *Wamp Server 3.2.6*, e para importação das informações de FPA que eram originárias de planilhas eletrônicas, foi utilizada a biblioteca *ExcelDataReader*.

Figura 19 – Divisão das camadas e suas relações



Fonte: Adaptado de (LUCIANO; ALVES, 2017)

Na figura 19 as linhas sólidas indicam associação direta e as tracejadas indicam associação indireta. Para um exemplificação mais detalhada podemos exemplificar que as operações de banco de dados como consultas de dados e informações se encontram na camada de *Model*, já o encapsulamento de dados e regras de negócio são contidos no *Controller*, por fim na camada *View* pode apresentar os dados ao usuário, juntamente das classes de interface do protótipo.

Este padrão de projeto, apesar de aumentar a abstração da construção do algoritmo, torna fácil a manutenção e desenvolvimento da aplicação, uma vez que pacotes modulares de rápido desenvolvimento podem ser utilizados. Além disto, os códigos desenvolvidos nesta arquitetura possuem otimização da velocidade entre as requisições feitas pelos comandos dos usuários, conforme apontado por Luciano e Alves (2017).

Abaixo a lista das classes utilizadas no protótipo:

3.3.5 Fluxo de execução

Uma vez arquitetado o formato de dados utilizados no projeto, através das telas visuais foi disponibilizado o registro e codificação das informações. Nesta etapa também houve o desenvolvimento da tela visual de execução do método.

Uma vez que os códigos da meta-heurística desenvolvidos na primeira etapa foram feitos em cima da linguagem do GNU Octave, uma transcrição foi feita para a linguagem C# de modo a viabilizar a execução neste ambiente.

Durante a execução, o algoritmo segue a sequência lógica ilustrada na Fi-

Tabela 2 – Classes do projeto e suas respectivas funções

| Classe | Função | Camada |
|------------------------|--|------------|
| Curso | Abstrair e Encapsular abstrações dos dados | Model |
| Disciplina | Abstrair e Encapsular abstrações dos dados | Model |
| Horario | Abstrair e Encapsular abstrações dos dados | Model |
| Professor | Abstrair e Encapsular abstrações dos dados | Model |
| Sala | Abstrair e Encapsular abstrações dos dados | Model |
| Turma | Abstrair e Encapsular abstrações dos dados | Model |
| CursoController | CRUD da classe Curso | Controller |
| DisciplinaController | CRUD da classe Disciplina | Controller |
| HorarioController | CRUD da classe Horario | Controller |
| ProfessorController | CRUD da classe Professor | Controller |
| SalaController | CRUD da classe Sala | Controller |
| TurmaController | CRUD da classe Turma | Controller |
| Listagem de Curso | Tela Visual | View |
| Listagem de Disciplina | Tela Visual | View |
| Listagem de Turma | Tela Visual | View |
| Listagem de Professor | Tela Visual | View |
| Listagem de Sala | Tela Visual | View |

Fonte: Elaborado pelo autor.

gura 1. Uma vez registrado todas informações pertinentes necessárias, é selecionado a quantidade de turmas a serem compiladas no programa, com isso é executado a solução do programa, codificando as informações e gerando uma solução inicial aleatória. O processo é repetido até obter uma população inicial de forma aleatória, os operadores de cruzamento e mutação são aplicados para gerar novos indivíduos, que são avaliados através da função objetivo e acrescidos a uma nova população. Dessa forma, a população corrente é totalmente substituída pela nova população através das iterações. Este processo se repetiu de acordo com o número de gerações definido no campo visual ou através de uma instrução forçada para parar a execução, resultando, portanto, no fim da busca por uma solução aceitável para o problema.

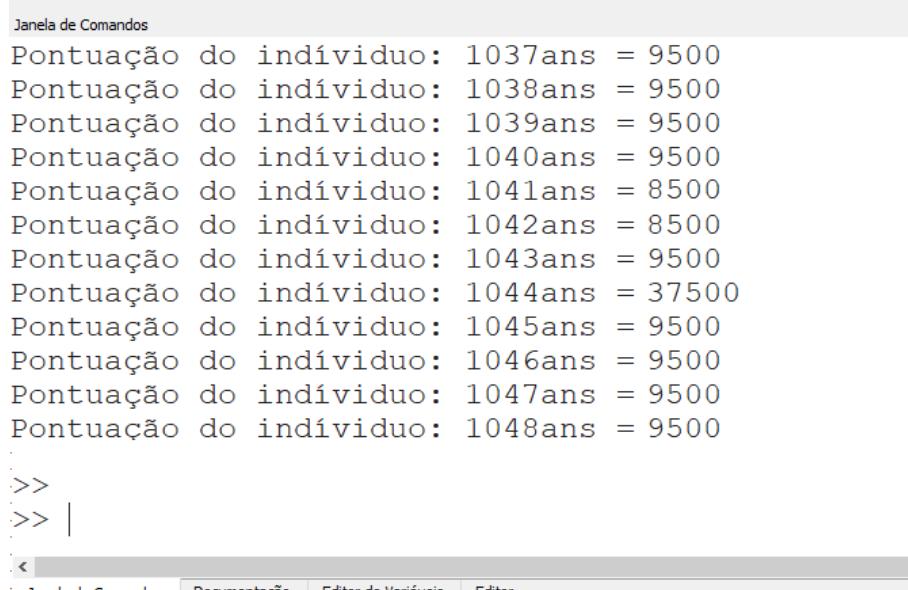
4 RESULTADOS

4.1 PRIMEIRA ETAPA

Nos testes realizados sobre os resultados desta etapa, o algoritmo foi aplicado a dois cursos, sendo 4 turmas do curso de Automação Industrial e 3 turmas do curso de Mecatrônica. Ambos os cursos são noturnos, com aula de segunda a sexta-feira, sendo que as turmas da mecatrônica possuem aulas também aos sábados de manhã. A solução inicial do problema é gerada de forma aleatória, conforme figura 14.

Durante a execução do algoritmo, uma tela de acompanhamento visual foi disponibilizada, de como a observar em tempo real a qualidade dos novos indivíduos gerados, assim, foi possível observar quando o comportamento do algoritmo, além de possibilitar visualizar ao obter um indivíduo minimamente factível, bastando que o mesmo tenha uma função objetivo < 10.000 pontos, conforme observado na figura 20.

Figura 20 – Função objetivo de cada indivíduo durante a execução.



```

Janela de Comandos
Pontuação do indivíduo: 1037ans = 9500
Pontuação do indivíduo: 1038ans = 9500
Pontuação do indivíduo: 1039ans = 9500
Pontuação do indivíduo: 1040ans = 9500
Pontuação do indivíduo: 1041ans = 8500
Pontuação do indivíduo: 1042ans = 8500
Pontuação do indivíduo: 1043ans = 9500
Pontuação do indivíduo: 1044ans = 37500
Pontuação do indivíduo: 1045ans = 9500
Pontuação do indivíduo: 1046ans = 9500
Pontuação do indivíduo: 1047ans = 9500
Pontuação do indivíduo: 1048ans = 9500

>>
>> |

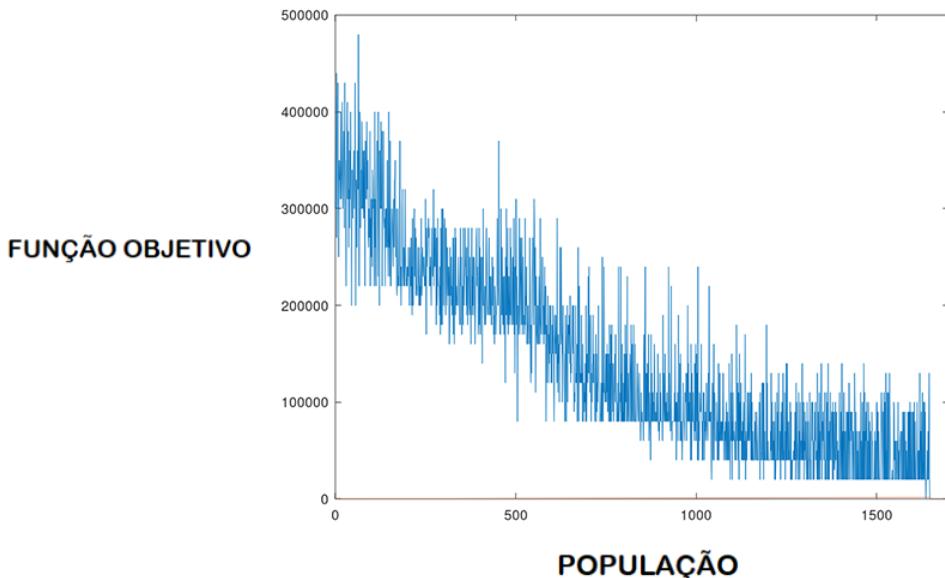
```

Janela de Comandos Documentação Editor de Variáveis Editor

Fonte: Elaborado pelo autor.

A Figura 21 exibe graficamente o comportamento da função objetivo para cada novo indivíduo gerado. É observado que a qualidade da solução é melhorada a cada iteração, até estabilizar, sendo atingida uma solução factível após cerca de 75 gerações. Em média, o tempo demandado para obtenção da solução foi de cerca de 6 minutos.

Um exemplo de solução encontrada é mostrado na Figura 22. A solução é apresentada de forma codificada, pois nesta etapa a decodificação das soluções não

Figura 21 – Função objetivo de cada indivíduo após a execução.

Fonte: Elaborado pelo autor.

foi implementada, a segunda etapa é a responsável por disponibilizar as informações de forma visual. Não há ocorrência de incompatibilidades graves na solução proposta, somente de aulas atribuídas em horário fora do horário preferencial dos professores.

Figura 22 – Solução obtida após 75 gerações, sendo o 1521º Individuo.

| | SEGUNDA | | | TERÇA | | | QUARTA | | | QUINTA | | | SEXTA | | | SABADO | | |
|----|---------|-----|-----|-------|-----|-----|--------|----|-----|--------|-----|-----|-------|-----|-----|--------|-----|-----|
| | P | D | S | P | D | S | P | D | S | P | D | S | P | D | S | P | D | S |
| T1 | 3 | 101 | 156 | 3 | 6 | 1 | 2 | 5 | 1 | 15 | 105 | 157 | 16 | 4 | 1 | -1 | -1 | -1 |
| T1 | 2 | 100 | 1 | 5 | 2 | 1 | 1 | 3 | 1 | 4 | 104 | 156 | 4 | 1 | 1 | -1 | -1 | -1 |
| T2 | 7 | 7 | 158 | 7 | 7 | 158 | 3 | 8 | 156 | 6 | 103 | 166 | 17 | 44 | 142 | -1 | -1 | -1 |
| T2 | 8 | 10 | 1 | 3 | 8 | 156 | 17 | 44 | 142 | 1 | 102 | 159 | 8 | 9 | 1 | -1 | -1 | -1 |
| T3 | 10 | 11 | 1 | 10 | 107 | 165 | 9 | 12 | 157 | 19 | 45 | 1 | 6 | 15 | 1 | -1 | -1 | -1 |
| T3 | 5 | 14 | 1 | 1 | 106 | 166 | 6 | 13 | 1 | 11 | 17 | 158 | 1 | 16 | 158 | -1 | -1 | -1 |
| T4 | 5 | 18 | 1 | 11 | 25 | 159 | 1 | 19 | 159 | 13 | 20 | 1 | 13 | 23 | 1 | -1 | -1 | -1 |
| T4 | 18 | 46 | 158 | 12 | 21 | 157 | 5 | 24 | 1 | 0 | 0 | 0 | 13 | 22 | 1 | -1 | -1 | -1 |
| T5 | 8 | 26 | 1 | 5 | 27 | 1 | 15 | 49 | 1 | 0 | 0 | 0 | 5 | 27 | 1 | 15 | 109 | 1 |
| T5 | 20 | 47 | 1 | 21 | 48 | 1 | 21 | 48 | 1 | 10 | 29 | 1 | 7 | 28 | 1 | 7 | 108 | 156 |
| T6 | 4 | 30 | 1 | 12 | 31 | 1 | 13 | 32 | 1 | 10 | 35 | 1 | 12 | 111 | 156 | 6 | 34 | 1 |
| T6 | 13 | 32 | 1 | 6 | 34 | 1 | 2 | 37 | 159 | 5 | 36 | 1 | 10 | 110 | 165 | 8 | 33 | 1 |
| T7 | 14 | 43 | 157 | 9 | 38 | 157 | 4 | 39 | 158 | 14 | 42 | 1 | 11 | 40 | 157 | 8 | 41 | 157 |
| T7 | 14 | 43 | 157 | 4 | 39 | 158 | 8 | 41 | 157 | 14 | 42 | 1 | 11 | 40 | 157 | 9 | 38 | 157 |

Fonte: Elaborado pelo autor.

Conforme observado na Figura 21, ao obter populações e indivíduos iniciais, a função objetivo fica em torno de 400.000, ao ser obtida novas gerações os novos indivíduos melhoram, justamente através das alterações dada pelo modelo metaheurístico.

A solução da Figura 22 é factível por não ter incompatibilidade grave, somente de atribuição fora de horário preferencial dos professores, resultando em uma função objetivo de apenas 4.500 pontos.

Os dados obtidos na figura acima, correspondem à valores pré- definidos em uma tabela de referência, para cada professor, disciplina e sala de aula é associado um número único, conforme definido nas figura 12 e 13. O número -1 associado à

tabela corresponde a não associação de informação, portanto, não houve necessidade organização e registro de informação.

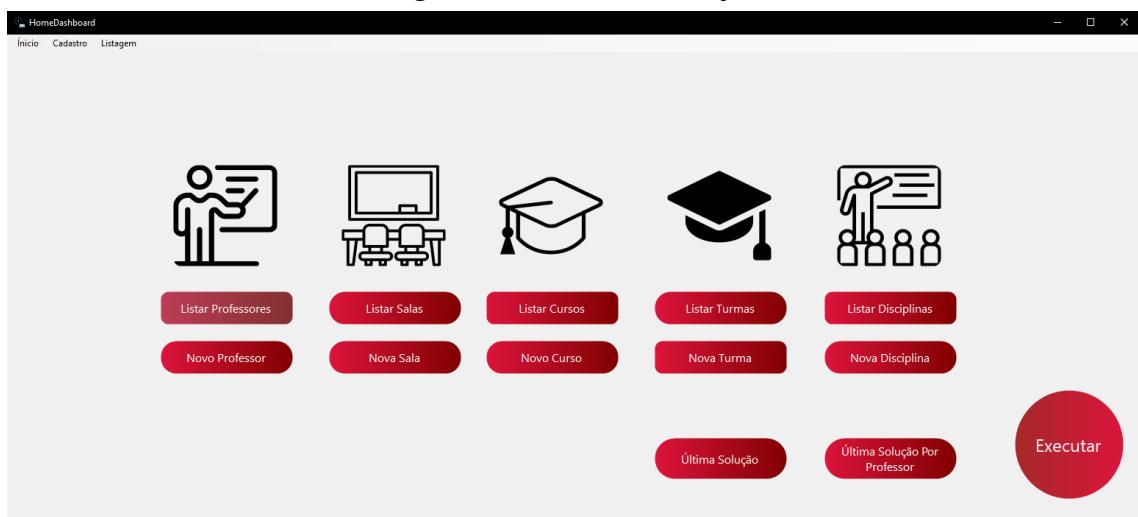
4.2 SEGUNDA ETAPA

4.2.1 Telas Visuais

O protótipo possui um módulo principal que possibilita acesso à todas telas do programa, conforme observado na figura 23, através desta tela é liberado o acesso nos demais módulos de cadastro de informação, onde são inseridas no banco de dados as informações referentes aos cursos, áreas de responsabilidade, docentes, disciplinas, salas e turnos.

Também há uma área responsável pela vinculação das disciplinas aos cursos, vinculação dos docentes com as disciplinas aos quais possuem afinidade e cadastramento por parte dos docentes de quais os dias e horários os mesmos não possuem disponibilidade para ministrar aulas. Por último, o protótipo possui o módulo onde o AG é parametrizado e executado para construir a grade horária.

Figura 23 – Tela Principal



Fonte: Elaborado pelo autor.

A figura 24 exibe a interface de registro de cada curso. Na área superior há os campos de nome, sigla e turno. Logo abaixo há a tabela de horário de funcionamento do curso, todas turmas registradas para o curso em específico, herdam as informações de seus horários, além disto, somente serão atribuídas disciplinas para locais onde há disponibilidade.

A figura 25 exibe a tela de listagem e busca dos cursos cadastrados, possibilitando a consulta de qualquer curso. As informações são listadas através das colunas, código, nome, sigla e turno, respectivamente. Através de um clique duplo em um curso

Figura 24 – Tela de Cadastro de Curso

| | | SEGUNDA | TERÇA | QUARTA | QUINTA | SEXTA | SÁBADO |
|----|-------------|---------|-------|--------|--------|-------|--------|
| M1 | 07:10-08:00 | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ |
| M2 | 08:00-08:50 | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ |
| M3 | 08:50-09:40 | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ |
| M4 | 10:00-10:50 | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ |
| M5 | 10:50-11:40 | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ |
| M6 | 11:40-12:30 | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ |
| T1 | 13:30-14:20 | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ |
| T2 | 14:20-15:10 | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ |
| T3 | 15:30-16:20 | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ |
| T4 | 16:20-17:10 | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ |
| N1 | 18:00-18:50 | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ |
| N2 | 19:00-19:50 | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ |
| N3 | 19:50-20:40 | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ |
| N4 | 20:55-21:45 | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ |
| N5 | 21:45-22:35 | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ |

Legenda : ✗ Indisponível ✓ Disponível

Salvar

Fonte: Elaborado pelo autor.**Figura 25 – Tela de Listagem Curso**

| Código | Nome do Curso | Sigla | Turno |
|--------|--|--------|-------|
| 15 | Tecnologia em Mecatrônica Industrial | TME | Noite |
| 14 | Técnico em Comércio Integrado ao Ensino Médio-EJA | PROEJA | Noite |
| 13 | Licenciatura em Matemática | MAT | Noite |
| 12 | Licenciatura em Física | FIS | Noite |
| 10 | Técnico em Administração Integrado ao Ensino Médio | ADM_I | Manha |
| 9 | Técnico em Administração | ADM | Noite |
| 8 | Engenharia de Computação | ENG | Tarde |
| 11 | Técnico em Automação Industrial | AUT | Noite |
| 16 | Tecnologia em Sistemas para Internet | TSI | Noite |
| 17 | Técnico em Informática Integrado ao Ensino Médio | INF_I | Manha |

Consultar Curso Cancelar

Fonte: Elaborado pelo autor.

selecionado, é exibido a tela de edição conforme a figura 24, porém desta vez com as informações preenchidas do curso.

A figura 26 exibe a tela de cadastro de cada disciplina. Para realizar o cadastro, é necessário que haja um prévio registro de professores, turmas e salas. Na

Figura 26 – Tela de Cadastro Disciplina

The screenshot shows a Windows application window titled 'F_CadastroDisciplina'. The interface includes fields for 'Nome' (Name), 'Sigla' (Code), 'Tipo' (Type), and 'Periodicidade' (Periodicity). Below these are dropdowns for 'Nº Aulas Semana' (Classes per week), 'Nº Aulas por Dia' (Classes per day), 'Nº Max de Blocos' (Max blocks), and 'A turma será fisicamente dividida?' (Will the class be physically divided?). Further down are dropdowns for 'Turma' (Class), 'Professor' (Teacher), and 'Sala' (Room). A section for 'Disciplina ofertada?' (Offered discipline) has a dropdown set to 'SIM'. At the bottom are 'Cancelar' (Cancel) and 'Salvar' (Save) buttons.

Fonte: Elaborado pelo autor.

área superior há informações de nome, sigla, tipo (teórica ou prática) e periodicidade respectivamente. Abaixo há informação da relação da quantidade de aula na semana, aulas por dia, a quantidade máxima de blocos para atribuir esta disciplina, além de informar se a disciplina será dividida ou não, caso sim, informar se a divisão é feita em conjunto com outra disciplina ou com professores em paralelo no mesmo dia. Logo abaixo, há espaço para a seleção da turma, professor e sala responsável por a disciplina.

Figura 27 – Tela de Listagem Disciplina

The screenshot shows a Windows application window titled 'F_ListagemDisciplina'. It features a search bar with 'Nome:' and a 'Pesquisar' (Search) button. Below is a table with columns: Código (Code), Disciplina (Name), Sigla (Code), Turma (Class), and Professor (Teacher). The table lists various disciplines with their respective codes and names. At the bottom are 'Consultar Disciplina' (View Discipline) and 'Cancelar' (Cancel) buttons.

| Código | Disciplina | Sigla | Turma | Professor |
|--------|-------------------------------------|-------|-------|-----------|
| 1 | ADMINISTRAÇÃO GERAL | ADMA1 | | |
| 2 | CONTABILIDADE | CONA1 | | |
| 3 | ECONOMIA E MERCADOS | ECMA1 | | |
| 4 | FUNDAMENTOS DE MATEMÁTICA ELEMENTAR | FMEA1 | | |
| 5 | INFORMÁTICA BÁSICA | INBA1 | | |
| 6 | METODOLOGIA DO TRABALHO CIENTÍFICO | MTCA1 | | |
| 7 | REDAÇÃO EMPRESARIAL | REMA1 | | |
| 8 | ADMINISTRAÇÃO DA PRODUÇÃO | ADPA2 | | |
| 9 | ADMINISTRAÇÃO MERCADOLÓGICA | AMCA2 | | |
| 10 | CUSTOS | CSTA2 | | |
| 11 | DIREITO E LEGISLAÇÃO TRABALHISTA | DLTA2 | | |
| 12 | INFORMÁTICA APLICADA | INFA2 | | |
| 13 | INGLÊS INSTRUMENTAL I | INGA2 | | |
| 14 | MATEMÁTICA FINANCEIRA | MAFA2 | | |
| 15 | TECNÍCIAS COMERCIAIS | TCMA2 | | |
| 16 | COMÉRCIO EXTERIOR | CFXA3 | | |

Fonte: Elaborado pelo autor.

A figura 27 exibe a tela de listagem da disciplina, onde contém dados como código, nome e sigla, respectivamente. De maneira análoga as demais telas visuais, com um duplo clique ou apertando o botão "Consulta Disciplina" é possível editar uma disciplina, exibindo assim a tela da figura 26, porém com os dados herdados da disciplina escolhida na tela anterior.

É possível observar através da figura 28 a interface onde são cadastradas as informações dos docentes. Na área superior há as informações principais dos docen-

Figura 28 – Tela de Cadastro Professor

The screenshot shows a software window titled 'F_CadastroProfessor_'. At the top left is a logo of a computer monitor with a bar chart. The top right has standard window controls (minimize, maximize, close). Below the title bar are input fields: 'Nome Completo' with 'Professor Eletronica 1' typed in, 'Prontuário' with 'B100000' typed in, 'Área' with 'Eletronica' typed in, and a 'Carregar FPA' button with a blue 'Selecionar ...' placeholder. A toggle switch labeled 'Afastado' is set to off. The main area is a grid representing weekly availability. The columns are labeled 'SEGUNDA', 'TERÇA', 'QUARTA', 'QUINTA', 'SEXTA', and 'SÁBADO'. The rows list time intervals from M1 (07:10-08:00) to N5 (21:45-22:35). Each cell contains either a red 'X' (Indisponível) or a green checkmark (Disponível). A legend at the bottom left defines the symbols. A large green 'Salvar' button is at the bottom right.

| | SEGUNDA | TERÇA | QUARTA | QUINTA | SEXTA | SÁBADO |
|----------------|---------|-------|--------|--------|-------|--------|
| M1 07:10-08:00 | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ |
| M2 08:00-08:50 | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ |
| M3 08:50-09:40 | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ |
| M4 10:00-10:50 | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ |
| M5 10:50-11:40 | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ |
| M6 11:40-12:30 | ✓ | ✓ | ✓ | ✗ | ✗ | ✗ |
| T1 13:30-14:20 | ✓ | ✓ | ✓ | ✗ | ✗ | ✗ |
| T2 14:20-15:10 | ✓ | ✓ | ✓ | ✗ | ✗ | ✗ |
| T3 15:30-16:20 | ✓ | ✓ | ✓ | ✗ | ✗ | ✗ |
| T4 16:20-17:10 | ✓ | ✓ | ✗ | ✓ | ✗ | ✗ |
| N1 18:00-18:50 | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ |
| N2 19:00-19:50 | ✗ | ✓ | ✓ | ✓ | ✓ | ✗ |
| N3 19:50-20:40 | ✗ | ✓ | ✓ | ✓ | ✓ | ✗ |
| N4 20:55-21:45 | ✗ | ✓ | ✓ | ✓ | ✓ | ✗ |
| N5 21:45-22:35 | ✗ | ✓ | ✓ | ✓ | ✓ | ✗ |

Legenda : ✗ Indisponível ✓ Disponível

Salvar

Fonte: Elaborado pelo autor.

tes, seu nome, prontuário e área , respectivamente. Há espaço para o carregamento do arquivo de FPA.

O arquivo de FPA, é um modelo preenchido semestralmente pelos docentes, no qual é informado sobre sua relação de disponibilidade semanal. Uma vez carregado, o software se responsabiliza por ler as informações e traduzir para a tabela de disponibilidade de horário. Da mesma forma a atualização pode ser feita através de clique manual. Os dias da semana no qual o docente possui restrições de horário são marcados em vermelho, os disponíveis são listados em verde, conforme a legenda abaixo da tabela. Em seguida clica-se em “Salvar”.

Na figura 29 exibe a consulta da relação dos professores disponíveis, através das colunas de código do professor, nome, prontuário e área. Além disso, exibe a possibilidade de filtragem de docente em específico. Na parte inferior, há o botão de acesso as informações detalhadas do professor, onde a tela da figura 28 é exibida com as informações do docente escolhido anteriormente.

A figura 30 apresenta o cadastro de cada sala, através desta tela é registrado a quantidade de salas e sua respectiva sigla.

Figura 29 – Tela de Consulta Professor

| Código | Nome | Prontuário | Área |
|--------|--------------|------------|---------------|
| 1 | Professor 1 | BI135574 | física |
| 2 | Professor 2 | BI14633X | Educacao |
| 3 | Professor 3 | BI111028 | Eletronica |
| 4 | Professor 4 | BI137133 | Informatica |
| 5 | Professor 5 | BI223281 | física |
| 6 | Professor 6 | SC205473 | Administracao |
| 7 | Professor 7 | BI243395 | Informatica |
| 8 | Professor 8 | BI101266 | Administracao |
| 9 | Professor 9 | BI215053 | Administracao |
| 10 | Professor 10 | BI209028 | Artes |
| 11 | Professor 11 | BI205151 | Administracao |
| 12 | Professor 12 | RI241568 | Eletronica |

Consultar professor FPA **Cancelar**

Fonte: Elaborado pelo autor.

Figura 30 – Tela de Cadastro Sala

Nome:

Quantidade:

Sigla:

Salvar

Fonte: Elaborado pelo autor.

Figura 31 – Tela de Listagem Sala

| Código | Sala | Sigla | Quantidade |
|--------|--|----------------|------------|
| 2 | Atelie de Artes | artes | 1 |
| 3 | Laboratorio de eletronica e instrumentacao | B156 | 1 |
| 4 | Laboratorio de dispositivos programaveis | B157 | 1 |
| 5 | Laboratorio de electricidade e comandos eletricos | B158 | 1 |
| 6 | Laboratorio de redes industriais | B159 | 1 |
| 7 | Laboratorio de processos de fabricacao | B165 | 1 |
| 8 | Laboratorio de hidraulica e pneumatica | B166 | 1 |
| 9 | Laboratorio de manufatura auxiliada por computador e ensaios ... | B168 | 1 |
| 10 | E-Lab1 (40 computadores) | E-Lab1 | 1 |
| 11 | E-LabInfo (20 computadores) | E-LabInfo | 3 |
| 12 | E-LabNote (Notebooks) | E-LabNote | 1 |
| 13 | Laboratorio Multidisciplinar de Ensino de Fisica | Lab Ens Fisica | 1 |
| 14 | Laboratorio de Fisica | lab Fisica | 2 |
| 15 | Laboratorio Ciencias da Natureza | Lab_Cie | 1 |
| 16 | Laboratorio de economia solidaria | LABSOL | 1 |
| 17 | Laboratorio de economia solidaria | I ARSOI | 1 |

Consultar Sala **Cancelar**

Fonte: Elaborado pelo autor.

Na listagem de sala é possível consultar cada sala pelo seu nome, sigla e quantidade pressionando o botão "Pesquisar". Igualmente é possível editar a sala realizando um duplo clique ou o botão de "Consulta Sala".

Para o cadastro de turmas, mostrado na 32, é necessário que haja um prévio cadastro de professores e disciplinas. Só então é possível fazer o registro da turma informando um nome, a quantidade de vagas, o professor regente, o curso e relacionar as disciplinas dessa turma com o professor responsável por lecioná-las

Figura 32 – Tela de Cadastro Turma

The screenshot shows a Windows-style application window titled "Cadastro de Turma". Inside, there are three input fields: "Nome*" (Name) with a placeholder "-----", "Curso*" (Course) with a dropdown menu showing "-----", and "Sigla" (Abbreviation) with a dropdown menu showing "-----". On the right side of the window is a large red button labeled "Salvar" (Save).

Fonte: Elaborado pelo autor.

Na listagem de turma é possível consultar cada turma pelo seu nome, sigla e quantidade pressionando o botão "Pesquisar". Igualmente é possível editar a turma realizando um duplo clique ou o botão de "Consulta Sala".

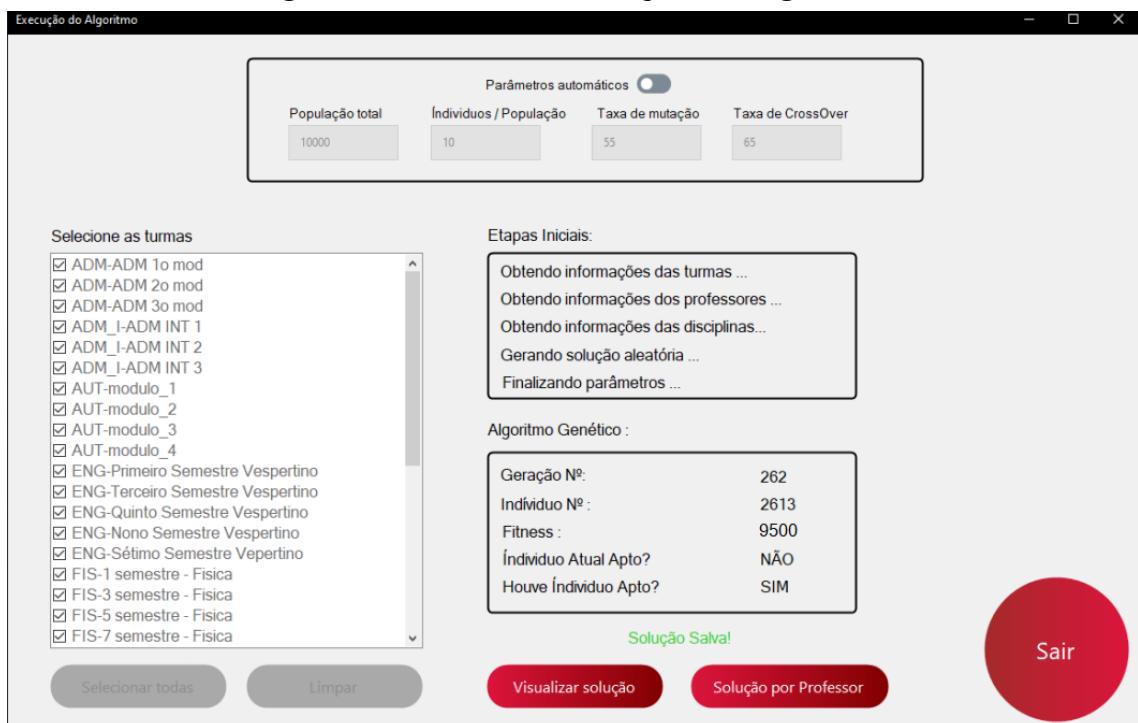
Após o registro devido das persistências de dados necessárias para aplicação do método, conforme a figura 17, o algoritmo pode ser executado através do acesso na tela da figura 34, na área superior há os parâmetros do algoritmo a ser executado, definido pelo usuário, com a opção de definição automática, no lado esquerdo há a lista com seleção das turmas a serem obtidas soluções, uma vez clicado no botão vermelho executar, o algoritmo realiza a codificação das informações armazenadas referentes as turmas, disciplinas, docentes e salas. Após a codificação ser encerrada, a etapa do AG se inicia, exibindo de forma visual o crescimento populacional e a qualidade de cada individuo gerado, o critério de parada é definido através da instrução forçada do usuário no botão vermelho, ou através da quantidade máxima de

Figura 33 – Tela de listagem Turma

The screenshot shows a Windows-style application window titled "F_ListagemTurma". At the top, there is a search bar with the placeholder "Nome :". Below it is a table with three columns: "Código" (Code), "Turma" (Class), and "Curso" (Course). The table contains 18 rows of data. At the bottom of the window are three buttons: "Pesquisar" (Search), "Consultar Turma" (View Class), and "Cancelar" (Cancel).

| Código | Turma | Curso |
|--------|----------------------------------|--|
| 3 | ADM-ADM 1o mod | Técnico em Administração |
| 4 | ADM-ADM 2o mod | Técnico em Administração |
| 5 | ADM-ADM 3o mod | Técnico em Administração |
| 6 | ADM_I-ADM INT 1 | Técnico em Administração Integrado ao Ensino Médio |
| 7 | ADM_I-ADM INT 2 | Técnico em Administração Integrado ao Ensino Médio |
| 8 | ADM_I-ADM INT 3 | Técnico em Administração Integrado ao Ensino Médio |
| 9 | AUT-modulo_1 | Técnico em Automação Industrial |
| 10 | AUT-modulo_2 | Técnico em Automação Industrial |
| 11 | AUT-modulo_3 | Técnico em Automação Industrial |
| 12 | AUT-modulo_4 | Técnico em Automação Industrial |
| 13 | ENG-Primeiro Semestre Vespertino | Engenharia de Computação |
| 14 | ENG-Terceiro Semestre Vespertino | Engenharia de Computação |
| 15 | ENG-Quinto Semestre Vespertino | Engenharia de Computação |
| 16 | ENG-Nono Semestre Vespertino | Engenharia de Computação |
| 17 | ENG-Sétimo Semestre Vespertino | Engenharia de Computação |
| 18 | FIS-1 semestre - Física | Ens. Científico em Física |

Fonte: Elaborado pelo autor.

Figura 34 – Tela de Execução do algoritmo

Fonte: Elaborado pelo autor.

população definido na área superior da tela. Uma vez obtido indivíduos aptos, ao algoritmo ser finalizado, o melhor individuo é salvo. Então é liberado o acesso visual à solução obtida, o acesso pode ser feito através dos botões inferiores vermelhos nas figuras 34 e 23.

4.3 DISCUSSÃO

Durante a primeira etapa, com os dados obtidos, características comportamentais do algoritmo foram observadas. O desempenho resultante na maioria dos casos não foi afetado pela má qualidade da solução inicial, sempre convergindo para soluções minimamente factíveis, além disto, o algoritmo obteve solução em um tempo consideravelmente rápido, uma vez que, estava voltado a somente dois cursos do campus, ainda assim, evidenciando a boa forma de codificação e desenvolvimento de código utilizado.

Ainda na etapa, para efeito de comparação, a solução implementada manualmente no semestre com as mesmas informações utilizadas, obteve uma função objetivo de 6.500 pontos, infringindo assim cerca de 65% dos horários preferenciais dos professores, a solução em questão, infringiu apenas 45%.

5 CONCLUSÃO

Durante a primeira etapa, os resultados foram obtidos através de simulações de horários geradas pelo algoritmo proposto no presente trabalho, que foram comparados aos horários montados anteriormente pelo coordenador do curso.

Conforme observado no gráfico da figura 21, é possível observar a variação de aptidão dos indivíduos durante a evolução ao longo das gerações. Além disso, é notório a evolução do algoritmo através das iterações, obtendo assim soluções melhores do que as iniciais, porém tendo problemas em otimizar, uma vez que chegou em soluções cuja incompatibilidades eram menores.

Esta etapa foi de crucial importância para o projeto, pois uma vez comprovado a eficácia utilizada na abordagem de solução para o problema em questão, bastou-se expandir o algoritmo de modo a suportar todos os cursos do campus, além da criação de uma interface visual para possibilitar o uso do software de maneira interativa para usuário final.

Durante a segunda etapa, a utilização de banco de dados para o registro de informação, ofereceu a possibilidade de utilização futura do software para demais semestres do ano. Nesta etapa houve visível problema de convergência de solução, uma vez que a quantidade de restrições foram expandidas, além das turmas utilizadas, foi possível observar o aumento de tempo para obtenção de uma solução no programa, em alguns casos o algoritmo entrava em um looping infinito, devido a quantidade de restrições de entrada aplicadas, apesar disto o algoritmo continuou com o comportamento de convergência mostrado na figura 21.

Desta forma, foi possível atingir os objetivos gerais e específicos do presente trabalho, além disso, os resultados obtidos demonstram que o algoritmo genético possui boa convergência e é capaz de solucionar o problema de geração de grade horária de forma rápida e precisa.

O modelo proposto é passível de ser aplicado tanto em instituições de ensino públicas quanto privadas em que futuramente, o modelo poderá ser estendido para instituições de ensino superior que trabalham no regime seriado.

REFERÊNCIAS

- ALMEIDA, M. W. d. S. **Utilização de algoritmos genéticos para montagem de horários acadêmicos com foco na blocagem de horários.** 22-23 p. Trabalho de Conclusão de Curso, 2015.
- ASC, T. **ASC.** 2022. Disponível em: <<https://www.asctimetables.com/>>. Acesso em: 22 de Outubro de 2021.
- BABAEI, H.; KARIMPOUR, J.; HADIDI, A. A survey of approaches for university course timetabling problem. **Computers & Industrial Engineering**, Elsevier, v. 86, p. 43–59, 2015.
- CHEN, R.-M.; SHIH, H.-F. Solving university course timetabling problems using constrictive particle swarm optimization with local search. **Algorithms**, Multidisciplinary Digital Publishing Institute, v. 6, n. 2, p. 227–244, 2013.
- CHRONOS. **Chronos.** 2022. Disponível em: <<https://www.cronostimetable.com/>>. Acesso em: 22 de Outubro de 2021.
- CISCON, L. A.; ALVARENGA, G. O problema de geração de horários: um foco na eliminação de janelas e aulas isoladas. In: **XXXVII Brazilian Symposium of Operational Research.** [S.l.: s.n.], 2005.
- EVEN, S.; ITAI, A.; SHAMIR, A. On the complexity of time table and multi-commodity flow problems. In: **16th Annual Symposium on Foundations of Computer Science (sfcs 1975).** [S.l.: s.n.], 1975. p. 184–193.
- FORGE, O. **Octave Forge: project samples.** Disponível em: <<https://sourceforge.net/projects/octave/>>. Acesso em: 21 de Outubro de 2021.
- GLOVER, F. Future paths for integer programming and links to artificial intelligence. **Computers & operations research**, Elsevier, v. 13, n. 5, p. 533–549, 1986.
- GOLDBERG, D. E. Genetic algorithms in search, optimization, and machine learning. **Reading: Addison-Wesley Publishing Company**, p. 412, 1989.
- GOTLIEB, C. C. The construction of class-teacher time-tables. **IFIP CONGRESS**, v. 13, p. 73–77, 1963. Amsterdam: North-Holland Publishing.
- GOUVEIA, L. B.; ARAÚJO, A. C. M. d.; TEIXEIRA, J. d. F.; SANTOS, G. T. Algoritmos genéticos: Aplicando a teoria a um estudo de caso / genetic algorithms: Applying theory to a case study. **Brazilian Journal of Development**, v. 7, n. 3, p. 21053–21077, 2021. ISSN 25258761, 25258761. Disponível em: <<https://www.brazilianjournals.com/index.php/BRJD/article/view/25510/20315>>.

- GRIDCLASS. **GridClass**. 2022. Disponível em: <<http://gridclass.com.br/>>. Acesso em: 22 de Outubro de 2021.
- HAMAWAKI, C. D. L. et al. Geração automática de grade horária usando algoritmos genéticos: o caso da faculdade de engenharia elétrica da ufu. Universidade Federal de Uberlândia, 2005.
- HERTZ, A. Finding a feasible course schedule using tabu search. **Discrete Applied Mathematics**, Elsevier, v. 35, n. 3, p. 255–270, 1992.
- HOLLAND, J. H. **Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence**. [S.I.]: MIT press, 1992.
- JONG, K. A. D. **An Analysis of the Behavior of a Class of Genetic Adaptive Systems**. Tese (Doutorado), USA, 1975. AAI7609381.
- KATOCH, S.; CHAUHAN, S. S.; KUMAR, V. A review on genetic algorithm: past, present, and future. **Multimedia Tools and Applications**, v. 80, n. 5, p. 8091–8126, Feb 2021. ISSN 1380-7501, 1573-7721. Disponível em: <<http://link.springer.com/10.1007/s11042-020-10139-6>>.
- KOTSKO, E. G. da S.; STEINER, D. M. T. A. Otimização na construção da grade horária escolar—uma aplicação—. 2003.
- LINDEN, R. **ALGORITMOS GENETICOS**. 3. ed. [S.I.]: CIENCIA MODERNA, 2012. v. 1. (3, v. 1). ISBN 9788539901951.
- LUCAS, D. C. **Algoritmos Genéticos: um estudo de seus conceitos fundamentais e aplicação no problema de grade horária**. Trabalho de Conclusão de Curso — Bachalerado em Informática, 2000.
- LUCIANO, J.; ALVES, W. J. B. Padrão de arquitetura mvc: Model-view-controller. **EPEQ Fafibe**, v. 1, n. 3a, p. 102–107, 2017.
- MICROSOFT. What's new in visual studio 2019. 2019. Disponível em: <<https://docs.microsoft.com/en-us/visualstudio/ide/whats-new-visual-studio-2019>>. Acesso em: 27 de Novembro de 2021.
- MOREIRA, J. P. de S.; COLNAGO, G. R. Otimização da grade de horários dos professores e disciplinas dos cursos do ifsp-câmpus cubatão. **REVISTA ACADÊMICA - ENSINO DE CIÊNCIAS E TECNOLOGIAS**, 2018.
- SCHAERF, A. A survey of automated timetabling. **Artificial Intelligence Review**, v. 13, n. 2, p. 87–127, Apr 1999. ISSN 1573-7462. Disponível em: <<https://doi.org/10.1023/A:1006576209967>>.
- SILVA, A. B. d. Alocação de recursos na resolução do problema da grade horária. Universidade Federal do Pampa, 2014.
- SOUZA, A. S. de; JR, J. E. P.; LOCH, G. V.; FRESSATO, A. A. Performance de dois solvers na resolução da metaheurística fix and optimize aplicado ao problema de high school timetabling. In: **II Simpósio de Métodos Numéricos em Engenharia**. [S.I.]: s.n., 2017.

WEARE, R.; BURKE, E.; ELLIMAN, D. A hybrid genetic algorithm for highly constrained timetabling problems. **Department of Computer Science**, 1995.

YANG, S.; JIANG, Y.; NGUYEN, T. T. Metaheuristics for dynamic combinatorial optimization problems. **IMA Journal of Management Mathematics**, Oxford University Press, v. 24, n. 4, p. 451–480, 2013.

GLOSSÁRIO

algoritmo: Sequência de passos lógicos voltados a um objetivo.

looping: Representar um erro na execução de determinado algoritmo, quando este passa a seguir repetidamente a mesma sequência de instruções, entrando em "repetição infinita".

software: Coleção de dados ou instruções que definem a forma que o mecanismo deve trabalhar.

solvers: Software matemático que tem como objetivo solucionar um problema matemático. Um solucionador pega as descrições do problema em algum tipo de forma genérica e calcula sua solução.

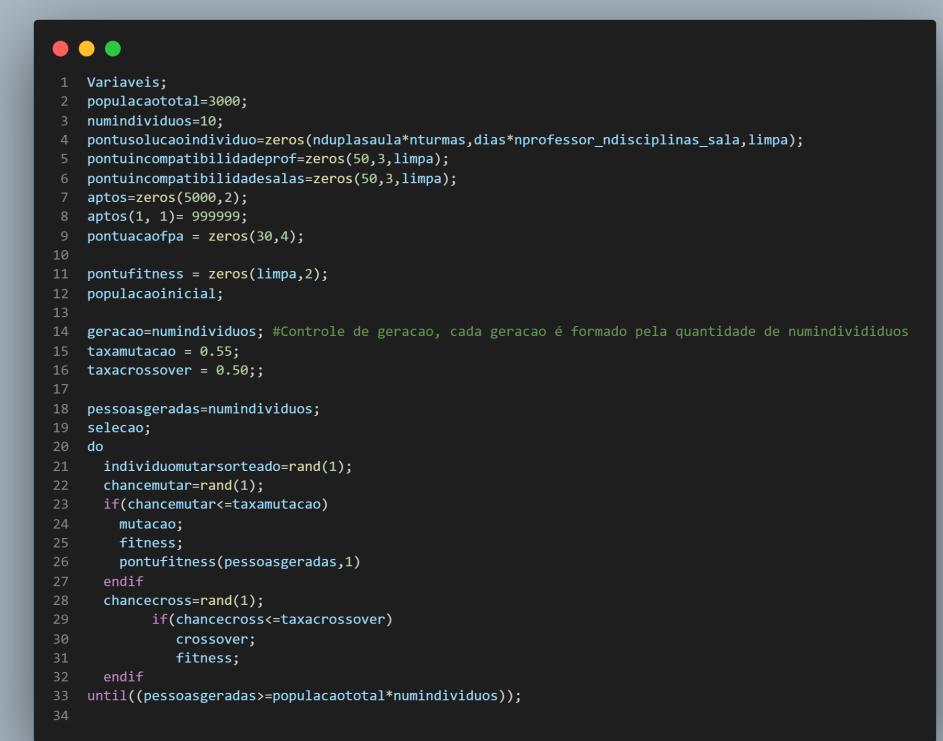
APÊNDICE A – Outras Técnicas de Busca

Existem diversas técnicas atualmente que também são utilizadas para resolver problemas de busca e otimização. Os conceitos abaixo visam esclarecer o funcionamento destas técnicas e dar um aprofundamento maior, visto que o uso de cada técnica foi mostrado na seção dos trabalhos relacionados ??

APÊNDICE B – Códigos Octave

Os códigos descritos neste apêndice exibem o desenvolvimento real do algoritmo, as figuras abaixo exibem o desenvolvimento real utilizado para a execução da solução do problema.

Figura 35 – Script de Execução Principal do Algoritmo Genético



```

1 Variaveis;
2 populacaototal=3000;
3 numindividuos=10;
4 pontosolucaoindividuo=zeros(nduplasaula*nTurmas,dias*nProfessor_ndisciplinas_sala,limpa);
5 pontuicompatibilidadeprof=zeros(50,3,limpa);
6 pontuicompatibilidadesalas=zeros(50,3,limpa);
7 aptos=zeros(500,2);
8 aptos(1, 1)= 999999;
9 pontuacaoofpa = zeros(30,4);
10
11 pontufitness = zeros(limpa,2);
12 populacaoinicial;
13
14 geracao=numindividuos; #Controle de geracao, cada geracao é formado pela quantidade de numindividuos
15 taxamutacao = 0.55;
16 taxacrossover = 0.50;;
17
18 pessoasgeradas=numindividuos;
19 selecao;
20 do
21     individuomutarsorteado=rand(1);
22     chancemutar=rand(1);
23     if(chancemutar<=taxamutacao)
24         mutacao;
25         fitness;
26         pontufitness(pessoasgeradas,1)
27     endif
28     chancecross=rand(1);
29     if(chancecross<=taxacrossover)
30         crossover;
31         fitness;
32     endif
33 until((pessoasgeradas>=populacaototal*numindividuos));
34

```

Fonte: Elaborado pelo autor.

Figura 36 – Script de Codificação das Disciplinas

```
1
2 aulascargas = [1    2    1
3 2    2    1
4 3    2    1
5 4    2    1
6 5    2    1
7 6    2    1
8 7    4    2
9 8    4    2
10 9   2    2
11 10  2    2
12 11  2    3
13 12  2    3
14 13  2    3
15 14  2    3
16 15  2    3
17 16  2    3
18 17  2    3
19 18  2    4
20 19  2    4
21 20  2    4
22 21  2    4
23 22  2    4
24 23  2    4
25 24  2    4
26 25  2    4
27 26  2    5
28 27  4    5
29 28  2    5
30 29  2    5
31 30  2    6
32 31  2    6
33 32  4    6
34 33  2    6
35 34  4    6
36 35  2    6
37 36  2    6
38 37  2    6
39 38  4    7
40 39  4    7
41 40  4    7
42 41  4    7
43 42  4    7
44 43  4    7
45 44  4    2
46 45  2    3
47 46  2    4
48 47  2    5
49 48  4    5
50 49  2    5
51 101 4   1
52 103 4   2
53 105 4   1
54 107 4   3
55 109 4   5
56 111 4   6
57 ];
58
59
60
61 b = size(aulascargas); #tamanho 55x3
62 c = b(1,1); # tamanho 55
63 d = aulascargas(c,1); # tamanho = 111
64
65 disciplinas= zeros(1,d);
66     for i=1:c;
67         disciplinas(1,aulascargas(i,1))= aulascargas(i,2);
68     endfor;
69
70
```

Fonte: Elaborado pelo autor.

Figura 37 – Script CrossOver

```

1 do
2 individuoscrossorteados=rand(1,2);
3 for cont = 1:umindividuos
4 if((individuoscrossorteados(1,1))>=ponturunqueado(cont,5)&& individuoscrossorteados(1,1)<=ponturunqueado(cont,6))
5 individuocross1=ponturunqueado(cont,2);
6 endif
7 if((individuoscrossorteados(1,2))>=ponturunqueado(cont,5)&& individuoscrossorteados(1,2)<=ponturunqueado(cont,6))
8 individuocross2=ponturunqueado(cont,2);
9 endif
10 endfor
11 until(individuocross1==individuocross2);
12
13 indicecross=randi(2);
14
15 [row1, column1] = find(pontufitness(:,2) == individuocross1);
16 [row2, column2] = find(pontufitness(:,2) == individuocross2);
17
18 dnaprincipal=randi(2);
19 individuoatual=individuoatual+1;
20
21 turmaspossiveis=[1,3,5,7,9,11,13];
22 turmasorteadas = turmaspossiveis(randperm(numel(turmaspossiveis)));
23
24 if(dnaprincipal==1)
25 pontusolucaoindividual(:, :, individuoatual)=pontusolucaoindividual(:, :, row1);
26 else
27 pontusolucaoindividual(:, :, individuoatual)=pontusolucaoindividual(:, :, row2);
28 endif
29
30 for i = 1:indicecross
31 sorteio=randi(2);
32 if(mod(i,2)==1)
33 if(sorteio==1)
34 pontusolucaoindividual(turmasorteadas(i):turmasorteadas(i)+1, :, individuoatual)=pontusolucaoindividual(turmasorteadas(i):turmasorteadas(i)+1, :, row1);
35 else
36 pontusolucaoindividual(turmasorteadas(i):turmasorteadas(i)+1, :, individuoatual)=pontusolucaoindividual(turmasorteadas(i):turmasorteadas(i)+1, :, row2);
37 endif
38 endif
39 endfor

```

Fonte: Elaborado pelo autor.**Figura 38 – Script Fitness**

```

1 Variaveis;
2
3 solucao inicial= pontusolucaoindividual(:, :, individuoatual);
4
5 pontuacaosalas;
6 pontuacao professores;
7 somapontuacaosalas = 0;
8 somapontuacao professores = 0;
9 pontuacao pafinal = 0;
10 linhapontuacaofpa=0;
11
12
13 pontuacao pfa_2;
14
15 if(!isempty(linhapontuacaofpa))
16 pontuacao pafinal = linhapontuacaofpa*500;
17 endif
18
19
20 if(!isempty(finalpontuacaosalas))
21 somapontuacaosalas = sum(finalpontuacaosalas(:, 3));
22 endif
23 if(!isempty(finalpontuacao professores))
24 somapontuacao professores = sum(finalpontuacao professores(:, 3));
25 endif
26 pontufitness(individuoatual,1)=somapontuacaosalas+somapontuacao professores+pontuacao pafinal; ##Somar as incompatibilidades de professor e de sala
27
28 pontufitness(individuoatual,2)=individuoatual; ## Armazenar o individuo;
29
30 if(!isempty(finalpontuacao professores))
31 pontuacompatibilidadeprof(1:rows(finalpontuacao professores),1:3,individuoatual)=finalpontuacao professores; ## Armazena a incompatibilidade do individuo
32 endif
33 if(!isempty(finalpontuacaosalas))
34 pontuacompatibilidadesalas(1:rows(finalpontuacaosalas),1:3,individuoatual)=finalpontuacaosalas; ## Armazeno a incompatibilidade;
35 endif
36
37 ultpt = somapontuacaosalas+somapontuacao professores+pontuacao pafinal;
38
39 if(ultpt-pontuacao pafinal==0 && pontuacao pafinal<aptos(comparaapto, 1))
40 printf("Individuo apto encontrado!:\n");
41
42 aptos(contaptos, 1)= linhapontuacaofpa;
43 aptos(contaptos, 2)= individuoatual;
44 comparaapto=contaptos;
45 contaptos=contaptos+1;
46 endif
47
48

```

Fonte: Elaborado pelo autor.

Figura 39 – Script Geração Inicial Aleatória 1

```
● ● ●

1 Variaveis;
2 if sabaut==1
3   for i=1:nTurmasAut*nduplasaula
4     for t=(dias-1)*nProfessor_ndisciplinas_sala+1:dias*nProfessor_ndisciplinas_sala
5       solucaoInicial(i,t)=-1;
6     endfor
7   endfor
8 endif
9 if sabmec==1
10   for i=nTurmasAut*nduplasaula+1:nduplasaula*nTurmas;
11     for t=(dias-1)*nProfessor_ndisciplinas_sala+1:dias*nProfessor_ndisciplinas_sala
12       solucaoInicial(i,t)=-1;
13     endfor
14   endfor
15 endif
16
17 auxaulascargas=aulascargas;
18 auxdisciplinas=zeros(nTurmas,nMaxDisciplinasPorTurma); #tam= 7x10
19 contador=1;
20 flagatrib=1;
21 segundavez=0;
22
23
24 while flagatrib~=0
25   flagatrib=0;
26   for i=1:nTurmas # para 1-7
27     for t=1:c; #para 1-c=55
28       #se as aulascargas(1-55,semestre) for=i
29       if aulascargas(t,3) == i && auxaulascargas(t,2)~=0
30         while (segundavez ==1) && (auxdisciplinas(i,contador) ~= 0)
31           contador = contador + 1;
32         endwhile
33         if(segundavez==1 && aulascargas(t,1)>100)
34           break;
35         endif
36         auxdisciplinas(i,contador)=aulascargas(t,1); #matriz 7x10
37         contador = contador + 1;
38         auxaulascargas(t,2) = auxaulascargas(t,2)-2;
39         flagatrib=1;
40       endif;
41     endfor;
42     contador = 1;
43   endfor;
44   segundavez=1;
45 endwhile
```

Fonte: Elaborado pelo autor.

Figura 40 – Script Geração Inicial Aleatória 2

```

1 k=1;
2
3 for i=1:nTurmas
4   vetorsorteio=0;
5   p=0;
6   for t=1:nMaxDisciplinasPorTurma
7     if auxdisciplinas(i,t)>0
8       p=p+1;
9
10    endif;
11  endfor;
12  vetorsorteio=randperm(p,p);
13  z=1;
14  for y=1:nduplasaula
15    for m=2:nProfessor_ndisciplinas_sala:nProfessor_ndisciplinas_sala*(dias) %1 ate 3*6
16      if z<=p
17        if solucaoInicial(k,m)==-1
18          else
19            solucaoInicial(k,m)=auxdisciplinas(i,vetorsorteio(z));
20            z=z+1;
21          endif
22        endif
23      endfor
24      k=k+1;
25    endfor
26  endfor;
27
28 flag=0;
29
30 for i=1:nTurmas*nduplasaula
31   for k=2:nProfessor_ndisciplinas_sala:nProfessor_ndisciplinas_sala*(dias)
32     if solucaoInicial(i,k)>=100 ### ESTUDAR GENERALIZAÇÃO
33       if mod(i,2)==0 %checagem de par
34         aux1=solucaoInicial(i-1,k);
35         if solucaoInicial(i-1,k)~=solucaoInicial(i,k)
36           solucaoInicial(i-1,k)=solucaoInicial(i,k);
37         if aux1~=0
38           for w=nProfessor_ndisciplinas_sala*dias-1:-nProfessor_ndisciplinas_sala:2
39             if solucaoInicial(i,w)==0 && flag==0
40               solucaoInicial(i,w)=aux1;
41               flag=1;
42             endif
43           endfor
44         endif
45       endif
46     else %impar
47       aux2=solucaoInicial(i+1,k);
48       if solucaoInicial(i+1,k)~=solucaoInicial(i,k)
49         solucaoInicial(i+1,k)=solucaoInicial(i,k);
50       if aux2~=0
51         for w=nProfessor_ndisciplinas_sala*dias-1:-nProfessor_ndisciplinas_sala:2
52           if solucaoInicial(i+1,w)==0 && flag==0
53             solucaoInicial(i+1,w)=aux2;
54             flag=1;
55           endif
56         endfor
57       endif
58     endif
59   endif
60   flag=0;
61 endfor
62 endfor
63 endfor

```

Fonte: Elaborado pelo autor.

Figura 41 – Script Geração Inicial Aleatória 3

```

● ● ●
1  for i=1:nTurmas*nDuplasAula
2    for k=2:nProfessor_nDisciplinas_Sala:nProfessor_nDisciplinas_Sala*(dias)
3      if(mod(i,2)==0 && solucaoInicial(i,k)>100)
4        solucaoInicial(i,k)=solucaoInicial(i,k)-1;
5      endif
6    endfor
7  endfor
8
9
10 for i=1:nTurmas*nDuplasAula
11   for j=2:nProfessor_nDisciplinas_Sala:nProfessor_nDisciplinas_Sala*(dias)
12     for k=1:21
13       for y=2:2:16
14
15         if(dadosDeEntrada(k,y)==1 && dadosDeEntrada(k,y)<100 && solucaoInicial(i,j)~-1)
16           if (dadosDeEntrada(k,y-1) == solucaoInicial(i,j))
17             solucaoInicial(i,j-1) = k;
18             solucaoInicial(i,j+1) = dadosDeEntrada(k,y);
19
20           endif
21         else
22           if(dadosDeEntrada(k,y)>100 && solucaoInicial(i,j)~-1 && dadosDeEntrada(k,y-1) == solucaoInicial(i,j))
23             solucaoInicial(i,j-1) = k;
24             solucaoInicial(i,j+1) = dadosDeEntrada(k,y);
25
26           endif
27         endif
28       endfor
29       flag=0;
30     endfor
31   endfor
32 endfor
33

```

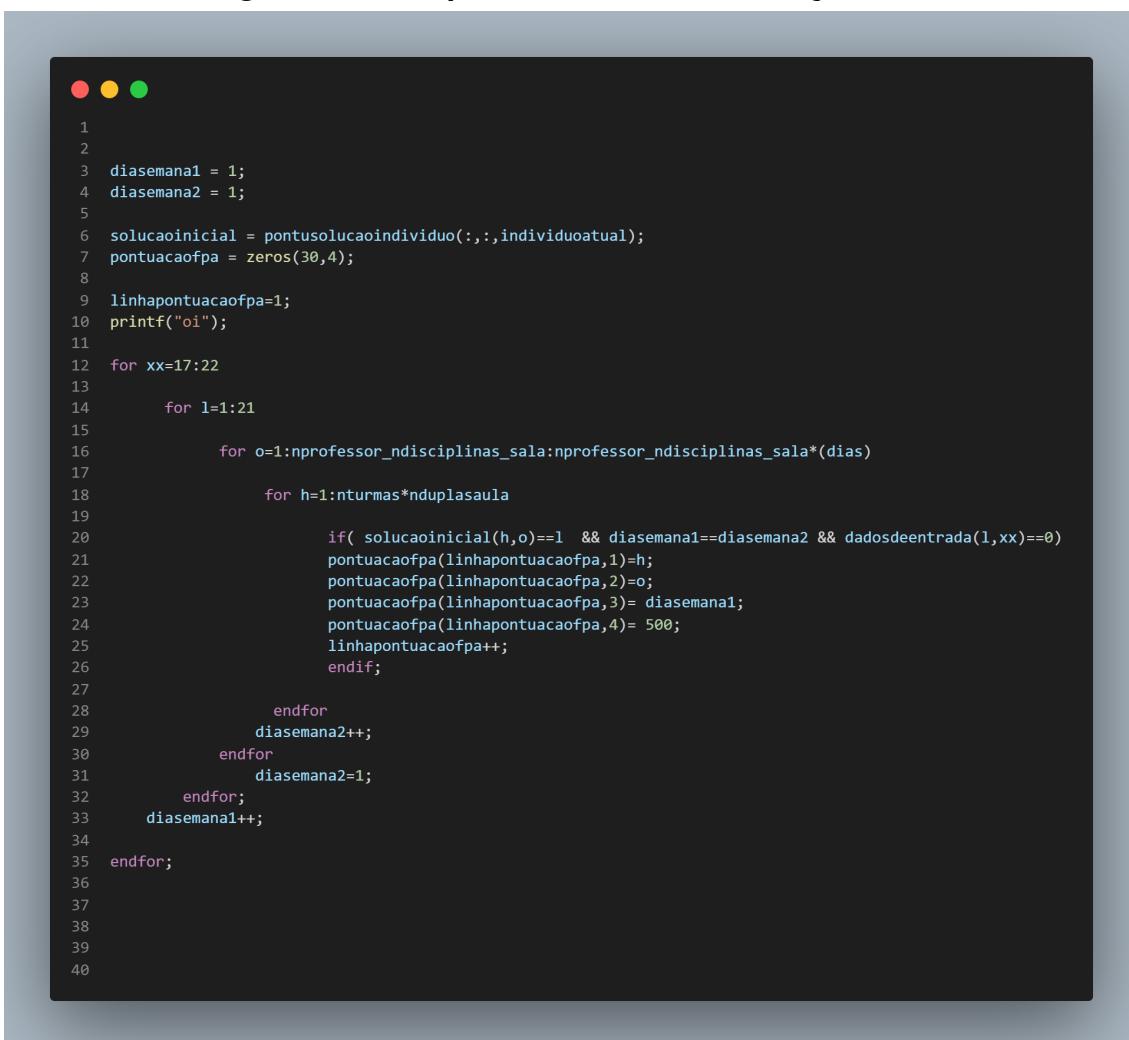
Fonte: Elaborado pelo autor.**Figura 42 – Script de Mutação**

```

● ● ●
1  for cont = 1:numIndividuos
2    if((individuoSorteados)->pontuacaoTotal && individuoMutar->pontuacaoTotal<-pontuacaoTotal)
3      individuoMutar = individuoSorteados;
4    endif
5  endfor
6  indicemutacao = 1;
7  individuoAtual=individuoMutar;
8  pontusolucionaIndividuo(:,individuoAtual)=pontusolucionaIndividuo(:,individuoMutar);
9  colunasDasSemana=[3,6,9,12,15,18];
10 auxsolo=[1,3,5,7,9,11,13];
11 auxxpliZeroes(2,3);
12 auxsoloZeroes(1,3);
13 entrou=0;
14 tipomutacao = rand(2);
15 do
16   diaSorteados = colunasDasSemana(rand(1,numel(colunasDasSemana)));
17   novodiasSorteados = turmas(rand(1,numel(colunasDasSemana)));
18   turmasSorteadas = turmas(rand(1,numel(turmas)));
19   until ((diaSorteados->novodiasSorteados) && (pontusolucionaIndividuo(turmasSorteadas,diaSorteados,individuoAtual))~-1) && (pontusolucionaIndividuo(turmasSorteadas,novodiasSorteados,individuoAtual))~-1)
20   if((pontusolucionaIndividuo(turmasSorteadas,diaSorteados-1,individuoAtual))>100) || (pontusolucionaIndividuo(turmasSorteadas,novodiasSorteados-1,individuoAtual))>100)
21     tipomutacao=1;
22   else
23     tipomutacao=rand(2);
24   endif
25   if(tipomutacao==1)
26     if((pontusolucionaIndividuo(turmasSorteadas,diaSorteados-1,individuoAtual))>100) || (pontusolucionaIndividuo(turmasSorteadas,novodiasSorteados-1,individuoAtual))>100)
27       auxup1(1,:)=pontusolucionaIndividuo(turmasSorteadas,diaSorteados-1,individuoAtual);
28       auxup2(1,:)=pontusolucionaIndividuo(turmasSorteadas,diaSorteados-2,individuoAtual);
29       pontusolucionaIndividuo(turmasSorteadas,diaSorteados-1,individuoAtual)=auxup1(1,:);
30       pontusolucionaIndividuo(turmasSorteadas,diaSorteados-2,individuoAtual)=auxup2(1,:);
31       pontusolucionaIndividuo(turmasSorteadas,diaSorteados-1,individuoAtual)=auxup1(1,:);
32       pontusolucionaIndividuo(turmasSorteadas,diaSorteados-2,individuoAtual)=auxup2(1,:);
33     else
34       coefficientetroca1 = round(rand);
35       coefficientetroca2 = round(rand);
36       auxsolo=pontusolucionaIndividuo(turmasSorteadas,coefficientetroca1,diaSorteados-2,diaSorteados,individuoAtual);
37       pontusolucionaIndividuo(turmasSorteadas,coefficientetroca1,diaSorteados-2,diaSorteados,individuoAtual)=pontusolucionaIndividuo(turmasSorteadas,coefficientetroca2,novodiasSorteados-2,individuoAtual);
38       pontusolucionaIndividuo(turmasSorteadas,coefficientetroca2,novodiasSorteados-2,individuoAtual)=auxsolo;
39     endif
40   else
41     if(tipomutacao==2)
42       auxinverter = pontusolucionaIndividuo(turmasSorteadas+1,diaSorteados-2,diaSorteados,individuoAtual);
43       pontusolucionaIndividuo(turmasSorteadas+1,diaSorteados-2,diaSorteados,individuoAtual)=pontusolucionaIndividuo(turmasSorteadas,diaSorteados-2,diaSorteados,individuoAtual);
44       pontusolucionaIndividuo(turmasSorteadas,diaSorteados-2,diaSorteados,individuoAtual)=auxinverter;
45     endif
46   endif
47

```

Fonte: Elaborado pelo autor.

Figura 43 – Script de Cálculo de Pontuação FPA

The screenshot shows a terminal window with a dark background and three colored window control buttons (red, yellow, green) at the top. The window contains a MATLAB script with line numbers from 1 to 40. The script initializes variables, sets up matrices for initial solutions and scores, and uses nested loops to calculate scores based on attendance and other criteria. It includes an 'oi' printf statement and various assignment and conditional statements.

```
1
2
3 diasemana1 = 1;
4 diasemana2 = 1;
5
6 solucao inicial = pontusolucaoindividuo(:,:,individuoatual);
7 pontuacaoofpa = zeros(30,4);
8
9 linhapontuacaoofpa=1;
10 printf("oi");
11
12 for xx=17:22
13
14     for l=1:21
15
16         for o=1:nprofessor_ndisciplinas_sala:nprofessor_ndisciplinas_sala*(dias)
17
18             for h=1:nturmas*nduplasaula
19
20                 if( solucao inicial(h,o)==1 && diasemana1==diasemana2 && dadosdeentrada(l,xx)==0)
21                     pontuacaoofpa(linhapontuacaoofpa,1)=h;
22                     pontuacaoofpa(linhapontuacaoofpa,2)=o;
23                     pontuacaoofpa(linhapontuacaoofpa,3)= diasemana1;
24                     pontuacaoofpa(linhapontuacaoofpa,4)= 500;
25                     linhapontuacaoofpa++;
26                 endif;
27
28             endfor;
29             diasemana2++;
30         endfor;
31         diasemana2=1;
32     endfor;
33     diasemana1++;
34
35 endfor;
36
37
38
39
40
```

Fonte: Elaborado pelo autor.

Figura 44 – Script de Cálculo de Pontuação Incompatibilidade de Professores

```

● ● ●

1 pontuacaoprofessores = zeros(50,3);
2 linhapontuacaoprofessores=1;
3
4 for j=1:nprofessor_ndisciplinas_sala:nprofessor_ndisciplinas_sala*(dias)
5   for i=1:nturmas*nduplasaula #i de 1-14
6     primeiravez=1;
7     if ((solucao inicial(i,j+1) > 100) &&(solucao inicial(i,j+1) != -1)) #####Realizo a contagem de penalidade das disciplinas > 100
8       for(p=i+1:nturmas*nduplasaula)
9         if(p+1)>15 break;
10        endif
11        if(solucao inicial(i,j)==solucao inicial(p,j))
12          if(primeiravez==1)
13            pontuacaoprofessores(linhapontuacaoprofessores,1)=1; ##Armazeno a linha da incompatibilidade
14            pontuacaoprofessores(linhapontuacaoprofessores,2)=j; ##Armazeno a coluna da incompatibilidade
15            pontuacaoprofessores(linhapontuacaoprofessores,3)= 10000; ##Armazeno a pontuacao da incompatibilidade
16            linhapontuacaoprofessores++;
17            primeiravez=0;
18        endif
19        pontuacaoprofessores(linhapontuacaoprofessores,1)=p; ##Armazeno a linha da incompatibilidade
20        pontuacaoprofessores(linhapontuacaoprofessores,2)=j; ##Armazeno a coluna da incompatibilidade
21        pontuacaoprofessores(linhapontuacaoprofessores,3)= 10000; ##Armazeno a pontuacao da incompatibilidade
22        linhapontuacaoprofessores++;
23      endif
24    endiff
25    else
26      if((solucao inicial(i,j+1)<100) &&(solucao inicial(i,j+1) != -1))
27        for(p=i+1:nturmas*nduplasaula)
28          if(p+1)>15 break;
29          endif
30          if(solucao inicial(i,j)==solucao inicial(p,j))
31            if((solucao inicial(p,+i) > 100) || (mod(i,2)==mod(p,2)))
32              if(primeiravez==1)
33                pontuacaoprofessores(linhapontuacaoprofessores,1)=i; ##Armazeno a linha da incompatibilidade
34                pontuacaoprofessores(linhapontuacaoprofessores,2)=j; ##Armazeno a coluna da incompatibilidade
35                pontuacaoprofessores(linhapontuacaoprofessores,3)=10000; ##Armazeno a pontuacao da incompatibilidade
36                linhapontuacaoprofessores++;
37                primeiravez=0;
38              endif
39              pontuacaoprofessores(linhapontuacaoprofessores,1)=p; ##Armazeno a linha da incompatibilidade
40              pontuacaoprofessores(linhapontuacaoprofessores,2)=j; ##Armazeno a coluna da incompatibilidade
41              pontuacaoprofessores(linhapontuacaoprofessores,3)= 10000; ##Armazeno a pontuacao da incompatibilidade
42              linhapontuacaoprofessores++;
43            endif
44            pontuacaoprofessores(linhapontuacaoprofessores,1)=p; ##Armazeno a linha da incompatibilidade
45            pontuacaoprofessores(linhapontuacaoprofessores,2)=j; ##Armazeno a coluna da incompatibilidade
46            pontuacaoprofessores(linhapontuacaoprofessores,3)= 10000; ##Armazeno a pontuacao da incompatibilidade
47            linhapontuacaoprofessores++;
48          endif
49        endif
50        #primeiravez=0;
51      endfor
52    endif
53  endfor
54 endfor
55
56 for(p=10:2:14)
57   for(j=10:2:14)
58     if(solucao inicial(p,13) == solucao inicial(j-1,16))
59       pontuacaoprofessores(linhapontuacaoprofessores,1)=j-1;
60       pontuacaoprofessores(linhapontuacaoprofessores,2)=16;
61       pontuacaoprofessores(linhapontuacaoprofessores,3)=10002;
62     endif
63   endfor
64 endfor
65
66 pontuacaoprofessores;
67 pontpfs= unique(pontuacaoprofessores,'rows');
68 idx2keep_columns = sum(abs(pontpfs),1)>0 ;
69 idx2keep_rows    = sum(abs(pontpfs),2)>0 ;
70
71 finalpontuacaoprofessores = pontpfs(idx2keep_rows,idx2keep_columns);
72 if(finalpontuacaoprofessores)
73 finalpontuacaoprofessores = sortrows(finalpontuacaoprofessores,2);
74 endif
75
76
77

```

Fonte: Elaborado pelo autor.

Figura 45 – Script de Cálculo de Pontuação Incompatibilidade de Salas

```
1 pontuacaosalas = zeros(50,3);
2 linhapontuacaosalas=1;
3 primeiravez = 1;
4
5 for j=3:nprofessor_ndisciplinas_sala:nprofessor_ndisciplinas_sala*(dias)
6     for i=1:nturmas*nduplasaula #i de 1-14
7         primeiravez=1;
8             if ((solucaoInicial(i,j-1) > 100) && (solucaoInicial(i,j)>100))
9
10                 for(p=i+1:nturmas*nduplasaula)
11                     if(p+1)>15 break;
12                 endif
13                     if(solucaoInicial(i,j)==solucaoInicial(p,j))
14                         if(primeiravez==1)
15                             pontuacaosalas(linhapontuacaosalas,1)=i; ##Armazeno a linha da incompatibilidade
16                             pontuacaosalas(linhapontuacaosalas,2)=j; ##Armazeno a coluna da incompatibilidade
17                             pontuacaosalas(linhapontuacaosalas,3)= 10000; ##Armazeno a pontuacao da incompatibilidade
18                             linhapontuacaosalas++;
19                             primeiravez=0;
20                         endif
21                         pontuacaosalas(linhapontuacaosalas,1)=p; ##Armazeno a linha da incompatibilidade
22                         pontuacaosalas(linhapontuacaosalas,2)=j; ##Armazeno a coluna da incompatibilidade
23                         pontuacaosalas(linhapontuacaosalas,3)= 10000; ##Armazeno a pontuacao da incompatibilidade
24                         linhapontuacaosalas++;
25                     endif
26                 endif
27             endif
28         endfor
29
30         elseif ((solucaoInicial(i,j-1)<100) && (solucaoInicial(i,j)>100))
31             for(p=i+1:nturmas*nduplasaula)
32                 if(p+1)>15 break;
33                 endif
34                 if(solucaoInicial(i,j)==solucaoInicial(p,j))
35                     if((solucaoInicial(p,j-1) > 100) || (mod(i,2)==mod(p,2)))
36                         if(primeiravez==1)
37                             pontuacaosalas(linhapontuacaosalas,1)=i; ##Armazeno a linha da incompatibilidade
38                             pontuacaosalas(linhapontuacaosalas,2)=j; ##Armazeno a coluna da incompatibilidade
39                             pontuacaosalas(linhapontuacaosalas,3)=10000; ##Armazeno a pontuacao da incompatibilidade
40                             linhapontuacaosalas++;
41                             primeiravez=0;
42                         endif
43                         pontuacaosalas(linhapontuacaosalas,1)=p; ##Armazeno a linha da incompatibilidade
44                         pontuacaosalas(linhapontuacaosalas,2)=j; ##Armazeno a coluna da incompatibilidade
45                         pontuacaosalas(linhapontuacaosalas,3)=10000; ##Armazeno a pontuacao da incompatibilidade
46                         linhapontuacaosalas++;
47                     endif
48                 endif
49                 #primeiravez=0;
50             endfor
51         endif
52     endfor
53 endfor
54
55 pontsls= unique(pontuacaosalas,'rows');
56 ###Remover os zeros
57 idx2keep_columns = sum(abs(pontsls),1)>0 ;
58 idx2keep_rows     = sum(abs(pontsls),2)>0 ;
59
60
61 finalpontuacaosalas = pontsls(idx2keep_rows,idx2keep_columns);
62 if(finalpontuacaosalas)
63     finalpontuacaosalas = sortrows(finalpontuacaosalas,2);
64 endif
65
```

Fonte: Elaborado pelo autor.

ANEXO A – TÍTULO

Nos anexos são colocados textos ou documentos não desenvolvidos pelo autor, mas que podem auxiliar na fundamentação e/ou comprovação do que foi argumentado no trabalho. Os anexos são identificados por letras maiúsculas consecutivas, travessão e seus respectivos títulos. Podem ser utilizadas letras maiúsculas dobradas quando o número de anexos esgotar as 26 letras do alfabeto.

Exemplos:

ANEXO A – Representação gráfica da contagem de células inflamatórias – Grupo II

ANEXO B – Representação molecular

....

ANEXO AA – Representação gráfica da contagem de células inflamatórias – Grupo XXVII

ÍNDICE

Consiste em uma lista de palavras ou frases ordenadas segundo determinado critério, que localiza e remete às informações contidas no texto. Deverá ser elaborado conforme a ABNT NBR 6034 (ASSOCIAÇÃO..., 2004)

Exemplos:

Academia, 51, 71, 88

ação, 35-7, 61,83

acaso, 22, 43, 54