
UNIVERSIDADE FEDERAL RURAL DE PERNAMBUCO


Matheus Paulo dos Santos Demiro

# Alocação otimizada de horários acadêmicos com disponibilidade restrita de professores usando algoritmos genéticos

**Recife**

Junho de 2022

Matheus Paulo dos Santos Demiro

# Alocação otimizada de horários acadêmicos com disponibilidade restrita de professores usando algoritmos genéticos

Artigo apresentado ao Curso de Bacharelado em Sistemas de Informação da Universidade Federal Rural de Pernambuco, como requisito parcial para obtenção do título de Bacharel em Sistemas de Informação.

Universidade Federal Rural de Pernambuco – UFRPE
Departamento de Estatística e Informática
Curso de Bacharelado em Sistemas de Informação

**Orientador: Cícero Garrozi**

Recife
Junho de 2022

Dados Internacionais de Catalogação na Publicação
Universidade Federal Rural de Pernambuco
Sistema Integrado de Bibliotecas
Gerada automaticamente, mediante os dados fornecidos pelo(a) autor(a)

D381a Demiro, Matheus Paulo dos Santos
Alocação otimizada de horários acadêmicos com disponibilidade restrita de professores usando algoritmos genéticos / Matheus Paulo dos Santos Demiro. - 2022.
48 f. : il.

Orientador: Cicero Garrozi.
Inclui referências e apêndice(s).

Trabalho de Conclusão de Curso (Graduação) - Universidade Federal Rural de Pernambuco, , Recife, 2022.

1. Escalonamento de horários acadêmicos. 2. Otimização combinatória. 3. Timetabling. 4. Algoritmos genéticos. I. Garrozi, Cicero, orient. II. Título

CDD

Matheus Paulo dos Santo Demiro

# Alocação otimizada de horários acadêmicos com disponibilidade restrita de professores usando algoritmos genéticos

Artigo apresentado ao Curso de Bacharelado em Sistemas de Informação da Universidade Federal Rural de Pernambuco, como requisito parcial para obtenção do título de Bacharel em Sistemas de Informação.

Aprovado em: 01 de Junho de 2022.

BANCA EXAMINADORA

Cícero Garrozi
Departamento de Estatística e Informática
Universidade Federal Rural de Pernambuco

Silvana Bocanegra
Departamento de Estatística e Informática
Universidade Federal Rural de Pernambuco

# Alocação otimizada de horários acadêmicos com disponibilidade restrita de professores usando algoritmos genéticos

**[Matheus Paulo dos Santos Demiro]** [1], **[Cícero Garrozi]** [2]

[1]Departamento de Estatística e Informática – Universidade Federal Rural de Pernambuco
Rua Dom Manuel de Medeiros, s/n, - CEP: 52171-900 – Recife – PE – Brasil

matheus.demiro@ufrpe.br, cicero.garrozi@ufrpe.br

***Resumo.*** *A geração de horários acadêmicos é uma das atividades mais complexas e árduas enfrentadas pelas instituições de ensino no começo de cada período letivo. Na maioria dos casos, a solução encontrada para esse problema, comumente chamado na literatura de "timetabling", é realizada de forma manual, o que torna o processo muito cansativo e moroso para as instituições. Este problema é considerado um grande desafio da otimização combinatória, devido ao amplo conjunto de variáveis e restrições envolvidas, sendo considerado um problema NP-Completo, onde não existe a possibilidade de encontrar uma solução por meio de métodos convencionais de programação. Este artigo trata do uso de técnicas de algoritmos genéticos para encontrar uma solução ótima ao problema do escalonamento de horários acadêmicos que leve em consideração as restrições do corpo discente e corpo docente, a fim de favorecer o rendimento acadêmico dos alunos e se adequar a disponibilidade dos professores. Para este trabalho espera-se desenvolver um algoritmo genético que seja capaz de obter resultados válidos que atendam as restrições do problema em um espaço de tempo razoavelmente considerável. Tecnicamente falando, é esperado que o algoritmo a partir de um conjunto de dados de entrada, processe e retorne uma solução que possua o maior valor de aptidão - menor número de infrações cometidas - dentre às gerações de indivíduos (soluções). Este artigo usa como base de dados a grade do curso de Sistemas de Informação da Universidade Federal de Rio Grande Norte. Após modificações na base e realizados os experimentos o algoritmo genético mostrou-se eficiente e conseguiu cumprir com os objetivos, gerando horários acadêmicos adequados e compatíveis com as restrições estabelecidas.*
***Palavras-chave****: Escalonamento de horários acadêmicos. Otimização combinatória. Timetabling. Algoritmos genéticos.*

***Abstract.*** *The generation of academic timetables is one of the most complex and arduous activities faced by educational institutions at the beginning of each academic period. In most cases, the solution found for this problem, commonly called "timetabling" in the literature, is performed manually, which makes the process very tiring and time-consuming for institutions. This problem is considered a great challenge in combinatorial optimization, due to the wide set of variables and constraints involved, being considered an NP-Complete problem, where there is no possibility of solution through conventional programming methods. This article deals with the use of genetic algorithm techniques to*

*find an optimal solution to the problem of scheduling academic schedules that takes into account the restrictions of the student and the faculty, in order to favor the academic performance of students and adapt to availability. from the students . teachers. For this work, it is expected to develop a genetic algorithm that is able to obtain valid results that meet the constraints of the problem in a reasonably considerable time. Technically, it is expected that the algorithm, from a set of input data, processes and returns a solution that has the highest fitness value - the lowest number of infractions committed - between generations of individuals (solutions). This article uses data from the Information Systems course grid at the Federal University of Rio Grande Norte as a base. After modifications in the base and the experiments were carried out, the genetic algorithm proved to be efficient and managed to achieve the objectives, generating adequate academic schedules and compatible with the established restrictions.*
***Keywords**: Academic timetable problem. Combinatorial optimization. Timetabling. Genetic algorithms.*

## 1. Introdução

No início de cada período letivo, é comum as equipes pedagógicas das instituições de ensino se depararem com o problema de montar uma grade de horários que satisfaça um conjunto amplo de restrições estabelecidas pela própria instituição, normalmente esse conjunto varia de acordo com o tipo de instituição e o cenário a ser aplicado. Esse problema é conhecido na literatura como *timetabling* e consiste, basicamente, em arranjar encontros entre docentes e discentes em um determinado tempo (dias da semana) e espaço (salas de aula) de modo que satisfaça o conjunto de restrições predefinidas pela instituição acadêmica [Maydana 2011].

Na maioria dos casos, o problema de escalonamento de horários é solucionado de forma manual, tornando o processo bastante cansativo e moroso para os responsáveis, devido a grande quantidade de variáveis e restrições. Normalmente, a grade horária resultante dessa tarefa manual não atende plenamente as restrições e nem as expectativas dos envolvidos (instituição, docentes e discentes). Importante salientar que a confecção de uma boa estrutura de horários pode melhorar a satisfação do corpo docente e permitir que a instituição de ensino seja mais eficiente na gestão de seus recursos, além de favorecer um melhor desempenho do corpo discente [Santos and Souza 2007]. Portanto, é essencial durante a montagem dos horários tentar atender as restrições e expectativas tanto dos docentes quanto dos discentes, a fim de favorecer o rendimento acadêmico dos alunos e as preferências dos professores [Almeida 2015].

Ao longo dos anos, diversos pesquisadores buscaram alternativas para automatizar e otimizar o processo de planejamento de uma grade de horários, a fim de minimizar a complexidade e o esforço humano. Tal complexidade é afetada de forma exponencial pela quantidade de variáveis e restrições estabelecidas no escalonamento, tornando inviável a busca por uma solução ótima em um espaço de tempo aceitável. Isso ocorre pelo fato do escalonamento de horário ser considerado um problema NP-Completo, onde não é possível encontrar um algoritmo determinístico que seja capaz de encontrar uma solução ótima em tempo polinomial [Cruz et al. 2019].

Para esse tipo de problema é inviável a construção de soluções genéricas de

automatização, pois as variáveis e restrições mudam de acordo com a instituição de ensino [Cruz et al. 2019]. Basicamente, cada instituição possuirá um conjunto de restrições específicas, podendo diminuir ou aumentar a complexidade do problema.

A complexidade na obtenção de uma solução ótima motivou pesquisadores e profissionais da área em utilizar uma metodologia baseada em heurísticas. De acordo com [Almeida 2015], uma heurística consiste em uma estratégia de busca por uma solução para o problema em tempo hábil, a fim de obter uma solução aceitável para o problema de forma mais rápida e eficaz, em comparação com a busca realizada de forma manual, na força bruta, buscando a melhor solução dentre as possíveis soluções [Almeida 2015]. Os métodos heurísticos não possuem uma garantia que a solução encontrada seja sempre ótima, entretanto, garante que tal solução esteja próxima ao ótimo global. Dentre os diversos métodos heurísticos utilizados na literatura para solucionar o problema do escalonamento de horários, é possível citar os algoritmos genéticos.

Com o avanço e abrangência da computação evolutiva, trabalhos manuais que seriam concluídos em um considerável espaço de tempo, acabam sendo realizados por sistemas inteligentes que são capazes de reduzir o consumo de tempo, além do trabalho e recursos anteriormente requisitados. Conforme citado anteriormente, na busca por otimização dos trabalhos manuais, principalmente da montagem de horários, surgem os algoritmos genéticos (AG), como uma ferramenta de busca inteligente que utilizam os fenômenos da evolução biológica, como cruzamento, seleção e mutação, para solucionar problemas reais que dispõem de um alto grau de complexidade. Normalmente, os AGs são utilizados para resolver problemas em que há um amplo conjunto de possíveis soluções e que a busca exaustiva, força bruta, seria impraticável ou até mesmo impossível de obter um resultado satisfatório e adequado às restrições do problema [Oliveira and Manzan 2022].

Este trabalho visa aplicar a técnica de algoritmo genético ao problema da montagem de horários acadêmicos, e utiliza a base de dados fornecida pela autora do trabalho [Almeida 2015], que são dados provenientes do curso de Sistemas de Informação da Universidade Federal do Rio Grande do Norte (UFRN), localizada na cidade de Caicó, como conjunto de entrada. A base de dados pode ser encontrada no trabalho [Almeida 2015]. O conjunto de entrada do algoritmo foi formado a partir de análises, reagrupamento e seleção dos dados disponibilizados (esse processamento dos dados será descrito detalhadamente na Seção 4).

Segundo [Almeida 2015], a ocorrência de aulas vagas ou intervalos entre aulas na grade de horário pode influenciar no desempenho acadêmico do corpo discente, podendo causar desmotivação nos alunos devido a ociosidade durante um longo período de tempo (uma hora-aula ou mais). Logo, é importante que a instituição de ensino leve em consideração o corpo discente e desta forma priorize a blocagem de horários durante a montagem dos horários, para evitar a ocorrência de fatores que venham a prejudicar o rendimento acadêmico dos alunos.

Mediante isso, o objetivo deste trabalho é utilizar a técnica de algoritmo genético para encontrar uma solução aceitável para o problema de escalonamento de horários acadêmicos, seguindo as restrições e regras associadas ao problema, além de garantir a blocagem de horário e, consequentemente, o favorecimento do rendimento acadêmico

do corpo discente. Também é objetivo deste trabalho assegurar que as restrições de disponibilidade do corpo docente sejam atendidas. Portanto, o intuito é manter o equilíbrio entre as restrições de alunos, quanto a ocorrência de aulas vagas, e professores, quanto a preferência de horários, a fim de encontrar uma solução que favoreça ambas as partes, pois é essencial para o processo de ensino-aprendizagem que as partes envolvidas estejam satisfeitas.

## 2. Referencial teórico

Esta seção descreve mais detalhadamente a respeito do problema de escalonamento de horários (*timetabling*) e aborda os principais conceitos de algoritmos genéticos. Além disso, explica a importância na definição de uma boa representação cromossômica para o problema.

### 2.1. Escalonamento de horários

O problema da geração de quadros de horários ou escalonamento de horários, também conhecido, em inglês, como "timetabling", é um problema de otimização que possui aplicações práticas e que pode ser encontrado facilmente no dia a dia, como por exemplo, planejamento de transporte público, escalonamento de médicos e horários de aula em instituições de ensino. Encontrar respostas para essas problemáticas podem solucionar diversos problemas enfrentados pela sociedade [Freitas et al. 2014].

Este problema é objeto de estudo das pesquisas de otimização devido a sua complexidade e o amplo conjunto de variáveis, dado o enorme espaço de possibilidades que devem ser avaliadas para aferir qual a solução ótima, ou seja, a melhor solução para o problema [Almeida 2015]. A maioria dos problemas de otimização devem seguir um conjunto de restrições, e isso não seria diferente para o problema de *timetabling*. Esse conjunto, que varia de acordo com o contexto do problema, influencia diretamente na complexidade do problema, pois quanto maior for o conjunto de restrições mais complexo será o problema e, consequentemente, levará mais tempo para se achar uma solução ótima. Portanto, a complexidade do problema é afetada exponencialmente pela quantidade de restrições e variáveis envolvidas, inviabilizando a busca por uma solução em um espaço tempo aceitável. Problemas desse tipo são categorizados como NP-Completo, onde não existe um algoritmo capaz de encontrar uma solução ótima em tempo polinomial [Cruz et al. 2019].

No caso do problema de escalonamento de horários acadêmicos, para realizar a montagem das grades de horário são necessários "os horários, a disponibilidade dos professores, as disciplinas a serem oferecidas, a quantidade de aulas de cada disciplina, as salas de aula disponíveis e os horários dos alunos." [Almeida 2015]. Essas são variáveis utilizadas pela maioria dos problemas de timetabling, mas não basta apenas isso, também é necessário levar em consideração as restrições associadas a tais variáveis, como por exemplo, nenhum professor pode ministrar duas aulas no mesmo horário, nenhuma sala pode receber duas aulas de disciplinas distintas no mesmo horário, entre outras restrições.

Na busca em solucionar tal problema, as instituições de ensino realizam a montagem de horários de forma manual e que atendam totalmente ou parcialmente as variáveis do problema e suas respectivas restrições, a fim de garantir a satisfação das partes envolvidas no processo, geralmente discentes e docentes. Como esse processo comumente é

realizado de forma manual, as instituições levam um tempo consideravelmente alto para elaboração dos horários [Freitas et al. 2014].

## 2.2. Algoritmos genéticos (AGs)

Ao longo dos anos técnicas computacionais foram criadas com o intuito de reproduzir o comportamento da natureza, um comportamento inteligente. Dentre essas técnicas estão os algoritmos genéticos, que são algoritmos inspirados nos princípios Darwinianos da evolução das espécies e da seleção natural, combinados com os conceitos da genética (cruzamento e mutação) [Almeida 2015]. De acordo com os princípios de Charles Darwin, a seleção é um processo em que os indivíduos mais adaptados ao ambiente são selecionados naturalmente e sobrevivem, conseguindo reproduzir e gerar novos descendentes (nova população). Dito isso, os AGs nada mais são que uma tentativa computacional de simulação desses princípios.

Os AGs são algoritmos meta-heurísticos que fazem uso de heurísticas de busca genérica para solucionar problemas de otimização. Esses algoritmos utilizam uma estratégia de busca que tenta encontrar o ótimo global utilizando mecanismos que evitem o confinamento em mínimos ou máximos locais [Becceneri et al. 2008]. Logo, os algoritmos genéticos por meio da junção dos princípios evolutivos das espécies, definidos por Darwin, e técnicas computacionais, aplicam técnicas heurísticas de busca que tentam imitar tais princípios para encontrar a solução ótima para determinado problema de otimização.

### 2.2.1. Conceitos

A área de inteligência computacional trabalha com o desenvolvimento de sistemas que simulam certos aspectos do comportamento inteligente, o comportamento dos seres humanos e outros animais. Como parte dessa área temos a computação natural, que permite a criação de um ambiente computacional que simula ecossistemas naturais criando populações com comportamentos semelhantes aos reinos animal e vegetal. Com isso, é possível a criação de soluções computacionais para problemas rotineiros da sociedade que demandam grande esforço [Almeida 2015].

Os algoritmos genéticos fazem parte da classe de algoritmos evolutivos (AEs), pois são baseados na teoria da evolução das espécies de Darwin e no conceito de genética. São algoritmos que fazem uso de heurísticas e estratégias de buscas eficientes baseadas nos princípios da seleção natural e da reprodução sexuada, e são capazes de encontrar soluções próxima da solução ótima para problemas complexos, com o mínimo de interferência humana, usando técnicas computacionais que simulam os processos naturais da evolução [Almeida 2015].

A teoria da evolução das espécies, constituída por Charles Darwin, diz que existe uma competição entre os indivíduos de uma população por recursos que são limitados. A luta pela sobrevivência faz com que os indivíduos mais aptos ao meio sobrevivam e consigam gerar seus descendentes, enquanto que os menos aptos tendem a diminuir sua descendência. Esse processo é chamado de seleção natural, conforme descrito na Seção 2.2 deste trabalho. Logo, à medida que os seres mais aptos repassam seus genes para seus descendentes (combinam os genes), surgem indivíduos com modificações que

podem afetar positivamente ou negativamente no seu processo de adaptabilidade ao meio ambiente, o que pode garantir ou não maior longevidade de sua espécie. Esse processo trata-se da evolução natural [Almeida 2015].

A computação evolutiva simula computacionalmente tais processos por meio de algoritmos evolutivos capazes de encontrar a melhor solução para determinado problema (seja complexo ou não), através de técnicas heurísticas e probabilísticas que imitam os princípios da evolução natural, onde uma população de soluções sofrem modificações (combinações) até que uma resposta seja encontrada [Almeida 2015].

Alguns termos e definições utilizados na biologia foram herdados pela computação evolutiva e são bastante utilizados na literatura. Pode-se citar os cromossomos, que na biologia são responsáveis por carregar informações sobre os seres vivos, enquanto que na computação os cromossomos artificiais são uma estrutura de dados que representa as características de um indivíduo [Almeida 2015]. Na Tabela 1 pode-se observar alguns termos importantes da computação evolutiva que possuem analogia com a natureza.

| **Natureza** | **Algoritmos genéticos** |
|---|---|
| Cromossomo | Palavra binária, vetor, etc |
| Gene | Característica do problema |
| Alelo | Valor da característica |
| Loco | Posição na palavra, vetor |
| Genótipo | Estrutura |
| Fenótipo | Estrutura submetida ao problema |
| Indivíduo | Solução |
| Geração | Ciclo |

**Tabela 1. Analogia de termos (Fonte: adaptada de [Almeida 2015])**

Durante o processo de evolução, os indivíduos da população passam por dois processos evolucionários: o cruzamento (crossover) e a mutação. O crossover nada mais é que o cruzamento entre dois cromossomos, onde ocorre uma troca de informações e a geração de novos cromossomos. Na computação essa troca ocorre da combinação de dois vetores (cromossomos) que geram novos cromossomos artificiais. Já o processo de mutação, na biologia, são modificações que ocorrem ao acaso no código genético dos indivíduos, seja motivada por um fator externo ou por erros durante a replicação do DNA. Nos algoritmos genéticos a mutação é utilizada para evitar que o algoritmo convirja rapidamente para um mínimo local e que novas características sejam exploradas.

### 2.2.2. Funcionamento dos AGs

Conforme citado anteriormente, a base dos algoritmos genéticos são os princípios da seleção natural, genética e teoria evolutiva de Darwin. Estes conceitos são simulados computacionalmente pelos AGs, que utilizam uma estratégia de busca probabilística inteligente, que se baseia no processo de evolução biológica, onde um conjunto de possíveis soluções é analisado e geradas novas soluções a partir de combinações do conjunto inicial [de Barros Montin 2022].

Os algoritmos genéticos geralmente são aplicados a problemas de otimização combinatória de grande escala, o seu funcionamento é simples, porém dependendo do contexto do problema, podem levar horas para encontrarem uma solução ótima.

O primeiro passo de execução de um AG é a geração da população inicial, que normalmente é gerada de forma aleatória e, as vezes, seguem algumas regras do problema [Almeida 2015]. A população é formada por indivíduos, que como descrito anteriormente são possíveis soluções para o problema, e para cada indivíduo (solução) é atribuído um valor numérico que define o seu nível de adaptabilidade, o fitness [Freitas et al. 2014]. Em outras palavras, o fitness determina a qualidade da possível solução.

Analogamente a regra da seleção natural, os AGs tendem a selecionar as soluções que possuem um maior valor de fitness (mais aptos) no decorrer do processo evolutivo do algoritmo. Após isso, a execução do AG é prosseguida pela aplicação dos operadores genéticos de crossover e mutação, visando gerar novas soluções a partir das soluções que já existem. Com isso, os operadores de seleção, crossover e mutação, nessa ordem, são executados até que uma condição de parada seja satisfeita, podendo tal condição ser a quantidade de vezes que o algoritmo é executado ou a quantidade de tempo decorrido da execução do algoritmo. Caso tal condição não seja alcançada, o algoritmo recomeça a execução a partir do operador de seleção [Almeida 2015].

### 2.2.3. Representação cromossômica

A representação cromossômica é uma etapa fundamental para os algoritmos genéticos, pois ela é uma maneira de representar os dados do problema em uma forma que consigam ser interpretados pelos computadores [Almeida 2015]. A sua importância se dá ao fato de que está diretamente relacionada com a qualidade dos resultados obtidos pelo AG, bem como o tempo computacional gasto para encontrar tais resultados [Lima et al. 2015].

Existem diversas formas de representação, a mais simples e mais usada pela literatura, a depender do contexto do problema, é a forma binária. Esse tipo de representação consiste em um vetor (cromossomo) com uma sequência de bits (cada bit é um gene), onde os valores dos bits são representados por 1 ou 0 [Almeida 2015]. Essa representação binária, embora tenha se mostrado eficaz em diversos problemas, não consegue codificar dados com precisão em aplicações mais complexas.

Como resultado, surgiram diversas formas de representação para problemas de otimização combinatória, como é caso da codificação por meio de vetores (listas) com números inteiros, a representação numérica (usada quando os cromossomos são representados por números reais), entre outras formas [Almeida 2015].

Apesar da existência de variadas formas de representação, não significa dizer que todas serão úteis para o problema alvo, pois como citado anteriormente, o tempo computacional para encontrar uma solução pode ser afetada drasticamente a depender do tipo de representação escolhida. Ou seja, a depender da representação ela pode tornar o problema ainda mais complexo. Logo, é essencial a escolha da melhor representação para o problema alvo. Isso pode ser feito por meio de testes manuais ou utilizando métodos que foram bem sucedidos na literatura, para que os resultados obtidos pelo AG sejam de qualidade e que o tempo computacional gasto não seja algo impraticável.

## 3. Trabalhos Relacionados

Existem diversos trabalhos na literatura que utilizam algoritmos genéticos para solucionar o problema do escalonamento de horários. Conforme citado anteriormente, este problema é um dos mais complexos em solucionar da otimização combinatória, sendo considerado um problema da classe NP-Completo. Por ser um problema enfrentado recorrentemente no cotidiano, principalmente pelas instituições de ensino, surgem diversos trabalhos e pesquisas que tentam encontrar modelos que otimizem a resolução de tal problemática.

Em [Oliveira and Manzan 2022] é apresentada uma solução para otimizar o processo de preparação de grade de horários das aulas dos cursos do Instituto Federal do Triângulo Mineiro (IFTM) - Campus UPT, localizado em Uberaba, Minas Gerais. Nesse campus a montagem de horários era realizada de forma manual, e por estar relacionada a diversas restrições como restrições de horários, aulas em diferentes unidades e agrupamento de aulas, era necessário investir em tempo e trabalho árduo para encontrar uma solução que atendesse as restrições estabelecidas. Dito isso, [Oliveira and Manzan 2022] propôs a modelagem de um sistema baseado em algoritmos genéticos para montagem dos horários de aulas atendendo as especificidades dos cursos do IFTM.

Para o desenvolvimento do AG foram consideradas restrições relacionadas ao corpo docente e as disciplinas lecionadas, tais como: um professor não pode ministrar duas disciplinas ao mesmo tempo, uma turma não pode ter aulas diferentes no mesmo horário, blocagem de horários de acordo com os cursos, prioridade para aulas duplas ou triplas, evitar janelas de horários, um professor deve ter um intervalo de no mínimo 11 horas entre sua última aula do dia e a primeira do próximo dia e restrições de horários dos professores. Também foram considerados os seguintes operadores genéticos: método da seleção por torneio, como operador de seleção dos pais; operador de cruzamento baseado em ponto de corte, onde um ponto de cruzamento é escolhido aleatoriamente e a partir deste ponto as informações dos pais são trocadas; operador de mutação tradicional modificado, onde eram escolhidos aleatoriamente dois dias e os mesmos eram trocados de posição em todas as turmas de um cromossomo.

Outro ponto importante do trabalho de [Oliveira and Manzan 2022], é a proposta de heurística para solucionar com maior rapidez a questão dos conflitos de horários dos professores. Observando que os operadores de cruzamento e mutação não resolviam a questão dos conflitos de horários, foi necessário empregar uma técnica mais robusta e dedicada a resolver os conflitos. A heurística foi aplicada a cada dez gerações do algoritmo e consistia em selecionar para cada turma de cada indivíduo o dia com mais choques de horários e o dia com menos choques de horários, e inverter os dias.

Segundo [Oliveira and Manzan 2022], os resultados obtidos mostraram que o AG conseguiu gerar grades de horários que atendessem as restrições do problema em um tempo menor comparado ao método manual. Além disso, a heurística implementada para solucionar os casos de choques de horários mostrou-se eficiente e conseguiu zerar a quantidade de choques de horários durante os testes, e também foi fundamental para obter resultados satisfatórios e com um menor tempo.

Em [Almeida 2015] é proposta uma solução algorítmica para resolver o problema de escalonamento de horários enfrentado pelo curso de Sistemas de Informação da Universidade Federal do Rio Grande do Norte (UFRN), localizada na cidade de Caicó (RN).

Segundo Almeida (2015), os intervalos vagos entre aulas, a ociosidade, pode acarretar na desmotivação dos discentes e, consequentemente, prejudicar no desempenho acadêmico deles. Com isso, devido a complexidade na montagem e no consequente problema em ter espaços vagos na grade de horário, foi proposto um modelo algorítmico baseado em algoritmo genético com o foco em impossibilitar a ocorrência de conflitos de horários e garantir a blocagem de aulas, a fim de favorecer o rendimento acadêmico do corpo discente. Para isso, o AG implementado desenvolvido utilizou os seguintes operadores genéticos: operador de seleção baseado no método da roleta, que consiste em escolher os pais mais aptos ou não; operador de cruzamento baseado em ordem, onde são selecionadas informações aleatórias do pai1 e herdadas para o filho1 e as informações que não foram selecionadas do pai1 são colocadas na ordem em que aparecem no pai2 e posteriormente inseridas no filho1; operador de mutação baseado em ordem por meio da permutação de genes, onde é realizada uma permutação em vários genes ao mesmo tempo. Porém, o operador de mutação sofreu adaptações ao problema devido às restrições definidas e ficou responsável em verificar se o indivíduo gerado não infringe nenhuma dessas restrições.

Segundo [Almeida 2015], o algoritmo mostrou ser capaz de encontrar horários que respeitem as restrições associadas ao problema e que podem ser utilizados como solução para o problema do escalonamento de horários tendo como foco principal a blocagem de horários. Além disso, o algoritmo se destaca em relação às demais soluções existentes na área, devido ao fato do algoritmo adaptar o funcionamento dos operadores genéticos de cruzamento e mutação dando ênfase ao objetivo do trabalho.

A plataforma [UniTime 2015] disponibiliza diversas bases de dados voltadas para o problema do escalonamento de horários, mais especificamente a problemas de atribuição de aulas a determinadas salas. O UniTime é um sistema distribuído que permite a construção de horários escolares que atenda às diversas necessidades organizacionais, permitindo a minimização de conflitos de horários. O sistema foi originalmente desenvolvido de modo colaborativo entre docentes, discentes e funcionários de universidades na América do Norte e na Europa. Atualmente a UniTime disponibiliza um software gratuito que pode ser utilizado por coordenadores de curso para a montagem de grades de horários.

No ano de 2019, a equipe do UniTime co-organizou a 4ª Competição Internacional de Escalonamento de Horários (em inglês, The Fourth International Timetabling Competition) [ITC 2019]. Competições desse tipo ocorrem desde o ano de 2003 e são fomentadas por pesquisadores e interessados. Nesse tipo de competição, as equipes de pesquisa competem pela construção de um solucionador de cronogramas de cursos e na resolução de problemas do mundo real. O objetivo principal da competição é motivar mais pesquisas sobre problemas complexos de escalonamento de horários de cursos universitários provenientes da prática. Para a 4ª edição da competição, o [UniTime 2015] disponibilizou as bases de dados que foram coletadas de instituições reais que usam o seu sistema.

Apesar do [UniTime 2015] estruturar e disponibilizar gratuitamente as bases de dados, o presente trabalho não fez uso de nenhuma delas, pois elas não continham atributos e requisitos compatíveis com o objetivo deste trabalho. Todas as bases estão relacionadas a problemas que envolvem atribuição de aulas a determinadas salas e alunos, o que

foge totalmente do escopo deste trabalho.

O modelo apresentado em [Almeida 2015] serviu como motivação para a elaboração do presente trabalho, pois o fato em priorizar a blocagem de horários para favorecer o corpo discente durante a montagem de horários, é um diferencial em relação aos demais trabalhos da área. Neste trabalho, além de garantir que as restrições dos discentes sejam atendidas, também garante que as preferências dos professores sejam contempladas, diferentemente do modelo desenvolvido em [Almeida 2015], onde todos os professores possuíam dedicação exclusiva.

## 4. Metodologia

Na seção anterior foram apresentados alguns trabalhos que lidam com o problema do escalonamento de horário e que utilizam algoritmos genéticos como estratégia de solução. Vale ressaltar que além dessas propostas, existem diversas outras soluções disponibilizadas no mercado de software na forma de produtos comerciais, gratuitos ou não.

A maioria dos trabalhos da literatura e dos softwares disponíveis no mercado que propõem solução para a problemática da montagem de horários, costumam priorizar os aspectos administrativos durante a estruturação dos horários. Ou seja, levam em conta restrições de professores, estrutura institucional (salas disponíveis) e as disciplinas que serão ofertadas, negligenciando as prioridades do corpo docente e influenciando negativamente em seu desempenho acadêmico [Almeida 2015].

Dentre os trabalhos apresentados na Seção 3, o [Almeida 2015] é o único que prioriza o desempenho do corpo docente para a montagem das grades de horário, tornando o docente um fator fundamental para o processo de ensino-aprendizagem. Este trabalho serviu de base para o presente artigo, devido aos interesses em comum de ambas as propostas. Dito isso, as estratégias utilizadas no trabalho [Almeida 2015] também foram aplicadas neste trabalho, porém sofreram algumas modificações e incrementos.

A estratégia em considerar apenas os alunos na estruturação da grade de horário só será útil em um contexto onde os discentes possuem dedicação exclusiva, e isso não é um cenário muito comum em instituições de ensino privadas. Pensando nisso e objetivando cobrir a maior quantidade possível de cenários, este trabalho propõe uma solução capaz de atender as restrições tanto do corpo docente quanto do corpo discente, a fim de favorecer no processo de ensino-aprendizagem e satisfazer as necessidades de ambas as partes envolvidas.

De acordo com [Almeida 2015], o aluno, além do professor, é peça essencial no processo de ensino-aprendizagem. Logo, é primordial também levar em consideração o corpo discente, além dos docentes, durante a montagem da grade de horários. Assim, para garantir um bom desempenho do corpo discente, é necessário que a equipe pedagógica desenvolva a sequência didática iterativa, que consiste em criar formas que despertem o interesse dos docentes no processo e ensino-aprendizagem. Portanto, segundo [Almeida 2015], uma forma de despertar tal interesse é diminuir a ociosidade do discente, ou seja, minimizar a ocorrência de aulas vagas na grade de horário.

Mediante isso, este trabalho tem o intuito em propor uma solução ao problema do escalonamento de horários acadêmicos utilizando algoritmos genéticos, com foco na otimização dos horários dos discentes, no cumprimento das restrições impostas pelos do-

centes e na minimização de intervalos entre aulas, buscando atingir o equilíbrio entre ambas as partes e contribuir para o processo de ensino-aprendizagem. Além disso, também é objetivo deste trabalho auxiliar as instituições no processo de montagem das grades de horários.

### 4.1. Base de dados

A base de dados utilizada neste trabalho foi disponibilizada pela autora do trabalho [Almeida 2015]. Esta base passou por um processo de análise, reagrupamento e seleção, antes de servir como conjunto de entrada para o algoritmo desenvolvido neste trabalho. Esse processamento foi necessário para montar uma base mais consolidada e com horários mais completos.

Os dados utilizados em [Almeida 2015] são referentes aos semestres letivos 2012.1, 2012.2, 2013.1, 2013.2, 2014.1, 2014.2, 2015.1 e 2015.2 do curso de Bacharelado em Sistemas de Informação (BSI) da UFRN. Este curso oferece disciplinas nos turnos matutinos e vespertinos, e é composto por oito períodos por semestre, onde cada período tem disciplinas específicas pré-definidas. Às vezes, esses cursos precisam ser oferecidos aos alunos que atrasaram seu progresso no curso. No turno matutino são oferecidas as disciplinas para alunos repetentes, enquanto que no contraturno (vespertino) são ofertadas as disciplinas para alunos regulares. Além disso, quando o semestre corrente é ímpar, as disciplinas regulares do turno vespertino são ofertadas nos períodos ímpares (1,3,5,7) e as disciplinas do contraturno (matutino) são ofertadas nos períodos pares (2,4,6,8). Quando o semestre corrente é par, ocorre o inverso [Almeida 2015].

Para este trabalho só foram consideradas as disciplinas ofertadas no turno vespertino e desconsideradas as disciplinas complementares. Além disso, inicialmente, foi considerado que os professores possuíam dedicação exclusiva, como em [Almeida 2015].

Durante o processo de análise da base, os dados foram submetidos a testes no algoritmo e percebeu-se que a solução ótima era encontrada rapidamente. Isso ocorreu pois os dados e restrições utilizadas não apresentavam muita complexidade e o problema ficou "fácil" de ser resolvido. Para evitar esse cenário, a base passou por um processo de seleção e reagrupamento, onde foram selecionados os períodos que apresentaram mais disciplinas regulares oferecidas e reagrupados em um único semestre letivo. Esse processo consistia em analisar cada período um a um e comparar com os períodos de mesmo nível nos demais semestres letivos, o período que apresentasse uma maior quantidade de disciplina e que fizesse sentido em ser realocado, era selecionado para fazer parte do novo semestre. O conjunto de dados das disciplinas e professores resultante desse processo pode ser visualizado no Apêndice D.

Apesar da mudança, o algoritmo ainda continuava encontrando soluções de forma muita rápida, ou seja, estava convergindo rapidamente. Logo, visando adicionar mais complexidade ao problema, foi retirado a dedicação exclusiva dos professores e eles passaram a ter uma lista de disponibilidade semanal, onde cada um possuía dias da semana em que estariam disponíveis para ministrar aula. A adição dessa variável aumentou consideravelmente a complexidade do problema e o algoritmo parou de convergir rapidamente, levando mais tempo para encontrar as possíveis soluções.

A base de dados deste trabalho é composta por 19 professores e 38 disciplinas, além de contar com dados gerados aleatoriamente de disponibilidade. Para auxiliar du-

rante os testes do algoritmo, foi utilizado um banco de dados SQL para armazenar o conjunto de dados de entrada. Na Tabela 2 é possível observar todos os professores e suas respectivas disciplinas que fizeram parte do conjunto de dados de entrada deste trabalho.

| **PROFESSORES** | **DISCIPLINAS** |
|---|---|
| GILSON | TGA,MTC,TGS,FSI,EMP |
| FABRICIO | POOI,POO2,ES1,PWEB |
| FLAVIUS | ALG,PROG,SAD |
| JOÃO PAULO | LOG,ED,AEX |
| LUIZ PAULO | II,ARQ |
| RICARDO | CEC |
| TACIANO | BD,ES2,PABD |
| LUCIANO | FMAT,PE,MATF |
| KARLIANE | OSM,GPS |
| JOÃO BORGES | SO,RED,SEG |
| JOSÉ ENÉAS | TSI |
| DÉSIO | CAL |
| DELSON | AL |
| CÉLIA | LPT |
| CARLOS | ETIC |
| TIAGO | PV |
| TALITHA | FIL |
| LEOMARQUES | ING |
| VYRNA | DIR |

**Tabela 2. Relação professores-disciplinas. (Fonte: o autor)**

Na Figura 1 é apresentado o diagrama do modelo do banco de dados utilizado para armazenar os dados do conjunto de entrada. O modelo foi projetado para ser algo simples e de pouca robustez, pois sua modelagem serviu apenas para auxiliar nos experimentos e resultados. As tabelas "Lesson" (em português, "turma") e "Avaliability" (em português, "disponibilidade") são responsáveis por armazenar os dados das turmas e a lista de disponibilidade dos professores, respectivamente. Já as tabelas "Teacher" (em português, "professor") e "Subject" (em português, "disciplina") servem para armazenar os dados dos professores e das disciplinas, respectivamente.

Em [Almeida 2015] não é informado nenhum dado em relação a disponibilidade dos professores, visto que todos possuíam dedicação exclusiva. Logo, para este trabalho foi necessário a criação do conjunto de dados de disponibilidade. Para isso foram definidas 10 amostras de dados de disponibilidade que foram criadas manualmente e com o máximo de disparidade possível, a fim de cobrir a maior quantidade de cenários possíveis.

Durante a análise da base de dados de disciplinas e seus respectivos professores, foi estipulado que um professor deveria ter no mínimo 3 dias de disponibilidade e no máximo 5 dias de disponibilidade, considerando que são até 3 aulas por dia. Esse mínimo mínimo de dias foi definido após a execução de alguns testes com diversas bases de disponibilidade que continham uma quantidade de dias inferior a 3. Os testes não obtiveram o sucesso que foi obtido com a quantidade mínima de 3 dias, além disso o algoritmo não

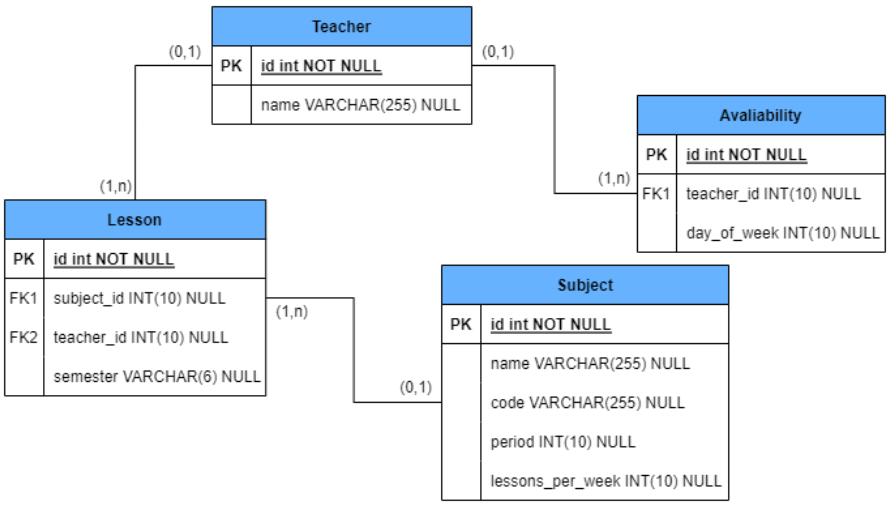



**Figura 1. Diagrama do modelo do banco de dados. (Fonte: o autor)**

conseguiu gerar bons resultados para algumas bases de disponibilidade, ou seja, o AG não conseguia encontrar uma solução que cumpria com todas as restrições. Com isso, foram definidas algumas regras para a criação das amostras, sendo elas:

1. Todo professor deve ter no mínimo 3 dias de disponibilidade;
2. Se um professor lecionar duas disciplinas em um mesmo período, então ele deve ter no mínimo 4 dias de disponibilidade;
3. Se um professor leciona 8 aulas na semana, então ele deve ter no mínimo 4 dias de disponibilidade;
4. Se um professor leciona mais do que 8 aulas na semana, então ele deve ter 5 dias de disponibilidade;
5. Se um professor leciona 8 aulas e possui duas disciplinas com 3 aulas na semana cada uma, então ele deve ter 5 dias de disponibilidade.

Com as regras estabelecidas, as 10 amostras foram criadas de forma manual e realizadas comparações para minimizar a ocorrência de dados repetidos entre as amostras. Essa estratégia para montagem das amostras que consiste em criar dados de disponibilidade dos professores com o máximo de disparidade possível, é para provar que o algoritmo é capaz de encontrar solução em diferentes cenários, aumentando a sua eficácia em diferentes contextos.

### 4.2. Algoritmo genético

O primeiro passo de um algoritmo genético é a definição da população inicial, onde o algoritmo cria uma população com uma quantidade n de indivíduos. Após isso, é iniciado o processo evolutivo da população, por meio dos operadores genéticos de seleção, mutação e cruzamento. Em seguida, a aptidão dos novos indivíduos gerados pelo processo de evolução é calculada. Caso a condição de parada seja satisfeita o algoritmo é interrompido, caso contrário aplica-se novamente os operadores genéticos. Isso se repete até que a condição de parada seja alcançada.

No Algoritmo 1, pode-se observar o pseudocódigo do algoritmo genético desenvolvido neste trabalho. Nas linhas 2 e 3, é possível notar os parâmetros genéticos de total

de gerações e tamanho da população que possuem os valores 2000 e 100, respectivamente. A primeira etapa do algoritmo é a criação da população inicial (representada pelo vetor "populacao") utilizando o método "gerarIndividuo()". Após essa etapa, o algoritmo inicia o processo evolutivo que se repete até que o critério de parada seja satisfeito. Este critério será satisfeito quando uma das seguintes situações ocorrerem: o algoritmo atinge o total de gerações (2000) ou quando ocorre a convergência do algoritmo (ou seja, a solução ótima é encontrada).

**Algorithm 1** : Pseudocódigo algoritmo genético

```
1:  totalGeracoes ← 2000;
2:  tamanhoPopulacao ← 100;
3:  populacao[tamanhoPopulacao];                    ▷ Array da população inicial
4:  for i = 1, ...,tamanhoPopulacao do
5:      populacao[i] ← gerarIndividuo();            ▷ Gerando indivíduo para a população
    inicial
6:  end for
7:  while criteorioDeParada == falso do
8:      novaPopulacao[tamanhoPopulacao];            ▷ Array da nova população
9:      for j = 1, ...,tamanhoPopulacao do
10:         pais ← selecionarPais(X);               ▷ Selecionando os pais
11:         filho_j ← cruzamento(pais);             ▷ Aplicando operador de cruzamento
12:         filho_j ← calcularAptidao(filho_j);     ▷ Calculando a aptidão do filho_j
13:         filho_j ← mutacao(filho_j);             ▷ Aplicando operador de mutação
14:         filho_j ← calcularAptidao(filho_j);     ▷ Recalculando a aptidão do filho_j
15:         novaPopulacao[j] ← filho_j;
16:     end for
17:     populacao ← selecaoSobreviventes(populacao, novaPopulacao);
18: end while
```

O processo evolutivo do algoritmo pode ser observado no Algoritmo 1 das linhas 10 até a 15, onde são chamados os operadores genéticos de seleção, cruzamento e mutação (nessa ordem), e o indivíduo resultante desse processo é armazenado no vetor "novaPopulacao". Este vetor armazena os novos indivíduos que são gerados durante o processo evolutivo. Também é possível observar que não existe taxa de cruzamento e nem de mutação, logo, o cruzamento e a mutação sempre ocorre na população e para todos os indivíduos.

Para a seleção de pais, representado na linha 10 como "selecionarPais", o algoritmo utilizou o método da roleta, que consiste em atribuir uma taxa de probabilidade de seleção baseado no nível de aptidão para os indivíduos da população e, aleatoriamente, selecionar dois indivíduos que serão os pais durante a etapa de cruzamento. Então, quanto maior a aptidão do indivíduo maior será a chance dele ser selecionado para a etapa de cruzamento. Os métodos de cruzamento e mutação podem ser visualizados com mais detalhes na Seção 4.4.

Após o processo evolutivo, o algoritmo na linha 17 executa o método "selecaoSobreviventes", que aplica uma seleção de sobreviventes baseado em fitness (aptidão) com estratégia elitista, onde é realizado uma junção dos vetores "populacao" e "novaPopula-

cao", e do vetor resultante dessa união são selecionados os melhores indivíduos que são repassados para a próxima geração. Dessa forma, não há perda dos bons indivíduos e suas características serão repassadas para as próximas gerações. Em seguida, caso a condição de parada não seja satisfeita o algoritmo retorna para a linha 8 e executa novamente o processo evolutivo (da linha 10 a 15).

### 4.3. Representação cromossômica

A representação cromossômica utilizada neste trabalho foi a mesma elaborada e apresentada no trabalho de [Almeida 2015], mas com algumas modificações.

No processo de definição da representação cromossômica, foi necessário compreender as regras básicas necessárias para a montagem de horários acadêmicos e as peculiaridades do cenário abordado neste trabalho. Basicamente, cada horário é formado por um conjunto de disciplinas, que são ministradas por professores, que possuem restrições de disponibilidade de horário, e frequentadas por alunos, isso remete ao conceito de turma (relação entre disciplina, discente e docente).

A base dados remete ao curso de BSI e esse curso apresenta algumas peculiaridades em sua estrutura de horário. O curso é diurno e composto por 8 períodos em cada semestre, apesar de que alguns semestres possam ter menos períodos caso não tenha novos alunos e alunos repetentes. Sabendo disso, cada cromossomo tem que descrever a estrutura de um conjunto de períodos e cada período deve ter informações sobre as turmas e seu cronograma. Cada turma tem informações sobre a disciplina, o semestre atual, o professor e quantidade de aulas da turma na semana.

A estrutura de dados utilizada para representar a turma foi um vetor, conforme apresentado na Figura 2, onde cada posição deste vetor possui as informações necessárias para representar uma turma.


**Figura 2. Representação de uma turma. (Fonte: adaptada de [Almeida 2015])**

Segundo [Almeida 2015], às aulas do curso de BSI da UFRN são oferecidas na maioria das vezes em blocos de duas-horas aula juntas, o que totaliza 6 horas aulas por dia, o equivalente a no máximo 3 aulas por dia. Dito isso, e que cada período é composto por um conjunto de turmas e horários, a estrutura de dados escolhida para representação dos períodos foi uma matriz (Figura 3), onde cada linha representa o dia da semana e cada coluna os blocos de aulas.

Na maioria dos cursos universitários, o conjunto de períodos ocorre no decorrer de um semestre letivo e dependendo da instituição pode ocorrer cerca de 2 ou mais semestres letivos no ano. Desta forma, para gerar uma solução para o problema do escalonamento de horário é necessário distribuir as turmas e seus horários em cada um dos períodos que estão dispostos no semestre letivo. Portanto, com base nas estruturas descritas anteriormente, a representação cromossômica utilizada neste trabalho é a junção das estruturas citadas alocadas em um vetor. Em outras palavras, a representação cromossômica do problema consiste em um vetor de matrizes (Figura 4), onde cada posição do vetor cor-

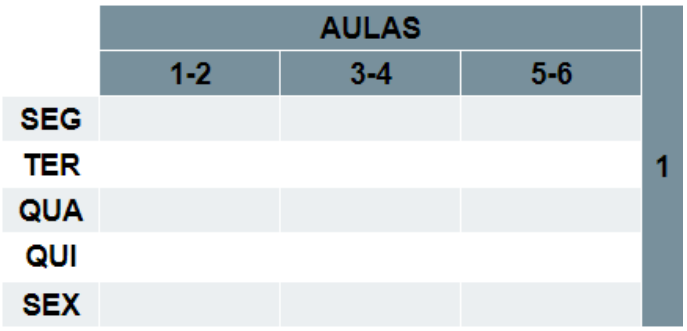



**Figura 3. Representação de um período. (Fonte: adaptada de [Almeida 2015])**

responde a uma matriz (período), e cada posição da matriz corresponde a um bloco de aula de uma turma (duas horas-aula).

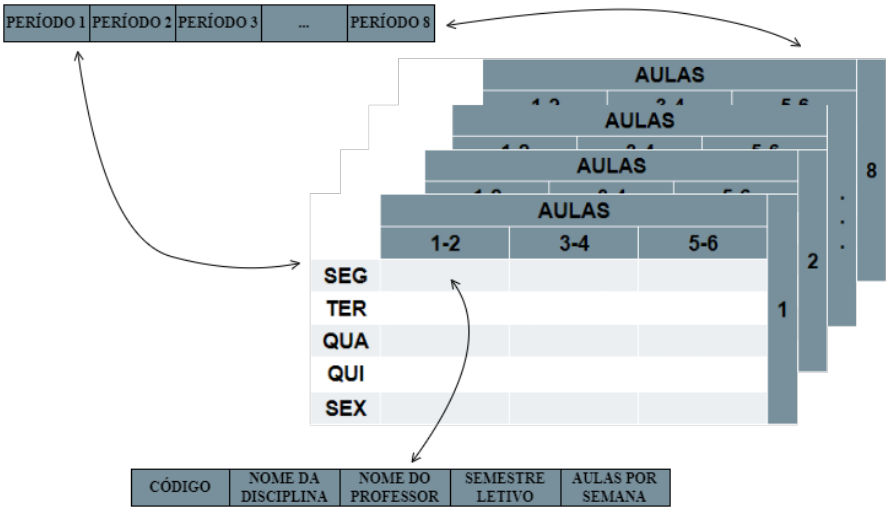



**Figura 4. Representação cromossômica. (Fonte: adaptada de [Almeida 2015])**

### 4.4. Operadores genéticos utilizados

Como estamos tratando da resolução de um problema utilizando técnicas computacionais por meio de algoritmos, é necessário identificar o tamanho do problema para conseguir enxergar sua complexidade e identificar os recursos a serem utilizados. O problema do escalonamento de horário é considerado NP-Completo, e como nenhum algoritmo específico é capaz encontrar a solução ótima em tempo polinomial, técnicas de busca heurística costumam ser utilizadas para encontrar uma solução viável [Almeida 2015].

O presente trabalho utiliza o mesmo estudo de caso realizado por [Almeida 2015], mas com algumas modificações quanto a estrutura e composição dos dados utilizados nos experimentos. Apesar disso, é possível por meio das variáveis envolvidas no problema de escalonamento de horário determinar o tamanho do espaço de busca através da Equação 1.

$$|S| = \sum_{p=1}^{n} \frac{e!}{(e - b_p!)} \tag{1}$$

onde $S$ é o espaço de busca; $p$ é o índice do período; $n$ é o número total de períodos (entre 4 e 8, em nosso estudo de caso); $b_p$ é o número de aulas que terá em cada semana de um período $p$ (entre 1 e 15, no nosso caso); e $e$ é a dimensão (tamanho) da estrutura de dados de cada período $p$ (no nosso caso 15 blocos) [Almeida 2015].

Conforme falado na Seção 2, que conceitua o problema da montagem de horários e descreve o funcionamento dos AGs, tem-se que após os passos de definição da representação cromossômica (descrita na Seção 4.3) e geração da população inicial, é necessário definir os operadores genéticos a serem utilizados pelo algoritmo.

Neste trabalho, assim como no de [Almeida 2015], foi utilizado a implementação de um AG simples com algumas modificações, principalmente nos operadores genéticos. Os operadores sofreram modificações para satisfazer as restrições estabelecidas para o problema, a fim de encontrar soluções viáveis. Os operadores genéticos utilizados para este trabalho foram de cruzamento e mutação.

O operador de cruzamento utilizado foi o baseado em ordem, onde era gerado aleatoriamente uma matriz de seleção composta de 1's e 0's com tamanho correspondente ao da matriz do período (Figura 5). Para seleção dos pais foi utilizado o método da roleta, que consiste em selecionar os pais mais aptos ou não, a fim de manter a diversidade da população.

| | AULAS | | |
|---|---|---|---|
| | 1-2 | 3-4 | 5-6 |
| SEG | 1 | 0 | 0 |
| TER | 0 | 1 | 0 |
| QUA | 1 | 0 | 1 |
| QUI | 0 | 1 | 1 |
| SEX | 0 | 1 | 0 |

**Figura 5. Matriz de seleção. (Fonte: adaptada de [Almeida 2015])**

A matriz de seleção da Figura 5 é gerada aleatoriamente e define uma etapa do operador de cruzamento em ordem que ocorre após a seleção dos pais. De acordo com o conceito deste operador, citado brevemente na Seção 3, quando o valor da posição i da matriz é 1, indica que o filho irá herdar a característica de mesma posição da matriz do pai1, caso seja 0 o filho herda do pai2. Essa estratégia pode ocasionar a perda de características relevantes dos pais ao longo das gerações, ou gerar anomalias.

Para evitar que essa perda de informações ocorra, foi utilizado um vetor auxiliar para guardar as características não herdadas do pai1. Em seguida, este vetor passa por um processo de ordenação que consiste em definir a ordem que os elementos do vetor aparecem na matriz de características do pai2. Com isso, o vetor passa a ter as características do pai2 e minimiza a perda de informações relevantes dos pais. Após a ordenação do vetor, as características do pai2 são herdadas pelo filho (cada posição i da matriz com valor 0 é preenchida na mesma posição da matriz do filho com a respectiva característica do vetor resultante). Na Figura 6 é mostrado o passo a passo do processo de cruzamento em ordem utilizado no algoritmo.

Seguindo o processamento do AG, após o filho ser gerado ele é submetido ao

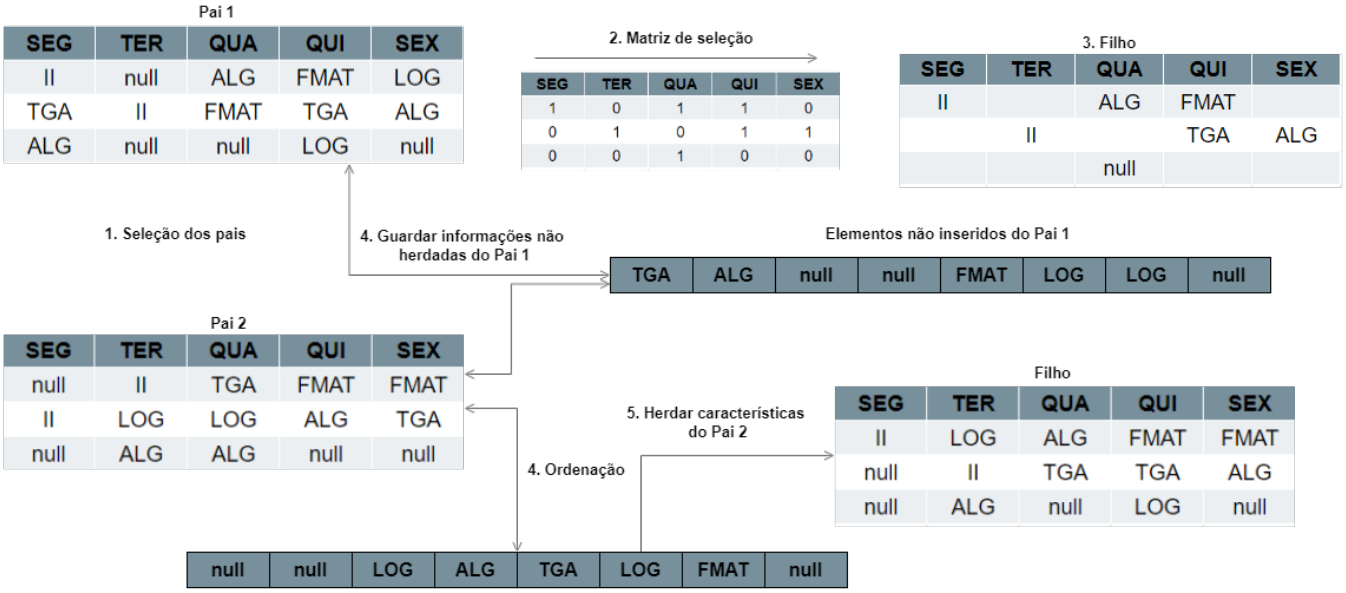



**Figura 6. Simulação do cruzamento em ordem. (Fonte: adaptada de [Almeida 2015])**

operador de mutação, a fim de explorar a diversidade genética, alterando uma ou mais característica do vetor. A mutação utilizada neste trabalho foi a baseada em ordem por meio da permutação de genes do mesmo período, citada brevemente na Seção 3. Este operador sofreu modificações para se adaptar ao problema, devido às restrições que foram estabelecidas. Logo, o operador ficou responsável por verificar se o indivíduo gerado durante o processo de montagem de horário não infringe nenhuma das restrições.

Na biologia, o processo de mutação de um indivíduo pode gerar certas anomalias, isso não seria diferente no contexto dos algoritmos genéticos. Para este trabalho, o algoritmo proposto utiliza um operador de mutação que atua na correção de anomalias que podem ser geradas durante a etapa de cruzamento dos indivíduos. Logo, este operador tem como principal objetivo corrigir os cromossomos mal formados, que infringe algumas das regras estabelecidas. Como este trabalho também visa a blocagem de horários (minimização de aulas vagas), o operador de mutação utilizado pelo algoritmo também ficou responsável em minimizar a ocorrência de aulas vagas, propondo horários com uma distribuição de blocos mais igualitária, uniforme, minimizando o intervalo entre aulas.

O operador de mutação realiza quatro etapas para cada um dos indivíduos da população. A primeira etapa é a verificação de aulas sequenciais de uma turma no mesmo dia. Esta verificação é realizada em todos os dias e períodos do indivíduo. Caso seja detectada a ocorrência de aulas de uma turma no mesmo dia, o algoritmo irá pegar o professor que leciona esta turma e selecionar uma lista de todas as turmas que não são ministradas por ele no período. Após isso, é selecionado aleatoriamente uma turma desta lista e realizada a troca de posição com a turma que foi detectada como incorreta pelo algoritmo.

Como este trabalho visa propor um algoritmo que também tenha como objetivo a blocagem de horários a favor do rendimento acadêmico dos discentes, se fez necessário

incluir algumas restrições em prol do corpo discente. Segundo [Almeida 2015], o curso de BSI da UFRN possui uma parcela significativa de alunos que moram em outras cidades e não podem ficar até o fim do sexto (último) horário. Com isso, foi definido eliminar a ocorrência de dias que possuam aulas apenas nos últimos horários.

Dito isso, a segunda etapa do operador de mutação é verificar a ocorrência dos eventos de aulas vagas entre aulas e de dias que possuam aulas apenas nos últimos horários para cada período do indivíduo. Caso seja detectada a ocorrência de algum desses eventos, o operador deve realizar uma permutação no mesmo período entre os blocos referentes ao mesmo dia, alocando a turma em um bloco (posição) anterior.

As etapas descritas nos parágrafos anteriores, objetivam a garantia da blocagem de horários e uma distribuição igualitária das aulas ao longo da semana, a fim de favorecer o rendimento acadêmico do discente (conforme falado na Seção 3). Na Figura 7 são ilustradas o funcionamento das etapas do operador de mutação que até agora foram descritas.

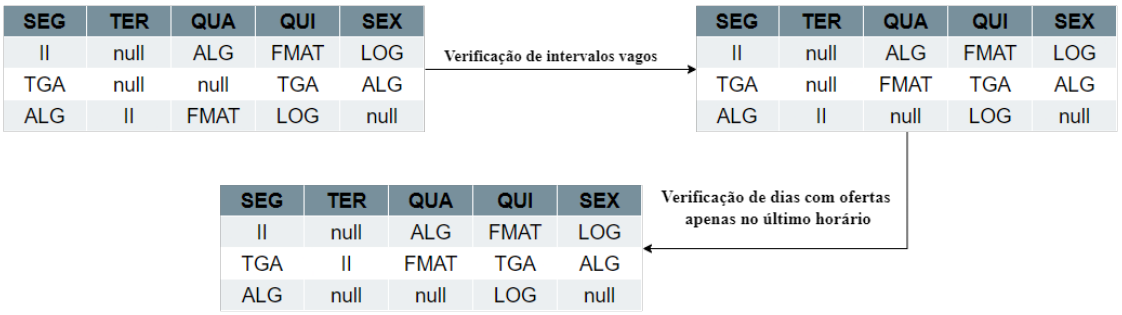



**Figura 7. Mutação - correção de anomalias. (Fonte: adaptada de [Almeida 2015])**

Após o processamento das etapas descritas anteriormente, o operador de mutação busca adequar os horários dos períodos às restrições de disponibilidade estabelecidas pelo corpo docente. O algoritmo percorre todos os dias de todos os períodos do indivíduo e busca por turmas que estão sendo ministradas em dias que o professor não tem disponibilidade. Caso essa infração seja detectada, é realizada uma permutação no mesmo período entre os dias de disponibilidade do professor e um bloco de aula da semana com o dia detectado como infração. Em outras palavras, o algoritmo seleciona aleatoriamente um dia da semana que o professor tenha disponibilidade e um bloco de aula da semana. Após isso é realizada a permutação entre o dia detectado como infração com o dia e bloco selecionados pelo algoritmo.

Como o operador de mutação considera as restrições de disponibilidade do corpo docente, para garantir que o processo de montagem de horário ocorra da forma correta, é necessário respeitar a restrição de que um mesmo professor não pode ministrar duas disciplinas distintas no mesmo horário de forma simultânea. Com isso, o operador de mutação também é responsável pela verificação de choques de horários entre professores.

A verificação de choques de horário entre professores é realizada da seguinte forma: todos os períodos são comparados entre si verificando se não existem turmas com o mesmo professor no mesmo horário. Ao detectar-se o choque de horário entre dois períodos, as posições do choque são armazenadas em uma tupla e um dos períodos é se-

lecionado aleatoriamente para sofrer um processo de permutação. Para este processo, são definidas algumas listas de prioridades, considerando o período selecionado, na seguinte ordem: aulas vagas no primeiro horário, aulas vagas no segundo horário e as demais turmas. Após isso, o algoritmo verifica, na ordem estabelecida, qual das listas possui elementos e a considera como base para realizar a permutação. Logo, a permutação irá selecionar aleatoriamente um dos elementos da lista de prioridade selecionada e realizar a permutação com as posições do choque de horário que foram armazenadas. Na Figura 8 é ilustrado o processo de correção de choque de horário.

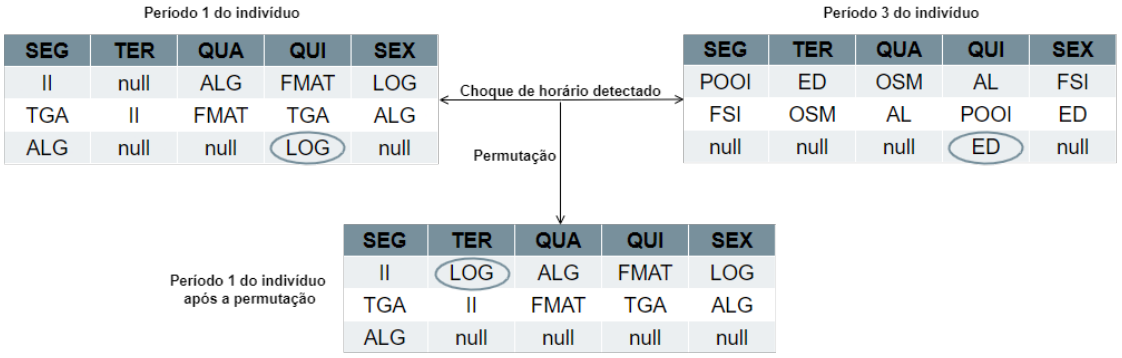



**Figura 8. Mutação - correção de choque de horário. (Fonte: adaptada de [Almeida 2015])**

Os operadores de seleção, cruzamento e mutação são etapas básicas de processamento de um AG e fundamentais para promover diversidade e maior cobertura de exploração por novas soluções. Após executar esses passos, o algoritmo proposto em [Almeida 2015] emprega um modelo geracional, onde a nova prole substitui a prole anterior, em outras palavras, os filhos substituem os pais. Esse modelo pode resultar na perda dos melhores indivíduos da população, podendo aumentar o tempo de convergência do algoritmo ou até mesmo não achar uma solução adequada ao problema.

Portanto, o presente trabalho utilizou uma seleção de sobreviventes baseado em fitness com estratégia elitista, para ao menos manter as chances de bons indivíduos não serem perdidos ao longo das gerações. Para este trabalho, a estratégia elitista funciona da seguinte forma: inicialmente a nova prole gerada é concatenada com a população antiga. Após isso, os indivíduos empatados com a maior aptidão são transferidos para a próxima geração e os demais indivíduos da população são selecionados usando o método da roleta.

### 4.5. Função de avaliação

Na biologia, após determinado indivíduo ser gerado passando pelos processos de cruzamento e mutação, novas características são agregadas ao seu DNA e de certa forma elas podem ou não contribuir para sua adaptação, sendo o meio ambiente o avaliador de tais características. Para os AGs esse procedimento de avaliação é realizado por uma função de avaliação, que é responsável por definir o nível de aptidão das soluções geradas. O valor de aptidão determina a qualidade da solução e influencia diretamente na etapa de seleção dos pais, pois soluções com maior nível de aptidão possuem maior probabilidade de serem selecionadas durante a etapa de cruzamento e, assim, gerar novas possíveis soluções.

A função de avaliação é dada pela soma de punições quando o indivíduo infringe alguma restrição do problema. A punição se faz necessária, pois os indivíduos gerados podem infringir determinadas restrições do problema e não se adequar aos critérios estabelecidos, o que pode influenciar no nível de aptidão da população. As punições são representadas de forma numérica, onde o valor determinado para cada restrição infringida é acrescida de 1. Os critérios estabelecidos foram os seguintes:

1. Último horário - segundo [Almeida 2015], os alunos do curso de BSI moram em cidades distantes e com isso dependem de transportes com saída em horário definido. Portanto, dias com aulas apenas no último horário resulta em um longo tempo de ociosidade dos discentes. Pode ter casos que o discente tenha que sair antes mesmo da aula acabar e perder o conteúdo ministrado da disciplina.
2. Aulas vagas - conforme cita [Almeida 2015] em seu trabalho, um longo tempo de ociosidade pode influenciar negativamente no desempenho acadêmico do corpo discente. Logo, minimizar a ocorrência de intervalos vagos entre aulas influencia positivamente no rendimento acadêmico dos alunos.
3. Aulas de uma mesma disciplina no mesmo dia - segundo [Almeida 2015], aulas de uma mesma disciplina no mesmo dia, são cansativas e leva o aluno a desmotivação. Em alguns casos existem disciplinas que devido a sua carga horária podem ocorrer com mais frequência durante o dia, mas elas não devem ser consideradas como referência para as demais. Logo, garantir uma distribuição igualitária das disciplinas é essencial para o desempenho do corpo discente.
4. Choque de horário - a restrição de choque de horário é fundamental quando trata-se de escalonamento de horário. Apesar do operador de mutação tentar minimizar a ocorrência de choques, caso ainda persista, uma punição é aplicada ao indivíduo.
5. Disponibilidade dos professores - no que diz respeito ao corpo docente, é fundamental durante a montagem de horário também levar em consideração as restrições dos professores, a fim de garantir suas preferências de horários para ministrar suas aulas. Atendendo a esta restrição e as outras citadas anteriormente, tanto os docentes quanto os discentes terão suas restrições satisfeitas e um ambiente propício para o desenvolvimento do processo ensino-aprendizagem. Assim, uma punição foi adotada quando o algoritmo não puder atender a questão da disponibilidade dos professores.

Mediante as punições citadas anteriormente, este trabalho definiu uma função de avaliação de maximização que é dada pelo inverso da soma de punições quando o indivíduo infringe alguma restrição do problema ou igual a 2 quando a soma de punições é igual a zero, como mostrado na Equação 2. Por ser uma função de maximização significa que quanto maior for o valor de aptidão (fitness) melhor será o indivíduo (solução). Esta função utiliza os mesmos parâmetros que a função apresentada em [Almeida 2015] com diferença apenas do acréscimo da variável "l" e por se tratar de uma função de maximização e não de minimização.

$$
\begin{aligned}
S(i) &= \sum_{p=1}^{x} (a_p + v_p + u_p + l_p + p_f) + (k * ch) \\
max\ F(i) &= \begin{cases} 2,\ se\ S(i) = 0 \\ \frac{1}{S(i)},\ caso\ contrário \end{cases}
\end{aligned}
\tag{2}
$$

onde $i$ é o indivíduo (solução); $p$ é o período; $x$ o número total de períodos; $a$ o número de aulas vagas no primeiro horário, desde que possua aula no segundo horário; v = número de aulas vagas entre aulas; $u$ o número de aulas que são ofertadas apenas no último horário em determinado(s) dia(s); $l$ o número de horários que possuem aulas de uma mesma disciplina no mesmo dia; $pf$ o número de infrações quanto a disponibilidade dos professores; $ch$ o número de choque de horários nos períodos; e $k$ o peso atribuído ao choque de horários.

De acordo com a Equação 2, é possível observar que $S(i)$ e $F(i)$ são inversamente proporcionais, pois quanto menor for $S(i)$ maior será o valor de $F(i)$ e, consequentemente, melhor será o indivíduo. Para este trabalho, foi definido que caso $S(i)$ seja igual 0, ou seja, o algoritmo encontrou uma solução que não infringiu nenhuma restrição, o valor da função $F(i)$ será igual 2. Caso contrário, com $S(i)$ maior do 0 o valor de $F(i)$ será o inverso de $S(i)$. Portanto, os melhores indivíduos possuem um fitness igual a 2, e os que apresentarem fitness igual 1, significa que o somatório de erros deu 1 e que ainda existem infrações sendo cometidas.

### 4.6. Implementação

Para a implementação do algoritmo genético e execução dos experimentos deste trabalho, foi utilizado a linguagem de programação Python 3.9.7 e o ambiente de desenvolvimento integrado PyCharm 2021.3.2 (Community Edition). Além disso, também contou com o auxílio de um banco de dados SQL para armazenamento da base de dados e tornar mais prático a execução dos experimentos.

A linguagem de programação Python foi escolhida devido a sua flexibilidade e praticidade de desenvolvimento. Além disso, essa linguagem é bastante utilizada na literatura para a implementação de algoritmos genéticos. Apesar da linguagem dispor de uma frameworks (bibliotecas) prontos para computação genética, o presente trabalho optou por uma implementação limpa, sem o auxílio de pacotes de terceiros. Essa decisão foi tomada para evitar dependências e ter controle total do algoritmo, favorecendo ajustes futuros.

Os parâmetros genéticos utilizados neste artigo foram os mesmos utilizados em [Almeida 2015], que foram os seguintes: o tamanho da população, que foi escolhido a partir de sugestão da literatura, de valor igual a 100; e a quantidade de gerações, que foi definida com valor igual a 2000.

O algoritmo proposto neste trabalho foi projetado para ser interrompido em duas situações: ao encontrar a solução mais aceitável, com fitness 2; ou caso alcançasse o limite máximo de gerações (2000). Essa situação de parada ao encontrar a solução ótima foi projetada para que os experimentos fossem executados com mais velocidade, devido a grande quantidade de 10 amostras de disponibilidade e ao prazo de entrega deste trabalho.

O algoritmo proposto neste trabalho segue a sequência lógica que é comum em qualquer implementação de AG, que seria na seguinte ordem: geração da população inicial, seleção dos pais, cruzamento, mutação, avaliação de cada indivíduo da nova população gerada e seleção de sobreviventes, se chegar ao fim da busca o algoritmo pára, caso contrário retoma a execução do passo de seleção e repete a sequência até finalizar a busca.

A etapa de seleção de sobreviventes implementada neste trabalho, funciona da seguinte forma: o algoritmo concatena a nova população com a antiga, seleciona os melhores indivíduos dessa junção e cria uma nova população com esses indivíduos e se estiver faltando indivíduos para completar o tamanho máximo da população, são selecionados indivíduos complementares por meio do método da roleta.

Para comprovar o funcionamento e validar a solução proposta neste trabalho, no final da execução, o algoritmo salva a grade de horários encontrada em um documento txt e exibe um gráfico para acompanhamento da evolução, com as médias de aptidão a cada geração processada.

## 5. Resultados

Os experimentos foram executados em duas máquinas independentes, sendo elas: um computador desktop AMD Ryzen 5 5600G 3.90 GHz com 16GB RAM e SSD M.2 2280 NVMe de 500 GB, e um notebook Intel Core i7-7500U 2.70 GHz com 8GB RAM e SSD A400 de 240 GB. Ambas as máquinas possuem Windowns 10 como SO.

Antes de realizar os experimentos, o conjunto de dados de entrada, conforme descrito na Seção 4.1, passou por uma série de processos até ficar consolidado e pronto para uso. Após ter a base consolidada foi adicionado os dados de disponibilidade dos professores, adicionando ao modelo do algoritmo uma nova variável com o intuito de adicionar mais complexidade ao problema.

Para os dados de disponibilidade foram definidas 10 amostras de dados que foram criadas baseadas nas propriedades descritas na Seção 4.1. As amostras contêm os dados referentes aos dias da semana de disponibilidade de cada professor, onde cada docente possui uma lista de dias úteis de preferência para ministrar suas aulas. Esses dias foram representados no banco de dados como um valor numérico inteiro que vai de 0 até 4, onde 0 representa segunda-feira, 1 terça-feira e assim por diante até a sexta-feira (4). Na Figura 9 é possível observar uma das 10 amostras utilizadas neste trabalho, o restante das amostras podem ser visualizadas no Apêndice A.

Analisando a Figura 9 é possível observar que ao lado do nome dos professores estão as respectivas disciplinas que eles ministram. A coluna dias informa os dias da semana de disponibilidade de cada professor. Esses dados de disponibilidade são permutações dos dias da semana, que foram gerados manualmente preservando sempre que possível a disparidade entre as amostras, a fim de cobrir o máximo de cenários possíveis.

O algoritmo proposto neste trabalho, assim como o apresentado no trabalho [Almeida 2015], se destaca em relação às demais soluções existentes na área, devido ao fato do algoritmo explorar o funcionamento dos operadores genéticos de cruzamento e, principalmente, de mutação com foco aos objetivos do trabalho. Geralmente a maioria das soluções o operador de mutação está atrelado a uma taxa de mutação, que determina a probabilidade de ocorrer mutação. Já neste trabalho o operador de mutação é independente de qualquer limitador de ocorrência e sempre é aplicado aos indivíduos da população, pois tal operador foi projetado para agir corrigindo os indivíduos que violem as restrições.

As simulações foram executadas seguindo alguns parâmetros genéticos que foram

| | PROFESSOR | DIAS | TOTAL DIAS |
|---|---|---|---|
| AMOSTRA 1 | GILSON (TGA,MTC,TGS,FSI,EMP) | SEG,TER,QUA,QUI,SEX | 5 |
| | FABRICIO (POOI,POO2,ES1,PWEB) | SEG,TER,QUI,SEX | 4 |
| | FLAVIUS (ALG,PROG,SAD) | SEG,TER,QUA,QUI,SEX | 5 |
| | JOÃO PAULO (LOG,ED,AEX) | SEG,QUA,SEX | 3 |
| | LUIZ PAULO (II,ARQ) | SEG,TER,SEX | 3 |
| | RICARDO (CEC) | TER,QUA,QUI | 3 |
| | TACIANO (BD,ES2,PABD) | SEG,TER,QUI,SEX | 4 |
| | LUCIANO (FMAT,PE,MATF) | TER,QUA,SEX | 3 |
| | KARLIANE (OSM,GPS) | SEG,QUI,SEX | 3 |
| | JOÃO BORGES (SO,RED,SEG) | TER,QUA,QUI,SEX | 4 |
| | JOSÉ ENÉAS (TSI) | SEG,QUA,SEX | 3 |
| | DÉSIO (CAL) | TER,QUA,QUI | 3 |
| | DELSON (AL) | SEG,QUA,SEX | 3 |
| | CÉLIA (LPT) | SEG,TER,SEX | 3 |
| | CARLOS (ETIC) | TER,QUA,SEX | 3 |
| | TIAGO (PV) | SEG,QUA,SEX | 3 |
| | TALITHA (FIL) | SEG,QUA,SEX | 3 |
| | LEOMARQUES (ING) | TER,QUA,SEX | 3 |
| | VYRNA (DIR) | SEG,QUA,QUI | 3 |

**Figura 9. Amostra 1 de disponibilidade. (Fonte: o autor)**

definidos durante a modelagem do algoritmo, conforme citado na Seção 4.6. Logo, todas as simulações foram executadas com tamanho da população igual a 100 e a quantidade de gerações igual a 2000. Ao todo foram executadas 10 simulações, correspondendo ao total de 10 amostras (Apêndice A).

Ao final de cada simulação o algoritmo gera um gráfico com as médias de aptidão de cada geração, que serve para acompanhar o desempenho do algoritmo ao longo das gerações. Na Figura 10 é possível observar o gráfico de resultado criado pelo algoritmo após o processamento da amostra 1 (Figura 9), os demais gráficos para o restante das amostras podem ser visualizados no Apêndice B.

De acordo com a Figura 10, é possível observar que ao longo das gerações o algoritmo consegue evoluir a média de aptidão da população, ou seja, consegue gerar melhores indivíduos com maior nível de aptidão durante o processo evolutivo. Isso comprova que o algoritmo conseguiu achar uma solução para a amostra 2 da Figura 9. Nesse caso, o algoritmo parou ao encontrar um indivíduo com valor de aptidão 2 na 864º geração e com média 0,189184 na última geração. Na Figura 11 está sendo apresentado o resultado do 2º período encontrado pelo algoritmo ao executar a amostra 1. Os resultados da grade de horários completa para a amostra 1 podem ser visualizados no Apêndice C.

A grade de horário gerada na Figura 11 é apenas um período de um total de 8 períodos da solução encontrada pelo algoritmo que obteve valor de aptidão 2. Analisando o horário é possível notar que as restrições definidas na Seção 4.5 foram respeitadas e

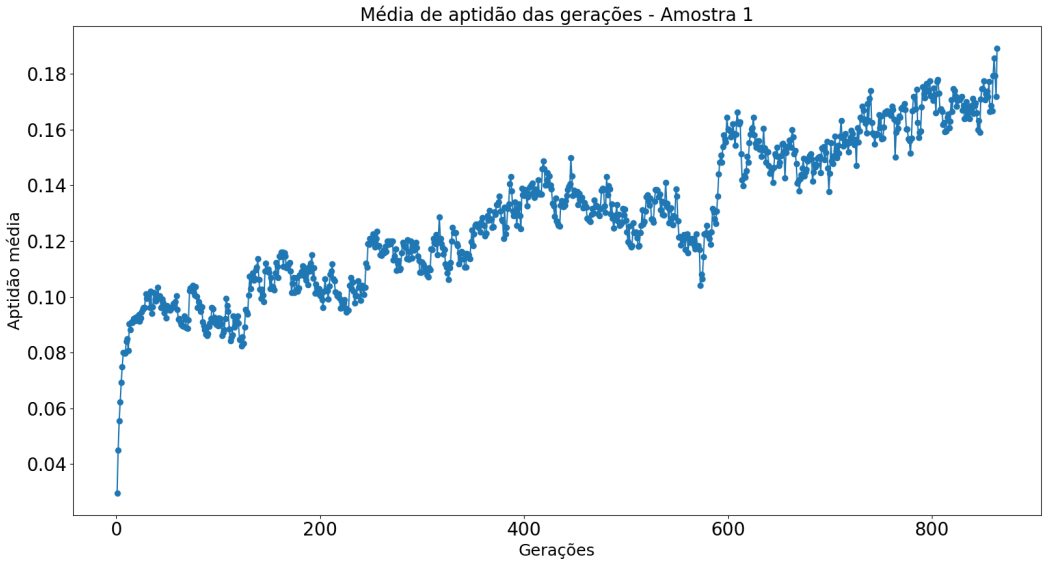



**Figura 10. Média de aptidão das gerações da amostra 1. (Fonte: o autor)**

| | AULAS | | | |
|---|---|---|---|---|
| | 1-2 | 3-4 | 5-6 | |
| SEG | LPT | PROG | null | |
| TER | MTC | PROG | null | 2 |
| QUA | TGS | CAL | PROG | |
| QUI | CAL | TGS | null | |
| SEX | LPT | MTC | null | |

**Figura 11. Horário gerado pelo algoritmo para o 2º período da amostra 1. (Fonte: o autor)**

que as disciplinas obtiveram uma boa distribuição na semana. Outro fator importante que merece destaque, é que antes de chegar na 2000ª geração o algoritmo conseguiu encontrar uma solução ótima para a amostra 1. Na Tabela 3 é detalhado os resultados que foram obtidos na execução de todas as amostras.

A Tabela 3 apresenta os resultados obtidos durante o processamento evolutivo do algoritmo com as amostras que foram geradas manualmente (Apêndice A). Vale ressaltar, que a coluna "Geração de parada" representa a última geração que o algoritmo criou ao encontrar um indivíduo com valor de aptidão 2 ou até executar a quantidade máxima de 2000 gerações. Já a coluna "Melhor aptidão" informa o valor de aptidão do melhor indivíduo da última geração processada pelo algoritmo.

Mediante os resultados expostos na Tabela 3, é possível afirmar que o algoritmo conseguiu encontrar solução ótima (valor de aptidão igual a 2) para todas as amostras definidas neste trabalho. Além disso, também é notável que para algumas amostras o algoritmo conseguiu convergir bem antes de alcançar a 2000ª geração, com destaque para

| Amostra | Geração de parada | Aptidão média da última geração | Melhor aptidão |
|---|---|---|---|
| 1 | 864 | 0,189184 | 2.0 |
| 2 | 1816 | 0,187291 | 2.0 |
| 3 | 333 | 0,234986 | 2.0 |
| 4 | 455 | 0,179098 | 2.0 |
| 5 | 48 | 0,139181 | 2.0 |
| 6 | 1625 | 0,442790 | 2.0 |
| 7 | 834 | 0,323438 | 2.0 |
| 8 | 496 | 0,159941 | 2.0 |
| 9 | 548 | 0,247523 | 2.0 |
| 10 | 1358 | 0,329141 | 2.0 |

**Tabela 3. Resultados das simulações com as amostras. (Fonte: o autor)**

amostra 5 que teve uma solução ótima obtida na 48ª geração. Isso ocorre devido as estratégias de aleatoriedade que são implementadas para a geração da população inicial e dos operadores genéticos do AG. Importante destacar que cada uma das amostras foram executadas apenas uma única vez e que o algoritmo obteve um tempo médio por iteração de aproximadamente 16,3 segundos. Esse tempo médio é apenas uma estimativa grosseira, pois as simulações foram executadas em máquinas que rodavam outros processos em paralelo além do algoritmo, o que impede uma definição exata do tempo de processamento do AG.

## 6. Discussões e Trabalhos futuros

O problema do escalonamento de horários normalmente é um trabalho árduo e moroso de ser solucionado manualmente, devido a sua alta complexidade que depende da quantidade de variáveis associadas ao contexto do problema. Portanto, utilizar métodos computacionais para automatizar a montagem de horários é uma forma de otimizar esse trabalho manual. A maioria das soluções encontradas na literatura para esse tipo de problema, costumam priorizar as restrições administrativas da instituição e dos docentes, relevando as preferências do corpo docente.

O presente trabalho tem o intuito em utilizar a técnica de algoritmo genético da computação evolutiva para tentar solucionar o problema da montagem de horários acadêmicos, por meio de uma alocação otimizada de horários levando em consideração as restrições do corpo discente, quanto a blocagem de horários acadêmicos, e de disponibilidade dos professores. Logo, o principal diferencial da solução proposta neste trabalho é considerar durante o processo de montagem das grades de horários os alunos e professores da instituição, a fim de garantir uma boa relação no processo ensino-aprendizagem.

Conforme citado na Seção 1, garantir a blocagem de horários com o intuito de minimizar os intervalos vagos entre aulas, influencia positivamente no desempenho acadêmico do corpo discente e também é um dos objetivos deste trabalho. Portanto, analisando os resultados apresentados na Seção 5, é possível afirmar que o algoritmo proposto neste trabalho conseguiu cumprir com os objetivos definidos, encontrando soluções para diferentes conjunto de dados de disponibilidade dos professores e respeitando as restrições para blocagem de horários a favor do rendimento acadêmico dos discentes.

Durante a análise dos gráficos do Apêndice B é perceptível que ao longo das gerações o processo evolutivo do algoritmo conseguiu evoluir a população inicial, ou seja, durante a execução foram gerados indivíduos com maiores níveis de aptidão e suas características conseguiram ser repassadas para a próxima geração. Outro detalhe importante, é que o algoritmo conseguiu encontrar uma solução ótima para todas as amostras geradas (conforme mostrado na Tabela 3), comprovando que o seu modelo é funcional e apresenta resultados satisfatórios que cumprem com os objetivos deste trabalho.

Para trabalhos futuros, sugere-se acrescentar ao algoritmo proposto novas variáveis a serem consideradas durante o processo de escalonamento de horários, como, por exemplo, disponibilidade e capacidade de salas de aula. Ademais, também é sugerido realizar testes com outros tipos de operadores genéticos e executar o algoritmo com novos dados de entrada para testar a sua eficácia e qualidade das respostas. Também sugere-se a criação de um sistema de informação para a Universidade Federal Rural de Pernambuco (UFRPE) utilizando o AG proposto para auxiliar supervisores de área e coordenadores de curso. Além disso, também podem ser realizadas mudanças a nível de implementação, que envolvem otimização do código fonte do projeto (disponibilizado em [Demiro 2022]), a fim de diminuir o tempo de resposta do algoritmo. Uma sugestão de implementação seria adicionar ao método que tenta corrigir as punições de disponibilidade dos professores, um funcionamento que previna tal correção de implantar erros de choques de horários. Por último, é sugerido realizar uma análise mais detalhada e com maior precisão quanto ao tempo médio de processamento do algoritmo.

## Referências


Almeida, M. W. d. S. (2015). Utilização de algoritmos genéticos para montagem de horários acadêmicos com foco na blocagem de horários. Graduação, Universidade Federal do Rio Grande do Norte, Caicó, RN.

Becceneri, J., Ramos, F., Campos Velho, H., Silva, J., Lorena, L., Vijaykumar, N., Santos, R., Rosa, R., and Travelho, J. (2008). Meta-heurísticas e otimização combinatória: Aplicações em problemas ambientais.

Cruz, R. F., dos Santos Júnio, G. P., Fontes, L. B., dos Anjos Santos, M., and da Silva, B. L. C. (2019). Geração automática de horário escolar com algoritmo genético. In *Revista Eixo*, volume 8, pages 230–241, Brasília, DF. Revista Eixo.

de Barros Montin, B. (2022). Algoritmo genético para problema generalizado de atribuição. Graduação, Universidade Federal do Rio Grande do Norte, Presidente Prudente, SP.

Demiro, M. (2022). Código fonte do algoritmo desenvolvido para este trabalho. disponível em: https://github.com/matheusdemiro/timetable-ga.

Freitas, C., Guimarães, P., Neto, M., and Barboza, F. (2014). Uma ferramenta baseada em algoritmos genéticos para a geração de tabela de horário escolar.

ITC (2019). International timetabling competition 2019. disponível em: https://www.itc2019.org/.

Lima, S. J. D. A. et al. (2015). Otimização do problema de roteamento de veículos capacitado usando algoritmos genéticos com heurísticas e representações cromossômicas alternativas.

Maydana, G. S. (2011). Geração automática de quadros de horários para o curso de ciência da computação da unipampa. Graduação, Universidade Federal do Pampa, Alegrete, RS.

Oliveira, A. L. and Manzan, J. R. G. (2022). Implementação de algoritmos genéticos para geração de grade horária de aula: caso iftm – campus avançado uberaba parque tecnológico. In *Brazilian Journal of Development*, volume 8, pages 21222–21237, Curitiba, PR. Brazilian Journal of Development.

Santos, H. G. and Souza, M. J. F. (2007). Programação de horários em instituições educacionais: Formulações e algoritmos. In *X X X I X SBPO - A Pesquisa Operacional e o Desenvolvimento Sustentável*, pages 2827–2882, Fortaleza, CE.

UniTime (2015). Unitime: University timetabling – comprehensive academic scheduling solutions. disponível em: https://www.unitime.org/.

# APÊNDICES

# APÊNDICE A - Amostras de dados de disponibilidade dos professores

| | PROFESSOR | DIAS | TOTAL DIAS |
|---|---|---|---|
| AMOSTRA 1 | GILSON (TGA,MTC,TGS,FSI,EMP) | SEG,TER,QUA,QUI,SEX | 5 |
| | FABRICIO (POOI,POO2,ES1,PWEB) | SEG,TER,QUI,SEX | 4 |
| | FLAVIUS (ALG,PROG,SAD) | SEG,TER,QUA,QUI,SEX | 5 |
| | JOÃO PAULO (LOG,ED,AEX) | SEG,QUA,SEX | 3 |
| | LUIZ PAULO (II,ARQ) | SEG,TER,SEX | 3 |
| | RICARDO (CEC) | TER,QUA,QUI | 3 |
| | TACIANO (BD,ES2,PABD) | SEG,TER,QUI,SEX | 4 |
| | LUCIANO (FMAT,PE,MATF) | TER,QUA,SEX | 3 |
| | KARLIANE (OSM,GPS) | SEG,QUI,SEX | 3 |
| | JOÃO BORGES (SO,RED,SEG) | TER,QUA,QUI,SEX | 4 |
| | JOSÉ ENÉAS (TSI) | SEG,QUA,SEX | 3 |
| | DÉSIO (CAL) | TER,QUA,QUI | 3 |
| | DELSON (AL) | SEG,QUA,SEX | 3 |
| | CÉLIA (LPT) | SEG,TER,SEX | 3 |
| | CARLOS (ETIC) | TER,QUA,SEX | 3 |
| | TIAGO (PV) | SEG,QUA,SEX | 3 |
| | TALITHA (FIL) | SEG,QUA,SEX | 3 |
| | LEOMARQUES (ING) | TER,QUA,SEX | 3 |
| | VYRNA (DIR) | SEG,QUA,QUI | 3 |

| | PROFESSOR | DIAS | TOTAL DIAS |
|---|---|---|---|
| AMOSTRA 2 | GILSON (TGA,MTC,TGS,FSI,EMP) | SEG,TER,QUI,SEX | 4 |
| | FABRICIO (POOI,POO2,ES1,PWEB) | SEG,QUA,QUI,SEX | 4 |
| | FLAVIUS (ALG,PROG,SAD) | SEG,TER,QUA,QUI,SEX | 5 |
| | JOÃO PAULO (LOG,ED,AEX) | SEG,TER,QUA,QUI | 4 |
| | LUIZ PAULO (II,ARQ) | SEG,TER,QUI | 3 |
| | RICARDO (CEC) | TER,QUA,SEX | 3 |
| | TACIANO (BD,ES2,PABD) | SEG,TER,QUA,QUI,SEX | 5 |
| | LUCIANO (FMAT,PE,MATF) | SEG,TER,QUA,QUI | 4 |
| | KARLIANE (OSM,GPS) | TER,QUA,SEX | 3 |
| | JOÃO BORGES (SO,RED,SEG) | SEG,QUI,SEX | 3 |
| | JOSÉ ENÉAS (TSI) | TER,QUA,QUI | 3 |
| | DÉSIO (CAL) | TER,QUA,QUI,SEX | 4 |
| | DELSON (AL) | SEG,QUI,SEX | 3 |
| | CÉLIA (LPT) | TER,QUI,SEX | 3 |
| | CARLOS (ETIC) | TER,QUA,QUI | 3 |
| | TIAGO (PV) | SEG,TER,QUI | 3 |
| | TALITHA (FIL) | SEG,TER,SEX | 3 |
| | LEOMARQUES (ING) | SEG,QUI,SEX | 3 |
| | VYRNA (DIR) | SEG,TER,QUI | 3 |

| AMOSTRA 3 | PROFESSOR | DIAS | TOTAL DIAS |
|---|---|---|---|
| | GILSON (TGA,MTC,TGS,FSI,EMP) | SEG,TER,QUI,SEX | 4 |
| | FABRICIO (POOI,POO2,ES1,PWEB) | SEG,QUA,QUI,SEX | 4 |
| | FLAVIUS (ALG,PROG,SAD) | SEG,TER,QUA,QUI,SEX | 5 |
| | JOÃO PAULO (LOG,ED,AEX) | SEG,TER,QUA,QUI | 4 |
| | LUIZ PAULO (II,ARQ) | SEG,TER,QUI,SEX | 4 |
| | RICARDO (CEC) | SEG,TER,QUA,SEX | 4 |
| | TACIANO (BD,ES2,PABD) | SEG,TER,QUA,QUI,SEX | 5 |
| | LUCIANO (FMAT,PE,MATF) | SEG,TER,QUA,QUI | 4 |
| | KARLIANE (OSM,GPS) | TER,QUA,QUI,SEX | 4 |
| | JOÃO BORGES (SO,RED,SEG) | SEG,TER,QUI,SEX | 4 |
| | JOSÉ ENÉAS (TSI) | SEG,TER,QUA,QUI | 4 |
| | DÉSIO (CAL) | TER,QUA,QUI,SEX | 4 |
| | DELSON (AL) | SEG,TER,QUI,SEX | 4 |
| | CÉLIA (LPT) | TER,QUA,QUI,SEX | 4 |
| | CARLOS (ETIC) | SEG,TER,QUA,QUI | 4 |
| | TIAGO (PV) | SEG,TER,QUI,SEX | 4 |
| | TALITHA (FIL) | SEG,TER,QUI,SEX | 4 |
| | LEOMARQUES (ING) | SEG,TER,QUI,SEX | 4 |
| | VYRNA (DIR) | SEG,TER,QUA,QUI | 4 |

| AMOSTRA 4 | PROFESSOR | DIAS | TOTAL DIAS |
|---|---|---|---|
| | GILSON (TGA,MTC,TGS,FSI,EMP) | SEG,TER,QUI,SEX | 4 |
| | FABRICIO (POOI,POO2,ES1,PWEB) | SEG,TER,QUA,SEX | 4 |
| | FLAVIUS (ALG,PROG,SAD) | SEG,TER,QUA,QUI,SEX | 5 |
| | JOÃO PAULO (LOG,ED,AEX) | SEG,TER,QUA,QUI | 4 |
| | LUIZ PAULO (II,ARQ) | TER,QUA,QUI,SEX | 4 |
| | RICARDO (CEC) | QUA,QUI,SEX | 3 |
| | TACIANO (BD,ES2,PABD) | SEG,TER,QUA,QUI,SEX | 5 |
| | LUCIANO (FMAT,PE,MATF) | SEG,TER,QUA,QUI | 4 |
| | KARLIANE (OSM,GPS) | SEG,TER,QUA,SEX | 4 |
| | JOÃO BORGES (SO,RED,SEG) | SEG,TER,QUI,SEX | 4 |
| | JOSÉ ENÉAS (TSI) | TER,QUA,QUI | 3 |
| | DÉSIO (CAL) | TER,QUA,QUI | 3 |
| | DELSON (AL) | SEG,QUI,SEX | 3 |
| | CÉLIA (LPT) | TER,QUI,SEX | 3 |
| | CARLOS (ETIC) | TER,QUA,QUI | 3 |
| | TIAGO (PV) | SEG,TER,QUI | 3 |
| | TALITHA (FIL) | QUA,QUI,SEX | 3 |
| | LEOMARQUES (ING) | SEG,QUI,SEX | 3 |
| | VYRNA (DIR) | SEG,TER,QUA | 3 |

AMOSTRA 5

| PROFESSOR | DIAS | TOTAL DIAS |
|---|---|---|
| GILSON (TGA,MTC,TGS,FSI,EMP) | SEG,TER,QUA,QUI,SEX | 5 |
| FABRICIO (POOI,POO2,ES1,PWEB) | TER,QUA,QUI,SEX | 4 |
| FLAVIUS (ALG,PROG,SAD) | SEG,TER,QUA,QUI,SEX | 5 |
| JOÃO PAULO (LOG,ED,AEX) | SEG,TER,QUA | 3 |
| LUIZ PAULO (II,ARQ) | SEG,TER,QUI | 3 |
| RICARDO (CEC) | TER,QUI,SEX | 3 |
| TACIANO (BD,ES2,PABD) | SEG,TER,QUA,QUI,SEX | 5 |
| LUCIANO (FMAT,PE,MATF) | SEG,QUA,QUI | 3 |
| KARLIANE (OSM,GPS) | TER,QUA,SEX | 3 |
| JOÃO BORGES (SO,RED,SEG) | SEG,TER,QUA,SEX | 4 |
| JOSÉ ENÉAS (TSI) | QUA,QUI,SEX | 3 |
| DÉSIO (CAL) | TER,QUA,SEX | 3 |
| DELSON (AL) | SEG,QUA,QUI | 3 |
| CÉLIA (LPT) | QUA,QUI,SEX | 3 |
| CARLOS (ETIC) | SEG,QUI,SEX | 3 |
| TIAGO (PV) | SEG,TER,QUA | 3 |
| TALITHA (FIL) | TER,QUA,QUI | 3 |
| LEOMARQUES (ING) | QUA,QUI,SEX | 3 |
| VYRNA (DIR) | SEG,QUA,SEX | 3 |

AMOSTRA 6

| PROFESSOR | DIAS | TOTAL DIAS |
|---|---|---|
| GILSON (TGA,MTC,TGS,FSI,EMP) | SEG,TER,QUA,QUI,SEX | 5 |
| FABRICIO (POOI,POO2,ES1,PWEB) | SEG,TER,QUA,QUI | 4 |
| FLAVIUS (ALG,PROG,SAD) | SEG,TER,QUA,QUI,SEX | 5 |
| JOÃO PAULO (LOG,ED,AEX) | TER,QUA,SEX | 3 |
| LUIZ PAULO (II,ARQ) | QUA,QUI,SEX | 3 |
| RICARDO (CEC) | TER,QUA,SEX | 3 |
| TACIANO (BD,ES2,PABD) | SEG,TER,QUA,QUI | 4 |
| LUCIANO (FMAT,PE,MATF) | SEG,TER,SEX | 3 |
| KARLIANE (OSM,GPS) | QUA,QUI,SEX | 3 |
| JOÃO BORGES (SO,RED,SEG) | SEG,TER,QUA,QUI | 4 |
| JOSÉ ENÉAS (TSI) | SEG,QUA,QUI | 3 |
| DÉSIO (CAL) | SEG,TER,QUI | 3 |
| DELSON (AL) | SEG,QUI,SEX | 3 |
| CÉLIA (LPT) | SEG,QUA,QUI | 3 |
| CARLOS (ETIC) | SEG,QUA,SEX | 3 |
| TIAGO (PV) | SEG,QUA,QUI | 3 |
| TALITHA (FIL) | TER,QUA,SEX | 3 |
| LEOMARQUES (ING) | SEG,QUA,QUI | 3 |
| VYRNA (DIR) | SEG,TER,QUA | 3 |

AMOSTRA 7

| PROFESSOR | DIAS | TOTAL DIAS |
|---|---|---|
| GILSON (TGA,MTC,TGS,FSI,EMP) | SEG,TER,QUA,QUI,SEX | 5 |
| FABRICIO (POOI,POO2,ES1,PWEB) | SEG,QUA,QUI,SEX | 4 |
| FLAVIUS (ALG,PROG,SAD) | SEG,TER,QUA,QUI,SEX | 5 |
| JOÃO PAULO (LOG,ED,AEX) | TER,QUA,SEX | 3 |
| LUIZ PAULO (II,ARQ) | SEG,TER,SEX | 3 |
| RICARDO (CEC) | SEG,TER,QUI | 3 |
| TACIANO (BD,ES2,PABD) | SEG,TER,QUA,QUI,SEX | 5 |
| LUCIANO (FMAT,PE,MATF) | QUA,QUI,SEX | 3 |
| KARLIANE (OSM,GPS) | TER,QUA,SEX | 3 |
| JOÃO BORGES (SO,RED,SEG) | SEG,TER,QUA,SEX | 4 |
| JOSÉ ENÉAS (TSI) | SEG,TER,QUI | 3 |
| DÉSIO (CAL) | SEG,QUA,SEX | 3 |
| DELSON (AL) | TER,QUA,QUI | 3 |
| CÉLIA (LPT) | TER,QUI,SEX | 3 |
| CARLOS (ETIC) | SEG,QUA,QUI | 3 |
| TIAGO (PV) | SEG,QUI,SEX | 3 |
| TALITHA (FIL) | SEG,QUA,QUI | 3 |
| LEOMARQUES (ING) | TER,QUI,SEX | 3 |
| VYRNA (DIR) | SEG,TER,SEX | 3 |

AMOSTRA 8

| PROFESSOR | DIAS | TOTAL DIAS |
|---|---|---|
| GILSON (TGA,MTC,TGS,FSI,EMP) | SEG,TER,QUA,QUI,SEX | 5 |
| FABRICIO (POOI,POO2,ES1,PWEB) | SEG,TER,QUA,SEX | 4 |
| FLAVIUS (ALG,PROG,SAD) | SEG,TER,QUA,QUI,SEX | 5 |
| JOÃO PAULO (LOG,ED,AEX) | SEG,QUA,QUI | 3 |
| LUIZ PAULO (II,ARQ) | QUA,QUI,SEX | 3 |
| RICARDO (CEC) | TER,QUI,SEX | 3 |
| TACIANO (BD,ES2,PABD) | SEG,QUA,QUI,SEX | 4 |
| LUCIANO (FMAT,PE,MATF) | TER,QUA,QUI | 3 |
| KARLIANE (OSM,GPS) | SEG,TER,SEX | 3 |
| JOÃO BORGES (SO,RED,SEG) | SEG,QUA,QUI,SEX | 4 |
| JOSÉ ENÉAS (TSI) | TER,QUI,SEX | 3 |
| DÉSIO (CAL) | TER,QUI,SEX | 3 |
| DELSON (AL) | SEG,TER,QUI | 3 |
| CÉLIA (LPT) | QUA,QUI,SEX | 3 |
| CARLOS (ETIC) | TER,QUI,SEX | 3 |
| TIAGO (PV) | SEG,TER,QUA | 3 |
| TALITHA (FIL) | SEG,TER,SEX | 3 |
| LEOMARQUES (ING) | SEG,TER,QUA | 3 |
| VYRNA (DIR) | SEG,TER,QUI | 3 |

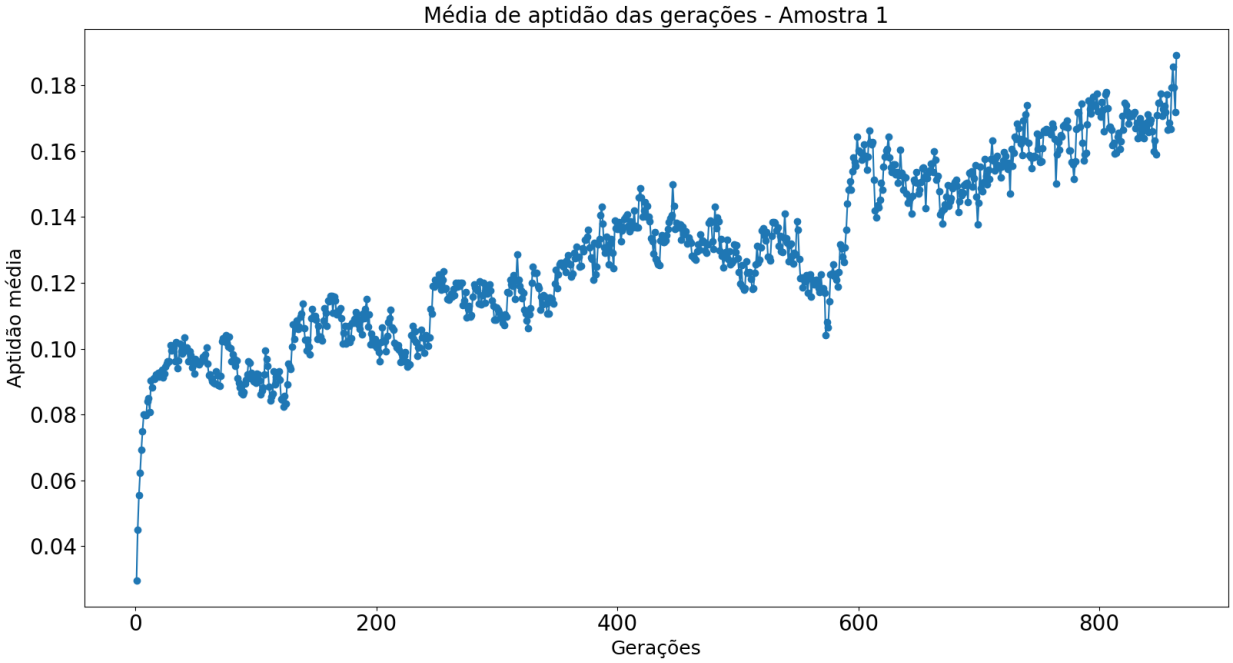
AMOSTRA 9

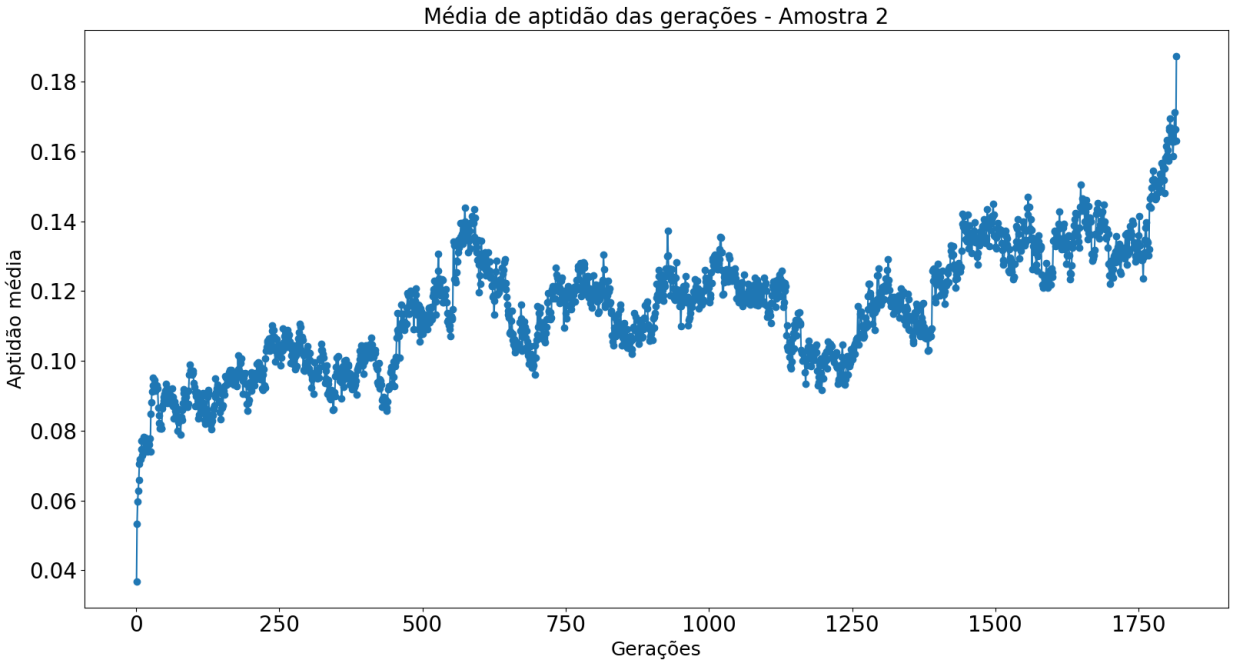
| PROFESSOR | DIAS | TOTAL DIAS |
|---|---|---|
| GILSON (TGA,MTC,TGS,FSI,EMP) | SEG,TER,QUA,QUI,SEX | 5 |
| FABRICIO (POOI,POO2,ES1,PWEB) | TER,QUA,QUI,SEX | 4 |
| FLAVIUS (ALG,PROG,SAD) | SEG,TER,QUA,QUI,SEX | 5 |
| JOÃO PAULO (LOG,ED,AEX) | TER,QUI,SEX | 3 |
| LUIZ PAULO (II,ARQ) | TER,QUA,SEX | 3 |
| RICARDO (CEC) | SEG,QUI,SEX | 3 |
| TACIANO (BD,ES2,PABD) | TER,QUA,QUI,SEX | 4 |
| LUCIANO (FMAT,PE,MATF) | SEG,QUI,SEX | 3 |
| KARLIANE (OSM,GPS) | SEG,QUA,SEX | 3 |
| JOÃO BORGES (SO,RED,SEG) | TER,QUA,QUI,SEX | 4 |
| JOSÉ ENÉAS (TSI) | SEG,QUI,SEX | 3 |
| DÉSIO (CAL) | QUA,QUI,SEX | 3 |
| DELSON (AL) | SEG,QUA,QUI | 3 |
| CÉLIA (LPT) | SEG,QUA,SEX | 3 |
| CARLOS (ETIC) | TER,QUI,SEX | 3 |
| TIAGO (PV) | TER,QUA,SEX | 3 |
| TALITHA (FIL) | SEG,TER,QUI | 3 |
| LEOMARQUES (ING) | SEG,QUA,SEX | 3 |
| VYRNA (DIR) | TER,QUA,SEX | 3 |

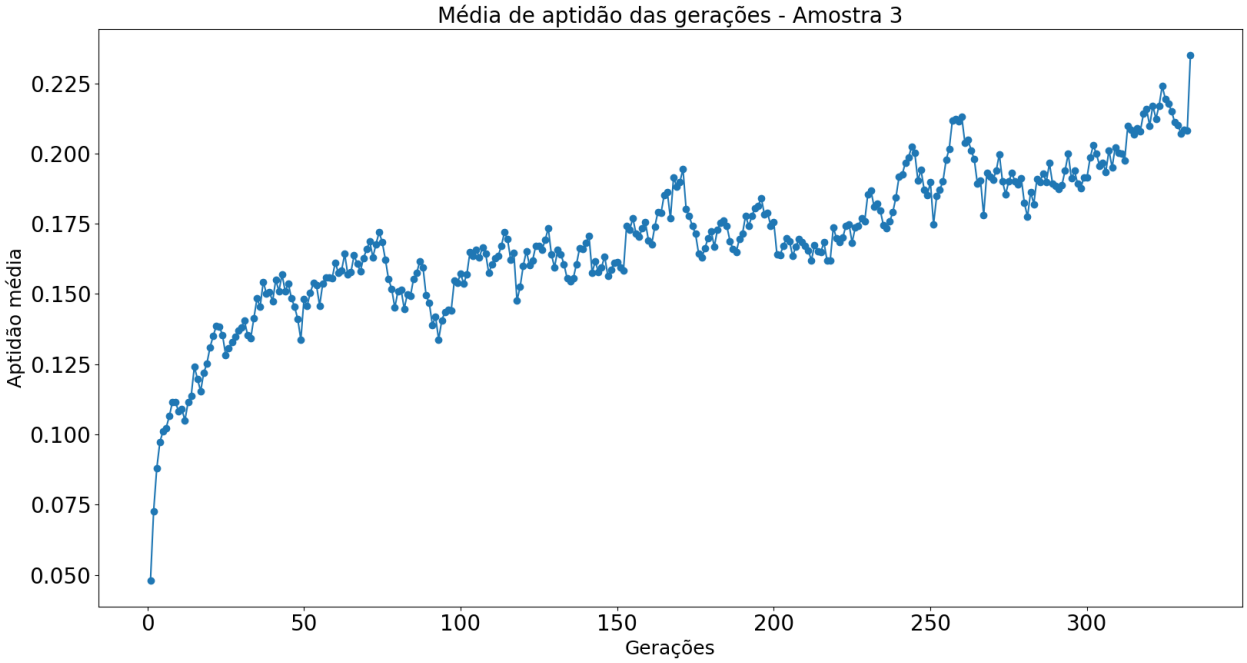
AMOSTRA 10

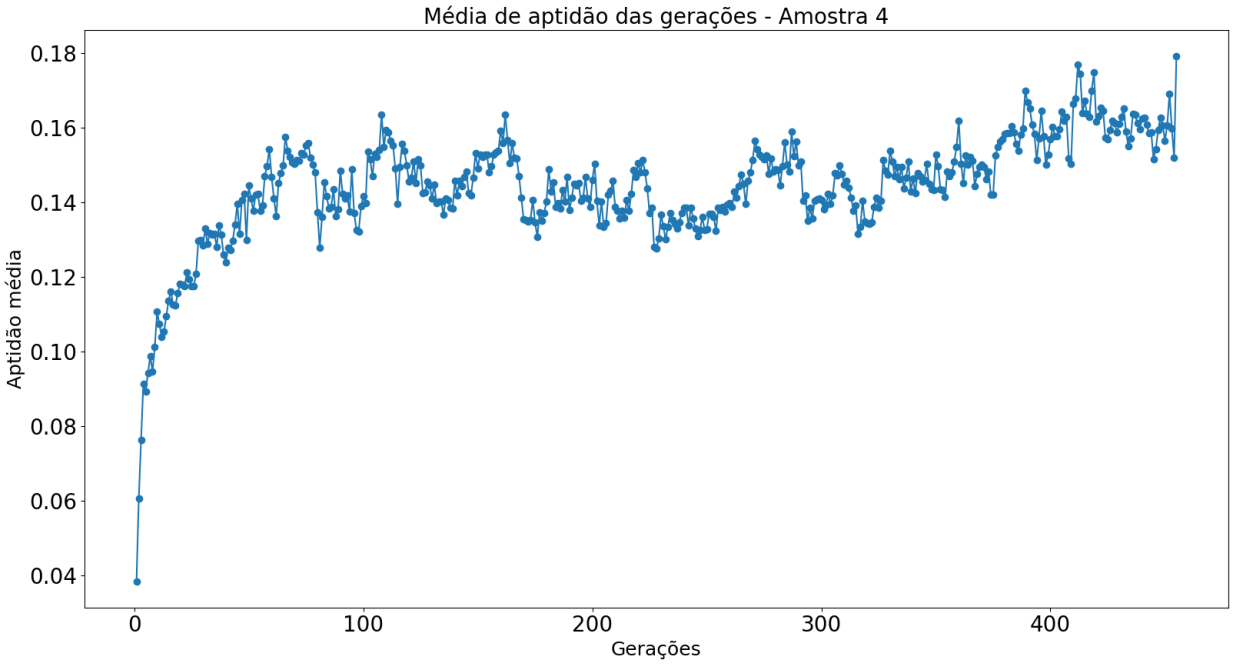
| PROFESSOR | DIAS | TOTAL DIAS |
|---|---|---|
| GILSON (TGA,MTC,TGS,FSI,EMP) | SEG,TER,QUA,QUI,SEX | 5 |
| FABRICIO (POOI,POO2,ES1,PWEB) | TER,QUA,QUI,SEX | 4 |
| FLAVIUS (ALG,PROG,SAD) | SEG,TER,QUA,QUI,SEX | 5 |
| JOÃO PAULO (LOG,ED,AEX) | SEG,QUA,QUI | 3 |
| LUIZ PAULO (II,ARQ) | QUA,QUI,SEX | 3 |
| RICARDO (CEC) | SEG,TER,SEX | 3 |
| TACIANO (BD,ES2,PABD) | SEG,TER,QUI,SEX | 4 |
| LUCIANO (FMAT,PE,MATF) | TER,QUA,SEX | 3 |
| KARLIANE (OSM,GPS) | SEG,QUA,QUI | 3 |
| JOÃO BORGES (SO,RED,SEG) | SEG,QUA,QUI,SEX | 4 |
| JOSÉ ENÉAS (TSI) | QUA,QUI,SEX | 3 |
| DÉSIO (CAL) | TER,QUI,SEX | 3 |
| DELSON (AL) | SEG,TER,SEX | 3 |
| CÉLIA (LPT) | SEG,QUI,SEX | 3 |
| CARLOS (ETIC) | SEG,TER,QUA | 3 |
| TIAGO (PV) | TER,QUI,SEX | 3 |
| TALITHA (FIL) | SEG,QUI,SEX | 3 |
| LEOMARQUES (ING) | TER,QUA,QUI | 3 |
| VYRNA (DIR) | QUA,QUI,SEX | 3 |

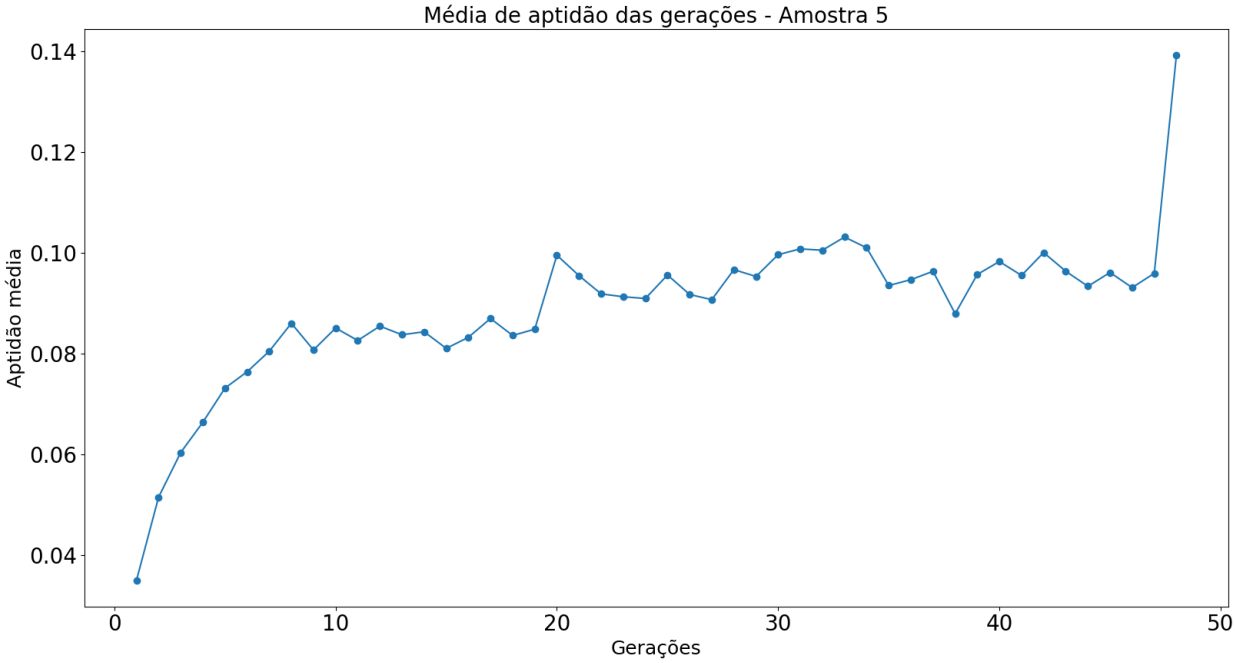
# APÊNDICE B - Gráficos com as médias de aptidão das gerações

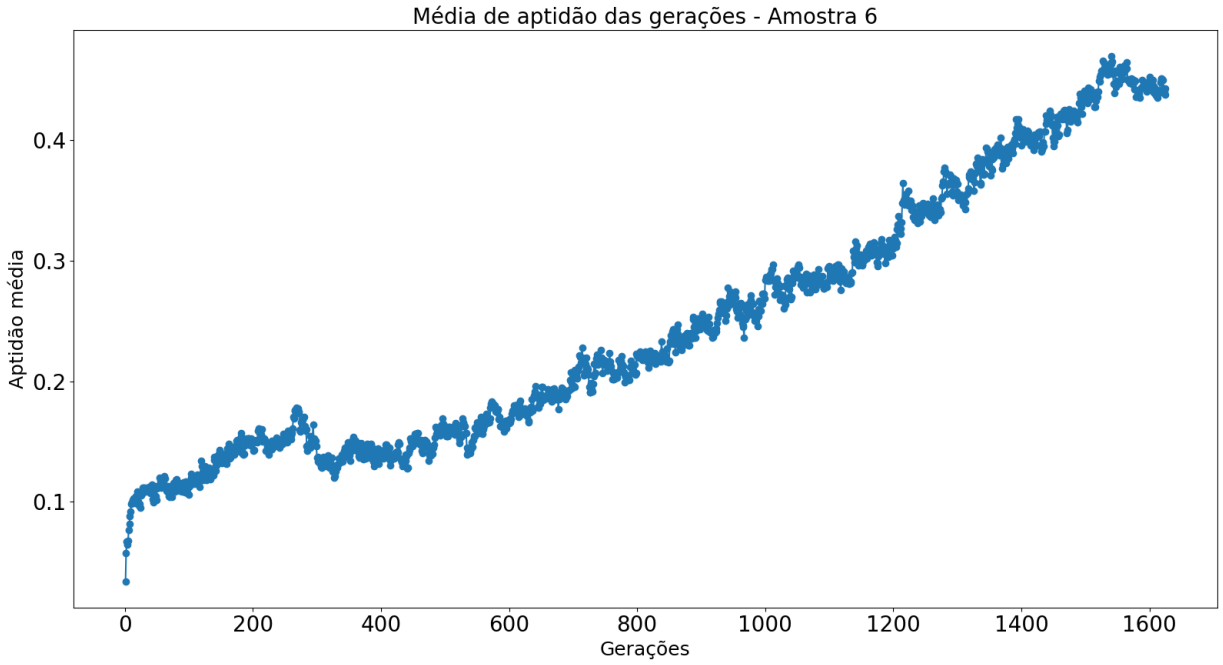

Média de aptidão das gerações - Amostra 1
Aptidão média
0.04
0.06
0.08
0.10
0.12
0.14
0.16
0.18
0
200
400
600
800
Gerações

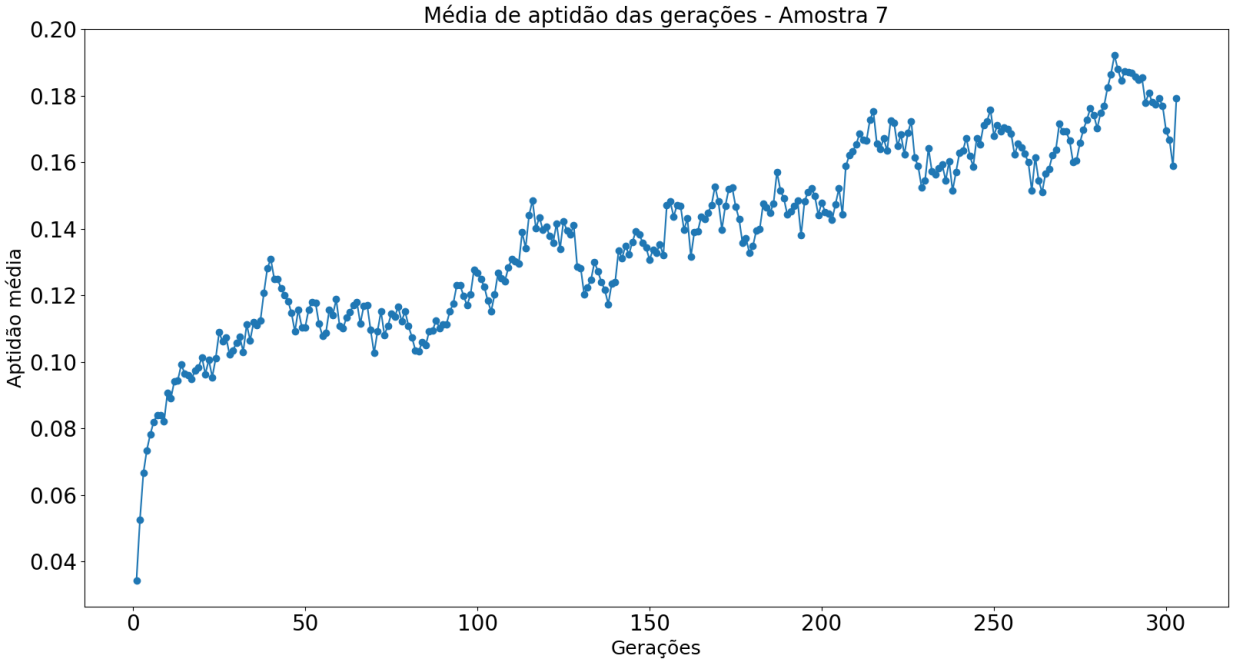

Média de aptidão das gerações - Amostra 2
Aptidão média
0.04
0.06
0.08
0.10
0.12
0.14
0.16
0.18
0
250
500
750
1000
1250
1500
1750
Gerações

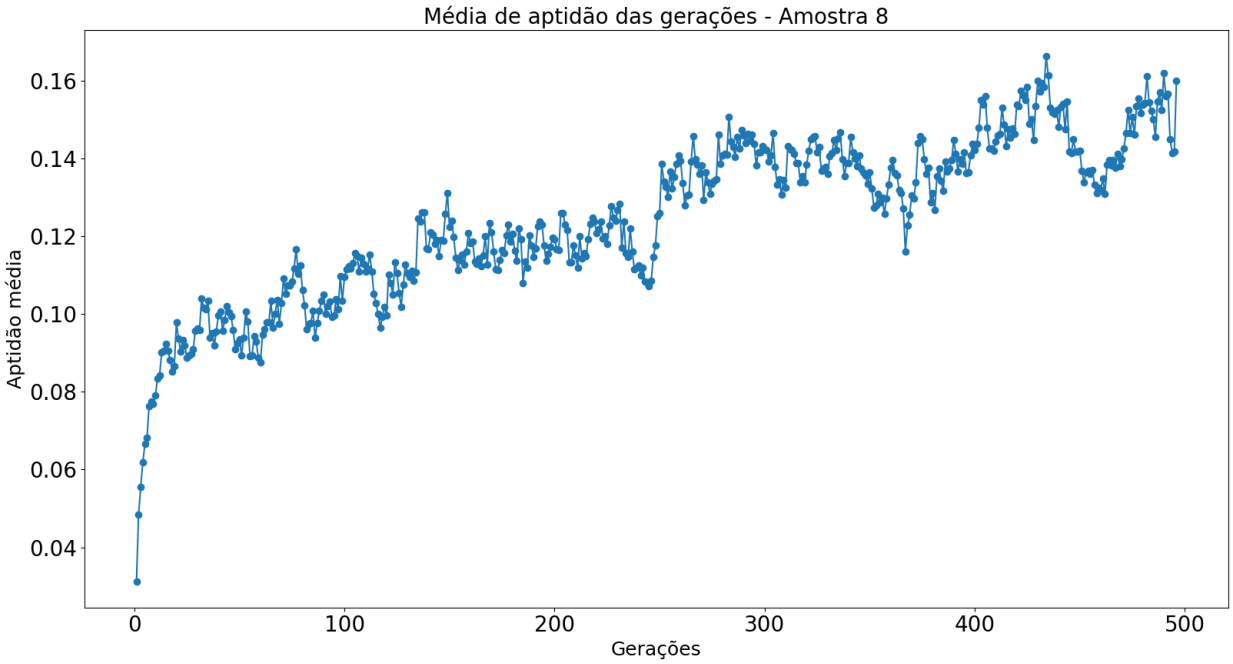

Média de aptidão das gerações - Amostra 3
Aptidão média
0.225
0.200
0.175
0.150
0.125
0.100
0.075
0.050
0
50
100
150
200
250
300
Gerações

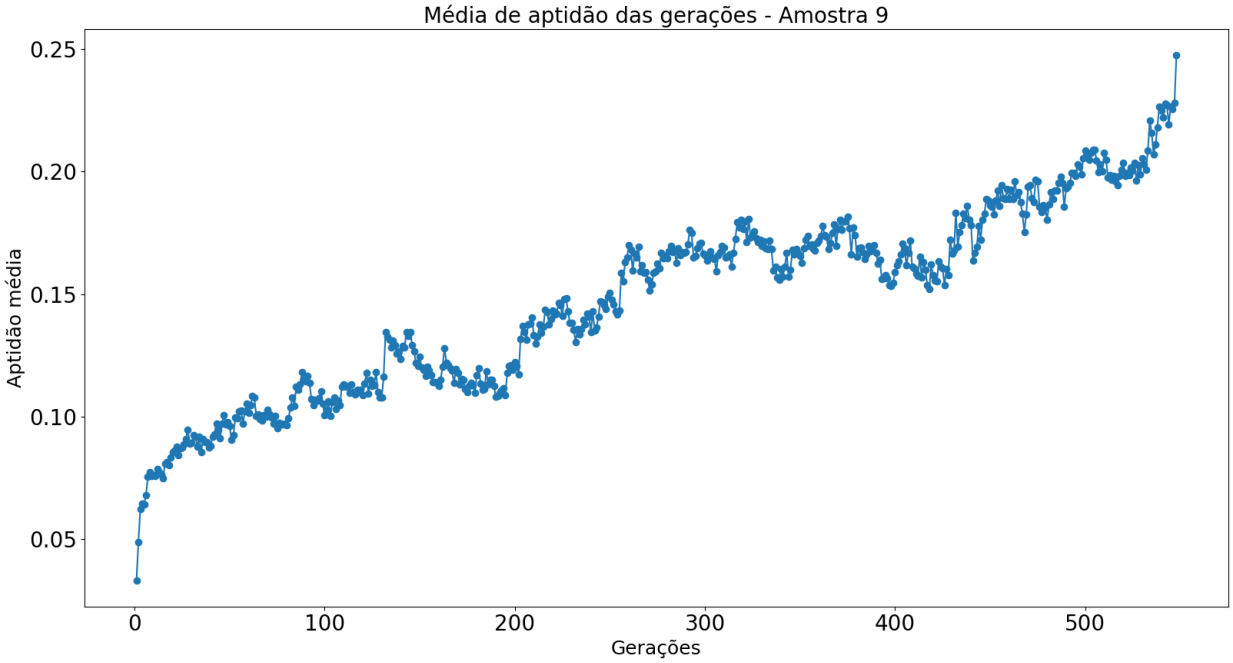

Média de aptidão das gerações - Amostra 4
Aptidão média
0.18
0.16
0.14
0.12
0.10
0.08
0.06
0.04
0
100
200
300
400
Gerações

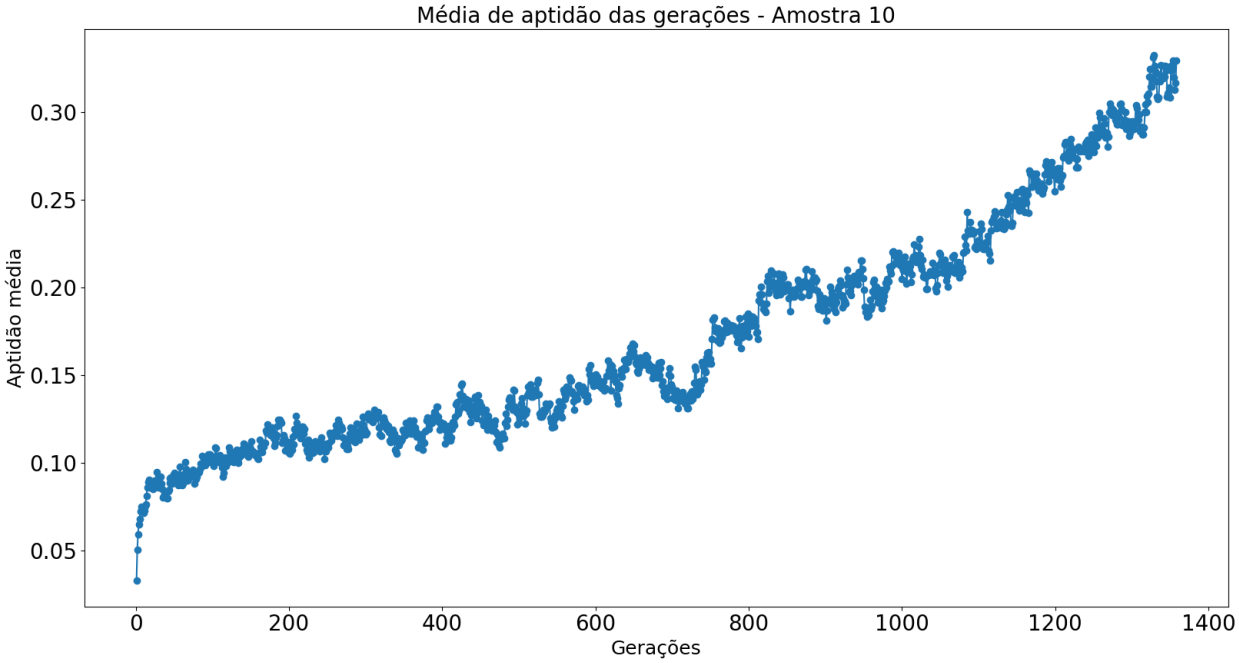

Média de aptidão das gerações - Amostra 5
Aptidão média
Gerações
0.04
0.06
0.08
0.10
0.12
0.14
0
10
20
30
40
50


Média de aptidão das gerações - Amostra 6
Aptidão média
Gerações
0.1
0.2
0.3
0.4
0
200
400
600
800
1000
1200
1400
1600


Média de aptidão das gerações - Amostra 7
Aptidão média
Gerações
0.20
0.18
0.16
0.14
0.12
0.10
0.08
0.06
0.04
0
50
100
150
200
250
300


Média de aptidão das gerações - Amostra 8
Aptidão média
Gerações
0.16
0.14
0.12
0.10
0.08
0.06
0.04
0
100
200
300
400
500


Média de aptidão das gerações - Amostra 9
Aptidão média
0.25
0.20
0.15
0.10
0.05
0
100
200
300
400
500
Gerações


Média de aptidão das gerações - Amostra 10
Aptidão média
0.30
0.25
0.20
0.15
0.10
0.05
0
200
400
600
800
1000
1200
1400
Gerações

APÊNDICE C - Grade de horários da amostra 1

| | AULAS | | | |
|---|---|---|---|---|
| | 1-2 | 3-4 | 5-6 | |
| SEG | ALG | LOG | II | |
| TER | ALG | II | null | 1 |
| QUA | ALG | TGA | FMAT | |
| QUI | null | null | null | |
| SEX | TGA | FMAT | LOG | |

| | AULAS | | | |
|---|---|---|---|---|
| | 1-2 | 3-4 | 5-6 | |
| SEG | LPT | PROG | null | |
| TER | MTC | PROG | null | 2 |
| QUA | TGS | CAL | PROG | |
| QUI | CAL | TGS | null | |
| SEX | LPT | MTC | null | |

| | AULAS | | | |
|---|---|---|---|---|
| | 1-2 | 3-4 | 5-6 | |
| SEG | AL | OSM | ED | |
| TER | POOI | null | null | 3 |
| QUA | ED | AL | FSI | |
| QUI | FSI | null | null | |
| SEX | POOI | ED | OSM | |

| | AULAS | | | |
|---|---|---|---|---|
| | 1-2 | 3-4 | 5-6 | |
| SEG | ARQ | POO2 | null | |
| TER | ARQ | POO2 | ING | 4 |
| QUA | PE | ING | BD | |
| QUI | ES1 | BD | null | |
| SEX | PE | ES1 | null | |

| | AULAS | | | |
|---|---|---|---|---|
| | 1-2 | 3-4 | 5-6 | |
| **SEG** | PWEB | EMP | null | |
| **TER** | PABD | EMP | null | **5** |
| **QUA** | ES2 | null | null | |
| **QUI** | SO | PWEB | null | |
| **SEX** | PABD | ES2 | SO | |

| | AULAS | | | |
|---|---|---|---|---|
| | 1-2 | 3-4 | 5-6 | |
| **SEG** | TSI | FIL | null | |
| **TER** | SEG | RED | CEC | **6** |
| **QUA** | CEC | SEG | TSI | |
| **QUI** | GPS | RED | null | |
| **SEX** | FIL | GPS | null | |

| | AULAS | | | |
|---|---|---|---|---|
| | 1-2 | 3-4 | 5-6 | |
| **SEG** | DIR | PV | null | |
| **TER** | MATF | null | null | **7** |
| **QUA** | PV | MATF | null | |
| **QUI** | DIR | null | null | |
| **SEX** | null | null | null | |

| | AULAS | | | |
|---|---|---|---|---|
| | 1-2 | 3-4 | 5-6 | |
| **SEG** | AEX | null | null | |
| **TER** | ETIC | null | null | **8** |
| **QUA** | ETIC | null | null | |
| **QUI** | SAD | null | null | |
| **SEX** | AEX | SAD | null | |

APÊNDICE D - Conjunto de dados de entrada de disciplinas e professores

| Disciplina | Período | Professor | Quantidade de aulas por semana |
|---|---|---|---|
| ALG | 1 | FLAVIUS | 3 |
| FMAT | 1 | LUCIANO | 2 |
| II | 1 | LUIZ PAULO | 2 |
| LOG | 1 | JOÃO PAULO | 2 |
| TGA | 1 | GILSON | 2 |
| CAL | 2 | DÉSIO | 2 |
| LPT | 2 | CÉLIA | 2 |
| MTC | 2 | GILSON | 2 |
| PROG | 2 | FLAVIUS | 3 |
| TGS | 2 | GILSON | 2 |
| AL | 3 | DELSON | 2 |
| ED | 3 | JOÃO PAULO | 3 |
| FSI | 3 | GILSON | 2 |
| OSM | 3 | KARLIANE | 2 |
| POOI | 3 | FABRÍCIO | 2 |
| ARQ | 4 | LUIZ PAULO | 2 |
| BD | 4 | TACIANO | 2 |
| ES1 | 4 | FABRÍCIO | 2 |
| ING | 4 | LEOMARQUES | 2 |
| PE | 4 | LUCIANO | 2 |
| POO2 | 4 | FABRÍCIO | 2 |
| EMP | 5 | GILSON | 2 |
| ES2 | 5 | TACIANO | 2 |
| PWEB | 5 | FABRÍCIO | 2 |
| PABD | 5 | TACIANO | 2 |
| SO | 5 | JOÃO BORGES | 2 |
| CEC | 6 | RICARDO | 2 |
| FIL | 6 | TALITHA | 2 |
| GPS | 6 | KARLIANE | 2 |
| RED | 6 | JOÃO BORGES | 2 |
| TSI | 6 | JOSÉ ENÉAS | 2 |
| SEG | 6 | JOÃO BORGES | 2 |
| DIR | 7 | VYRNA | 2 |
| MATF | 7 | LUCIANO | 2 |
| PV | 7 | TIAGO | 2 |
| AEX | 8 | JOÃO PAULO | 2 |
| ETIC | 8 | CARLOS | 2 |
| SAD | 8 | FLAVIUS | 2 |