UNIVERSIDADE FEDERAL DO PAMPA

ALEXANDRE BRASIL DA SILVA

Alocação de Recursos na Resolução do Problema da Grade Horária

Bagé
2014

ALEXANDRE BRASIL DA SILVA

Alocação de Recursos na Resolução do Problema da Grade Horária

Trabalho de Conclusão de Curso apresentado ao Curso de Engenharia de Computação da Universidade Federal do Pampa, como requisito parcial para obtenção do Título de Bacharel em Engenharia de Computação.

Orientador: Carlos Michel Betemps

Coorientador: Milton Heinen

Bagé
2014

ALEXANDRE BRASIL DA SILVA

Alocação de Recursos na Resolução do Problema da Grade Horária

Trabalho de Conclusão de Curso apresentado ao Curso de Engenharia de Computação da Universidade Federal do Pampa, como requisito parcial para obtenção do Título de Bacharel em Engenharia de Computação.

Trabalho de Conclusão de Curso defendido e aprovado em: , agosto de .

Banca examinadora:

_____

Prof. MSc. Carlos Michel Betemps
Orientador
UNIPAMPA

_____

Prof. MSc. Sandra Dutra Piovesan
UNIPAMPA

_____

Prof. MSc. Gerson Alberto Leiria Nunes
UNIPAMPA

RESUMO

O problema da confecção da grade horária é recorrente na maioria das instituições de ensino. Ele ocorre sempre no início de seus períodos letivos e é necessário grande esforço para a construção de uma solução para o mesmo. Encontrar esta resposta pode ser uma tarefa trabalhosa, que muitas vezes demanda uma mão de obra grande, e todos os envolvidos neste processo devem estar atentos ao que está sendo feito, pois é uma questão complexa em que muitas vezes uma "boa solução" é difícil de ser encontrada. Este trabalho apresenta a implementação de uma técnica heurística – algoritmos genéticos – implementada em um protótipo de *software* em que foi utilizada linguagem de programação JAVA e banco de dados MySQL. Esta técnica, que baseia-se na teoria da evolução dos seres vivos proposta por Darwin, varre o espaço de busca das soluções de forma aleatório guiada, fazendo com que as soluções mais aptas sejam "descobertas" dentro do conjunto de soluções possíveis para este problema. Em uma boa resposta para este problema, os recursos concernentes à grade horária devem ser combinados e alocados de maneira eficiente e que ofereçam possibilidade de aplicação real na instituição de ensino. Testes realizados com o protótipo mostraram que a técnica é capaz de resolver problemas de alocação e combinação de recursos, trazendo bons resultados para o escopo deste trabalho.

Palavras-chaves: Grade horária, algoritmos genéticos, alocação de recursos.

ABSTRACT

The problem of preparing the timetable is recurrent in most educational institutions. It always occurs at the beginning of their academic periods and great effort is needed to build a solution for the same. Find this response can be a laborious task that often requires a great handwork, and everyone involved in this process must be attentive to what is being done, it is a complex issue that often a "good solution" is difficult to be found. This work presents the implementation of a heuristic technique, genetic algorithms, in a prototype software that was implemented with the JAVA programming language and MySQL database This technique, which is based on the theory of evolution proposed by Darwin, scans the search space of solutions in random way guided, making the fittest solutions discovered within the set of possible solutions to this problem. In a good answer to this problem, the concerning timetable resources should be combined and allocated efficiently and offer the possibility of real application in educational institutions. Tests conducted with the prototype showed that the technique is capable of solving problems of allocation and combination of resources, bringing good results for the scope of this work.

Keywords: Grid time, genetic algorithms, resource allocation.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

AG – Algoritmo Genético

COEF – Comissão de Organização do Espaço Físico

DAO – Data Access Object

RAM – Random Access Memory

Gb – Gygabytes

SQL – Structred Query Language

IFTM – Instituto Federal do Triângulo Mineiro

EC – Engenharia de Computação

EERA – Engenharia de Energias Renováveis e Ambiente

SGBD – Sistema Gerenciador de Banco de Dados

HTML – HyperText Markup Language

SUMÁRIO

1 INTRODUÇÃO

No mundo real, muitas atividades são dependentes de recursos limitados. Dessa forma, a alocação correta e eficiente destes recursos finitos é imprescindível para que as respostas destas atividades possam ser encontradas. Alocar uma frota de ônibus em escalas de linhas e horários [11], alocar policiais, médicos ou enfermeiros em escalas de serviço, realizar o correto roteamento de veículos em sistemas de transporte e logística [12], gerenciar recursos computacionais conforme restrições/prioridades impostas pelo sistema operacional [16] são questões que dependem exclusivamente da quantidade de recursos disponíveis e de como e onde serão aplicados.

No transporte público, o problema da alocação de veículos em linhas e horários oferece pontos bem complexos e que devem ser levados em consideração no momento de serem estudados. Itens como número de passageiros, duração média de cada viagem e tabela de horário a ser cumprida são alguns recursos que devem ser aplicados da melhor maneira possível, levando-se em consideração sempre a capacidade dos veículos, quantidade máxima de horas trabalhadas e período de repouso obrigatório tanto dos motoristas quanto dos carros. Esforço semelhante é aplicado na constituição de escalas de serviço de pessoal (médicos, policiais, etc) em que os recursos disponíveis (mão de obra) devem ser aplicados de forma a compor os quadros horários de trabalho, fazendo com que determinados indivíduos não fiquem sobrecarregados com mais horas trabalhadas do que outros; da mesma forma ocorre no roteamento de veículos em que é preciso escolher trechos de rotas, que podem muitas vezes não ser os mais curtos, para aplicá-los no caminho que deverá ser percorrido pelo veículo.

Quando se tem poucos recursos para serem aplicados em poucos locais, isto não chega a ser um problema complexo, pois visto o número de combinações possíveis as mesmas podem ser feitas e enumeradas, escolhendo-se ao final a solução que melhor se encaixa ao problema. Porém sua complexidade pode aumentar, conforme se aumentam os recursos e os locais onde os mesmos serão aplicados, isso acaba tornando impraticável a ação da verificação manual das soluções, pois o número de combinações pode crescer de forma fatorial em relação aos recursos disponíveis. Geralmente isto ocorre nos problemas de otimização combinatória, onde o elevado número de recursos e combinações possíveis entre os mesmos podem causar uma “explosão” de soluções aptas.

Em um contexto educacional, as instituições de ensino necessitam confeccionar sua grade de horários antes do início de seus períodos letivos, realizando uma alocação de professores em disciplinas, horários e salas. Este problema é conhecido como "problema de geração da grade horária".

Recursos como quantidade de professores com suas respectivas restrições de horário e afinidade com disciplinas, quantidade de salas disponíveis, disciplinas que necessitam de laboratórios de uso específico, dentre outras, devem ser levadas em consideração para uma alocação eficiente. Dentro desta realidade, podem ser definidas, por exemplo, algumas das seguintes combinações na aplicação destes recursos: para qual disciplina um professor será alocado, em qual sala esta turma será alocada, qual período do dia letivo será destinado para determinada turma. Tudo isto respeitando regras simples como não alocar um professor, concomitantemente, em mais de uma sala ou não alocar mais de uma turma na mesma sala, no mesmo período em que possuam disciplinas diferentes. Em suma, a geração da grade horária precisa atender as mais variadas restrições relativas a este processo, levando sempre em consideração as variáveis como professores, disciplinas, salas e horários.

O número de combinações possíveis que podem ocorrer no momento da confecção de uma grade horária é diretamente proporcional ao fatorial do número de turmas/salas disponíveis [13]. Impondo as restrições do parágrafo anterior, o problema pode se tornar bem mais complexo, visto que será necessário filtrar e descartar as grades que não satisfazem as condições impostas.

Partindo do pressuposto que uma instituição possui jornadas educacionais de 4 períodos por turno de dia trabalhado, durante 5 dias na semana, totalizam-se 20 períodos semanais para uma única turma. Esse número pode gerar um espaço de busca igual (4 x 5)!, ou seja, $2{,}43 \times 10^{18}$ combinações possíveis de serem efetuadas nesta grade horária.

Como apresentado em [6], a formulação matemática que mostra o tamanho do espaço de pesquisa de um problema de elaboração de horários de aulas é expresso pela equação 1.

$$TEP = (NDS \times NAD \times NSA)^{(ND \times NAP)} \qquad \text{Equação (1)}$$

onde:

TEP é o tamanho do espaço de pesquisa;

NDS é o número de aulas diárias para o curso envolvido na distribuição das aulas consideradas;

NSA é o número de salas de aulas disponíveis para utilização das aulas;

ND é o número de disciplinas pertencentes ao horário de aula;

NAP é o número de aulas a serem distribuídas para cada uma das disciplinas.

Sendo que, para a geração de uma grade horária que contemple mais de um curso simultaneamente, esta equação deve ser aplicada a cada um dos cursos em questão, e a resposta final sobre o tamanho do espaço de busca/pesquisa total, é o somatório dos espaços de busca/pesquisa de cada um dos cursos envolvidos.

Geralmente, o desenvolvimento de grade horária dentro das instituições de ensino pode ocorrer de duas maneiras: manualmente ou empiricamente [14]. O processo manual implicaria em colocar parte da mão de obra disponível da instituição de ensino para trabalhar neste problema. Esta tarefa pode demorar dias até que seja encontrado um resultado considerado "bom". Porém, se erros forem cometidos no início desta atividade, teremos, ao final, uma solução ruim e todo o resultado poderá ser invalidado, sendo, desta forma, necessário o recomeço da tarefa, e ocasionando a perda de dias de trabalho. Por outro lado, o processo empírico significaria contar com a experiência obtida nos anos/semestres anteriores e aproveitar o máximo possível dos horários já alocados nos períodos de semestres/anos passados, modificando pouca coisa e aplicando-os nas grades correntes.

Solucionando o problema desta última forma, pode-se ter o malefício de uma solução "engessada" ao longo dos anos, que no momento em que for preciso aplicar alterações severas na grade utilizada, será necessário o retrabalho desde o seu início, ocasionado a confecção de uma solução totalmente nova para o problema em questão.

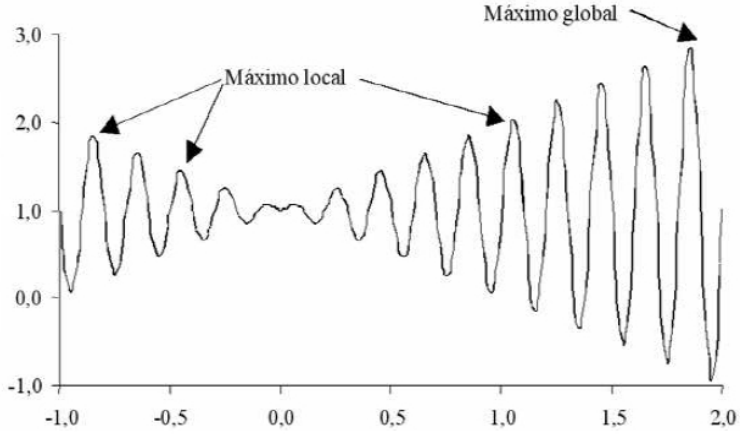
Para resolução deste problema, assim como os demais problemas apresentados anteriormente, afirma-se que pode não existir uma única solução correta, mas sim um conjunto amplo de soluções dentro do seu espaço de busca que igualmente podem ser utilizadas como resposta. Pode, de certa forma, haver diferenças entre elas, mesmo assim todas podem solucioná-lo de forma satisfatória. Deste modo, é esperado a aplicação de um método que possa filtrar/priorizar as melhores respostas para o problema, ou respostas que consigam chegar bem perto de uma solução ótima.

Assim como [11] comenta, é importante lembrar os conceitos de máximos locais e de máximos globais, pois são essenciais em problemas de busca que envolvem otimização e combinação de recursos. Um máximo global realmente representa uma solução qualificada em relação a todo o conjunto de soluções possíveis para o problema, enquanto o máximo local é uma boa solução perante as soluções vizinhas, mas que não aborda todo o espaço de

soluções possíveis. Muitas vezes a busca por um máximo global pode ser inviável, quando o problema apresenta muitas variáveis/recursos. Um exemplo de máximos locais e máximo global é observado na figura 1.

Em tese, este problema da grade horária é um dos clássicos problemas computacionais, cuja sua solução acaba por determinar uma sequência de encontros entre alunos, docentes e a combinação entre os mesmos, salas e disciplinas. Ele é classificado como um problema do tipo NP-difícil, pois é mais fácil verificar a qualidade de uma solução apresentada com base no resultado que é procurado, do que obter analiticamente tal solução em tempo polinomial[10].

Figura 1 - Exemplo de pontos de máximos locais e máximo global



Fonte: Ferreira; Corrêa (2010)

1.1 **Objetivos**

1.1.1 **Objetivo Geral:**

Abordar o tema proposto da geração de grade horária para uma instituição de ensino. Por conseguinte, desenvolver um protótipo de *software* que consiga resolver de forma satisfatória o referido problema.

1.1.2 **Objetivos Específicos:**

Para que o objetivo geral deste trabalho pudesse ser alcançado, foram propostos uma série de objetivos específicos, elencados a seguir:

– Estudar as diversas técnicas utilizadas em problemas de alocação de recursos e otimização combinatória;

– Estudar grades de horários de semestres passados do campus Bagé da Universidade Federal do Pampa;

– Analisar a grade curricular dos cursos de graduação da Universidade Federal do Pampa – Campus Bagé, bem como os professores de cada área;

– Mapear as restrições à serem impostas no momento da geração da grade horária;

– Definir uma linguagem de programação e um sistema gerenciador de banco de dados para a implementação do protótipo;

– Realizar a identificação e a implementação das técnicas que busquem a solução mais adequada para o problema;

– Desenvolver um protótipo que utilize a(s) solução(ões) escolhida(s);

– Realizar um estudo de caso nos dados da grade horária dos cursos de graduação da Universidade Federal do Pampa – Campus Bagé.

1.2 **Justificativa**

Sabendo-se que os problemas de alocação e combinação de recursos demandam, na maioria das vezes, grande poder computacional e um tempo elevado de processamento, visto a grande quantidade de combinações que podem existir entre os recursos e os locais onde os mesmos deverão ser alocados, a implementação de uma possível solução, em particular para o problema da grade horária, se faz necessário e é de relevante importância no âmbito das

instituições de ensino.

Observando-se que a Unipampa, campus Bagé, atualmente utiliza a forma empírica de confecção da grade horária, algo custoso, demorado e que, dependendo das mudanças a serem realizadas de um semestre para o outro, pode tornar-se em uma forma manual de confecção. O presente trabalho buscou a automatização deste processo.

1.3 **Metodologia**

Inicialmente, foi realizada uma pesquisa e levantamento de literatura para verificação do que já foi produzido e implementado em relação a problemas de grade horária no âmbito acadêmico.

Foi necessário realizar o levantamento das restrições que possuem necessidade de serem impostas na grade horária, e aplicá-las ao domínio do problema de grade horária em questão. Esse levantamento ocorreu em primeiro plano com a coordenação do curso de Engenharia de Computação do campus Bagé. Após, foi expandido aos outros cursos deste campus, para que soluções interligadas entre as grades de diferentes cursos fossem geradas.

A posteriori, o problema em si foi modelado buscando estruturar os requisitos levantados, de forma que ajudem a explicar as características do protótipo, suas funcionalidades, seu comportamento e a melhor forma de implementação. Os dados e restrições foram relacionados entre si, para que a auxiliassem na correta construção do projeto.

Em seguida, a implementação do protótipo ocorreu em linguagem de programação que atendeu satisfatoriamente as demandas e requisitos elencados para o protótipo, que fazem parte do escopo do problema. O banco de dados escolhido tem licença de utilização livre, com suporte à linguagem SQL e ferramentas que auxiliam o seu gerenciamento de forma fácil e simplificada.

Escolhidos a linguagem de programação e o sistema gerenciador de banco de dados, foram analisados os dados disponibilizados pela Unipampa – Campus Bagé – como: dados sobre a grade curricular dos cursos de graduação, quadro docente, turmas dos cursos e estrutura física disponível, como salas de aulas e laboratórios de uso específicos utilizados nas disciplinas.

Após o protótipo, estar funcional, foram realizados testes, trazendo-os para o plano

do problema em questão, que é a distribuição dos horários dos cursos de graduação, do Campus Bagé, de forma satisfatória e funcional. Os resultados obtidos foram avaliados tanto em desempenho como em acertos nas grades. Foram levadas em consideração o número de colisões de horário que ocorreram, número de restrições quebradas e os tipos das restrições que não foram atendidas.

1.4 **Estrutura do trabalho**

A estrutura do trabalho está dividida entre os seguintes capítulos:

O capítulo 1, no qual é apresentada a introdução, com uma contextualização do problema, os objetivos gerais do trabalho, uma justificativa, e uma breve explanação da metodologia do trabalho.

No capítulo 2, são mostrados os conceitos gerais e a revisão da literatura, a descrição da técnica de busca aplicada ao contexto do problema, suas estruturas e seus modelos e uma breve explicação do desenvolvimento.

A apresentação e a descrição do desenvolvimento do trabalho são parte do capítulo 3, que contém os trabalhos correlatos, levantamento das restrições, coleta dos dados e informações relativos ao problema, modelagem dos dados e das estruturas, implementação do protótipo de *software* na linguagem de programação escolhida e a implementação da técnica escolhida.

No capítulo 4 é mostrado o protótipo, os dados experimentais e os resultados, e no capítulo 5 a conclusão do trabalho e os trabalhos futuros.

2 CONCEITOS GERAIS E REVISÃO DA LITERATURA

Segundo [13], todos os semestres, instituições de ensino precisam definir os horários das disciplinas de acordo com a disponibilidade de seus professores. Ele afirma ainda que podem haver outras restrições, como turmas que possuem alunos em comum, preferência para que alunos estudem sempre no mesmo bloco, etc.

Para [2], esse conjunto de restrições podem variar de instituição para instituição, sendo observadas sempre as características do ambiente e as características das pessoas envolvidas neste processo; que geralmente são os coordenadores de curso. Ele também afirma, que na maioria das vezes, a montagem do horário é feita de forma manual ou empírica, conforme já visto, e que esta tarefa pode necessitar de um tempo consideravelmente alto até que seja finalizada, sendo extremamente suscetível a erros.

De acordo com [9], a satisfação de todos os requisitos desejáveis na confecção de uma grade horária pode ser, e geralmente é, impossível de se realizar na prática. Até mesmo porque, muitas vezes, vários pedidos de professores podem ser conflitantes, acabando por tornar a grade não compacta, trazendo períodos livres durante a semana, ou até mesmo, aulas dispersas para alguns dos docentes. Ele ainda afirma que estes requisitos devem ser divididos em diferentes prioridades (importantes e não importantes).

Seguindo este mesmo raciocínio, [13] e [5] afirmam que as restrições aplicadas ao problema podem ser classificadas em dois tipos de restrições: *hard constraint* e *soft constraint*. Uma *hard constraint* é uma restrição que não pode ser quebrada, pois inviabilizaria a resposta final, como a alocação de um professor para ministrar aulas em duas salas concomitantemente. Algo impossível de ser satisfeito. Por outro lado, uma *soft constraint* seria uma restrição desejável de ser aplicada, porém se a mesma não puder ser atendida, não torna a resposta da solução incorreta. Isto seria a questão de alocar as aulas de alunos de um determinado semestre num mesmo bloco de um prédio, para conforto dos mesmos, ou de um professor que preferencialmente quer ter suas disciplinas agrupadas em dois dias da semana.

Por vezes, devido ao alto número de recursos/restrições, particularmente no problema de grade horária, são utilizadas técnicas baseadas em algoritmos heurísticos construtivos, que consistem basicamente em alocar aula a aula, disciplina a disciplina, docente a docente, até que o quadro horário seja confeccionado por completo. Utilizando uma implementação deste

tipo, significa que se deve priorizar a alocação de casos mais restritivos, para que, posteriormente, possam ser alocados os casos com menos restrições e que terão uma prioridade menor. Esta implementação tem alto impacto na codificação do algoritmo no momento em que as restrições necessitarem que suas prioridades sejam modificadas [14].

Ainda segundo [14], apesar da aparente facilidade de implementação do método descrito acima, o crescente aumento/variação de recursos, de locais disponíveis para alocação e a variação de restrições sobre os recursos existentes, tornam a sua utilização pouco eficiente. Deste modo, diversas técnicas têm sido propostas e utilizadas neste problema, como, por exemplo, a de programação inteira, problema de fluxo de redes ou sendo utilizadas técnicas com coloração de grafos.

Ultimamente, a inteligência artificial, através de suas técnicas dentro de espaços de busca, tem se mostrado uma forte ferramenta na resolução de problemas de otimização NP-difíceis na alocação de recursos, e estão sendo largamente empregados neste contexto. Dentre estes podem ser enumerados a busca tabu, as estratégias evolucionárias, os algoritmos genéticos, algoritmos meméticos, o resfriamento simulado, GRASP, entre outras [14].

Segundo [18], a busca tabu é uma heurística baseada e originada nos anos 70. Ela é uma técnica que guia a busca de modo que se possa permitir que a mesma consiga escapar de máximos locais. Durante sua execução, é mantida uma lista com os melhores indivíduos encontrados. Desta forma, mantêm-se o espaço de busca otimizado. Após a aplicação de uma heurística é verificado se o indivíduo encontrado é uma solução boa, em caso negativo, ele é removido da lista, quando o mesmo atinge seu tempo de expiração (prazo tabu), este pode ser removido da lista. O problema apesentado está na calibração do tamanho da lista, do tempo de aspiração dos indivíduos, e de, apresar de apresentar tentativas de escapar dos máximos locais, não raro esta técnica acaba por não produzir soluções ideais. Para [11], ela é muito sensível aos seus parâmetros de entrada.

O resfriamento simulado, ainda segundo [18], é uma técnica que tenta simular o processo de *annealing* dos cristais. Esta técnica surgiu da observação do resfriamento gradativo de certos materiais, que, apartir de uma alta temperatura inicial leva o material ao estado mínimo de energia. Esses estados são caracterizados por uma perfeição estrutural caso o resfriamento não tivesse sido gradativo. Na simulação, a temperatura fixa aquece o material e faz com que suas partículas (recursos que devem ser alocados) sejam agitadas e gradativamente as mesmas começam a ocupar seus locais no indivíduo. A verificação da

temperatura de aquecimento e o quanto a temperatura deve diminuir a cada iteração são elementos difíceis de serem parametrizados, e para que um máximo global consiga ser encontrado, depende muito de como é a qualidade do primeiro indivíduo da execução (que é codificado de forma aleatória) sendo que muitas vezes esta técnica pode ficar "presa" em um máximo local por causa do seu primeiro indivíduo.

Para [17], a técnica de *Hill Climbing* começa visitando a vizinhança de uma solução inicial e, uma nova solução somente é aceita em substituição à existente, se a mesma apresentar um resultado melhor ou igual à solução atual. O nome se deve a analogia, para um problema de maximização, ao ato de uma pessoa escalar uma montanha, onde a mesma deve somente subir a encosta, e nunca descê-la.

Em seu trabalho, [11] nos mostra o método de GRASP (*greedy randomized adaptative search procedure*), que é uma técnica de busca adaptativa através de um algoritmo guloso. A cada iteração é construída uma solução viável, e em seguida, é aplicado um processo de busca na vizinhança para um possível refinamento da resposta. A primeira solução da técnica é construída de forma totalmente gulosa, com o objetivo de gerar uma solução viável. As melhores soluções a cada iteração passam a fazer parte de uma lista restrita de candidatos. Após isso, é então realizada uma busca local nesta lista para achar então as melhores soluções. O autor nos fala que este método é muito dependente do método de busca local para o aperfeiçoamento da solução.

Sobre os algoritmos meméticos, [5] apresenta como sendo uma técnica baseada nos algoritmos genéticos, onde as unidades básicas da implementação são os memes e, ao contrário dos genes, podem ser modificados/aprimorados durante a vida, utilizando para isso uma função de aprimoramento que é executada com uma frequência pré-fixada.

Seguindo o mesmo pensamento de [5], em que deseja-se anular de alguma forma os máximos locais, e não ter uma resposta encontrada em um tempo exponencial em relação ao tamanho da entrada do problema, escolheu-se os algoritmos genéticos por uma série de vantagens listadas a seguir:

– podem resolver problemas complexos de forma confiável;

– a construção de um AG e de seus modelos é algo geralmente simples de ser implementado;

– são extensíveis à outros problemas;

– existe uma certa facilidade de combinação com outros métodos de busca, principalmente no que se refere a questão de otimização dos seus parâmetros de execução;

– existe a plena possibilidade de implementação que tire vantagem de arquiteturas paralelas.

2.1 **Algoritmos Genéticos**

A história dos algoritmos genéticos começa na década de 1940, quando os cientistas começavam a se inspirar na natureza para criar este ramo da inteligência artificial. As pesquisas se desenvolveram mais nas áreas das questões cognitivas e na compreensão dos processos de raciocínio e aprendizado até o final dos anos 50. A partir daí, os pesquisadores começaram também a voltar seus esforços em busca de modelos de sistemas genéricos que pudessem gerar soluções candidatas para problemas que eram difíceis demais para serem resolvidos computacionalmente [13].

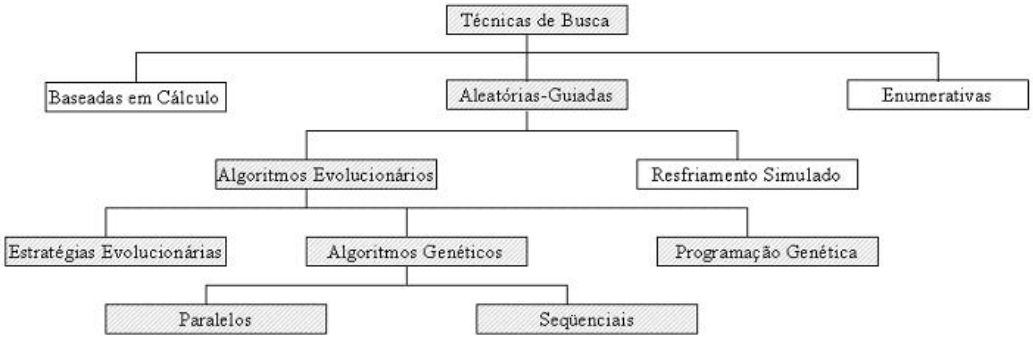
Ainda segundo [13], em 1957, George Box apresentou seu primeiro esquema de associação de problemas de otimização com a teoria da evolução natural das espécies, proposta por Darwin em 1859.

Uma outra tentativa de utilizar processos evolutivos para resolução de problemas foi feita por Rechenberg em 1965, quando ele desenvolveu as estratégias evolucionárias. Sua pesquisa tinha ampla fundamentação teórica e foi aplicada com sucesso a vários problemas práticos. Mesmo não possuindo conceitos amplamente aceitos atualmente, como cruzamento entre indivíduos e populações maiores, o trabalho de Rechenberg pode ser considerado pioneiro, por ter introduzido as técnicas evolucionárias ao tratamento de problemas do cotidiano [13].

No final da década de 1960, outro pesquisador, com o nome de John Holland, trabalhou nas falhas e aprimorou o que havia sido pesquisado por Rechenberg. Contendo operações que compunham itens como necessários à sobrevivência das espécies, sendo estas o cruzamento, a mutação, a seleção natural, as grandes populações, entre outras, Holland pode ser considerado como o “inventor” dos algoritmos genéticos [13].

> Holland estudou formalmente a evolução das espécies e propôs um modelo heurístico computacional que quando implementado poderia oferecer boas soluções para problemas extremamente difíceis que eram insolúveis computacionalmente até aquela época. (LINDEN, 2012, p. 38).

Figura 2 - Diagrama de posicionamento dos algoritmos genéticos dentro das técnicas de busca da inteligência artificial.



Fonte: Linden (2012)

Dentre os métodos enumerados na figura 2, destacam-se os algoritmos evolucionários. São técnicas de busca aleatório guiadas, que utilizam o estado atual do algoritmo, mais informações aleatórias, para guiá-lo na montagem de como será seu próximo estado futuro na pesquisa pela solução do problema. Isto faz com que parte do caminho até uma das "boas" soluções fosse cortado, e um atalho fosse criado. Estas técnicas baseiam seus modelos computacionais nos processos naturais de evolução como ferramentas para resolução de problemas.

A principal diferença entre as técnicas de busca e a otimização tradicional é que estes algoritmos operam sobre uma série de soluções candidatas que podem ser uma das boas soluções possíveis para o problema, refinando cada vez mais seu conjunto de soluções conforme o tempo passa, enquanto a otimização tradicional tenta construir a melhor reposta, podendo muitas vezes necessitar de um tempo exponencial ou fatorial em relação ao tamanho dos dados de entrada.

Dentro do ramo de estudo dos algoritmos evolucionários, foram estudados para o problema de grade horária os algoritmos genéticos, que são uma técnica que têm se destacado como solução de uma grande gama de problemas devido a sua simplicidade e facilidade de implementação.

2.1.1 **Estruturas dos algoritmos genéticos**

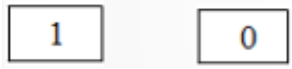
Segundo [13], os algoritmos genéticos são um ramo dos algoritmos evolucionários (subdivisão da área da inteligência artificial) que podem ser definidos como uma técnica

baseada em uma metáfora do processo biológico da evolução natural. Pode-se dizer que eles são uma técnica de otimização global, ou seja, eles não ficam restritos a apenas um determinado local do espaço de soluções, pois tentam varrer de forma aleatório guiada, a maior parte do espaço total de buscas.

Assim como na evolução natural, os algoritmos genéticos possuem diversas etapas e diversos elementos que se assemelham com a teoria da evolução proposta por Darwin, em que os indivíduos mais aptos, acabam tendo maiores probabilidades de se reproduzir. Os algoritmos genéticos compreendem os seguintes elementos: gene, locus, genótipo, fenótipo, indivíduo, população, geração, operadores genéticos (reprodução dos indivíduos), grau de aptidão e os métodos de seleção.

Gene: é a parcela indivisível do indivíduo, em conjunto com outros determinados genes produz determinada característica para o indivíduo (aqui identificados por algarismos binários) – figura 3;

Figura 3 - As parcelas indivisíveis do AG, os genes



Fonte: Acervo próprio

Locus: é a posição que um gene ocupa dentro de um indivíduo.

Genótipo: é a estrutura composta por uma dupla, ou um conjunto de genes.

Fenótipo: é a característica exteriorizada do genótipo, ou seja, o que se pode ver de forma externa.

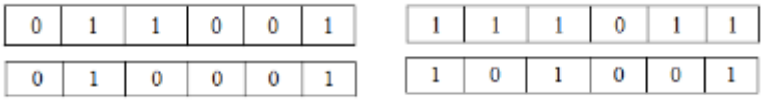
Indivíduo: é formado pelo conjunto de genes, e este conjunto se torna uma das soluções candidatas que podem ser a resposta para o problema trabalhado (exemplificado por um número binário de 6 *bits*) – figura 4;

Figura 4 - O indivíduo composto por um conjunto de genes

Fonte: Acervo próprio

População: é o conjunto de vários indivíduos coexistindo no mesmo período de tempo – figura 5;

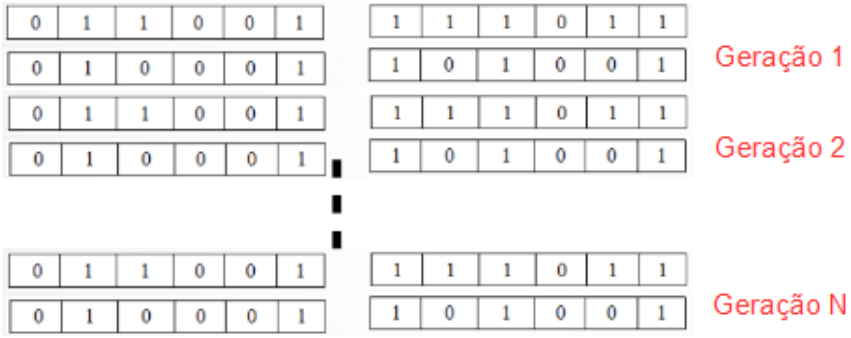
Figura 5 - Conjunto de indivíduos formando uma população



Fonte: Acervo próprio

Geração: é a variação de diferentes indivíduos da população com o passar do tempo – figura 6;

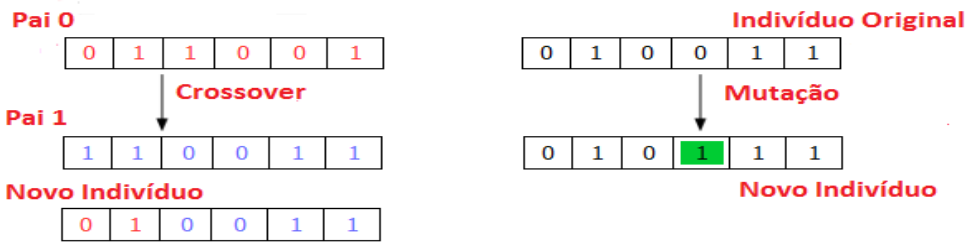
Figura 6 - Variação das populações com o passar do tempo, formando as gerações do presente e do passado.



Fonte: Acervo próprio

Operadores Genéticos: são os métodos reprodutivos, que podem ser de forma sexuada, ou de forma assexuada. A forma sexuada é a forma com que os indivíduos se reproduzem através do cruzamento entre si, trocando material genético (genes) para que seja possível a criação de novos indivíduos. A este método dá-se o nome de *crossover* ou operador de recombinação, e pode ser observado na figura 7.

Figura 7 - C*rossover* juntamente com mutação



Fonte: Acervo próprio

Já ao método assexuado dá-se o nome de mutação, é onde um indivíduo cria uma cópia de si mesmo, porém durante a criação desta cópia, determinado gene aleatório é modificado em relação a sua matriz genética original, fazendo com que o mesmo deixe de ser uma cópia e seja um novo indivíduo.

Na figura 7, observa-se que a troca de material genético entre dois indivíduos se deu de forma a se cruzarem em um único ponto, ou seja, este é o chamado *crossover* de 1 ponto. Porém a implementação pode ser feita em dois pontos, chamado de *crossover* de 2 pontos, ou em 3 pontos, que é chamado de *crossover* de 3 pontos. Conforme [13], acima de 3 pontos não existem ganhos comprovados de qualidade na solução obtida como resposta. A figura 8, representa o chamado *crossover* uniforme, onde pode-se observar que a troca de material genético entre os pais se dá de forma aleatória, através de uma matriz de recombinação. Este é um dos tipos de *crossover* em que se pode obter as melhores soluções. Ele será discutido posteriormente.

Figura 8 - Crossover uniforme

Fonte: Acervo próprio

Função de Avaliação: Também conhecida como função de *fitness* ou função objetivo. Nos algoritmos genéticos, esta é a fase que consome maior tempo de processamento, pois é o método que avalia um a um os indivíduos, conforme a qualidade que apresentam em relação a resposta esperada para o problema. Esta "qualidade" é transformada em um valor numérico, para que se possa saber quais indivíduos são mais aptos dentro de uma população.

Método de Seleção: Imprescindível para a evolução do AG, o método de seleção ocorre baseado na avaliação realizada anteriormente. Ele deve delimitar quais são os indivíduos que devem ter mais chances de levar seu material genético às gerações futuras e quais sucumbirão na atual geração, não deixando descendentes. Importante salientar que isto não implica que os indivíduos mais bem avaliados tenham que obrigatoriamente passar seu material genético à

frente, mas sim que eles tenham maiores chances (probabilidade maior do que os indivíduos com avaliação menor) de deixarem descendentes. Entre os tipos de métodos de seleção, pode-se citar os seguintes:

Método do Elitismo → Os melhores indivíduos de uma geração são passados diretamente para a geração seguinte, sem efetuar cruzamento com outros indivíduos ou mutações em sua estrutura.

Método do Torneio → Indivíduos aleatórios são escolhidos dentro da população e os mesmos "duelam" entre si, levando em consideração o valor de suas avaliações calculadas pela função de avaliação. A disputa é para ver qual dos dois indivíduos sorteados aleatoriamente tem o direito de usar algum dos operadores genéticos (*crossover* ou mutação).

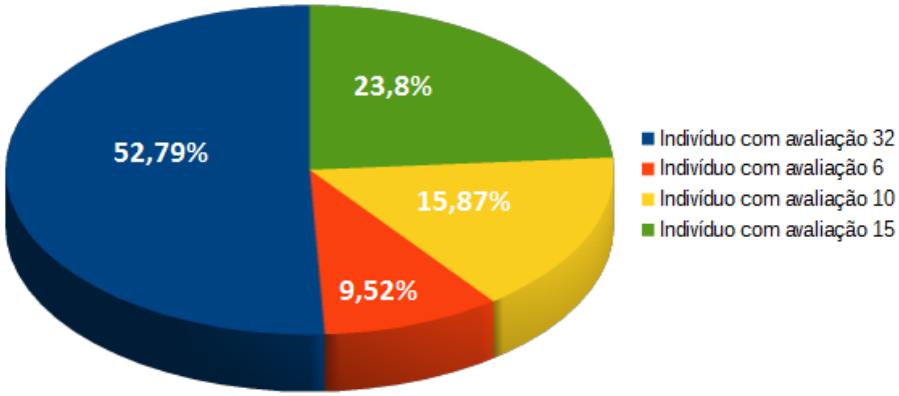
Método da Seleção por *Ranking* → Estabelece-se uma ordem decrescente pelo valor da avaliação de cada indivíduo e, a partir disto escolhe-se os primeiros do ranking para serem efetuados os operadores genéticos sobre os mesmos. Em seu relato, [13] fala que este método, apesar de ser de fácil implementação, pode ser prejudicial para o algoritmo, pois muitas vezes resulta em uma convergência genética precoce. Convergência genética é quando as gerações começam a evoluir somente para determinada região do espaço de busca, não contemplando outras partes deste espaço. Na seleção por *Ranking*, isso pode acontecer porque os piores elementos podem não ter chance de se reproduzirem, e se alguns destes elementos possuírem uma parte de sua carga genética essencial para a solução do problema, essa carga não chegará até as gerações futuras e a resposta convergirá de forma rápida, porém não tão completa.

Método da Roleta Viciada → Método que consiste em emular computacionalmente uma roleta de jogos viciada. Uma população (conjunto de indivíduos) possui, por exemplo, avaliações nos valores de 32, 6, 15, 10, cujo o somatório é 63, este número corresponde aos 360º de uma roleta. O indivíduo que possui avaliação 32 ocupará 52,79% da roleta, ou seja, 190º. O indivíduo com avaliação 6 ocupará 9,52% da roleta, ou seja, 34,27º. O indivíduo de avaliação 15 ocupará 23,8% da roleta, ou seja, 85,71º e o indivíduo de avaliação 10 ocupará 15,87% da roleta, ou seja, 57,14º. Observa-se que o indivíduo que terá mais chances de ser sorteado é o que possui avaliação 32, seguido do que possui avaliação 15, sendo o próximo o que possui avaliação 10 e o indivíduo que possui menor avaliação, ou seja, 6, possui a menor chance de ser selecionado entre todos os outros. Este método de seleção é mais lento perto dos outros 3, mas é o que dá mais chances de os indivíduos menos aptos serem escolhidos para que possam levar sua carga genética até as gerações futuras. Dessa forma, pode-se prevenir a

convergência genética prematura e aumentar o espaço de busca a ser explorado. Uma pseudo representação da divisão da roleta entre os indivíduos no método da roleta viciada pode ser melhor observado na figura 9.

Segundo [13], e como já comentado, os métodos de seleção mais rápidos (elitismo, torneio, *ranking*), possuem convergência genética prematura e varredura de um espaço de busca muitas vezes menor que o necessário. Nestes métodos, os indivíduos com aptidão baixa são privados do cruzamento, sendo que muitas vezes, parte de sua carga genética teria contribuição importante para a resposta final do problema.

Figura 9 - Exemplo do método da Roleta Viciada



Fonte: Acervo próprio

Já os métodos de seleção mais lentos (roleta viciada), são o inverso em relação aos métodos mais rápidos. O tempo de execução é muito prejudicado, já que pode haver o cruzamento entre indivíduos com aptidão muito baixa, formando, dessa forma, muitos indivíduos ruins. Porém, a varredura do espaço de busca é bem mais ampla do que os outros métodos, já que desta forma uma parte do espaço de busca inexplorado passa ser visitado e a possibilidade de encontrar indivíduos mais aptos para a melhor solução do problema é consequentemente maior.

2.1.2 **Modelo de algoritmo genético**

Conforme [13], a forma com que os indivíduos serão codificados para representação

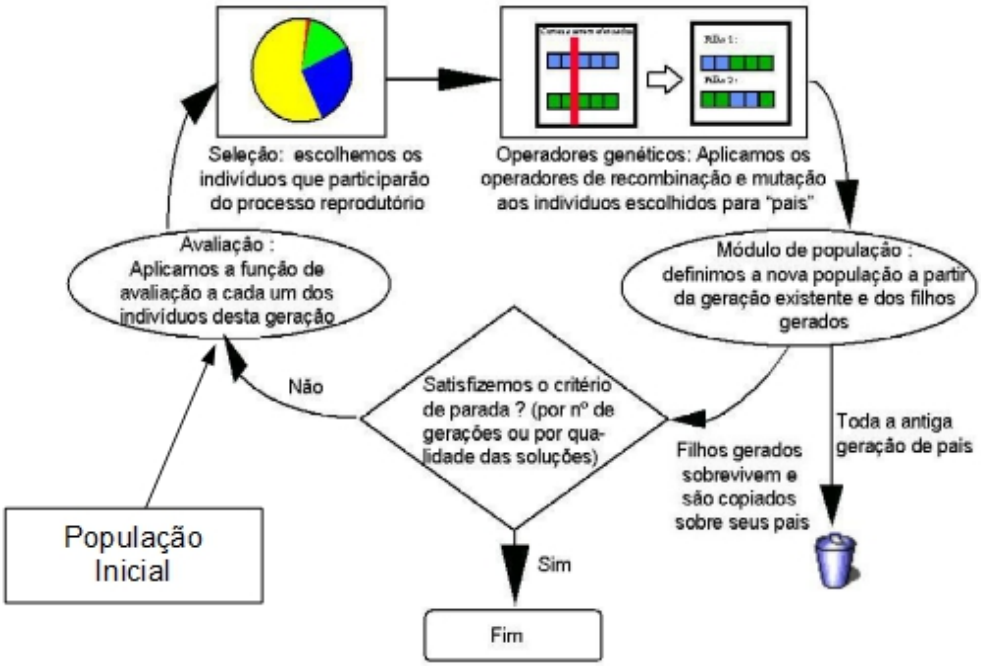
das informações do problema dentro do AG é parte de grande importância dentro da implementação do algoritmo. Deve-se modelar a estrutura para que se possa trabalhar com todas as informações relevantes do problema que serão manipuladas pela implementação.

Esta ação se resume, basicamente, em traduzir a informação do problema em questão para um modelo em que seja viável seu tratamento pelo computador, quanto mais adequada ela for, melhor será a qualidade dos resultados obtidos.

Uma boa definição da representação das informações no indivíduo minimiza a quantidade de testes necessários para que possa ser feita a avaliação do mesmo, diminuindo, desta forma, o tempo de execução do algoritmo, visto que a avaliação dos indivíduos é um dos pontos que consome maior tempo de processamento dentro do AG.

Pode-se observar na figura 10 o fluxograma de como é o funcionamento básico de um algoritmo genético, contendo suas diversas etapas.

Figura 10 - Esquema do funcionamento de um algoritmo genético



Fonte: Adaptado de Linden (2012)

2.2 **Trabalhos correlatos**

Segundo [3], um indivíduo com boa avaliação satisfaz questões como colisões dos

docentes na grade horária e leva em consideração as disponibilidades dos professores nos diversos períodos para que ministrem suas aulas. Para ele, quando uma grade possui dois períodos consecutivos da mesma disciplina, ela é pontuada positivamente com 10 pontos, quando existem 3 períodos consecutivos da mesma disciplina, são acrescentados 5 pontos, e quando o agrupamento passa para 4, a pontuação adicionada passa a ser 15 pontos. Pode-se observar que este tipo de abordagem favorece as aulas agrupadas sempre com dois períodos consecutivos em relação aos outros agrupamentos que podem ocorrer.

O mesmo acontece no momento da alocação de um docente. O docente pode ter ou não preferência sobre um determinado período, se o mesmo for alocado em um período em que possui preferência, serão creditados 2 pontos na avaliação do indivíduo, e se for alocado em um período que não são os preferenciais, 1 ponto é adicionado.

Para [3], o AG tem sua execução finalizada quando um número predeterminado de gerações é contabilizado.

O autor de [10], em sua tese de mestrado, também utilizou os algoritmos genéticos na resolução do problema da geração de horários. As ferramentas utilizadas por ele foram o banco de dados Firebird 1.5 e o ambiente de desenvolvimento Borland Delphi 7. O sistema desenvolvido neste trabalho possui diversos módulos, sendo que um dos principais é o módulo em que são configurados diversos parâmetros do AG. Ele disponibiliza os seguintes parâmetros para configuração: quantidade máxima de gerações a serem seguidas, quantidade de indivíduos em cada geração, taxa de cruzamento, taxa de mutação, tipo de cruzamento (um ou dois pontos), método de seleção dos pais (torneio ou roleta).

Um dos principais pontos de atenção na implementação de um AG é a modelagem das soluções/indivíduos. Em sua implementação, [10] usa como representação dos indivíduos de uma população uma matriz de 25 colunas por 6 linhas, onde a primeira coluna é o primeiro horário da segunda-feira, a segunda coluna é o segundo horário da segunda-feira, e a vigésima quinta coluna é o último horário da sexta-feira. A primeira linha é o primeiro semestre, a segunda é o segundo semestre, até a sexta linha, que é o sexto semestre da grade curricular baseada no curso de Análise e Desenvolvimento de Sistemas do IFTM.

Foi observado que [18] comparou soluções exatas, heurísticas e mistas. Ele afirma que em seus diversos testes e experimentos, o método heurístico (AG) pecou em várias soluções, apesar de serem boas e aplicáveis ao problema, não apresentaram muitas vezes aulas geminadas, que para muitas instituições é indispensável, visto que seus horários já são

culturalmente confeccionados levando em consideração este item.

A representação dos indivíduos foi implementada com um vetor em [8]. Ele utiliza cruzamentos de apenas um ponto, sendo que os semestres são divididos em cadeias dentro deste vetor. A mutação ocorre com a troca/*swap* de professores e disciplinas dentro dos genes, parecidos com o que ocorre nos algoritmos meméticos. A grande limitação desta implementação é trabalhar com a geração de uma grade horária para um único curso por vez, e aparentemente, trabalhar com grades que possuam um número pré-determinado de créditos por semestre. Caso este número fosse modificado, os métodos como a implementação trabalha devem ser alterados.

Igualmente ao caso anterior, [10] afirma que sua solução foi implementada também para ser executada apenas com um curso por vez, e como já comentado, com cursos de até seis semestres. Os operadores genéticos escolhidos por ele são os cruzamentos de um ou dois pontos que devem ser parametrizados antes da execução do AG, assim como os métodos de seleção utilizados são a roleta, ou o torneio, que devem ser escolhidos entre um ou outro previamente. O operador genético de cruzamento implementado realiza a troca de genes entre os indivíduos. Isto pode fazer com que o "esquema" de um semestre seja desfeito, retirando aulas da carga horária de determinada disciplina e alocando aulas a mais para outra, tornando o indivíduo inválido. Por conta disto, toda vez que um indivíduo inválido é gerado pelo cruzamento, o mesmo é desprezado e um outro indivíduo é gerado em seu lugar. Isto pode gerar um tempo de processamento grande até que todos os indivíduos possam ser gerados de forma satisfatória.

Em [14], o operador de cruzamento teve uma alta complexidade de implementação, visto que assim como [10], durante os cruzamentos, a ocorrência de sobreposição de turmas no indivíduo gerado ocorreram com considerável frequência.

Assim como já mostrado, em [3] também ocorrem perda de horas/aula durante o cruzamento dos indivíduos, sendo que também se faz necessário verificar se os indivíduos são válidos (se a carga horária dos mesmos está correta). A implementação de [3] atua somente sobre um curso isolado da instituição, não referindo as disponibilidades usadas de um determinado professor sobre outros cursos em que o mesmo já foi alocado. Nela ainda existe a limitação de haver apenas um professor titular para cada disciplina, que deve ser previamente alocado antes do início da execução do AG. Ele ainda diz que cada fase de um curso possui apenas uma turma por turno.

De outra forma, [5] afirma que a estrutura de sua implementação de AG é extremamente flexível no que diz respeito a ser possível a alocação de qualquer número de cursos, sendo que os mesmos podem possuir qualquer número de semestres, disciplinas, tipos de disciplinas e tipos de recursos. Porém, existe a necessidade de adaptar o modelo às necessidades de outras instituições de ensino para a estrutura funcional das mesmas, visto que o AG sugerido pelo autor é altamente modelado para as necessidades da Universidade Federal de Santa Catarina.

Apesar de não ser o foco deste trabalho, foram verificadas algumas produções acadêmicas, como [18], que menciona o *software* Lingo, e [4], que menciona o *software* Time'sCool, o *software* Tablix (software que utiliza AG) e o *software* Cronos.

O *software* Time'sCool, para [4], é um software que apesar de se mostrar bem funcional, tem recursos bem básicos, sendo que não dispõe de interações mais aprofundadas que permitam uma boa adequação à realidade das instituições de ensino. É uma interface *web* e de uso livre. Quando [4] menciona o *software* Tablix, refere-se a um sistema que possui versões para Linux, Windows e MacOS, interface e manual de instruções somente em inglês e sem suporte técnico disponível, e [4] ainda fala em uma interface um pouco confusa no momento da inclusão das informações pertinentes ao problema. Também é de uso livre. Para [4], o *software* mais completo é o Cronos, que possui restrições quanto aos professores não ministrarem aulas em determinados horários/dias como sendo indesejados ou inviáveis, e ainda possibilita a criação de aulas geminadas. Os resultados podem ser gerados em formato de planilhas eletrônicas do tipo *Microsoft Office Excel* ou formato HTML. Não foi constatado a parametrização de salas ou laboratórios e no site da empresa, [4] não encontrou informações se seria possível uma customização do sistema.

Em sua dissertação de mestrado, [18] fala do *software* Lingo, que resolve o problema através de métodos exatos matemáticos, porém ele afirma que o custo financeiro deste *software* é extremamente elevado, além de ser necessário um grande conhecimento prévio sobre o comportamento da ferramenta.

Como em outros casos, a solução utilizada por [7] também gera grades somente para 1 único curso e [15] trabalha com uma heurística de formulação matemática, que pode até mesmo ser aplicada ao *Microsoft Office Excel* e em cursos com grade horária pequena, se mostrou muito eficiente.

3 DESENVOLVIMENTO DO TRABALHO

O problema da grade horária, como já descrito anteriormente, contém uma série extensa de recursos que precisam ser combinados e alocados em determinados *slots* de tempo, porém sempre levando em consideração, determinadas restrições que podem ser impostas a todos ou a alguns dos recursos durante sua alocação.

Desta forma, para auxiliar a encontrar a solução deste problema, e como foco principal deste trabalho, foi implementado, em um protótipo de software, a técnica de busca aleatório guiada conhecida como algoritmos genéticos.

3.1 Desenvolvimento do protótipo

Já descrito anteriormente, um dos objetivos secundários propostos neste trabalho refere-se à escolha da linguagem de programação para o desenvolvimento do protótipo de software e para a implementação da técnica selecionada para resolução do problema.

Utilizando as mesmas ideias que [1] e [14], a linguagem de programação apurada para o desenvolvimento da interface e aplicação da heurística escolhida foi Java, por ser robusta, livre, possuir ambiente de desenvolvimento gratuito e ser multiplataforma.

Assim como sugerido em [8], todas as informações sobre disciplinas, docentes, cursos, etc, têm necessidade de serem guardadas em um local para que possam ser resgatadas posteriormente de forma íntegra. Para isso, foi selecionado o gerenciador de banco de dados MySQL, um dos bancos mais populares da atualidade, que possui licença GPL e utiliza a linguagem SQL, *structured query language* ou, linguagem de consultas estruturadas.

3.2 Levantamento das restrições

No contexto da Unipampa, pode-se observar as seguintes restrições para que uma solução de qualidade possa ser encontrada:

– A instituição possui laboratórios que são de uso específico para algumas disciplinas dos seus cursos, sendo que duas disciplinas que necessitam de aula num mesmo laboratório deste tipo, não podem ser alocadas concomitantemente;

– Como regra, duas aulas com disciplinas diferentes não podem ser alocadas numa mesma sala de aula, num mesmo período;

– Um mesmo professor não pode ser alocado para ministrar aulas para duas turmas num mesmo período de tempo em salas diferentes;

– A alocação de períodos, por definição para este trabalho, vai da segunda-feira pela manhã até a sexta-feira à noite;

– O balanceamento de carga horária de professores deve ser levado em consideração no momento da geração da grade horária, para que o mesmo possa ser feito da forma mais igualitária possível, assim como a distribuição das salas também deve ser feita de maneira semelhante;

– Existem determinados dias e determinados períodos em que alguns docentes não podem ministrar aulas, devido aos mais variados motivos (pessoais, profissionais, etc);

– Existem determinados dias e/ou períodos em que os docentes têm preferência por ministrarem suas aulas.

3.3 **Coleta das informações do escopo do problema**

As informações referentes ao funcionamento do algoritmo genético devem ser mantidas em uma base de dados, para que possam ser utilizados sempre que for preciso gerar uma nova grade horária. A consistência e relação entre estes dados é de extrema importância para que a solução satisfatória do problema possa ser encontrada.

Através de um levantamento feito com planilhas que continham as grades curriculares dos cursos de graduação e grades de horários de semestres passados, notou-se que a Unipampa, em seu campus Bagé, possui atualmente aulas divididas em 15 períodos (10 períodos diurnos, 4 períodos noturnos e 1 período entre noturno e diurno), 334 disciplinas que atendem um total de 12 cursos, sendo que estas disciplinas estão vinculadas à determinadas áreas (13 áreas ao total), que são áreas responsáveis pela organização e execução das mesmas nos cursos do campus.

Existem disciplinas que podem ser ministradas por mais de um professor, e um professor pode ministrar mais de uma disciplina, criando assim uma vinculação de N disciplinas para N professores, e também existem disciplinas em que somente um professor pode ministrar aulas, sendo a vinculação de N disciplinas para 1 professor.

O que se espera é que, contendo estes dados organizados e relacionados da forma correta, ou seja, disciplinas vinculadas às suas respectivas áreas de responsabilidade, restrições de dias e horários enumeradas para os seus referidos professores e disciplinas vinculadas ao curso de maneira correta, obtenha-se um resultado com afinidade entre professores e disciplinas de maneira satisfatória, onde a grade horária final esteja bem próxima da resposta esperada, podendo ser aplicada aos cursos da referida instituição de ensino.

Dados como quantidade de salas disponíveis e indisponíveis, capacidade, quantidade de laboratórios e seus usos específicos foram coletados através de planilhas eletrônicas no formato *Microsoft Excel*, junto à Comissão de Espaço Físico e Obras do campus Bagé, da Unipampa.

Todas estas informações foram inseridas na base de dados proposta neste trabalho, para que possam ser resgatadas pelo protótipo a qualquer momento em que se queira realizar um experimento, não necessitando que os dados tenham de ser colhidos das planilhas e organizados novamente. A base de dados em questão será explanada na próxima subseção.

3.4 **Modelagem do Banco de Dados**

O banco de dados do protótipo foi modelado a fim de comportar todas as informações levantadas junto as planilhas distribuídas pela antiga coordenação do curso de Engenharia da Computação do campus Bagé, e informações sobre espaço físico com a COEF.

Ao todo, foram criadas 9 tabelas dentro do banco de dados MySQL. São elas:

Tabela docentes: possui todos os docentes do campus Bagé, separados pelas "áreas de responsabilidades" ao qual pertencem.

Tabela area_responsabilidade_disciplinas: conjunto de "áreas de responsabilidades" que são responsáveis por manter as disciplinas do campus junto com seus docentes.

Tabela disciplinas: cadastro referente a todas as disciplinas ministradas no campus da universidade.

Tabela afinidade_disciplinas: tabela que contém as informações sobre quais docentes (existentes no universo de docentes cadastrados na tabela **docentes**) estão aptos a ministrar determinada disciplina que pertence ao universo existente na tabela **disciplinas**.

Tabela cursos: contém informações sobre os diversos cursos oferecidos pela universidade.

Tabela disciplinas_cursos: tabela que mostra quais disciplinas fazem parte do currículo de determinado curso dentro do universo de disciplinas presentes na tabela **disciplinas**.

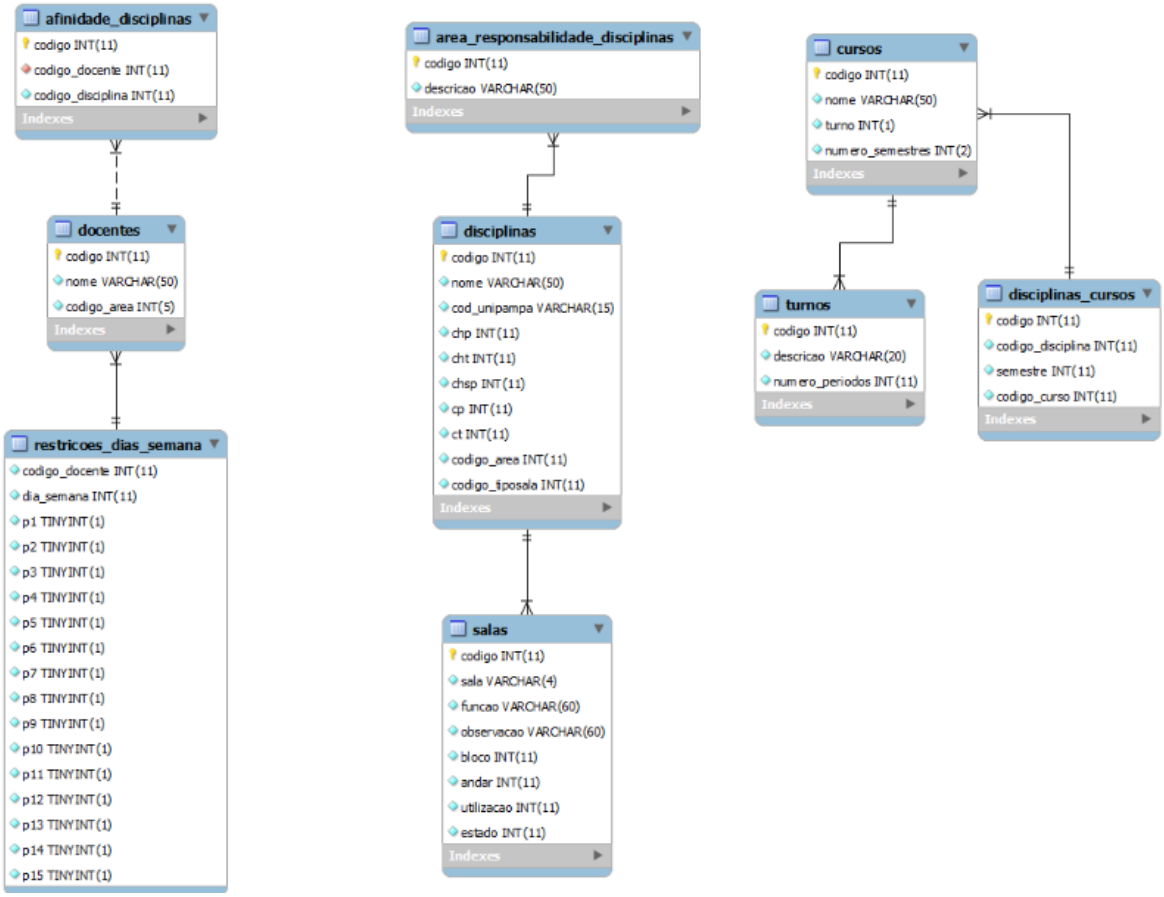
Tabela restricoes_dias_semana: possui todas as informações sobre restrições de dias e horários em que os docentes (oriundos da tabela **docentes**) não podem ministrar aulas.

Tabela salas: tabela com todas as salas do campus Bagé da Unipampa.

Tabela turnos: tabelas em que a Unipampa possui períodos letivos.

O relacionamento das tabelas elencadas acima, que fazem parte do banco de dados do protótipo, pode ser observado na figura 11.

Figura 11 - Modelo de Entidade Relacionamento do banco de dados do protótipo



Fonte: Acervo próprio

Através da figura 11, seguindo o relacionamento entre as tabelas do banco de dados, é possível verificar todos os docentes que são aptos a ministrar determinada disciplina, quais disciplinas possuem vínculo a determinado curso, as restrições que determinados docentes possuem, as áreas de responsabilidade com suas respectivas disciplinas, disciplinas que necessitam de salas específicas e as salas de aula do campus da universidade.

3.5 **Implementação do protótipo em Java**

A implementação do protótipo baseou-se no padrão de arquitetura MVC, que é o modelo visão-controlador. Este é o modelo de arquitetura de *software* que separa a representação, da informação e da interação do usuário com estas duas últimas.

A modelagem do MVC, para este trabalho, foi feita implementando 3 camadas, sendo elas: camada *Bean* (controlador), camada *View* (visão), e camada DAO (modelo).

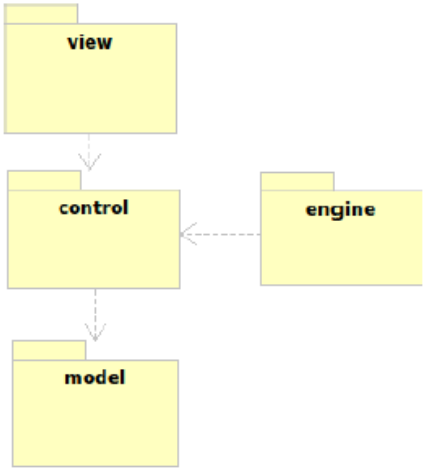
Camada *Bean* → Camada de objetos de negócio, responsável pelo encapsulamento dos dados.

Camada *View* → Camada de apresentação visual dos dados.

Camada DAO → Camada de persistência entre a aplicação e o banco de dados.

Para comunicação entre o banco de dados MySQL e o protótipo em Java, foi utilizada a biblioteca *Mysql Connector Java* 5.1.18, e para importação das informações que eram originárias das planilhas eletrônicas, foi utilizada a biblioteca *JexcelAPI*.

Figura 12 - Divisão das classes em pacotes



Fonte: Adaptado de Ferreira; Vieira (2011)

Para uma divisão das classes em pacotes, assim como o realizado em [19], ficariam no pacote *model* as classes pertencentes à camada DAO, pois são as classes responsáveis pelo acesso e gestão dos dados, formando assim uma camada de abstração do SGBD. No pacote *control* as classes da camada *Bean*, que são as classes que possuem as regras de negócio. No pacote *view* as classes da camada *View,* que são as classes que apresentam os dados ao usuário, juntamente das classes de *interface* do protótipo. Por último, no pacote *engine* as classes que se referem diretamente ao algoritmo genético, ou seja, o núcleo do protótipo. A

figura 12 mostra esta divisão.

O protótipo possui 31 classes, que podem ser observadas na tabela 1 e 2:

Classe	Função	Pacote
AfinidadeDisciplinasBean	Camada de controle MVC	*control*
AfinidadeDisciplinasDAO	Persistência entre o banco e o protótipo	*model*
AreaResponsabilidadeDisciplinaBean	Camada de controle MVC	*control*
AreaResponsabilidadeDisciplinaDAO	Persistência entre o banco e o protótpo	*model*
Conexao	Realiza a conexão junto ao MySQL	*model*
Coordenadas	Contém coordenadas de linha/coluna na grade horária	*control*
CursoBean	Camada de controle MVC	*control*
CursoDAO	Persistência entre o banco e o protótipo	*model*
DisciplinaBean	Camada de controle MVC	*control*
DisciplinaCursoBean	Camada de Controle MVC	*control*
DisciplinaCursoDAO	Persistência entre o banco e o protótipo	*model*
DisciplinasDocentes	Mantém as relações entre disciplinas e docentes	*control*
DocenteBean	Camada de controle MVC	*control*
DocenteDAO	Persistência entre o banco e o protótipo	*model*
FormAfinidadeDisciplinas	Camada *view* MVC	*view*
FormAreaResponsabilidadeDisciplina	Camada *view* MVC	*view*
GradeHoraria	Formulário de geração da grade horária	*view*
Individuo	Classe que instância o indivíduo do AG	*Engine*
Main	Classe principal do protótipo	*View*
FormCursos	Camada *view* MVC	*view*
FormDisciplina	Camada *view* MVC	*View*
FormDisciplinaCurso	Camada *view* MVC	*view*
FormDocente	Camada *view* MVC	*view*

Tabela 1 - Tabela contendo as classes do protótipo e suas funções

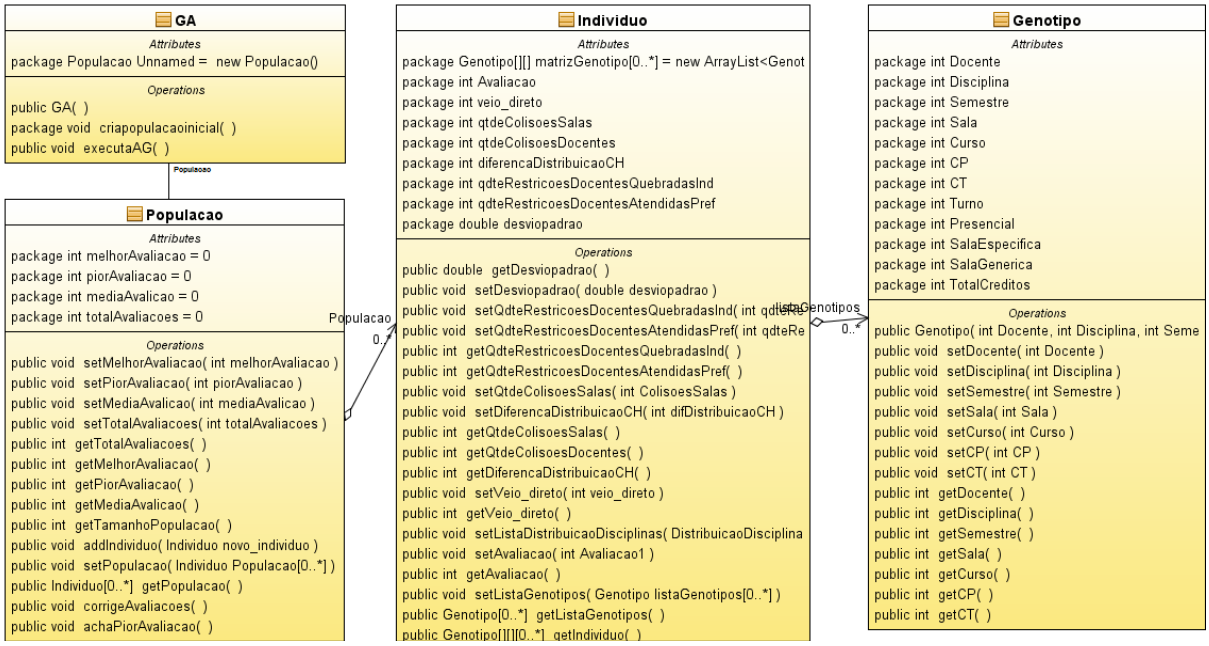
Classe	Função	Pacote
FormGeraGrade	Camada *view* MVC	*view*
FormPendenciasDisciplinasDocentes	Camada *view* MVC	*view*
FormPrincipal	Formulário principal	*view*
FormRestricoesDiasSemana	Camada *view* MVC	*view*
FormSalas	Camada *view* MVC	*view*
FormTurnos	Camada *view* MVC	*view*
GA	Classe principal do AG	*engine*
Genotipo	Classe que mantém as informações do genótipo	*engine*
Populacao	Classe que mantém a população da AG	*engine*
RestricoesDiasSemana	Classe que mantém as restrições dos docentes	*control*
RestricoesDiasSemanaBean	Camada de controle MVC	*control*
RestricoesDiasSemanaDAO	Persistência entre o banco de dados e o protótipo	*model*
SalasBean	Camada de controle MVC	*view*
SalasDAO	Persistência entre o banco e o protótipo	*model*

Tabela 2 - Tabela contendo as classes do protótipo e suas respectivas funções

Um diagrama de classes, do pacote que é responsável pela execução em si do algoritmo genético, pode ser vista na figura 13, na qual nota-se a relação entre as mesmas durante a execução da heurística utilizada neste trabalho. Pode-se observar as 4 classes responsáveis pelo núcleo do algoritmo genético. São elas GA, Populacao, Individuo, Genotipo.

"GA" é a classe que diretamente executa o algoritmo genético através do método "executaAG()". A classe "Populacao" mantém a lista de indivíduos pertencentes à determinada geração, e os índices de avaliação de seus indivíduos. A classe Individuo contém a lista de matrizes que possuem os genes e métodos como "funcaoAvaliacao()", que é a função que avalia o indivíduo, ou seja, o indivíduo avalia-se a si próprio conforme todos os parâmetros da função de avaliação e também a classe "Genotipo", que é a classe responsável por carregar todos os genes que serão inseridos nos indivíduos com as informações de docentes, disciplinas, salas, cursos, semestre oriundas do banco de dados.

Figura 13 - Diagrama de classes do pacote *engine*



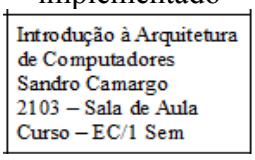
Fonte: Acervo próprio

3.6 **Implementação do Algoritmo Genético**

Segundo [13], a representação dos indivíduos, assim como dos genes é de fundamental importância para que o AG consiga resultados de forma satisfatória. A representação é a maneira de traduzir as informações de nosso problema para que seja viável seu tratamento pelo computador. Apesar de a maioria da literatura disponível indicar a representação binária para a utilização dos algoritmos genéticos, ela na verdade é totalmente arbitrária, sendo que muitos autores indicam a utilização de uma representação que mais se ajusta ao problema em questão.

Neste trabalho, o gene (menor parte formadora e indivisível de um indivíduo) é representado por uma tupla que possui as seguintes informações: código da disciplina, código do curso ao qual a disciplina está relacionada, código do docente responsável por ministrar a aula, turno em que a disciplina é oferecida, semestre do curso ao qual pertence a disciplina, sala em que a mesma deverá ser ministrada, se é uma disciplina presencial, créditos práticos e créditos teóricos. Durante a implementação em java, essa tupla foi chamada de genótipo (figura 14).

Figura 14 - Exemplo do Gene de um indivíduo no AG implementado



	Segunda	**Terça**	**Quarta**	**Quinta**	**Sexta**
Manhã					
Manhã					
Tarde					
Tarde					
Intervalo Turnos					
Noite	Geometria Analítica Michéli Peters 4109 - Sala de Aula Curso – EC/1 Sem	Cálculo I Karlla Morales 4309 - Sala de Aula Curso – EC/1 Sem	Algoritmos e Programação Sandra Piovesan 2306 - Lab. Informática Curso – EC/1 Sem	Introdução à Engenharia de Computação Sandra Piovesan 4211 - Sala de Aula Curso – EC/1 Sem	Introdução à Arquitetura de Computadores Sandro Camargo 2103 - Sala de Aula Curso – EC/1 Sem
Noite	Produção Acadêmico-Científica Cássia 1307 - Sala de Aula Curso – EC/1 Sem	Introdução à Arquitetura de Computadores Sandro Camargo 2103 - Sala de Aula Curso – EC/1 Sem	Algoritmos e Programação Sandra Piovesan 4105 - Sala de Aula Curso - EC/1 Sem	Geometria Analítica Michéli Peters 4109 - Sala de Aula Curso - EC/1 Sem	Cálculo I Karlla Morales 4309 - Sala de Aula Curso - EC/1 Sem

Tabela 3 - Exemplo de matriz representando o 1º semestre do curso de EC (Noturno)

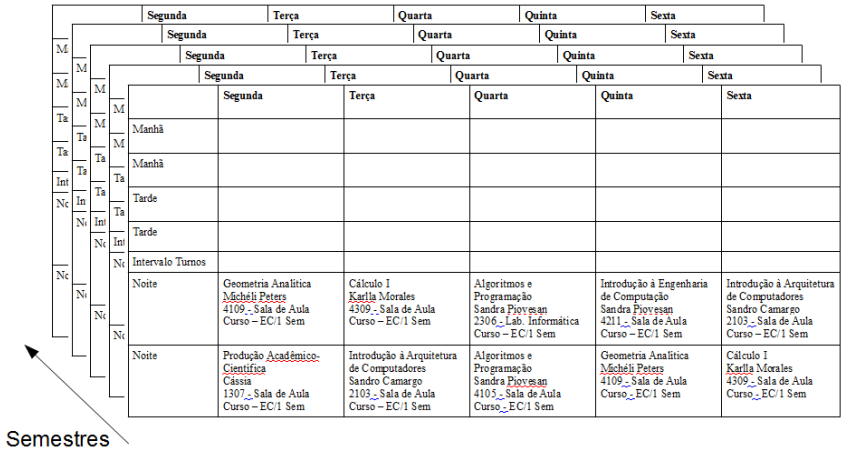
Um conjunto desses genes forma o indivíduo, que é uma das soluções candidatas a resposta do problema. O indivíduo foi modelado de maneira a assemelhar-se com uma lista de matrizes, em que cada célula de determinada matriz contém um gene se for um período ocupado, ou um espaço em branco se for um horário vago. Cada matriz desta lista é do tamanho de 7 linhas por 5 colunas, sendo que as 5 colunas representam os 5 dias letivos em que são possíveis a alocação de aulas e as 7 linhas representam, nesta ordem, os períodos matutinos de antes do intervalo, os períodos matutinos após o intervalo, os períodos vespertinos antes do intervalo, os períodos vespertinos após o intervalo, o período entre os turnos diurno e noturno, os períodos noturnos antes do intervalo e os dois últimos períodos da noite. Esta divisão foi escolhida desta forma pois a Unipampa aloca as aulas das disciplinas com no mínimo dois períodos de aulas consecutivos. Desta forma, o número de alocações a serem feitas cai pela metade, pois em vez de ser necessária a alocação período a período, ela é feita a cada dois períodos. As tabelas 3 e 4 exemplificam estas matrizes.

	Segunda	Terça	Quarta	Quinta	Sexta
Manhã	Transferência de Calor e Massa II Edson Chiaramonti 2104 - Sala de Aula Curso – EERA/7 Sem		Tecnologia de Sistemas Fotovoltáicos Jocemar Parizzi 1105 - Lab. Energia Eólica Curso – EERA/7 Sem	Eletrônica de Potência Jocemar Parizzi 2209 - Sala de Aula Curso – EERA/7 Sem	Eletrônica de Potência Jocemar Parizzi 2209 - Sala de Aula Curso – EERA/7 Sem
Manhã	Laboratório de Máquinas Elétricas Carlos Guilherme 2204 - Lab. EERA - Eletr. e Autom. Curso – EERA/7 Sem			Transferência de Calor e Massa II Edson Chiaramonti 2104 - Sala de Aula Curso – EERA/7 Sem	Máquinas Elétricas Carlos Guilherme 4103 - Sala de Aula Curso – EERA/7 Sem
Tarde	Tecnologia de Sistemas Fotovoltáicos Jocemar Parizzi 4205 - Sala de Aula Curso – EERA/7 Sem		Máquinas Elétricas Carlos Guilherme 4103 - Sala de Aula Curso – EERA/7 Sem		Sistemas Digitais Aplicado Fábio Tomm 4308 - Sala de Aula Curso – EERA/7 Sem
Tarde	Gestão e Planejamento Ambiental Helvio Rech 4309 - Sala de Aula Curso – EERA/7 Sem	Eletroquímica Luis Brudna 1307 - Sala de Aula Curso – EERA/7 Sem			Sistemas Digitais Aplicado Fábio Tomm 4308 - Sala de Aula Curso – EERA/7 Sem
Intervalo Turnos					
Noite					
Noite					

Tabela 4 - Exemplo de matriz representando o 7º semestre do curso de EERA (Diurno)

Cada matriz da lista de matrizes do indivíduo corresponde a um semestre do curso ao qual está sendo gerada a grade. Se o algoritmo estiver sendo executado com múltiplos cursos, a matriz do último semestre de um dos cursos selecionados para geração da grade horária é seguida pela matriz referente ao primeiro semestre do próximo curso que está selecionado para execução do AG.

Figura 15 - Indivíduo completo, com todas suas matrizes

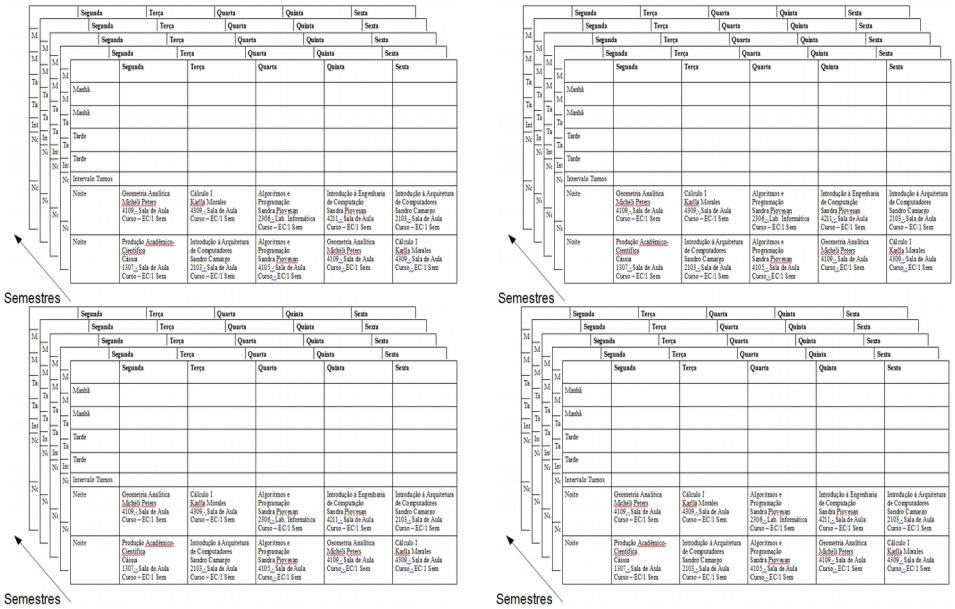


Fonte: Acervo próprio

A população do algoritmo genético é o conjunto de indivíduos que podem ser solução

do problema, ou seja, na implementação em questão é uma lista de indivíduos. A figura 16 mostra um exemplo próprio de uma pequena população de 4 indivíduos do AG implementado (exemplificado na figura 15).

Figura 16 - Exemplo de população do AG implementado



Fonte: acervo próprio

3.6.1 **Formação da população inicial**

Quando a execução do algoritmo genético é iniciada, é neste momento em que a população inicial é criada indivíduo a indivíduo. Para que isto aconteça, são seguidos uma série de critérios relacionados a seguir.

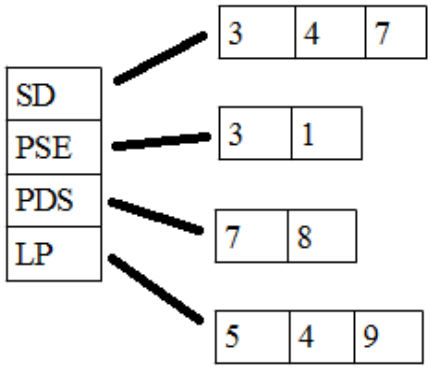
Primeiramente, uma lista com todas as disciplinas dos cursos selecionados para comporem o AG é carregada em memória. Se, por exemplo, o curso de EC e o curso de EERA estão selecionados, a disciplina de Calculo II aparecerá duas vezes na lista, sendo uma vez para EC e outra vez para EERA, pois os dois cursos possuem esta disciplina em seus currículos.

Posteriormente é feita a distribuição dos docentes nesta lista de disciplinas, seguindo a lista de afinidades preestabelecida (figura 17). Se houver uma disciplina que não possua afinidade com nenhum docente, o AG será impedido de ser executado até que seja atribuído

pelo menos a afinidade de um docente a esta disciplina. Para ajudar a manter a distribuição de forma que seja a mais igualitária possível entre todos os docentes, a cada disciplina que é feita uma atribuição de docente, o conjunto dos docentes que possuem afinidade com a disciplina em questão é reunido e o docente com menor número de alocações é escolhido, já que é mantida a informação de quantas disciplinas já foram atribuídas a determinado docente. Se existem vários docentes dentro deste conjunto com o mesmo número de atribuições, e estes estão entre os que foram menos alocados, então é realizado um sorteio entre eles para que um deles possa ser combinado com esta disciplina. Desta forma não assegura-se que a carga horária será igual, apenas ajuda na divisão das disciplinas. A questão da carga horária será avaliada mais a frente, através da função de avaliação do AG.

Em seguida, a lista de disciplinas é percorrida novamente a fim de combinar as salas em que as aulas serão ministradas. Se no momento do cadastro da disciplina, for pré-definida uma sala, por exemplo, um laboratório; a mesma já será alocada nesta sala pré-determinada. Caso contrário, serão reunidas as salas que possuem o menor número de alocações (por exemplo, as salas com menos alocações são 5 salas, e foram alocadas 1 única vez), e será realizado um sorteio entre elas, para que uma seja utilizada com esta aula.

Figura 17 - Exemplo de uma lista das disciplinas com a afinidade dos docentes. SD possui afinidade com os docentes 3, 4, 7; PSE com os docentes 3, 1; PDS com os docentes 7, 8; e LP com os docentes 5, 4, 9



Fonte: Acervo próprio

Após isso, novamente é percorrida a lista de disciplinas, distribuindo-as na grade de seus respectivos semestres, dentro de seus respectivos cursos. Como neste trabalho a alocação mínima de uma disciplina será de dois períodos; esta distribuição se dará com base nesta informação. Será feita uma divisão do número de créditos totais da disciplina (créditos

práticos somados aos créditos teóricos) pela alocação mínima (dois). Se a parte inteira do resultado desta divisão for um, então a disciplina era de dois ou três créditos e necessita ser alocada em apenas um *slot* de tempo. Se o resultado for 2, necessita de dois *slots* na grade do semestre (que equivalem a 4 horas/aula), ou se o resultado da divisão for 3, necessita de 3 *slots* (6 horas/aula). No momento em que se sabe quantos *slots* são necessários para alocação, eles são escolhidos aleatoriamente dentro da grade, e a disciplina é alocada nos mesmos. Isso é feito em todas as disciplinas da lista de disciplinas.

Durante a distribuição dos docentes nesta primeira população, é buscado sempre, em primeiro caso, não alocar o docente mais de uma vez no mesmo semestre, fazendo com que, por exemplo, ele venha a ministrar duas disciplinas no primeiro semestre. Porém, há casos em que isso não pode ser adequado, como se em um mesmo semestre duas disciplinas possuírem afinidade somente com um docente.

Todas as distribuições/combinações citadas acima são realizadas cada vez que um novo indivíduo é criado, e indivíduos são criados tantas vezes quanto é o tamanho da população pré-determinado na execução do AG. Ao final da criação de toda essa população inicial, temos o que se chama de primeira geração do AG. Pode-se afirmar que todos os indivíduos podem ser soluções válidas para o problema, pois todos possuem a quantidade de aulas corretas por disciplinas. O que vai nos dizer se os indivíduos são soluções boas ou ruins (colisões de salas, de docentes, preferências de horários de docentes atendidas, indisponibilidades de docentes atendidas, etc) é quando a função de avaliação for aplicada a toda a população inicial.

3.6.2 **Cálculo da avaliação dos indivíduos da população**

A próxima etapa do AG é o cálculo da avaliação de todos os indivíduos pertencentes à população. Neste momento, são levadas em consideração todas as restrições vinculadas aos docentes referentes aos dias e horários em que os mesmos não possuem disponibilidade de horário para ministrar aulas, ou as preferências que os mesmos possuem, colisões entre disciplinas alocadas numa mesma sala num mesmo horário, colisão entre disciplinas alocadas para um mesmo professor num mesmo período e fator de distribuição de carga horária entre os docentes. Colisões de salas, colisões de docentes, quebras de restrições em que o docente

não poderia ministrar aulas e o fator de distribuição da carga horária entre os docentes são penalizações que o indivíduo recebe durante a avaliação. Por disciplina alocada em suas grades, o indivíduo é bonificado com um valor fixo, e quando uma preferência de horário de um docente é atendida, da mesma forma o indivíduo também recebe um bônus que é somado ao valor de sua avaliação final. (Ver tabela 5).

Parâmetro	*Penalização/Bônus*	*Peso por indicidência*
Preferência de dia/horário atendida	Bônus	Ajustável
Alocação de uma disciplina	Bônus	Fixo
Quebra de restrição de horário docente	Penalização	Ajustável
Fator distribuição carga horária entre docentes	Penalização	Ajustável
Colisão de docentes/salas no mesmo horário	Penalização	Ajustável

Tabela 5 - Parâmetros da função de avaliação

A função de avaliação pode ser descrita por:

$$f(x) = \sum bônus - \sum penalizações \qquad \text{Equação (2)}$$

O cálculo do fator de distribuição de carga horária entre os docentes que participam da grade é o calculo do desvio padrão desta distribuição, e ele ocorre da seguinte forma:

$$Média\ Aritmética = \frac{Número\ total\ de\ créditos\ da\ grade\ horária}{Número\ de\ docentes} \qquad \text{Equação (3)}$$

$$Desvio\ padrão = \sqrt{\frac{\sum_{i=0}^{n} (x_n - Média\ Aritmética)^2}{n-1}} \qquad \text{Equação (4)}$$

Onde n é o número de docentes, e x_n é o número de disciplinas alocadas para o *n-ésimo* docente.

3.6.3 **Métodos de seleção**

No presente trabalho foram aplicados dois métodos simultâneos de seleção: o elitismo e o método da roleta viciada.

O método do elitismo é aplicado antes do método da roleta, visando com que um percentual dos indivíduos mais bem avaliados da população atual passem para a próxima geração diretamente, sem participarem de cruzamentos ou mutações. Isto nos assegura que, pelo menos, os indivíduos menos aptos da próxima geração terão no mínimo a mesma avaliação dos melhores indivíduos da geração anterior.

Após ser realizado o elitismo, o método da roleta é executado. Ele consiste em selecionar 2 pais entre os indivíduos da população (incluindo os que foram selecionados pelo método do elitismo) e realizar o cruzamento entre os mesmos. Visto que a chance de a roleta selecionar o mesmo indivíduo para determinado cruzamento certas vezes é muito alta, e um cruzamento deste tipo certamente seria perda de tempo pois geraria um indivíduo descendente igual aos seus pais (o que não é bom para a diversidade genética necessária para exploração do espaço de busca), nesta implementação o cruzamento somente ocorre quando os pais selecionados são indivíduos diferentes.

3.6.4 **Operador de cruzamento – *Crossover* uniforme**

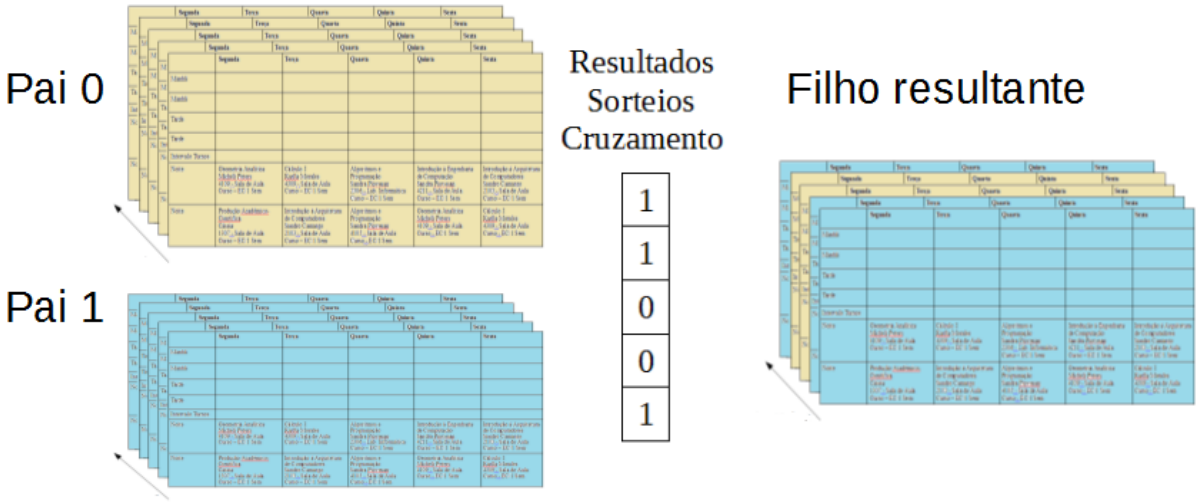
O tipo de cruzamento implementado é o *crossover* uniforme, pelo fato de o mesmo explorar melhor a diversidade genética do que os cruzamentos de um e dois pontos. Em primeira instância, ele seria aplicado dentro das matrizes, afetando diretamente as grades horárias dos semestres, porém implementado desta forma, ele mostrou-se extremamente destrutivo, ignorando os esquemas (por exemplo, número de aulas de determinada disciplina) das grades dos semestres, muitas vezes, duplicando o número de aulas de determinada disciplina, e retirando aulas de outras.

Em virtude do exposto, foi implementado o *crossover* uniforme entre as matrizes, pois o mesmo mantém os esquemas das grades, não interferindo na distribuição das disciplinas, sendo que as trocas de materiais genéticos acontecem carregando toda a matriz referente a determinado semestre. Este *crossover* pode ser observado na figura 18.

Após selecionados os pais, é sorteado uma primeira vez um número entre 0 (zero) e 1, se o número 0 for sorteado, a primeira grade do pai (matriz na posição zero da lista de matrizes do indivíduo) é selecionada para ser a primeira grade do filho. Se em vez de zero, o

número sorteado fosse 1, a primeira grade do filho seria originária da mãe. No segundo sorteio entre 0 e 1, da mesma forma é selecionada qual será a segunda grade horária que fará parte da lista de grades do filho (matriz na posição 1 na lista de matrizes do indivíduo), e assim por diante. O número de sorteios realizados é o mesmo número do tamanho da lista de matrizes dos pais, sempre seguindo o conceito de ao ser sorteado 0 (zero), a matriz que vai para o filho é do pai, e se for sorteado 1, a matriz que vai para o filho é da mãe.

Figura 18 - Crossover uniforme entre as matrizes dos indivíduos



Fonte: Acervo próprio

3.6.5 **Operador de Mutação**

O operador de mutação, como já mencionado anteriormente, é de extrema importância para o aumento da diversidade genética, principalmente devido ao tipo de cruzamento escolhido neste trabalho, que impossibilita o cruzamento de informações dentro das matrizes. A confecção do arranjo das disciplinas dentro das matrizes de determinado semestre é feito somente no momento da criação da primeira população. Através da mutação, a maneira como as disciplinas foram distribuídas desde a primeira população pode ser modificada, possibilitando, assim, atingir locais no espaço de busca que não poderiam ser visitados pelas grades originais da população inicial.

Este operador é aplicado a todos os indivíduos originados da população anterior que não foram selecionados para o *crossover*. Ele visita cada gene do indivíduo, em todas as

matrizes, e baseado na sua chance de mutação pré-determinada antes da execução do AG, como veremos mais a frente, ele troca ou não o gene de lugar com outro em uma posição diferente, escolhido de forma aleatória, porém na mesma matriz.

A chance de mutação é um dos parâmetros que devem ser informados no momento da geração da grade horária. Normalmente deve ser uma chance baixa, entre 0,5% e 1%. Se for parametrizado com um valor muito alto, muitas trocas entre disciplinas serão feitas, e o que pode ter sido construído de positivo ao longo das gerações pode ser destruído, tornando, desta forma, o AG em um *random walk* (como se o AG "passeasse" pelo espaço de soluções sem um caminho a seguir) ignorando a evolução que vinha sendo atingida.

A figura 19 demonstra a mutação em uma matriz de um indivíduo.

Figura 19 - Exemplo de mutação em grade horária

	Segunda	**Terça**	**Quarta**	**Quinta**	**Sexta**
Noite	Geometria Analitica Michéli Peters 4109 – Sala de Aula Curso – EC/1 Sem	Cálculo I Karlla Morales 4309 – Sala de Aula Curso – EC/1 Sem	Algoritmos e Programação Sandra Piovesan 2306 – Lab. Informática Curso – EC/1 Sem	Introdução à Engenharia de Computação Sandra Piovesan 4211 – Sala de Aula Curso – EC/1 Sem	Introdução à Arquitetura de Computadores Sandro Camargo 2103 – Sala de Aula Curso – EC/1 Sem
Noite	Produção Acadêmico Científica Cássia 1307 – Sala de Aula Curso – EC/1 Sem	Introdução à Arquitetura de Computadores Sandro Camargo 2103 – Sala de Aula Curso – EC/1 Sem	Algoritmos e Programação Sandra Piovesan 4105 – Sala de Aula Curso – EC/1 Sem	Geometria Analitica Michéli Peters 4109 – Sala de Aula Curso – EC/1 Sem	Cálculo I Karlla Morales 4309 – Sala de Aula Curso – EC/1 Sem

	Segunda	**Terça**	**Quarta**	**Quinta**	**Sexta**
Noite	Algoritmos e Programação Sandra Piovesan 4105 – Sala de Aula Curso – EC/1 Sem	Cálculo I Karlla Morales 4309 – Sala de Aula Curso – EC/1 Sem	Algoritmos e Programação Sandra Piovesan 2306 – Lab. Informática Curso – EC/1 Sem	Introdução à Engenharia de Computação Sandra Piovesan 4211 – Sala de Aula Curso – EC/1 Sem	Introdução à Arquitetura de Computadores Sandro Camargo 2103 – Sala de Aula Curso – EC/1 Sem
Noite	Produção Acadêmico Científica Cássia 1307 – Sala de Aula Curso – EC/1 Sem	Introdução à Arquitetura de Computadores Sandro Camargo 2103 - Sala de Aula Curso – EC/1 Sem	Geometria Analitica Michéli Peters 4109 – Sala de Aula Curso – EC/1 Sem	Geometria Analitica Michéli Peters 4109 – Sala de Aula Curso – EC/1 Sem	Cálculo I Karlla Morales 4309 – Sala de Aula Curso—EC/1 Sem

Fonte: Acervo próprio

3.6.6 **Parametrização do Algoritmo Genético**

Antes do início da execução do AG, alguns parâmetros devem ser escolhidos como, por exemplo, os cursos que deverão participar da geração da grade de horários, de forma que seja obtido uma boa solução para o problema em questão, que é a geração de uma grade que

contemple os cursos do campus Bagé da Unipampa.

Nesta etapa é parametrizado o tamanho das populações que farão parte do algoritmo, bem como o número máximo de gerações que o mesmo deve atingir. Deve-se parametrizar o percentual de elitismo, se for realizado com o valor 0 (zero), o elitismo é desconsiderado, tornando o método da roleta o único a ser aplicado à população. Se for parametrizado com um valor muito alto, ocorrerá uma rápida convergência genética e o algoritmo não passará por todo o espaço de busca, podendo, muitas vezes, ficar "preso" em algum máximo local.

A chance de mutação (parâmetro), como já visto, deve ser parametrizado com um valor pequeno, pois ela tem um alto poder de destruir esquemas montados. Porém, se for parametrizada com zero, as mutações nunca ocorrerão e arranjos entre as disciplinas de uma mesma matriz que não foram atendidos durante a criação da população inicial nunca serão alcançados. Através deste parâmetro é que serão verificadas célula a célula de todas as matrizes do indivíduo, onde poderão ocorrer as mutações.

A probabilidade de *crossover* (parâmetro) é o percentual de vezes que a ocorrência de cruzamentos pode vir a acontecer sobre uma população do AG, descontado os elementos do elitismo. Ligado a ele existe o parâmetro da probabilidade de mutação, sendo que a soma dos dois deve ser 100%. Se for parametrizado uma probabilidade de 80% de *crossover*, a probabilidade de mutação será 20%. Se um dos indivíduos for "contra" a probabilidade de *crossover*, quer dizer que este sofrerá uma mutação para poder gerar um novo indivíduo, sendo que cada um dos seus genes (cada uma de suas células de suas grades horárias) terá a chance de mutação igual ao valor já parametrizado (como já falado, entre 0,5% e 1%).

Também serão parametrizados os pesos que serão aplicados a cada colisão de sala e a cada colisão de docente. Da mesma forma, os pesos das preferências de horários por alocação feita que satisfaz as preferências dos docentes e por alocação que é feita em um período/dia em que o docente não pode ministrar aula.

3.6.7 **Condição de parada**

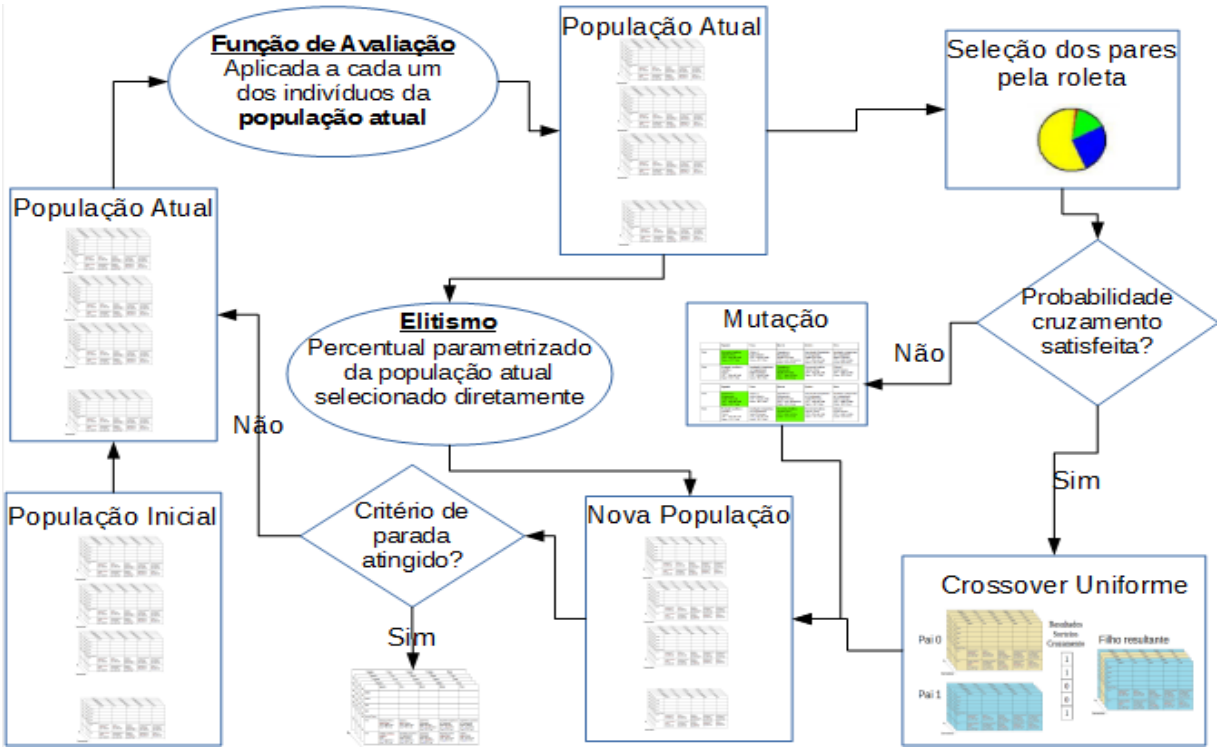
Na implementação deste trabalho a condição de parada ocorre quando o algoritmo atinge o número máximo de gerações que fora parametrizado antes de sua execução.

Atingido o critério para a parada do algoritmo, o indivíduo que possuir o valor de avaliação mais alto, na última geração, é considerado como resposta para o problema.

O conteúdo deste indivíduo, suas matrizes com seus genes, é exportado para um arquivo de extensão xls (planilha eletrônica padrão *Microsoft Office*) assim como o histórico da evolução das gerações. Já as informações da última geração presente no AG, parametrização geral do algoritmo, o tempo de execução, número de colisões de salas e de docentes, número de períodos em que docentes foram alocados mas que estavam marcados como indisponíveis, o número de alocações de docentes que conseguiram ser realizadas conforme a sua preferência são enviadas para um arquivo de texto. Da mesma forma, quando a execução o algoritmo gera uma solução com colisão de salas e docentes (que são os principais tipos de colisões e que devem ser evitados), elas ficam também listadas em um arquivo de texto.

Para melhor entendimento do fluxograma desta implementação de algoritmo genético, a figura 20 apresenta o mesmo, com todas as suas etapas descritas.

Figura 20 - Fluxograma do algoritmo genético implementado



Fonte: Acervo próprio

4 PROTÓTIPO IMPLEMENTADO, EXPERIMENTOS E RESULTADOS

Neste capítulo pode-se observar o protótipo implementado, experimentos realizados e resultados provenientes destes experimentos.

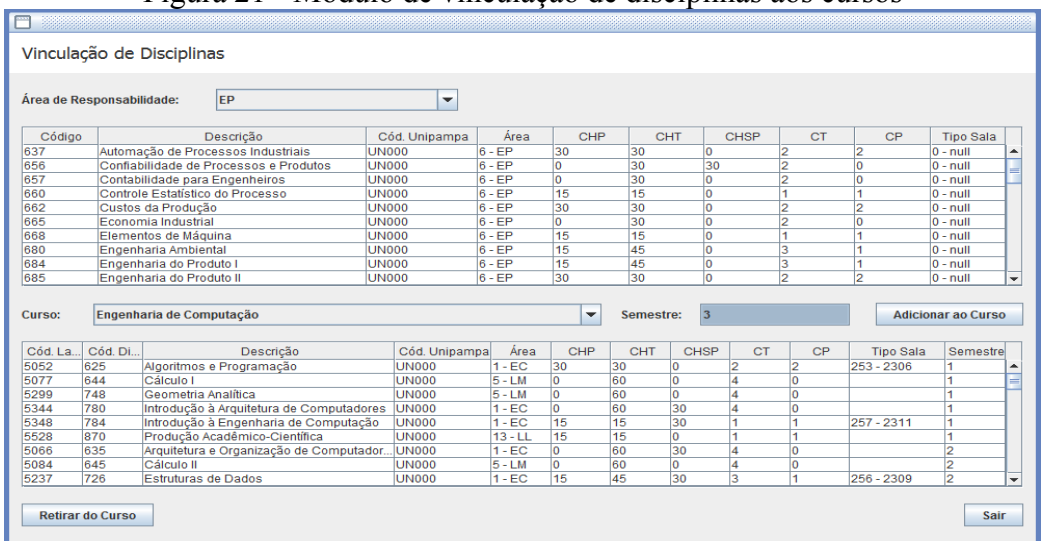
4.1 Protótipo Implementado

Para implementação do protótipo, foi utilizado o ambiente de desenvolvimento integrado NetBeans7.4 e o *widget toolkit Swing* como interface gráfica do protótipo com o usuário.

O protótipo possui um módulo de cadastro, onde são inseridas no banco de dados as informações referentes aos cursos, áreas de responsabilidade, docentes, disciplinas, salas e turnos. Também possui uma parte responsável pela vinculação das disciplinas aos cursos, vinculação dos docentes com as disciplinas aos quais possuem afinidade e cadastramento por parte dos docentes de quais os dias e horários os mesmos não possuem disponibilidade para ministrar aulas. Por último, o protótipo possui o módulo onde o AG é parametrizado e executado para construir a grade horária.

Abaixo, pode-se verificar algumas imagens dos principais módulos do protótipo.

Figura 21 - Módulo de vinculação de disciplinas aos cursos



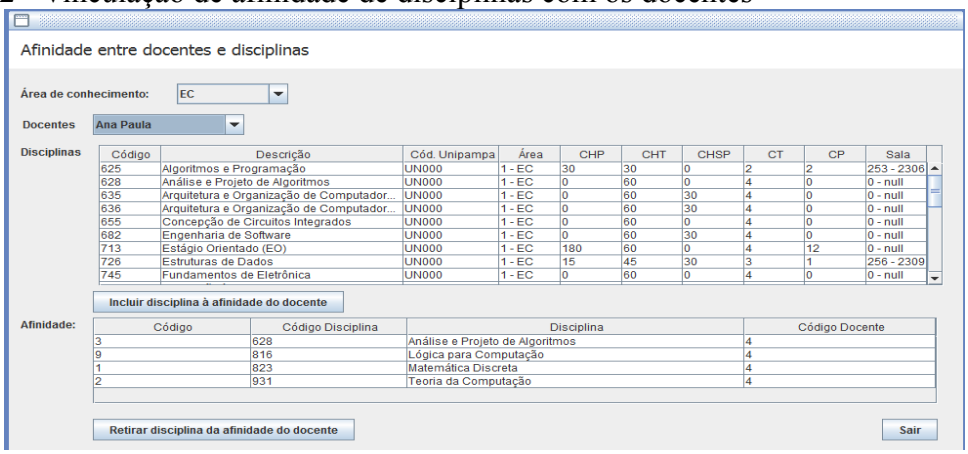
Código	Descrição	Cód. Unipampa	Área	CHP	CHT	CHSP	CT	CP	Tipo Sala
637	Automação de Processos Industriais	UN000	6 - EP	30	30	0	2	2	0 - null
656	Confiabilidade de Processos e Produtos	UN000	6 - EP	0	30	30	2	0	0 - null
657	Contabilidade para Engenheiros	UN000	6 - EP	0	30	0	2	0	0 - null
660	Controle Estatístico do Processo	UN000	6 - EP	15	15	0	1	1	0 - null
662	Custos da Produção	UN000	6 - EP	30	30	0	2	2	0 - null
665	Economia Industrial	UN000	6 - EP	0	30	0	2	0	0 - null
668	Elementos de Máquina	UN000	6 - EP	15	15	0	1	1	0 - null
680	Engenharia Ambiental	UN000	6 - EP	15	45	0	3	1	0 - null
684	Engenharia do Produto I	UN000	6 - EP	15	45	0	3	1	0 - null
685	Engenharia do Produto II	UN000	6 - EP	30	30	0	2	2	0 - null

Cód. La...	Cód. Di...	Descrição	Cód. Unipampa	Área	CHP	CHT	CHSP	CT	CP	Tipo Sala	Semestre
5052	625	Algoritmos e Programação	UN000	1 - EC	30	30	0	2	2	253 - 2306	1
5077	644	Cálculo I	UN000	5 - LM	0	60	0	4	0		1
5299	748	Geometria Analítica	UN000	5 - LM	0	60	0	4	0		1
5344	780	Introdução à Arquitetura de Computadores	UN000	1 - EC	0	60	30	4	0		1
5348	784	Introdução à Engenharia de Computação	UN000	1 - EC	15	15	30	1	1	257 - 2311	1
5528	870	Produção Acadêmico-Científica	UN000	13 - LL	15	15	0	1	1		1
5066	635	Arquitetura e Organização de Computador...	UN000	1 - EC	0	60	30	4	0		2
5084	645	Cálculo II	UN000	5 - LM	0	60	0	4	0		2
5237	726	Estruturas de Dados	UN000	1 - EC	15	45	30	3	1	256 - 2309	2

Fonte: Acervo próprio

A figura 21 mostra o módulo onde vincula-se disciplinas aos cursos. Primeiramente é selecionada a Área de responsabilidade/conhecimento de onde a disciplina que deseja vincular a um curso é proveniente. Após a lista das disciplinas da Área de responsabilidade ser apresentada, é selecionada a disciplina na tabela. Em seguida seleciona-se o curso ao qual deseja-se vincular a disciplina selecionada e escolhe-se o semestre ao qual ela faz parte no curso, assim é realizado um clique no botão “Adicionar ao curso” para que a vinculação seja efetivada.

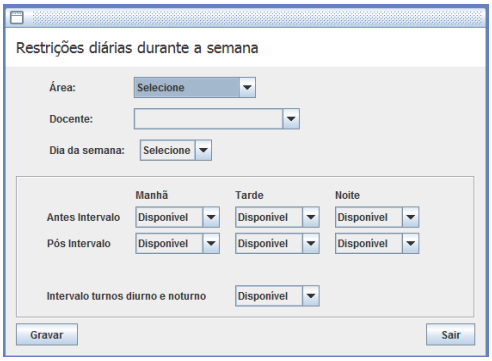
Figura 22 - Vinculação de afinidade de disciplinas com os docentes



Fonte: Acervo próprio

A figura 22 mostra o local onde é cadastrada a afinidade entre os docentes e as disciplinas que os mesmos podem ministrar. Primeiramente é selecionada a área de conhecimento ao qual o docente pertence, após é selecionada a disciplina que se deseja atribuir a afinidade e clica-se em “Incluir disciplina à afinidade do docente”. As disciplinas com as quais o docente já possui afinidade serão listadas na grade inferior.

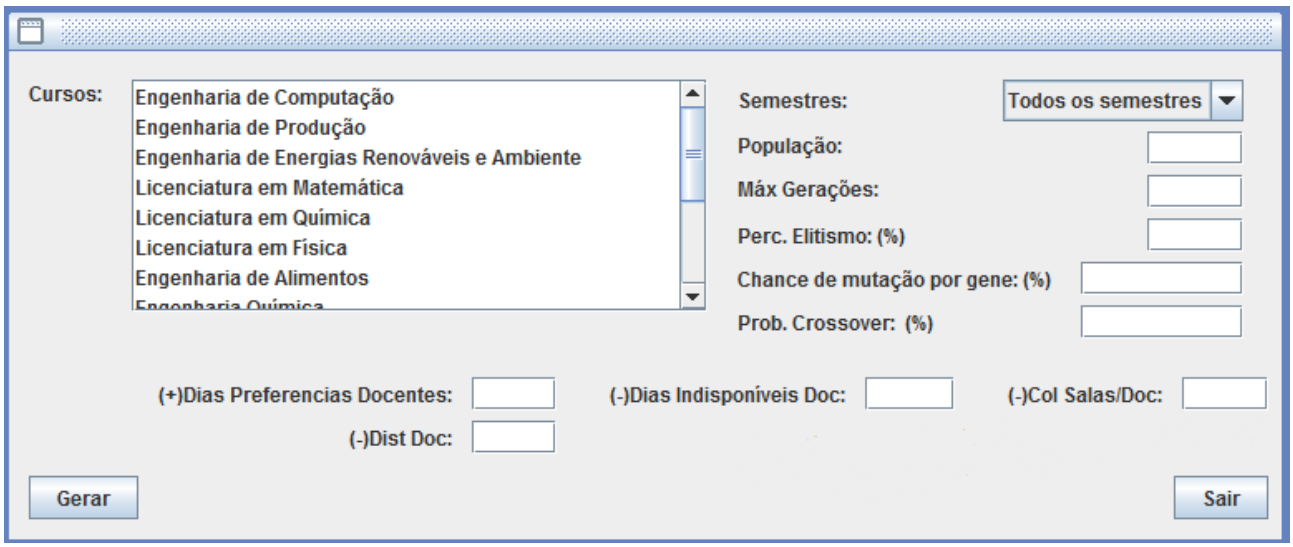
Figura 23 - Módulo de cadastro de restrições de docentes



Fonte: Acervo próprio

É possível observar na figura 23 a interface onde são cadastradas as restrições de horários dos docentes. Novamente é selecionada a área de conhecimento do docente, o docente, o dia da semana no qual o mesmo possui restrições de trabalho e são marcados os períodos referentes aos horários das restrições. Em seguida clica-se em “Gravar”.

Figura 24 - Etapa construtora da grade horária



Fonte: Acervo próprio

A parte mais importante e de maior ênfase deste trabalho é observada na figura 24. Nesta etapa é onde o AG é parametrizado e onde o mesmo é posto em execução. O curso ao qual deseja-se confeccionar a grade deve ser selecionado, assim como se a grade deve conter somente semestres pares, semestres ímpares, ou todos os semestres. A quantidade de indivíduos que compõem a população é informado no campo “População”, o número máximo de gerações que o AG deve percorrer deve ser informado em “Máx. Gerações”, o percentual de elitismo é percebido pelo campo “Perc. Elitismo”, o percentual da chance que cada um dos genes de um indivíduo podem sofrer mutação é informado em “Perc. Mutação”. A probabilidade de ocorrência de *crossover* é informado no campo “Prob. *Crossover*”.

Os próximos campos correspondem a parametrização da função de avaliação do AG. “Dias Preferenciais Docentes” corresponde ao peso do bônus que será dado ao indivíduo por alocação de docente que seja realizada em seu dia/horário de preferência. “Dias Indisponíveis Doc.” é o peso negativo quando o indivíduo possui docentes que foram alocados em períodos que foram marcados como indisponíveis para o mesmo. “Col. Salas/Doc.” é peso negativo

associado a cada colisão de salas e docentes que ocorrem nas grades de horário de um indivíduo da população e “Dist Doc” refere-se ao peso negativo que é multiplicado pelo valor resultante do desvio padrão da distribuição de carga horária entre os docentes. Respectivamente, estes são os campos do formulário do protótipo que deve ser preenchido antes da execução do AG em si, mostrados na figura 24.

4.2 **Experimentos e Resultados**

Os experimentos foram realizados em uma plataforma *Windows* 8.1, com processador Intel Core I7 e 8 Gb de memória RAM.

Após a realização de diversos testes, chegou-se configuração de parâmetros da função de avaliação para execução do AG conforme mostra a tabela 6. Durante estes testes, verificou-se que a parametrização não pode sofrer amortizações, pois dessa forma algumas das penalizações podem ser suprimidas pelas bonificações, fazendo com que o algoritmo não retorne a solução esperada.

Parêmetro	*Peso atribuído*
Penalização por alocação de docente em período indisponível	120
Penalização sobre o cálculo do desvio padrão na distribuição de disciplinas	2000
Penalização por ocorrência de colisão entre salas e horários dos professores	700
Bônus por alocação de docente em período preferencial	30

Tabela 6 - Tabela com ajustes dos parâmetros da função de avaliação.

Foram importados da planilha de horários do 1º semestre do ano de 2014 do campus Bagé da Unipampa, disponível em seu sítio da *internet*, os dados para simulação das preferências dos dias e períodos que os docentes possuem para ministrar suas aulas. Isto foi baseado nas alocações contantes na referida planilha, de forma com que se um docente foi alocado, por exemplo, na segunda-feira, nos dois últimos períodos da noite e na terça-feira, nos dois primeiros períodos da tarde, para efeito de simulação, estes são os seus períodos preferenciais. Contabilizou-se um total de 638 períodos preferenciais e foram adicionados aleatoriamente 24 períodos divididos entre 4 docentes, em que os mesmos estariam indisponíveis para ministrar suas disciplinas.

O quadro abaixo mostra o resumo de uma das melhores execuções do protótipo:

```
+--Resultado AG--+
Tamanho população: 1000
Máximo de gerações: 3000
Percentual Elitismo: 10
Chance de mutação: 0.5%
Probabilidade de Crossover: 65%
Avaliação melhor indivíduo: 30709
Total de colisões de salas: 0
Total de colisões de docentes: 0
Total de Disciplinas Alocadas: 248
Total de Períodos Alocados: 429
Desvio padrão na distribuição de disciplinas por docente: 1.9106173650410312
Total de quebras de restrições de docentes em horários indisponíveis: 0
Total de alocações em horários preferenciais de docentes: 293
Tempo Total: 9402.4 segundos (~ 2 h e 30 min)
```

Observa-se que do total de períodos alocados, 68,3% foram em horários que os docentes possuíam preferências por ministrarem suas aulas. Os outros 31,7% eram indiferentes quanto aos períodos destinados a eles. Todos os períodos em que os docentes encontravam-se indisponíveis foram preservados.

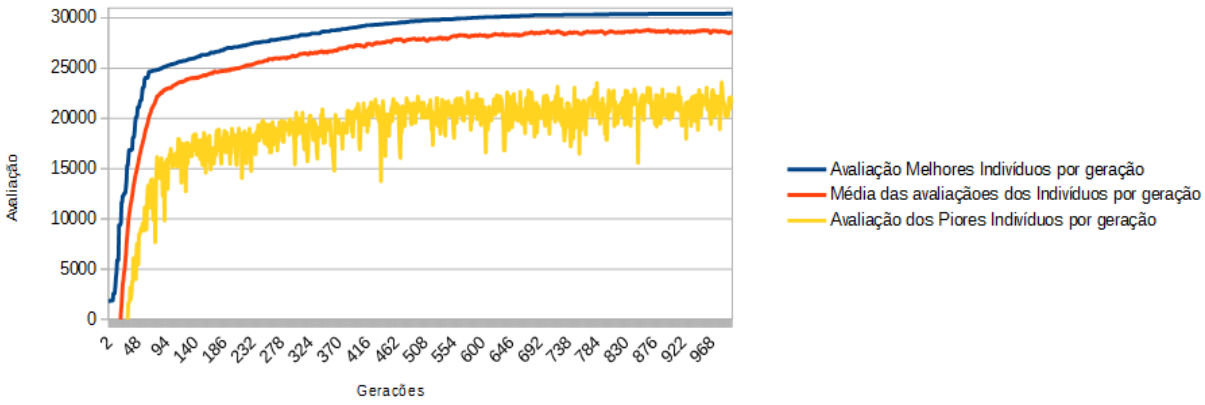
Restrições importantes que podem invalidar a aplicação de uma grade de horários, como a alocação de uma sala/laboratório para duas disciplinas diferentes concomitantemente, e um professor ser alocado para ministrar aulas em duas ou mais salas ao mesmo tempo, foram satisfeitas.

Os testes foram realizados gerando as grades horárias para os semestres ímpares dos cursos de Engenharia da Computação, Engenharia de Produção, Engenharia de Energias Renováveis e Ambiente, Licenciatura em Matemática, Licenciatura em Química, Licenciatura em Física, Engenharia de Alimentos, Engenharia Química e Licenciatura em Letras – línguas Adicionais, todos do campus Bagé da Unipampa.

A curva que mostra a evolução das aptidões dos indivíduos com o passar das gerações pode ser observada na figura 25. Nota-se o rápido crescimento da curva, isso acontece por causa das colisões de salas e docentes, que por possuírem o 2º maior valor na parametrização do AG, são tratadas primeiramente até serem sanadas. Posteriormente pode-se observar um declínio na taxa crescimento dessa curva, o que nos leva a perceber que o algoritmo faz com que passem a serem exploradas as soluções que, após não possuírem mais

colisões entre salas e docente, coíbem as alocações em que professores estariam indisponíveis para ministrarem suas aulas. Em seguida, busca as soluções que contemplem também suas alocações preferenciais, que é o parâmetro com menor peso. Isto segue até o final de sua execução.

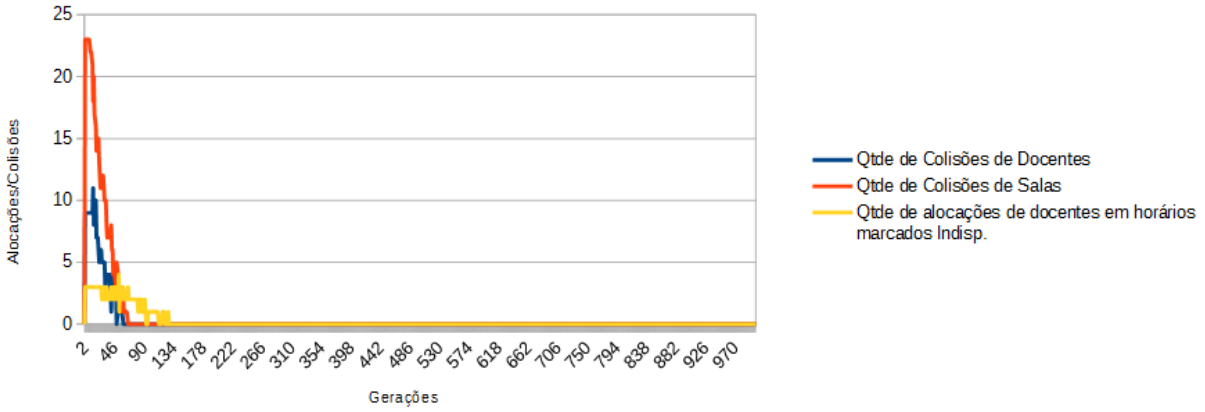
Figura 25 - Gráfico com a evolução das avaliações do indivíduos do AG, limitado a mil gerações



Fonte: Acervo próprio

Para confrontar esta informação, pode-se observar o gráfico da figura 26, onde nota-se que tanto a curva das colisões de salas, quanto a curva das colisões de docentes decaem de forma tão rápida quanto a curva da avaliação dos melhores indivíduos cresce na figura 25. Sendo que ainda se observa na figura 26 a queda também acentuada do número de alocações de docentes em períodos em que os mesmos estão indisponíveis.

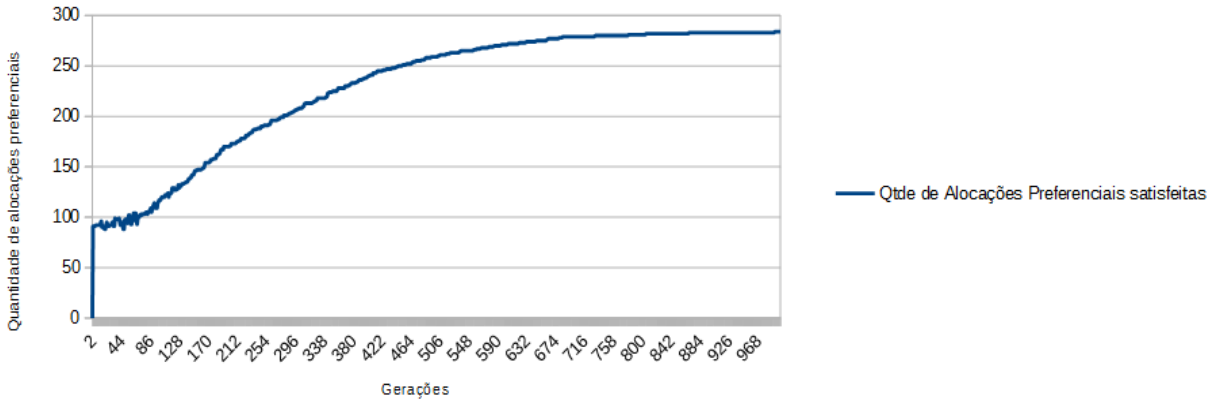
Figura 26 - Gráfico da quantidade de alocações de salas/docentes com colisão, e quantidade de alocação de docentes em períodos marcados como indisponíveis, limitado a mil gerações



Fonte: Acervo próprio

Quando a curva do melhor indivíduo da figura 25 começa a diminuir sua taxa de crescimento, acontece a busca do AG em alocar os docentes que ainda não estão seus em seus horários preferenciais. Isto pode ser confirmado pela figura 27, que é gráfico que mostra o crescimento do número de alocações dos docentes em horários de sua preferência. Esta figura ainda mostra que no início da execução do algoritmo, algumas gerações são passadas sem que o número da quantidade de alocações aumente de forma substancial. Este crescimento ocorre logo após este pequeno período e prossegue até o final da execução do algoritmo.

Figura 27 - Gráfico da quantidade de alocações preferenciais satisfeitas, limitado a mil gerações



Fonte: Acervo próprio

5 CONCLUSÕES E TRABALHOS FUTUROS

A técnica de exploração de espaço de busca mostrou-se funcional em relação ao problema proposto, sendo que grades horárias aplicáveis foram geradas com sucesso. Grades que não poderiam ser aplicadas (com colisões de salas e horários de docentes) também foram geradas, porém somente ocorreram casos isolados, com populações pequenas, que limitavam muito o espaço de busca do AG.

Pode-se observar que tanto o tamanho da população quanto o número máximo de gerações foram itens impactantes no desempenho do algoritmo, pois o tempo de execução está diretamente relacionado com estes, principalmente por causa do número de vezes com que as funções de cruzamento, mutação e avaliação são executadas.

Todavia, quanto maior o tamanho das populações, maior ser torna o espaço de busca e a diversidade genética. Na maioria dos testes, um espaço de buscas grande encontrou soluções bem mais adequadas do que com populações pequenas. Algumas simulações com populações de 300 indivíduos e número máximo de 300 gerações apontaram para grades de horários que não possuíam colisões de docentes/salas e nem ocorriam quebras de períodos em que os docentes estariam indisponíveis, porém poucos períodos preferenciais foram alocados. Isto mostra que a partir de um certo número de gerações o algoritmo busca algum tipo de ajuste fino para a sua solução, como pôde ser observado nas figuras 25, 26 e 27.

Pode-se ressaltar que a distribuição da carga horária em muitos testes pareceu prejudicada, por casos de disciplinas que acabaram sendo alocadas para os “Professores Substitutos”. Como as informações de disciplinas, horários preferenciais e docentes foram importados de planilhas com os horários dos semestres passados e do semestre atual, sempre que nelas apareciam a informação de que o docente era um professor substituto, o protótipo lia essa informação como se esse docente fosse um docente efetivo do quadro da universidade, e atribuía-lhe afinidades para essas disciplinas, o que muitas vezes pode ter sido motivo de o mesmo ter sido escolhido para ministrar várias disciplinas, pois acabou possuindo várias afinidades.

Como trabalho futuro, visa-se abranger elementos como a ideia de alocação e divisão dos alunos em turmas, que não foi o foco deste trabalho. Essa divisão, possibilitará a alocação de turmas de diferentes cursos numa mesma sala quando a disciplina entre ambos for igual, e

isto ajudaria tanto na alocação de salas, bem como, principalmente na melhor distribuição da carga horária entre os docentes.

Duas formas de distribuição de disciplinas que deveriam ser coibidas pela implementação é a ocorrência de 4 créditos de uma mesma disciplina em turnos diferentes do dia (por exemplo, uma disciplina for alocada com 2 créditos pela manhã e 2 créditos pela tarde), e também a alocação de 8 créditos num mesmo dia, quando durante a semana podem aparecer dias com apenas 2 ou 4 créditos alocados.

Assim como existem restrições de alocação para docentes, com seus dias preferenciais e indisponíveis, poderia ocorrer algo semelhante em relação às disciplinas, podendo haver restrições deste tipo a serem configuradas no protótipo, como por exemplo a disciplina de Desenho Técnico ser preferencialmente oferecida nos quatro períodos da manhã.

Pode-se vislumbrar a execução deste algoritmo de modo a explorar o potencial de arquiteturas paralelas, como comentado durante o texto, pois como visto, várias atividades são realizadas de modo que não dependem de outras funções do sistema, como a função de avaliação, que em uma população de 3000 indivíduos, é executada 3000 vezes somente para uma única geração. Certamente uma solução que trabalhe de forma paralela obterá uma considerável otimização no tempo de execução do algoritmo.

E ainda a implementação de uma função que consiga mapear melhor o andamento da execução e AG, e fazer com que o critério de parada possa ocorrer antes do número de gerações máximo ser alcançado, se de alguma forma for elucidado que os indivíduos não possuem mais chance de desenvolverem-se, poupando desta forma, ciclos de execução.

6 REFERÊNCIAS

[1] CAMARGO, L. de M., et al. **ALOCAÇÃO DE PROFESSORES USANDO ESCALONADOR DE TIMETABLING SWICTBOARD**. V MOSTRA DE INICIAÇÃO CIENTÍFICA E IV MOSTRA DE PESQUISA DE PÓSGRADUAÇÃO DA IMED. Passo Fundo, 2011.

[2] CHERZE, C. Freitas, et al. **Uma Ferramenta Baseada em Algoritmos Genéticos para Geração de Tabela de Horário Escolar**. 1Faculdade Ruy Barbosa (FRB) – Salvador.

[3] CORREIA, Rodrigo. **SISTEMA GERADOR DE GRADE HORÁRIA DE PROFESSORES USANDO ALGORITMOS GENÉTICOS**. Trabalho de Conclusão de Curso (Bacharel em Ciências da Computação), UNIVERSIDADE REGIONAL DE BLUMENAU, Blumenau, 2013.

[4] DA CUNHA, Frederico Almeida. **O ALGORITMO GENÉTICO E A PROBLEMÁTICA DA ELABORAÇÃO DE GRADES DE HORÁRIOS**. (Projeto de Iniciação Científica) Faculdades Integradas Simonsen. Rio de Janeiro, 2012.

[5] DE OLIVEIRA, Osmar Braz Júnior. **Otimização de Horários em Instituições de Horários em Instituições de Ensino Superior Através de Algoritmos Genéticos**. Dissertação (Mestrado em Engenharia de Produção), Universidade Federal de Santa Catarina, Florianópolis, 2000.

[6] FERREIRA, Hélio de L. J.; CORRÊA, Marcos V. **Implementação de uma Sistema de Alocação de Professores e Disciplinas em Grades Horárias Utilizando Algoritmos Genéticos**. Trabalho de Conclusão de Curso (Bacharel em Ciências da Computação), Universidade do Anhembi Morumbi, São Paulo, 2010

[7] FREITAS, C. C. **Uma Ferramenta Baseada em Algoritmos Genéticos para a Geração de Tabela de Horário Escolar**, Sétima Escola Regional de Computação Bahia-Sergipe, 2007.

[8] FUCILINI, Tiago P.; MARUANI, Eli; REBONATTO Marcelo T. ***Timetabling*** **com algoritmos genéticos: resultados, restrições e exploração do paralelismo**. Revista Hífen, v. 32, nº 62, Uruguaiana, 2008.

[9] GAMBINI, Haroldo Santos, **Formulações e Algoritmos para o Problema de**

Programação de Horários em Escolas. Tese (Doutorado em Otimização Combinatória e Inteligência Artificial), Universidade Federal Fluminense, Niterói, 2007.

[10] GERVÁSIO, Jairo de F. **Computação Evolutiva Aplicada ao Problema de Grade Horária: o Caso do Curso de Análise e Desenvolvimento de Sistemas do IFTM**. Dissertação (Mestrado em Ciências), Uberlândia, 2012.

[11] GIORDANO, Gilberto Filho. **SOLUÇÃO HÍBRIDA DE *TIMETABLING* APLICADA AO ESCALONAMENTO DE VEÍCULOS DE TRANSPORTE PÚBLICO**. Monografia (Bacharel em Ciência da Computação), Centro Universitário La Salle, Canoas, 2008.

[12] HEINEN, Milton Roberto; SANTOS, Fernando Osório. **Algoritmos Genéticos aplicados ao Problema de Roteamento de Veículos**, HÍFEN,Vol. 30, No 58, 2006.

[13] LINDEN, R. **Algoritmos Genéticos. 3ª Edição**. Rio de Janeiro: Editora Ciência Moderna Ltda., 2012.

[14] LOBO, Eduardo Luis Miranda. **UMA SOLUÇÃO DO PROBLEMA DE HORÁRIO ESCOLAR VIA ALGORITMO GENÉTICO PARALELO**. Dissertação (Mestrado em Modelagem Matemática e Computacional), Centro Federal de Educação Tecnológica de Minas Gerais, Belo Horizonte, 2005.

[15] OLIVEIRA, Eder Abensur; CAVALCANTE, Rafael Oliveira. **UM MÉTODO HEURÍSTICO CONSTRUTIVO PARA O PROBLEMA DA GRADE HORÁRIA ESCOLAR**. Revista Eletrônica Operacional para o Desenvolvimento, v.4, n.2, Rio de Janeiro, 2012.

[16] PINTO, A.R.; DANTAS, M.A.R.. **Servidor Genético: Uma abordagem de balanceamento de carga baseada em algoritmo de aprendizado de máquina genético para agregados de computadores**. Universidade Federal de Santa Catarina

[17] SIMONETTI, Geraldo Bello. **Abordagem do Problema de Programação de Grade Horária Sujeito a Restrições Utilizando Coloração de Grafos**. Dissertação (Mestrado em Informática), Universidade Federal do Espírito Santo, Vitória, 2007.

[18] TEIXEIRA, Anderson Goés. **OTIMIZAÇÃO NA DISTRIBUIÇÃO DA CARGA HORÁRIA DE PROFESSORES – MÉTODO EXATO, MÉTODO HEURÍSTICO, MÉTODO MISTO E INTERFACE** –. Dissertação (Mestrado em Ciências), Universidade

Federal do Paraná, Curitiba, 2005.

[19] VIEIRA, F.; MACEDO, H. **Sistema de Alocação de Horários de Cursos Universitários: Um Estudo de Caso no Departamento de Computação da Universidade Federal de Sergipe**, SCIENTIA PLENA VOL. 7, NUM. 3, 2011.