

## SUMÁRIO

<b>1</b>	<b>INTRODUÇÃO</b>	<b>14</b>
1.1	Contextualização	14
1.2	Objetivos	14
1.2.1	Objetivos Gerais	15
1.2.2	Objetivos Específicos	15
1.3	Justificativa	16
1.4	Organização do Trabalho	16
<b>2</b>	<b>REFERENCIAL TEÓRICO</b>	<b>17</b>
2.1	Problemas de Otimização	17
2.2	Heurística	18
2.2.1	Busca Tabu	19
2.2.2	Recozimento Simulado	21
2.2.3	Algoritmos genéticos	22
2.3	TimeTabling	23
2.4	Trabalhos Relacionados	25
<b>3</b>	<b>METODOLOGIA</b>	<b>26</b>
3.1	Ferramentas Utilizadas	27
3.1.1	Sistema de Gerenciamento de Banco de Dados	28
3.1.2	Ferramentas Back-end	28
3.1.3	Ferramentas Front-end	29
3.1.4	IDE	30
<b>4</b>	<b>SISTEMA DESENVOLVIDO</b>	<b>31</b>
4.1	Modelagem	31
4.1.1	Diagramas de caso de uso	31
4.1.2	Diagrama de Entidade Relacionamento	32
4.1.3	Diagramas de classe	33
4.2	Algoritmo Genético	34
4.2.1	Indivíduo	35
4.2.2	População	36
4.2.3	Operadores genéticos	37
4.2.4	Definição da função objetivo	39
4.2.5	Fluxo do algoritmo	41
<b>5</b>	<b>RESULTADOS OBTIDOS</b>	<b>43</b>
<b>6</b>	<b>CONSIDERAÇÕES FINAIS</b>	<b>44</b>
6.1	Trabalhos futuros	44

<b>REFERÊNCIAS . . . . .</b>	<b>45</b>
------------------------------	-----------

# **1 INTRODUÇÃO**

## **1.1 Contextualização**

explicar o problema passo a passo

falar das restrições

O trabalho consiste na distribuição das disciplinas dos diversos cursos de graduação e pós-graduação apresentados pelos colegiados, em salas, através do sistema que será criado todo início de semestre assim que todos os colegiados já tenham enviado suas necessidades para aquele semestre.

Para realização da alocação, faz-se necessário o conhecimento das salas existentes para atender a demanda, além de conhecer sua capacidade e suas características. A partir deste passo, faz-se necessário o levantamento das solicitações enviadas pelos colegiados, para alocar as disciplinas em salas adequadas de forma a atender todas as solicitações de forma otimizada.

O principal problema atualmente é a falta de salas adequadas para a distribuição das disciplinas, uma vez que as turmas atuais estão com um número de alunos matriculados acima da capacidade das salas, devido a este problema o sistema deve propor um relatório de disciplinas que não atenderam a capacidade do prédio para que o responsável pela alocação possa negociar em prédios de outros cursos a alocação das disciplinas que não poderão ser alocadas no prédio que o sistema executou a alocação.

## **1.2 Objetivos**

O tratamento do problema de geração de horários em escolas carece de bons trabalhos na literatura. Apesar de se encontrar ferramentas disponíveis, poucas tratam de maneira eficiente as restrições reais existentes em escolas. Com este trabalho objetiva-se:

### *1.2.1 Objetivos Gerais*

O capítulo a seguir descreve o sistema desenvolvido neste trabalho, que tem como objetivo otimizar a alocação de disciplinas nas específicas salas do prédio de uma universidade, neste caso o prédio da FAFICH UFMG, diretamente facilitando a vida do gerente. Por se tratar de um problema específico do local fica difícil encontrar tecnologias disponíveis para otimização do problema, neste o desenvolvimento de um sistema que atenda todas as necessidades exigidas é de grande valia para o responsável pela alocação.

Este trabalho envolve conhecimentos de análise de sistemas e desenvolvimento almejando um sistema capaz tratar e otimizar a execução do problema de alocação de salas de uma universidade. Este trabalho possui um grande valor uma vez que pode facilitar a vida do responsável pela alocação das salas, por se tratar de um trabalho manual, trabalhoso e que para a execução são necessárias em torno de trinta horas que poderiam estar sendo utilizadas para uma tarefa mais importante.

### *1.2.2 Objetivos Específicos*

Objetivos específicos:

- Desenvolvimento do sistema.
- Apresentação de um modelo matemático acerca do problema;
- Implementação de um algoritmo que solucione o modelo apresentado considerando as restrições mais comuns para a geração de um horário de qualidade.
- Otimizar o tempo do gestor.
- Eficiência na geração dos relatórios.

### **1.3 Justificativa**

### **1.4 Organização do Trabalho**

Este trabalho está definido da seguinte forma, foi dividido em seis capítulos, sendo este capítulo 1 e mais cinco outros.

O capítulo 2 apresenta o referencial teórico do trabalho, descrevendo os conceitos utilizados para o desenvolvimento do projeto proposto: conceitos de TimeTable e Heurísticas.

No capítulo 3 é apresentada a metodologia do sistema e as tecnologias adotadas para desenvolvimento da solução.

No capítulo 4 iremos descrever e citar detalhadamente as características e propostas de desenvolvimento do sistema desenvolvido, proposto para este trabalho.

A conclusão deste trabalho e considerações finais são mostrados nos capítulos 5 e 6.

Se tiver anexo explicar cada anexo.

## 2 REFERENCIAL TEÓRICO

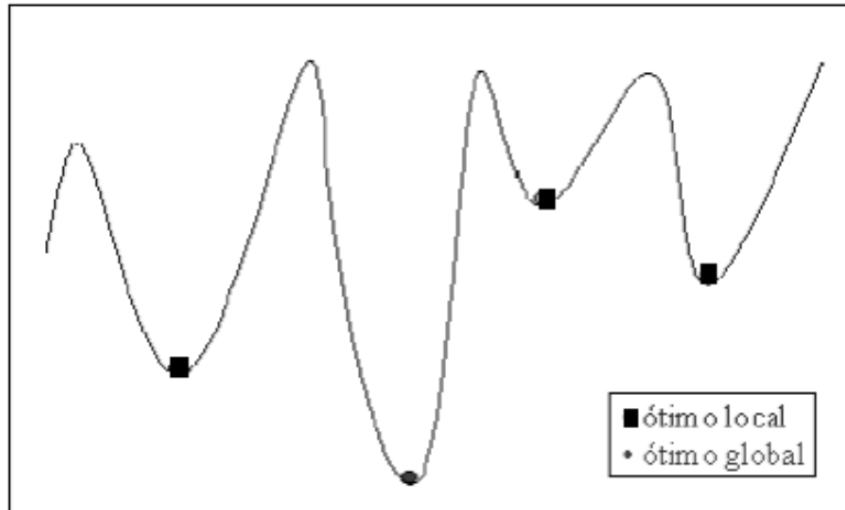
### 2.1 Problemas de Otimização

Otimização é o processo de encontrar a melhor solução, também chamada de solução ótima para um determinado problema (TIMÓTEO, 2005).

De acordo com (STEIGLITZ; PAPADIMITRIOU, 1982) a constituição de um problema de otimização se deve aos termos vizinhança, ótimo local e ótimo global. O termo vizinhança se trata de um subconjunto do conjunto do problema, ótimo local pode ser tratado como o melhor resultado em uma vizinhança, e ótimo global é a melhor solução encontrada no conjunto de acordo com a função objetivo.

De acordo com a figura 1 pode se observar a relação entre ótimo local e ótimo global em um problema típico de minimização.

**Figura 1 – Representação de um problema de minimização com ótimos locais**



Fonte: (TIMÓTEO, 2005)

Segundo (STEIGLITZ; PAPADIMITRIOU, 1982), os problemas de otimização são divididos em duas categorias, problemas com variáveis contínuas e problemas com variáveis discretas. Problemas com variáveis discretas também podem ser conhecidos como Problemas de Otimização Combinatória (POC).

Conforme cita (RAUPP, 2003), o problema de otimização combinatória pode ser denominado como a ação de maximizar ou minimizar uma função objetiva de diversas variáveis sujeita

a um conjunto de restrições, dentro de um contexto.

De acordo com (RAO, 1984) problemas do tipo POC tratam do estudo matemático para encontrar agrupamentos, arranjos ou a seleção ótima de objetos discretos, logo, não permitindo, a utilização de métodos clássicos de otimização contínua para sua resolução.

Segundo (GOLBARG; LUNA, 2000) a ocorrência de problemas de otimização combinatória podem acontecer em diversas áreas, projetos de sistemas de distribuição de energia elétrica, posicionamento de satélites, roteamento ou escalonamento de veículos, sequenciamento de genes e DNA, classificação de plantas e animais.

De acordo com (CISCON, 2006) em problemas de otimização combinatória, cujo universo de dados é grande e existe um grande número de combinações, o que torna inviável a análise de todas soluções possíveis em um tempo adequado, utilizamos as heurísticas, também conhecidas como algoritmos heurísticos, que são métodos que compõem uma gama de soluções para problemas de otimização combinatória.

## 2.2 Heurística

O termo heurística é derivado do grego *heuriskein*, o que significa descobrir ou achar. De acordo com (TIMÓTEO, 2005) o significado da palavra em pesquisa operacional vai um pouco além da raiz etimológica. Segundo (STEIGLITZ; PAPADIMITRIOU, 1982), heurísticas são consideradas métodos de aproximação ou métodos de busca de solução, deve se levar em consideração que não exista uma garantia formal de seu desempenho e uma garantia de que estas heurísticas que iram encontrar uma solução. As heurísticas, apesar de não garantirem encontrar a solução ótima para um problema, procuram por soluções consideradas de boa qualidade em um tempo computacional razoável.

Segundo (EVANS; MINIEKA, 1992) heurísticas são necessárias para implementação de problemas NP Difícil, caso deseje-se resolver tais problemas em um tempo razoável.

Ressalta-se que dentre as heurísticas, as chamadas meta-heurísticas, merecem especial atenção pois adotam técnicas para amenizar, a dificuldade que os métodos heurísticos têm de escapar dos ótimos locais. As meta-heurísticas podem partir em busca de regiões mais promissoras no espaço de soluções, além disto, as meta-heurísticas possuem grande abrangência, podendo ser aplicada à maioria dos problemas de otimização combinatória.(NASCIMENTO; SILVA;

ALVARENGA, 2005)

Segundo (OLIVEIRA, 2006) uma heurística é a instanciação de uma meta-heurística, ou seja, a aplicação da mesma em um problema específico de otimização.

Como exemplos de meta-heurísticas temos Busca Tabu (*Tabu Search*), Otimização por Colônias de Formigas (*Ant Colony Optimization*), Recozimento Simulado (*Simulated Annealing*) e Algoritmo Genético (*Genetic Algorithm*).

### 2.2.1 Busca Tabu

Busca tabu (BT) é uma meta-heurística adaptativa, que utiliza uma estrutura de memória através de uma lista, contendo um histórico de evolução para evitar que o processo de busca forme ciclos, ou seja, o retorno a um ótimo local previamente visitado (SOUZA, 2000), (ARMENTANO; BRANCHINI, 2013) e (SUBRAMANIAN et al., 2006).

Segundo (SUBRAMANIAN et al., 2006) BT foi desenvolvida por (GLOVER, 1986) com o objetivo de encontrar soluções para problemas de programação linear. Ao formalizar a técnica, o autor publicou uma serie de trabalhos envolvendo diversas aplicações da meta-heurística.

Basicamente o funcionamento do BT a feito partir da definição de uma população inicial  $S_0$ , o algoritmo explora cada iteração, de um subconjunto  $V$  da vizinhança  $N(S)$  da solução corrente  $S$ . O membro  $S'$  de  $V$  com melhor valor nessa região segundo a função  $f(.)$  torna-se a nova solução corrente mesmo que  $S'$  seja pior que  $S$  isto é, que  $f(S') > f(S)$  para um problema de minimização(SOUZA, 2000). A figura 2 representa o pseudocódigo do algoritmo da Busca Tabu.

Segundo (ARMENTANO; BRANCHINI, 2013) o algoritmo tem um intensivo uso de memória o que é uma característica essencial deste método. Para o autor o uso da memória pode ajudar a intensificar a busca em regiões com grande chances de se encontrar o resultado, ou até mesmo, diversificar a busca através de regiões inexploradas.

De acordo com (ARMENTANO; BRANCHINI, 2013) devemos adotar alguns procedimentos para que o processo de busca tenha um melhor resultado, listas tabu dinâmicas, passagens por regiões planas, intensificação, diversificação, *path relinking*. Listas tabu dinâmicas tem como objetivo evitar que o algoritmo entre em processo de ciclo. Passagens por regiões planas pode levar o algoritmo a pensar que não existem melhoras significativas na qualidade das soluções



**Figura 2 – Representação do pseudocódigo do algoritmo da Busca Tabu**

```

procedimento  $BT(f(\cdot), N(\cdot), A(\cdot), |V|, f_{min}, |T|, BTmax, s)$ 
1   $s^* \leftarrow s;$            {Melhor solução obtida até então}
2   $Iter \leftarrow 0;$        {Contador do número de iterações}
3   $MelhorIter \leftarrow 0;$  {Iteração mais recente que forneceu  $s^*$ }
4   $T \leftarrow \emptyset;$    {Lista Tabu}
5  Inicialize a função de aspiração  $A$ ;
6  enquanto  $(f(s) > f_{min} \text{ e } Iter - MelhorIter < BTmax)$  faça
7     $Iter \leftarrow Iter + 1;$ 
8    Seja  $s' \leftarrow s \oplus m$  o melhor elemento de  $V \subset N(s)$  tal que
      o movimento  $m$  não seja tabu ( $m \notin T$ ) ou
       $s'$  atenda a condição de aspiração ( $f(s') < A(f(s))$ );
9     $T \leftarrow T - \{\text{movimento mais antigo}\} + \{\text{movimento que gerou } s'\};$ 
10   Atualize a função de aspiração  $A$ ;
11    $s \leftarrow s';$ 
12   se  $(f(s) < f(s^*))$  então
13      $s^* \leftarrow s;$ 
14      $MelhorIter \leftarrow Iter;$ 
15   fim-se;
16 fim-enquanto;
17  $s \leftarrow s^*;$ 
18 Retorne  $s$ ;
fim  $BT$ ;

```

**Fonte:** (SOUZA, 2000)

e atingir o critério de parada, para evitar esta situação é necessário aumentar o tamanho da lista tabu enquanto o algoritmo estiver passando pela região plana e voltar a reduzir quando houver mudança no valor da função objetiva. Intensificação são técnicas utilizadas para concentrar os esforços da pesquisa em regiões consideradas promissoras. Diversificação é uma técnica que utiliza memória de longo prazo para redirecionar a pesquisa para regiões que ainda não foram suficientemente exploradas. PathRelinking trata a intensificação de incorporar atributos de soluções de boa qualidade (chamadas de soluções elite), em seguida explora caminhos que contenham uma ou mais soluções de elite.

Ainda segundo (ARMENTANO; BRANCHINI, 2013) uma característica importante do método é que a solução final tem pouca ou nenhuma dependência da escolha feita para a solução inicial, isso graças aos mecanismos implementados pelo método, que fogem de ótimos locais.

### 2.2.2 Recozimento Simulado

Técnica de busca local probabilística, proposta originalmente por (KIRKPATRICK; JR.; VECCHI, 1983), que se fundamenta em uma analogia com a termodinâmica, ao simular o resfriamento de um conjunto de átomos aquecidos.

Isto é, conforme (NORONHA, 2003) em analogia a física da matéria: levando um cristal a sua temperatura de fusão, as moléculas estão desordenadas e se agitam livremente. Ao resfriar-se a amostra de maneira infinitamente lenta, as moléculas vão adquirir a estrutura cristalina estável que tem um nível de energia mais baixo possível.

Segundo (SOUZA; MARTINS; ARAÚJO, 2002) o processo se inicia com um membro qualquer do espaço de soluções, normalmente gerado aleatoriamente, e seleciona um de seus vizinhos randomicamente. Se este vizinho for melhor que o original ele é aceito e substitui a solução corrente. Se ele for pior por uma quantidade  $\Delta$ , ele é aceito com uma probabilidade  $e^{-\Delta/T}$ , onde  $T$  decresce gradualmente conforme o progresso do algoritmo. Esse processo é repetido até que  $T$  seja tão pequeno que mais nenhum movimento seja aceito. A melhor solução encontrada durante a busca é tomada como uma boa aproximação para a solução ótima. Originalmente, Simulated Annealing foi derivado de simulações em termodinâmica e por esta razão o parâmetro  $T$  é referenciado como temperatura e a maneira pela qual ela é reduzida é chamada de processo de resfriamento.

A figura 3 representa o pseudocódigo do algoritmo Simulated Annealing.

**Figura 3 – Representação do pseudocódigo do algoritmo Simulated Annealing**

```

procedimento SA( $f(\cdot)$ ,  $N(\cdot)$ ,  $\alpha$ ,  $SA_{max}$ ,  $T_0$ ,  $s$ )
1   $s^* \leftarrow s$ ;           {Melhor solução obtida até então}
2   $IterT \leftarrow 0$ ;       {Número de iterações na temperatura T}
3   $T \leftarrow T_0$ ;        {Temperatura corrente}
4  enquanto ( $T > 0$ ) faça
5      enquanto ( $IterT < SA_{max}$ ) faça
6           $IterT \leftarrow IterT + 1$ ;
7          Gere um vizinho qualquer  $s' \in N(s)$ ;
8           $\Delta = f(s') - f(s)$ ;
9          se ( $\Delta < 0$ )
10             então
11                  $s \leftarrow s'$ ;
12             se ( $f(s') < f(s^*)$ ) então  $s^* \leftarrow s'$ ;
13             senão
14                 Tome  $x \in [0, 1]$ ;
15                 se ( $x < e^{-\Delta/T}$ ) então  $s \leftarrow s'$ ;
16             fim-se;
17         fim-enquanto;
18          $T \leftarrow \alpha \times T$ ;
19          $IterT \leftarrow 0$ ;
20     fim-enquanto;
21      $s \leftarrow s^*$ ;
22 Retorne  $s$ ;
fim SA;
```

**Fonte: (SOUZA; MARTINS; ARAÚJO, 2002)**

Conforme (AARTS; KORST, 1988) a analogia com a otimização (combinatória ou não) é bastante direta. Os estados da matéria são as soluções realizáveis, a quantidade objetiva substitui a energia, os estados metaestáveis da matéria sendo ótimos locais e a estrutura cristalina corresponde ao ótimo global.

### 2.2.3 Algoritmos genéticos

Conforme cita (OLIVEIRA, 2005), os algoritmos genéticos foram introduzidos por (HOLLAND, 1975), com intuito de aplicar a teoria da evolução das espécies elaborada por (DARWIN, 1968) utilizando os conceitos da evolução biológica como genes, cromossomos, cruzamento, mutação e seleção na computação procurando explicar rigorosamente processos de adaptação em sistemas naturais e desenvolver sistemas artificiais (simulados em computador) que mantenham os mecanismos originais, encontrados em sistemas naturais.

Segundo (OLIVEIRA, 2005), o processo de evolução executado por um algoritmo genético corresponde a um procedimento de busca no espaço de soluções potenciais para o problema e, como enfatiza (MICHALEWICZ; SCHOENAUER, 1996), esta busca requer um equilíbrio entre dois objetivos aparentemente conflitantes: a procura das melhores soluções na região que se apresenta promissora ou fase de intensificação e a procura de outra região ou exploração do espaço de busca, também conhecida como diversificação.

Ainda segundo (OLIVEIRA, 2005), os algoritmos genéticos têm se mostrado ferramentas poderosas para resolver problemas onde o espaço de busca é muito grande e os métodos convencionais se mostraram ineficientes.

Mitchel (MITCHELL, 1998) cita que a terminologia biológica é muito importante para a compreensão do funcionamento dos algoritmos genéticos. Eis os principais termos:

- Cromossomo: estrutura que representa uma determinada característica da solução ou a própria solução;
- Gene: característica particular de um cromossomo. O cromossomo é composto por um ou mais genes.
- Alelo: valor de determinado gene;
- Locus: determinada posição do gene no cromossomo;
- Genótipo: estrutura que codifica uma solução. Um genótipo pode ser formado por um

ou mais cromossomos;

- Fenótipo: decodificação ou o significado da estrutura;
- Fitness: significa aptidão. O quanto o indivíduo é apto para determinado ambiente;

As principais características que diferenciam os algoritmos genéticos de métodos tradicionais são (GOLDBERG, 1989):

- Parâmetros: os algoritmos genéticos trabalham com a codificação dos parâmetros e não com os parâmetros propriamente;
- Número de soluções: os algoritmos genéticos trabalham com uma população de indivíduos (representando um conjunto de soluções) e não com uma única solução;
- Avaliação das soluções: os algoritmos genéticos utilizam informações de custo ou recompensa penalizando ou premiando determinadas características das soluções;
- Regras: os algoritmos genéticos utilizam regras probabilísticas e não determinísticas;

O algoritmo genético é uma forma da estratégia gerar-e-testar realizando os testes baseados nos parâmetros da evolução biológica. Uma desvantagem notável é a variação dos operadores genéticos do algoritmo em cada problema. Dessa forma, para resolução de determinado problema, torna-se necessário um estudo particular a respeito do mesmo.

O algoritmo genético atua sobre uma população fazendo com que esta evolua de acordo com uma função de avaliação. O funcionamento é iterativo iniciando com a geração de uma população inicial que pode ser aleatória ou não, seguida do processo de avaliação, seleção, cruzamento e mutação, que ocorre a cada iteração até que seja atingido algum critério de parada. Os passos gerais de um algoritmo genético são ilustrados na figura Figura XXXX. Cada passo pode ser realizado de várias maneiras e pode variar de problema para problema (TIMÓTEO, 2005).

Figura Etapas de um Algoritmo Genético Básico

## 2.3 TimeTabling

Segundo (LUEZUTE; KRIPKA, 2013) os problemas de programação de horários (PPH), também conhecidos como *TimeTabling*, são os problemas que mais se destacam nas organizações acadêmicas. De acordo com (SCHAERF, 1999) estes problemas são divididos em três categorias *school timetabling*, *course timetabling* e *examination timetabling*.

*School TimeTabling*: Se trata basicamente da geração de horários semanais, em escolas de segundo grau, onde deve-se evitar os choques entre os horários das disciplinas e que cada professor receba apenas uma turma para cada horário. Neste caso o aluno recebe um número fixo de disciplinas a serem cursadas.

*Course TimeTabling*: Diz respeito à alocação de aulas de uma universidade típica. Neste problema os alunos podem escolher as matérias em que vão se matricular, portando o problema tem como objetivo minimizar os possíveis choques entre as disciplinas, professores e horários disponibilizados pela instituição de ensino.

*Examination TimeTabling*: Aborda o problema de programação de horários dos exames da instituição, de maneira que, disciplinas que tenham alunos em comum, distanciem o máximo possível as datas dos exames.

Segundo (PINHEIRO; OLIVEIRA, 2001) o problema de programação de horários vem sendo abordado desde a década de 60, sendo que os primeiros trabalhos a se destacarem foram realizados na década de 80.

O Problema de Alocação de Salas (PAS) também conhecido como *Classroom Assignment* é tratado como parte do problema de programação de cursos universitários *course time-tabling*. Segundo (MARINHO, 2005) varias instituições universitárias se deparam com o PAS durante o início de cada semestre letivo, este problema é considerado NP-Difícil por (EVEN; ITAI; SHAMIR, 1975) e (CARTER; TOVEY, 1992), com isto, a determinação da solução ótima do problema, em um período de tempo aceitável se torna uma tarefa difícil.

Segundo (LUEZUTE; KRIPKA, 2013) o problema deve considerar que as disciplinas dos cursos universitários já tenham seus horários de início e de término definidos. O problema se resume então na alocação das disciplinas às salas desta universidade respeitando os horários destas disciplinas e as demais restrições exigidas.

Segundo (SOUZA, 2000) boa parte das universidades ainda resolvem este problema de forma manual, o que torna o processo árduo e demorado, podendo levar vários dias para ser concluído.

Uma vez que é de extrema dificuldade encontrar a solução ótima do PAS em tempo razoável, este problema é normalmente tratado através de técnicas heurísticas, que apesar de não garantirem encontrar a solução ótima do problema, são capazes de retornar uma solução de qualidade em um tempo adequado. As meta-heurísticas surgiram como uma alternativa para amenizar a dificuldade que os métodos heurísticos tem de escapar dos chamados ótimos

locais.(NASCIMENTO; SILVA; ALVARENGA, 2005).

Segundo (EVEN; ITAI; SHAMIR, 1975) o PAS pertence a classe de Problemas de Otimização Combinatória (POC).

## **2.4 Trabalhos Relacionados**

trabalho do marinho que usa tabu. trabalho da silvia que usa Recozimento Simulado (Simulated Annealing). trabalho da leonardo que usa Algoritmos Genéticos (AG). achar algum trabalho que utiliza o algoritmo das formigas. falar porque o trabalho do cara se assemelha ao meu trabalho.



todo início de semestre na instituição de ensino escolhida.

PQ do algoritmo genetico segundo renand dupas a escolha de algoritimos geneticos se dá através de uma comparação com outras técnicas e através disso de uma evidenciação das vantagens de sua utilização , logo o algoritmo genetico foi escolhido para este trabalho.

População inicial = –UMA NOVA ABORDAGEM PARA AUMENTAR A DIVERSIDADE.pdf

modelo matematico —APLICAÇÃO DE MODELO MATEMÁTICO, ABORDAGEM HEURÍSTICA E MÉTODO MISTO NA OTIMIZAÇÃO DA PROGRAMAÇÃO DE HORÁRIO DOS PROFESSORESTURMAS.pdf

restrições do problema e modelo matematico <http://www.dcc.ufla.br/infocomp/artigos/v4.3/art08.pdf>

Foi realizada uma análise dos requisistos atravez de entrevista com o stakeholder, após as entrevistas a modelagem de dados foi realizada de acordo com a demanda do projeto, para o desenvolvimento destes diagramas foi utilizadas a UML (Universal Modeling Language). Os seguintes diagramas foram desenhados, diagrama de caso de uso, diagrama de classe e diagrama de entidade relacionamento. Foram escolhidos os seguintes diagramas para que o sistema tenha uma documentação minima tendo em vista que o foco do trabalho é a resolução do problema de timetable através da utilização do algoritmo genetico.

Por se tratar de um sistema complexo, antes de iniciar a implementação do sistema fez-se necessário a sua modelagem. Segundo Elmasri Navathe (2005) as metodologias de modelagem de dados de objetos como UML (Universal Modeling Language – Linguagem de Modelagem Universal) estão se tornando cada vez mais populares no projeto e engenharia de software. Essas metodologias vão além do projeto de um banco de dados, especificando o projeto detalhado dos módulos de software e suas interações, utilizando vários tipos de diagramas.

### 3.1 Ferramentas Utilizadas

Este trabalho conta com a utilização de tecnologias proprias para o desenvolvimento de sistemas web, foram utilizadas as seguintes ferramentas: Para SGBD foi o utilizado PostgreSQL; No back-end foi utilizado Java e o *framework* Play!; No front-end as tecnologias utilizadas foram HTML, CSS, JavaScript e *framework* AngularJS e a IDE utilizada Eclipse.



### 3.1.1 Sistema de Gerenciamento de Banco de Dados

O SGBD escolhido foi o PostgreSQL pelo fato de ser uma ferramenta open-source e que trabalha perfeitamente com o framework escolhido Play!, uma vez que utilizado em projetos anteriores não foram apresentados conflitos entre o framework e o SGBD. A seguir pode ser notar que é uma ferramenta robusta e que tem visão no mercado internacional.

O PostgreSQL é um poderoso sistema gerenciador de banco de dados objeto-relacional de código aberto. Tem mais de 15 anos de desenvolvimento ativo e uma arquitetura que comprovadamente ganhou forte reputação de confiabilidade, integridade de dados e conformidade a padrões. Roda em todos os grandes sistemas operacionais. É totalmente compatível com ACID, tem suporte completo a chaves estrangeiras, junções (JOINS), visões, gatilhos e procedimentos armazenados (em múltiplas linguagens). Inclui a maior parte dos tipos de dados do ISO SQL:1999, incluindo INTEGER, NUMERIC, BOOLEAN, CHAR, VARCHAR, DATE, INTERVAL, e TIMESTAMP. Suporta também o armazenamento de objetos binários, incluindo figuras, sons ou vídeos. Possui interfaces nativas de programação para C/C++, Java, .Net, Perl, Python, Ruby, Tcl, ODBC, entre outros, e uma excepcional documentação.(POSTGRESQL, 2013)

Para a resolução do problema de timetable foi escolhido o algoritmo genético, a escolha do algoritmo foi devida a grande utilização do mesmo para resolução de problemas do tipo NP-difícil que foram encontrados na literatura.

### 3.1.2 Ferramentas Back-end

Foi escolhida uma linguagem de programação Java por ser orientada a objeto. Também foi escolhido o *Play! framework*, para que o desenvolvimento aconteça de forma mais rápida, fácil e eficiente.

Java foi criada pela Sun Microsystems para desenvolver inovações tecnológicas em 1992, time liderado por James Gosling. O Java utiliza do conceito de máquina virtual, onde existe, entre o sistema operacional e a aplicação, uma camada extra responsável por traduzir, mas não apenas isso - o que sua aplicação deseja fazer para as respectivas chamadas do sistema operacional, onde ela está rodando no momento. Sua aplicação roda sem nenhum envolvimento com o sistema operacional, sempre conversando apenas com a JVM - Java Virtual Machine

(CAELUM, 2013).

Em 2009 a Oracle comprou a Sun, fortalecendo a marca. A Oracle sempre foi, junto com a IBM, uma das empresas que mais investiram e fizeram negócios através do uso da plataforma Java. Em 2011 surge a versão Java 7 com algumas pequenas mudanças na linguagem (CAELUM, 2013).

The Play! É um moderno framework MVC de alta produtividade, que utiliza Java e Scala para o desenvolvimento web, open-source , utiliza templates, hibernate e JUnit em sua arquitetura. Existe duas versões do framework Play! 1 e Play2! este trabalho utiliza a versão 1 do framework(PLAY!, 2013).

### 3.1.3 Ferramentas Front-end

As ferramentas de Front-end descritas abaixo, foram escolhidas devida a grande utilização na web grande parte dos sites contem HTML, CSS ou JavaScript em algum trexo de seu código, foi escolhido tambem o framework AngularJS para que o desenvolvimento ocorra de maneira agil e mais rapida.

HTML que é defindo por (*HyperText Markup Language*) ou linguagem de marcação, é uma linguagem que é utilizada no desenvolvimento de paginas web (W3C, 2013 a).

Cascading Style Sheets (CSS) é uma tecnologia utilizada para adicionar estilos como cores, fontes, espaçamentos em documentos escritos em uma linguagem de marcação como exemplo o HTML (W3C, 2013 b).

JavaScript é uma linguagem de script utilizada no desenvolvimento de paginas na web, atualmente é a principal linguagem para programação client-side em navegadores web. Todas as paginas de HTML modernas estão usando JavaScript para adicionar funcionalidades e para se comunicar com os webServers(W3SCHOOLS, 2013).

Angularjs é um *framework JavaScript* construido e mantido pelo grupo de engenheiros do Google, ele usa o HTML como uma *template engine*, tudo isso no intuito de fornecer uma solução completa para o cliente-side de sua aplicação. Além disso tem total compatibilidade com as bibliotecas javascript mais utilizadas, como jQuery. É um novo conceito para desenvolvimento de web apps client-site.(MENDES, 2013)

### 3.1.4 IDE

O Eclipse é uma IDE (*integrated development environment*). Diferente de uma RAD(*Rapid Application Development*), onde o objetivo é desenvolver o mais rápido possível através do arrastar-e-soltar do mouse, onde montanhas de código são gerados em background, uma IDE te auxilia no desenvolvimento, evitando se intrometer e fazer muita mágica (CAELUM, 2013).

O Eclipse é a IDE líder de mercado. Formada por um consórcio liderado pela IBM, possui seu código livre. A última versão é a 4.3, mas com qualquer versão posterior a do 3.1 você terá suporte ao Java 5, 6 e 7 (CAELUM, 2013).

Esta IDE foi escolhida devido ao grande reconhecimento mundial, por sua eficiência ao se trabalhar com a linguagem de programação Java, por ser open-source e pela existência de várias ferramentas criadas pela comunidade, para o auxílio no desenvolvimento de softwares.

## 4 SISTEMA DESENVOLVIDO

O capítulo descreve o sistema desenvolvido neste trabalho, tem como objetivo o sistema, otimizar a alocação das disciplinas envidas pelos colegiados em salas de uma universidade, neste caso as salas do prédio FAFICH na UFMG. Por se tratar de um problema específico do local torna-se difícil o trabalho de encontrar tecnologias disponíveis para otimização do problema, neste caso o desenvolvimento de um sistema que atenda todas as necessidades é de grande importância para o responsável pela alocação.

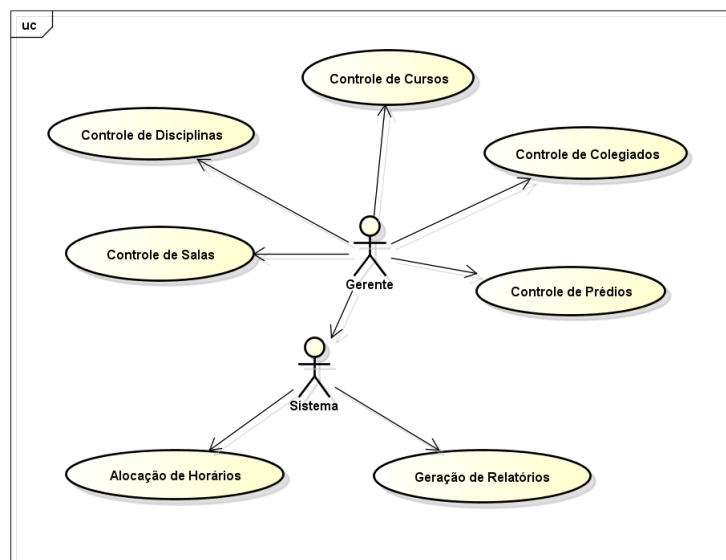
### 4.1 Modelagem

#### 4.1.1 Diagramas de caso de uso

A Figura XX descreve todas as funcionalidades que o sistema possui, essas funcionalidades foram divididas em 2 atores "Gerente" e "Sistema" cada um ligado com suas respectivas funcionalidades, porém, o "Gerente" pode acessar o ator "Sistema" para ter acessos funcionalidades que são encontradas no mesmo. O sentido das setas informa o que cada ator pode acessar no sistema.

As funcionalidades controle de turnos, controle de horários, controle de prédios, controle de salas, controle de cursos, controle de colegiados, controle de períodos e controle de disciplinas são disponibilizadas através de módulos CRUD para que o responsável tenha total controle sobre o que será alocado, onde e como.

Já as funcionalidades alocação de horários e geração de relatórios, são rotinas executadas pelo "Sistema". A funcionalidade alocação de horários é uma rotina que utiliza conceitos de algoritmo genético para encontrar a melhor solução do problema e a funcionalidade geração de relatórios mostra para o "Gerente" o melhor resultado de alocação encontrado pelo algoritmo.

**Figura 6 – Diagrama de Caso de Uso**

**Fonte: Desenvolvido pelo autor**

#### 4.1.2 Diagrama de Entidade Relacionamento

Na Figura XX mostra o relacionamento das tabelas no sistema através do diagrama de entidade relacionamento.

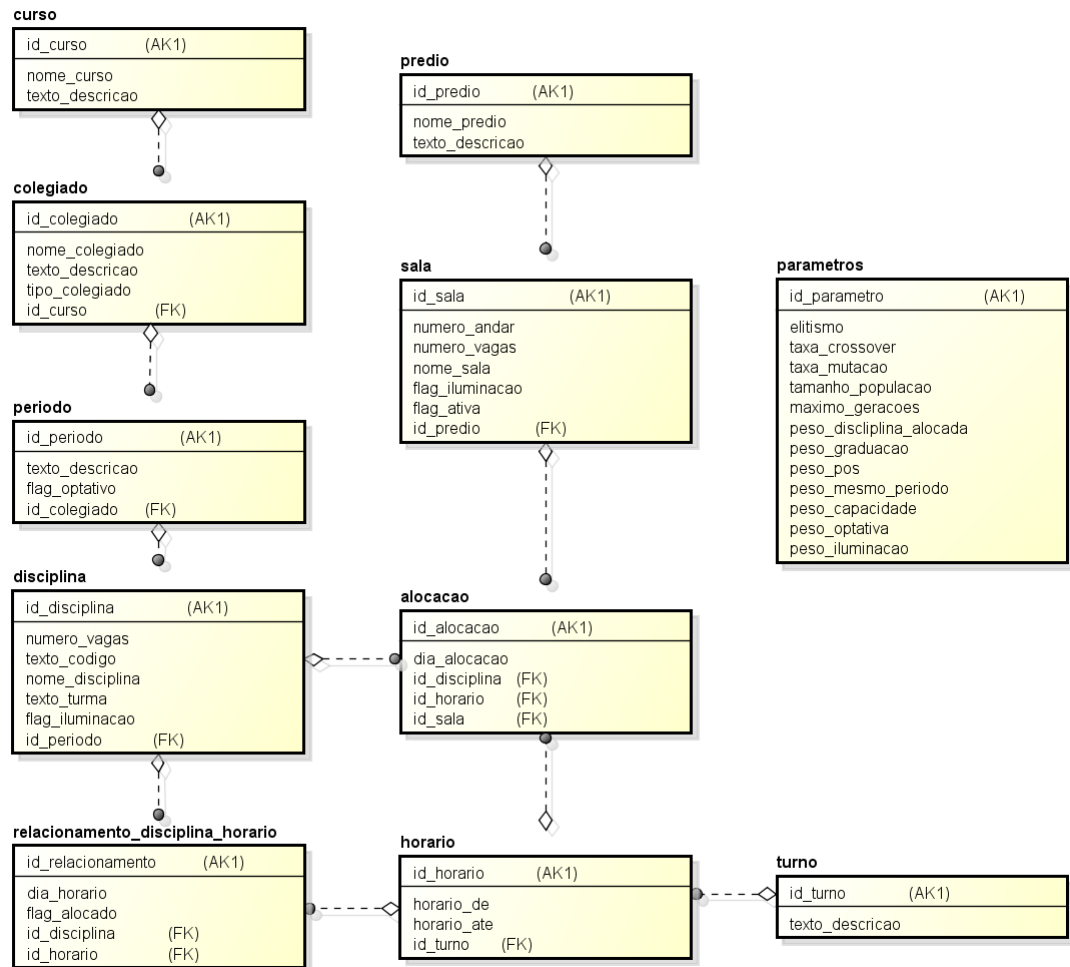
As tabelas curso, colegiado, período, disciplina, sala, prédio, turno, horário são utilizadas para armazenar as informações dos respectivos objetos que são controlados pelo CRUD e suas respectivas funcionalidades.

A tabela relacionamento\_disciplina\_horário, salva as obrigatoriedades dos horários das disciplinas uma vez que todos os horários são montados pelos colegiados e não pelo responsável pela alocação das disciplinas nas salas.

A tabela alocação contém o relacionamento de disciplina, horário e sala o que corresponde a melhor alocação encontrada pelo algoritmo. O item disciplina pode ter o seu valor como nulo o que significa que em um específico horário e em uma determinada sala não existe disciplina alocada.

A tabela parâmetro é utilizada apenas para guardar os últimos parâmetros utilizados na execução do algoritmo genético.

**Figura 7 – Diagrama de Entidade Relacionamento**



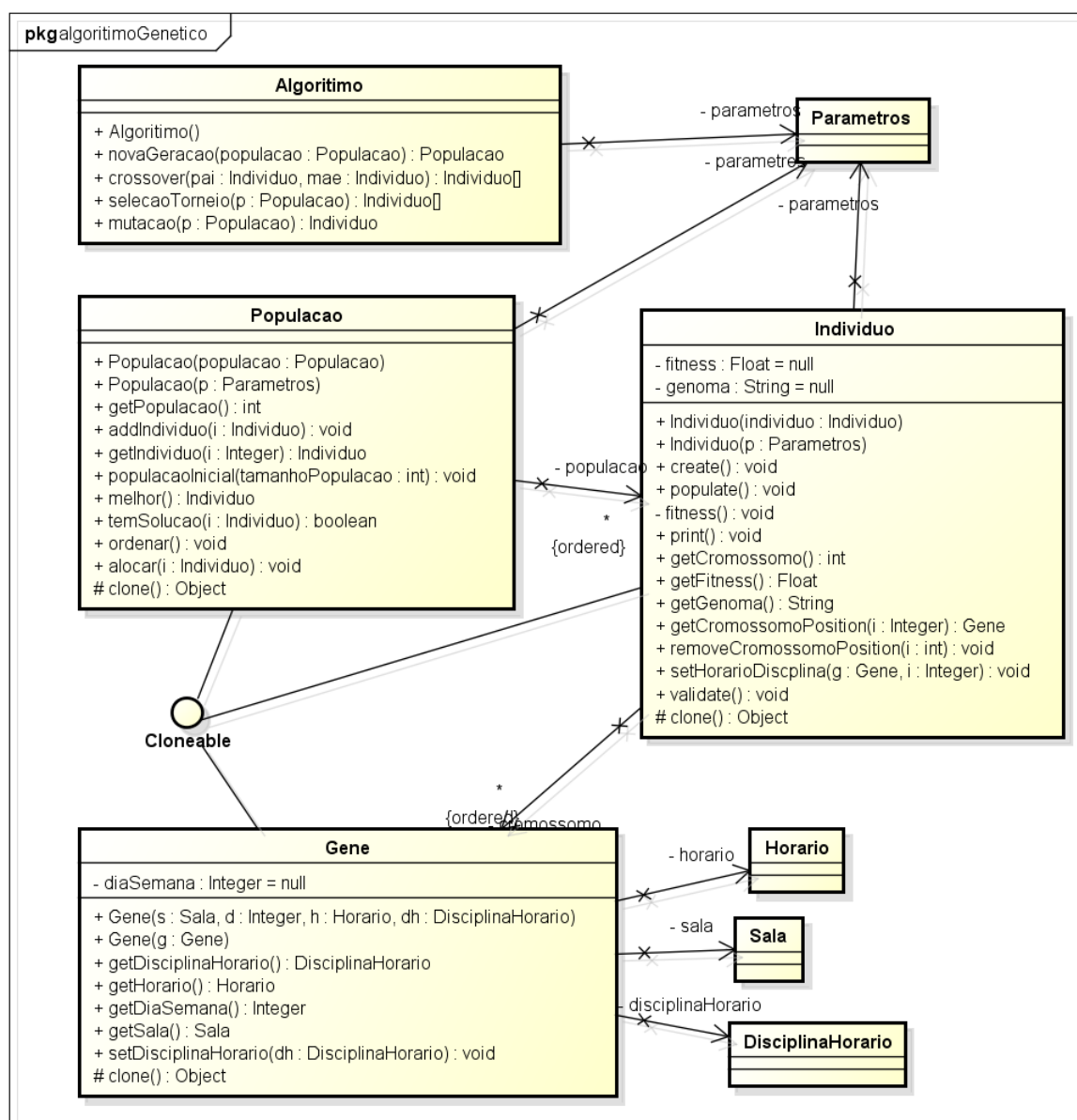
Fonte: Desenvolvido pelo autor

#### 4.1.3 Diagramas de classe

Este diagrama é das classes do algoritmo genético, as classes do sistema em geral não serão abordados. Quando terminar o código revisar e explicar.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Donec vestibulum mauris at velit varius aliquet. Nulla at risus vehicula, tempus orci sagittis, molestie nibh. Fusce sodales sollicitudin viverra. Aliquam erat volutpat. Nullam nec odio mi. Suspendisse vel mauris felis. In mi diam, auctor vel feugiat ut, pulvinar ac ipsum. In a convallis arcu. Integer a augue accue rutrum mauris. Phasellus id massa a lorem semper placerat eget eu urna.

### Figura 8 – Diagrama de Classe



**Fonte: Desenvolvido pelo autor**

## 4.2 Algoritmo Genético

Após a execução da modelagem do sistema com pleno conhecimento do problema e o levantamento bibliográfico sobre algoritmos genéticos, foi feito o relacionamento do problema com os termos da biologia. Foram encontradas varias fontes que agregaram valor para o desenvolvimento do trabalho, porem algumas modificação foram feitas para que o modelo tratado por outros autores funcionasse adequadamente para a resolução do problema proposto para este trabalho. A seguir serão apresentados os itens da biologia utilizados no desenvolvimento do

algoritmo juntamente com sua ligação com o problema.

#### 4.2.1 *Indivíduo*

Alguns trabalhos tratam os termos indivíduo e cromossomo como a mesma representação biológica, neste trabalho o termo indivíduo é composto pelo cromossomo juntamente com a pontuação adquirida após a execução do método de calculo de fitness, já o termo cromossomo se refere a combinação dos genes do indivíduo, a figura XX demonstra a representação de um indivíduo.

**Figura 9 – Representação Indivíduo**

Indivíduo								
Fitness	Cromossomo							
12.0	1	null	null	3	4	null	...	7

Fonte: Desenvolvido pelo autor

Um cromossomo é uma sequência de genes, neste trabalho esta sequência tem um tamanho fixo e pode ser medido pela seguinte formula (número de Salas \* número de Horários \* número de dias da semana). Inicialmente o individuo contem todos as variáveis relacionamentos de obrigatoriedade nulas, estas variáveis serão preenchidas randomicamente na criação da população inicial que em breve será explicada. O cromossomo do indivíduo preenchido com os relacionamentos de obrigatoriedades representa uma alocação, está alocação é medida pela sua pontuação de fitness, podendo ser ou não o resultado do problema. Os valores utilizados na representação do cromossomo na figura XX são os ID's do relacionamento de obrigatoriedade entre horário e disciplina, em caso de horário vago em um determinado gene esta variável terá o valor nulo.

**Figura 10 – Representação Cromossomo**

Cromossomo							
1	null	null	3	4	null	...	7

Fonte: Desenvolvido pelo autor

O termo gene representado pela figura XX possui uma combinação de quatro variáveis sala, dia da semana, horário e o relacionamento de obrigatoriedade "disciplina horário". As



variáveis sala, horário e dia da semana são fixas, não podem ser nulas uma vez que todas as combinações possíveis destas três variáveis formam um cromossomo que tem um tamanho fixo para todos os indivíduos.

Um gene com o relacionamento "disciplina horário" igual a nulo, representa um horário vago para aquela combinação específica de sala, horário e dia da semana.

**Figura 11 – Representação Gene**

Gene			
Sala	Dia da Semana	Horário	Relacionamento
2	3	4	2

Gene			
Sala	Dia da Semana	Horário	Relacionamento
2	3	5	null

**Fonte: Desenvolvido pelo autor**

O fitness do indivíduo é calculado através da função objetivo que será abordada em outro tópico brevemente explicando se trata de um valor que vai de 0 a 100 onde 100 é a pontuação máxima de do indivíduo está nota é alcançada quando todos os requisitos de alocação desejados foram atendidos.

#### 4.2.2 População

Uma população, quando relacionada aos termos genéticos se trata de um conjunto de indivíduos, também representa uma interação do algoritmo genético, ou seja uma geração. A manipulação da população e de suas propriedades é feita através de parâmetros enviados antes da execução do algoritmo, estes parâmetros são elitismo, taxa de crossover, taxa de mutação, tamanho da população e número máximo de gerações.

De acordo com os parâmetros passados é criada a população inicial. Para cada indivíduo criado é utilizado um método para inserir randomicamente todos os registros de relacionamento de obrigatoriedade entre disciplina e horário em cada um dos genes que previamente foram criados como nulo, estes indivíduos são criados até a população atinga o tamanho da população que deve ser igual ao parâmetro tamanho da população.

Para cada geração é criada uma nova população a partir da população criada na geração anterior, se o operador genético elitismo estiver como valor verdadeiro, iniciamos esta nova

população com 20% dos melhores indivíduos da população anterior, os melhores indivíduos de uma População são indicados pelas maiores pontuações de fitness de da população anterior.

Durante a criação da nova população podemos ter duas operações ocorrendo mutação e crossover, para a execução destes operadores genéticos são utilizadas porcentagens enviadas pelos parâmetros do algoritmo. Para que estas operações genéticas aconteçam são utilizados valores randomicos para serem comparados com as taxas de mutação e crossover. Em cada interação da criação desta nova população são selecionados por torneio os pais que serão utilizados no crossover, se a condição da taxa de execução for verdadeira os pais serão descartados e os filhos gerados a partir dos genes dos pais, serão adicionados na nova população, em caso de falso os pais serão adicionados na nova população.

Para se utilizar a operação de mutação novamente é utilizado outro valor randomico se a condição for verdadeira um indivíduo da população anterior é selecionado e sofrerá a mutação genética, o individuo antes da mutação genética é descartado e o novo indivíduo que sofreu a mutação é adicionado na nova população. O fluxo de uma nova população é descrito na figura XX.

#### 4.2.3 Operadores genéticos

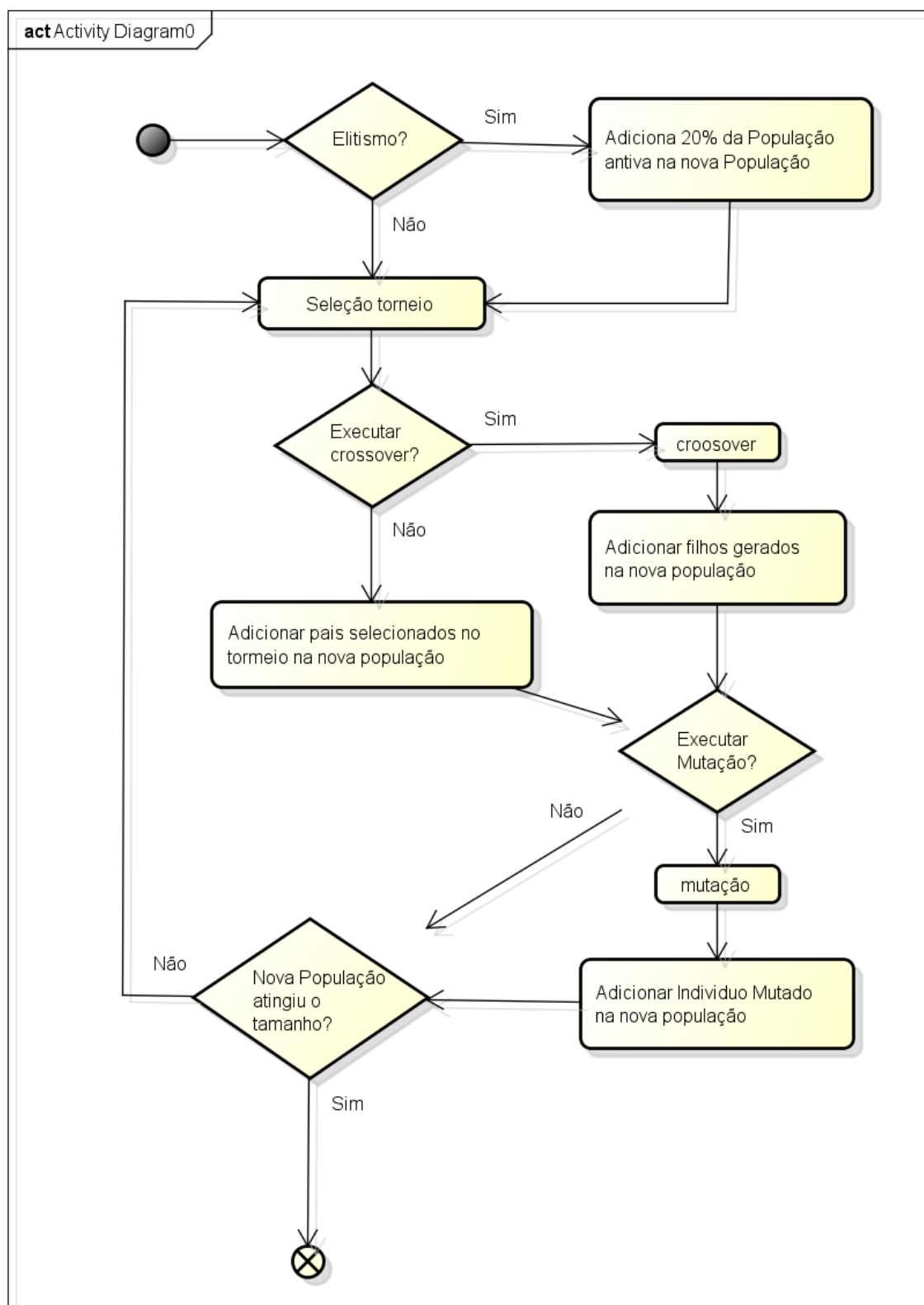
Este trabalho utiliza elitismo como operador genético ao se inciar uma nova população, vinte por cento dos melhores indivíduos são escolhidos para compor a nova população. O elitismo é calculdo atravez da função objetiva criada para o problema especifico do trabalho.

Para selecionar os inviduos para realizar o crossover é utilizado metodo de seleção por torneio, são escolhidos três Indivíduos da população anterior, e destes três são escolhidos os dois com maior pountuação de fitness, os dois Indivíduos selecionados são enviados para o crossover.

Mutacao é a inversão genetica dos genes de um Indivíduo escolhido randomicamente da população anterior, apos a realização da mutação genetica o individuo é inserido na nova população.

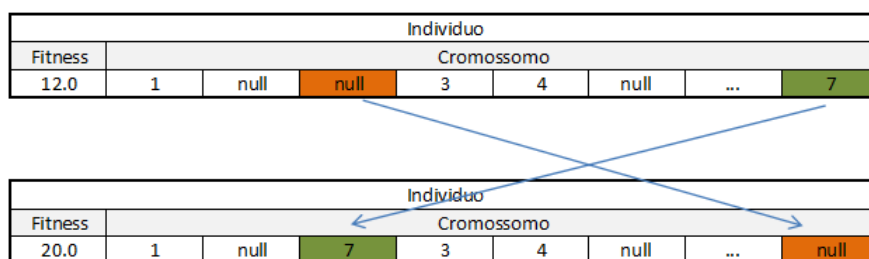
Primeiramente são escolhidos dois genes do cromossomo, apos a escolha randomica dos itens a serem trocados, é feita a troca dos genes e retornado um ndividuo que tem a composição genetica apos a alteração, Apos a mutação este Indivíduo recebe uma nova nota de fitness de

**Figura 12 – Fluxo Nova População**



Fonte: Desenvolvido pelo autor

acordo com a sua nova sequência de Genes e sua adaptação no ambiente, está nota pode ser maior ou menor do que a anterior.

**Figura 13 – Representação Mutação**

Fonte: Desenvolvido pelo autor

Crossover é o cruzamento dos individuos selecionados pela seleção torneio. Explicar com figuras. apos o cruzamento dos individuo os dois filhos gerados são inseridos na nova população.

#### 4.2.4 Definição da função objetivo

Somatorio disso

Para o calculo do fitness foram definidos pesos para modelagem da função, estes pesos podem ser configurados de acordo com a necessidade da alocação.

Graduação alocada ganha 05 de peso

Pós graduação alocada ganha 03 de peso

Periodos na mesma sala cada um ganha 05 de peso \* o numero do periodo

Quandtidade de vagas igual a da sala 05 de peso

não optativa ganha 5

optativa ganha 3

iliminacao atendida 5

Criar a função matematica com as legendas conforme o trabalho 117.pdf

Falar o numero de salas, o numero de horarios, o numero de curos o numero de colegia- dos o numero de periodos o numero de disciplinas para cada colegiado.....

restrições

falar um pouco das restrições e enumeralas

As disciplinas não podem ser alocadas em horarios direfentes dos que já foram pré definidos pelo colegiado.

As disciplinas devem ter apenas a quantidade de alocações necessárias.

As disciplinas devem respeitar a capacidade da sala.

As disciplinas não optativas tem preferencia de alocação na mesma sala.

Preferencias por salas claras ou escuras

Restrição 1

Fitness

Para se iniciar o calculo do fitness são verificados todos os horarios já alocados somando os pesos se adequados.

para cada gene

se tem horario alocado

horario bate

capacidade da turma

turma graduacao

optativa

iluminacao

soma tudo

fim se tem alocação

soma tudo

fim para cada gene

Calculo do fitness01  $\text{fitness01} = \text{somatorio} \times 100 / \text{colocar algum valor para dividir}$  não sei ainda

calculo fitness02 penaliza disciplinas com mais alocação do que se deve

para cada gene

para cada disciplina

soma

fim

Calculo fitness02  $= \text{fitness01} \times (1 - (\text{total alocados} - \text{total necessario} / \text{dividir pelo numero possivel de alocações}))$  fim

calculo fitness03 penalidade por capacidade

para cada gente

se a sala tiver capacidade diferente

fim

calculo fitness03  $= \text{fitness02} \times (1 - (\text{numero de erros} / \text{numero de possiveis alocações}))$

calcula fitness04 preferências clara ou escura

para cada gene

se tiver com o optativo errado

fim

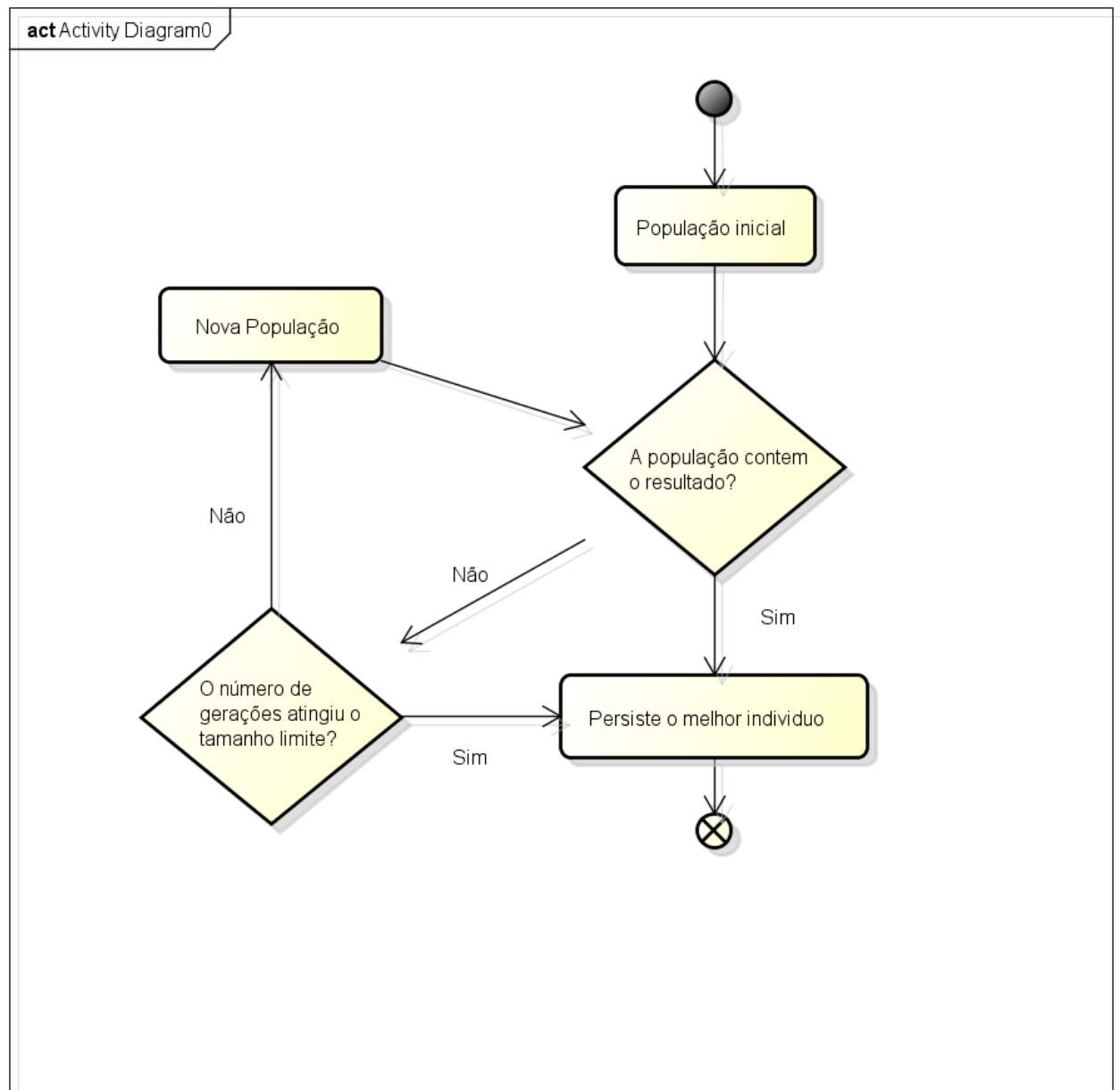
$\text{fitness04} = \text{fitness03} \times (1 - (\text{numero de erros} / \text{numero de possíveis alocações}))$

O fitness04 é o resultado final

#### 4.2.5 Fluxo do algoritmo

parametros Algoritmo

O fluxo do algoritmo conforme a imagem XX é iniciado pela criação da população inicial, para cada interação do algoritmo é verificado se a população contém o resultado e se o algoritmo não atingiu o número de gerações pré definidas. Se as duas condições forem falsas o algoritmo cria uma nova População de acordo com a figura XX

**Figura 14 – Fluxo Algoritmo**

Fonte: Desenvolvido pelo autor

## **5 RESULTADOS OBTIDOS** Depois do sistema implementado

Entrada processamento e saída

ajuste do algoritmo de alocação

falar do algoritmo que não evolui sem o crossover so com mutação.



## **6 CONSIDERAÇÕES FINAIS**

\*Relembra ro problema e comprar como o resultadoo bjetido.

Discussão dos resultados obtidos na pesquisa, onde se verificam as observações pessoais do autor. Poderá também apresentar sugestões de novas linhas de estudo. A conclusão deve estar de acordo com os objetivos do trabalho. A conclusão não deve apresentar citações ou interpretações de outros autores.

### **6.1 Trabalhos futuros**

Criar um DW para geração dos relatorios de acordo com a dimensão escolhida.

Utilização de outros algoritimos para a resolução do problema ex. algoritimos evolutivos formiga entre outros.

Pegar o feed back do usuario para melhoria na interface, e do algoritimo.

## REFERÊNCIAS

- AARTS, E.; KORST, J. Simulated annealing and boltzmann machines. New York, NY; John Wiley and Sons Inc., 1988.
- ARMENTANO, V. A.; BRANCHINI, R. M. Uma introdução à busca tabu. 2013.
- CAELUM. *Apostila do curso FJ-11 - Java e Orientação a Objetos*. 2013. Disponível em: <<http://www.caelum.com.br/apostila-java-orientacao-objetos>>. Acesso em: 29 set. 2013.
- CARTER, M. W.; TOVEY, C. A. When is the classroom assignment problem hard? *Operations Research*, INFORMS, v. 40, n. 1-Supplement-1, p. S28–S39, 1992.
- CISCON, L. A. O problema de geração de horários: Um foco na eliminação de janelas e aulas isoladas. 2006.
- DARWIN, C. On the origin of species by means of natural selection. 1859. *See also*: <http://www.literature.org/authors/darwin-charles/the-origin-of-species>, 1968.
- EVANS, J. R.; MINIEKA, E. *Optimization algorithms for networks and graphs*. [S.l.]: CRC Press, 1992.
- EVEN, S.; ITAI, A.; SHAMIR, A. On the complexity of time table and multi-commodity flow problems. In: IEEE. *Foundations of Computer Science, 1975., 16th Annual Symposium on*. [S.l.], 1975. p. 184–193.
- GLOVER, F. Future paths for integer programming and links to artificial intelligence. *Computers & Operations Research*, Elsevier, v. 13, n. 5, p. 533–549, 1986.
- GOLBARG, M. C.; LUNA, H. P. L. Otimização combinatória e programação linear. *Rio de Janeiro: Campus*, 2000.
- GOLDBERG, D. Genetic algorithms in optimization, search and machine learning. *Addison Wesley, New York. Eiben AE, Smith JE (2003) Introduction to Evolutionary Computing. Springer. Jacq J, Roux C (1995) Registration of non-segmented images using a genetic algorithm. Lecture notes in computer science*, v. 905, p. 205–211, 1989.
- HOLLAND, J. H. *Adaptation in natural and artificial systems: An introductory analysis with applications to biology, control, and artificial intelligence*. [S.l.]: U Michigan Press, 1975.
- KIRKPATRICK, S.; JR., D. G.; VECCHI, M. P. Optimization by simulated annealing. *science*, Washington, v. 220, n. 4598, p. 671–680, 1983.
- LUVEZUTE, R. M.; KRIPKA, K. M. Simulated annealing aplicado ao problema de alocação de salas com deslocamentos mínimos. 2013.
- MARINHO, E. *Heurísticas busca tabu para o problema de programação de tripulações de ônibus urbano*. Tese (Doutorado) — Master's Thesis, Universidade Federal Fluminense, 2005.

- MENDES, W. *AngularJS um framework para facilitar sua vida*. 2013. Disponível em: <<http://www.slideshare.net/WilsonMendes/angularjs-um-framework-para-facilitar-sua-vida>>. Acesso em: 29 set. 2013.
- MICHALEWICZ, Z.; SCHOENAUER, M. Evolutionary algorithms for constrained parameter optimization problems. *Evolutionary computation*, MIT Press, v. 4, n. 1, p. 1–32, 1996.
- MITCHELL, M. An introduction to genetic algorithms (complex adaptive systems). A Bradford Book, 1998.
- NASCIMENTO, A. S.; SILVA, R. M. S.; ALVARENGA, G. B. Uma aplicação de simulated annealing para o problema de alocação de salas. *INFOCOMP Journal of Computer Science*, v. 4, n. 3, p. 59–66, 2005.
- NORONHA, T. Uma abordagem sobre estratégias metaheurísticas. 2000. *Projeto Orientado–Universidade Federal do Rio Grande do Norte. Rio Grande do Norte. Disponível em: <http://www.sbc.org.br/reic/edicoes/2001e1/cientificos/UmaAbordagemSobreEstrategiasMetaheuristicas.pdf>*, 2003.
- OLIVEIRA, A. C. de. Uso do algoritmo genético e recozimento simulado para o problema de alocação de salas. *Monografia, Departamento de Ciência da Computação, Universidade Federal de Lavras*, 2006.
- OLIVEIRA, H. Algoritmo evolutivo no tratamento do problema de roteamento de veículos com janela de tempo. *Monografia, Departamento de Ciência da Computação, Universidade Federal de Lavras*, 2005.
- PINHEIRO, P.; OLIVEIRA, J. A. Um ambiente de apoio a construção de horário escolar na web: modelagem, implementação e aplicação nas escolas de ensino médio. *XXXIII Simpósio Brasileiro de Pesquisa Operacional”, Campos do Jordão, SP*, 2001.
- PLAY! *The High Velocity Web Framework For Java and Scala*. 2013. Disponível em: <<http://www.playframework.com/>>. Acesso em: 29 set. 2013.
- POSTGRESQL. *Sobre o PostgreSQL*. 2013. Disponível em: <<http://www.postgresql.org.br/sobre>>. Acesso em: 29 set. 2013.
- RAO, S. S. *Optimization : theory and applications*. New Delhi: Wiley Eastern, 1984. ISBN 0-85226-780-0. Disponível em: <<http://opac.inria.fr/record=b1092847>>.
- RAUPP, M. Introdução à otimização linear. *LNCC, Rio de Janeiro. Notas de Aulas, Curso de Verão LNCC*, 2003.
- SCHAERF, A. A survey of automated timetabling. *Artificial intelligence review*, Springer, v. 13, n. 2, p. 87–127, 1999.
- SOUZA, M. J. F. Programação de horários em escolas: uma aproximação por metaheurísticas. *Rio de Janeiro*, 2000.

SOUZA, M. J. F.; MARTINS, A. X.; ARAÚJO, C. R. d. Experiências com simulated annealing e busca tabu na resolução do problema de alocação de salas. 2002.

STEIGLITZ, K.; PAPADIMITRIOU, C. H. Combinatorial optimization: Algorithms and complexity. *Prentice Hall, New Jersey., UV Vazirani (1984). On two geometric problems related to the travelling salesman problem. J. Algorithms*, v. 5, p. 231–246, 1982.

SUBRAMANIAN, A. et al. Aplicação da metaheurística busca tabu na resolução do problema de alocação de salas do centro de tecnologia da ufpb. *Anais do XXVI Encontro Nacional de Engenharia de Produção*, p. 1, 2006.

TIMÓTEO, G. T. S. *Desenvolvimento de um Algoritmo Genético para a Resolução do Timetabling*. 2005.

W3C. *HTML 4.01 Specification*. 2013 a. Disponível em: <<http://www.w3.org/TR/html4>>. Acesso em: 29 set. 2013.

W3C. *Cascading Style Sheets*. 2013 b. Disponível em: <<http://www.w3.org/Style/CSS>>. Acesso em: 29 set. 2013.

W3SCHOOLS. *JavaScript Tutorial*. 2013. Disponível em: <<http://www.w3schools.com/js/>>. Acesso em: 29 set. 2013.